

# Kollmorgen Automation Suite

## KAS IDE User Guide



**Document Edition: U, December 2021**

**Valid for KAS Software Revision 3.07**

Valid for AKD firmware version: 01-21-00-000

Valid for AKD2G firmware version: 02-08-00-000

Part Number: 959715



For safe and proper use, follow these instructions. Keep for future use.

# Trademarks and Copyrights

## Copyrights

Copyright © 2009-2021 Kollmorgen

Information in this document is subject to change without notice. The software package described in this document is furnished under a license agreement. The software package may be used or copied only in accordance with the terms of the license agreement.

This document is the intellectual property of Kollmorgen and contains proprietary and confidential information. The reproduction, modification, translation or disclosure to third parties of this document (in whole or in part) is strictly prohibited without the prior written permission of Kollmorgen.

## Trademarks

- KAS and AKD are registered trademarks of [Kollmorgen](#).
- [Kollmorgen](#) is part of the [Altra Industrial Motion](#) Company.
- EnDat is a registered trademark of Dr. Johannes Heidenhain GmbH
- EtherCAT is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH
- Ethernet/IP is a registered trademark of ODVA, Inc.
- Ethernet/IP Communication Stack: copyright (c) 2009, Rockwell Automation
- HIPERFACE is a registered trademark of Max Stegmann GmbH
- PROFINET is a registered trademark of PROFIBUS and PROFINET International (PI)
- SIMATIC is a registered trademark of SIEMENS AG
- Windows is a registered trademark of Microsoft Corporation
- [PLCopen](#) is an independent association providing efficiency in industrial automation.
- Codemeter is a registered trademark of [WIBU-Systems AG](#).
- SyCon® is a registered trademark of [Hilscher GmbH](#).

Kollmorgen Automation Suite is based on the work of:

- [7-zip](#) (distributed under the terms of the LGPL and the BSD 3-clause licenses - [see terms](#))
- The [C++ Mathematical Expression Library](#) (distributed under [the MIT License](#))
- [curl](#) software library
- [JsonCpp](#) software (distributed under the MIT License - [see terms](#))
- [Mongoose](#) software (distributed under the MIT License - [see terms](#))
- [Qt](#) cross-platform SDK (distributed under the terms of the LGPL3; Qt source is available on KDN)
- [Qwt](#) project (distributed under the terms of the [Qwt License](#))
- [U-Boot](#), a universal boot loader is used by the AKD PDMM and PCMM (distributed under the [terms](#) of the GNU General Public License). The U-Boot source files, copyright notice, and readme are available on the distribution disk that is included with the AKD PDMM and PCMM.
- [Zlib](#) software library

All other product and brand names listed in this document may be trademarks or registered trademarks of their respective owners.

## Disclaimer

The information in this document (Version U published on 11/30/2021) is believed to be accurate and reliable at the time of its release. Notwithstanding the foregoing, Kollmorgen assumes no responsibility for any damage or loss resulting from the use of this help, and expressly disclaims any liability or damages for loss of data, loss of use, and property damage of any kind, direct, incidental or consequential, in regard to or arising out of the performance or form of the materials presented herein or in any software programs that accompany this document.

All timing diagrams, whether produced by Kollmorgen or included by courtesy of the PLCopen organization, are provided with accuracy on a best-effort basis with no warranty, explicit or implied, by Kollmorgen. The user releases Kollmorgen from any liability arising out of the use of these timing diagrams.

# 1 Table of Contents

---

<b>1</b>	<b>Table of Contents</b>	<b>3</b>
<b>2</b>	<b>Document History</b>	<b>20</b>
<b>3</b>	<b>Preface</b>	<b>23</b>
<b>3.1</b>	<b>System Requirements for KAS IDE and KAS Simulator</b>	<b>23</b>
<b>3.2</b>	<b>Learning Kollmorgen Automation Suite</b>	<b>24</b>
3.2.1	Where to Start	24
3.2.1.1	Beginner	24
3.2.1.2	Advanced User	24
3.2.1.3	Online	24
3.2.2	Read the KAS Manuals	24
<b>4</b>	<b>Introducing Kollmorgen Automation Suite</b>	<b>26</b>
<b>4.1</b>	<b>Key Features</b>	<b>28</b>
4.1.1	Integrated Development Environment	30
4.1.2	KAS Runtime	31
4.1.3	KAS Simulator	32
<b>4.2</b>	<b>Looking at Kollmorgen Automation Suite</b>	<b>33</b>
4.2.1	Physical View	33
4.2.2	Logical View	34
4.2.3	Architectural View	35
<b>4.3</b>	<b>KAS Breakdown</b>	<b>37</b>
4.3.1	Human-Machine Interface	39
4.3.2	Controllers	40
4.3.2.1	AKD PDMM and PCMM Hardware	40
4.3.2.2	PDMM or PCMM Rotary Switch	41
4.3.2.3	PDMM & PCMM Web Server	41
4.3.2.4	Controller Performance	42
4.3.3	Communication and Fieldbus	45
4.3.3.1	Fieldbus	45
4.3.3.2	Motion bus	45
4.3.4	Input Output Bus	45
4.3.5	Drives	46
4.3.6	Kollmorgen Motors	46
<b>4.4</b>	<b>Different Implementations</b>	<b>47</b>
4.4.1	Single-Axis with PDMM	47
4.4.2	Multi-Axis with PDMM or PCMM	48
<b>5</b>	<b>Describing KAS Graphical User Interface</b>	<b>49</b>
<b>5.1</b>	<b>Windows and Panels Overview</b>	<b>49</b>
5.1.1	Main Window	49
5.1.1.1	About toolboxes	50
5.1.2	Project Explorer	51
5.1.2.1	System	52
5.1.2.2	Controller	53
5.1.2.3	HMI Device	62
5.1.2.4	SafePLC2 Project	62

---

5.1.3 Libraries .....	66
5.1.3.1 Function Blocks .....	66
5.1.3.2 Controls .....	66
5.1.3.3 Properties .....	66
5.1.4 Dictionary .....	67
5.1.4.1 Variables tab .....	67
5.1.4.2 Enum Tab .....	81
5.1.4.3 Bit Fields Tab .....	83
5.1.5 Information and Logs .....	85
5.1.5.1 Log Messages .....	85
5.1.5.2 Log Messages Settings .....	87
5.1.5.3 Find and Replace Tab .....	92
5.1.5.4 Find and Replace Operations .....	95
5.1.5.5 Browse Variable Tab .....	97
5.1.5.6 Breakpoints tab .....	99
5.1.5.7 Compiler Output .....	100
5.1.6 Watch Window .....	102
5.1.6.1 Multiple Watch Windows .....	102
5.1.6.2 Access Structure and Arrays .....	103
5.1.6.3 Add Variable .....	103
5.1.6.4 Add an Expression .....	104
5.1.6.5 Force a Variable .....	106
5.1.7 AKD Drive .....	108
5.1.7.1 WorkBench Toolbar .....	108
5.1.7.2 WorkBench Status Bar .....	108
5.1.8 Status Bar .....	109
5.1.8.1 Local Version .....	109
5.1.8.2 Controller Version .....	109
5.1.8.3 Drives State .....	109
5.1.8.4 Controller State .....	110
5.1.8.5 Connection State .....	110
5.1.8.6 Color Codes .....	110
<b>6 Getting Started .....</b>	<b>112</b>
<b>6.1 KAS IDE to Runtime Compatibility .....</b>	<b>112</b>
<b>6.2 KAS PLC Code to Runtime Compatibility .....</b>	<b>113</b>
<b>6.3 KAS IDE to Project File Compatibility .....</b>	<b>114</b>
<b>6.4 Starting the KAS IDE .....</b>	<b>115</b>
6.4.1 Find KAS IDE Version .....	116
6.4.2 Access Help System .....	116
6.4.3 KAS GUI .....	117
6.4.4 KAS IDE Log Window .....	117
6.4.4.1 Log Information .....	117
6.4.5 Create a Project .....	117
6.4.6 Add and Configure a Controller .....	118
6.4.6.1 Add the Controller .....	118
6.4.6.2 Configure the Controller .....	119
6.4.7 Configure EtherCAT Motion Bus .....	122

6.4.7.1 EtherCAT Scan and Association .....	122
6.4.8 Overview: Create Programs .....	139
6.4.8.1 Overview: Create Variables .....	140
6.4.8.2 Overview: Create and/or Call Functions & Function Blocks .....	141
6.4.8.3 Overview: Using Constants .....	142
6.4.8.4 Overview: Map Input & Output to Variables .....	142
6.4.9 Adding Motion .....	143
6.4.9.1 Design Motion with Pipe Network .....	143
6.4.9.2 Design Motion with PLCOpen Axis .....	151
6.4.9.3 Design Coordinated Motion .....	160
6.4.10 Add an HMI Device .....	170
6.4.10.1 Create KVB Project .....	170
6.4.10.2 Map Variables to HMI .....	171
6.4.10.3 Design KVB Panel with Kollmorgen Visualization Builder .....	173
6.4.11 Working with SafePLC2 Projects .....	174
6.4.12 Overview: Compile, Connect, Download, and Run .....	175
6.4.13 Advanced Project Creation .....	175
6.4.14 Add and Configure a Servo Drive .....	175
6.4.14.1 Add the Drive .....	175
6.4.14.2 Configure the AKD2G/AKD Drive .....	176
6.4.14.3 Save Parameters to Non-Volatile Memory .....	178
6.4.14.4 AKD Setup Wizard... .....	178
6.4.14.5 Configure Onboard I/O .....	179
6.4.14.6 Digital Input Mode .....	179
6.4.15 Add and Configure a Stepper Drive .....	181
6.4.15.1 AKT2G-SM-Lxx Stepper Drive I/O Tab .....	181
6.4.15.2 AKT2G-SM-Lxx Stepper Drive Configuration Tab .....	182
6.4.16 Add and Configure I/O Slices .....	186
6.4.16.1 Add an AKT (K-Bus) I/O Coupler .....	186
6.4.16.2 Add an AKT (K-Bus) I/O Slice .....	187
6.4.17 Add an AKT2G Stepper Drive .....	187
6.4.17.1 Configure the I/O Slice .....	188
6.4.18 Add & Configure Third Party Devices .....	189
6.4.18.1 Add Modules to Third Party EtherCAT Devices .....	191
6.4.18.2 Modules — General Properties Tab .....	192
6.4.19 Use Pre-defined Libraries .....	193
6.4.20 Create and Use Custom Libraries .....	194
6.4.20.1 Create the Custom Library .....	194
6.4.20.2 Use the Custom Library .....	195
6.4.21 Adding Cam Profiles .....	198
6.4.21.1 Create Cam Profiles .....	198
6.4.21.2 Use Cam Profiles .....	201
6.4.21.3 Cam Profile Switching .....	202
6.4.22 Define Scheduling .....	203
6.4.22.1 Periodicity .....	203
6.4.22.2 Order of Execution .....	203
6.4.22.3 Define the PLC Cycle .....	204

---

6.4.22.4 About Parent-Child relationships and execution order .....	207
<b>6.5 Running the Project .....</b>	<b>209</b>
6.5.1 Set the Compilation Options .....	209
6.5.1.1 Conditional Compiling .....	210
6.5.2 Compile the Application .....	212
6.5.3 Launch KAS Simulator .....	213
6.5.4 Connect to the Controller .....	214
6.5.4.1 Actions to Prevent Compatibility Issues .....	215
6.5.4.2 Application Status Bar .....	215
6.5.4.3 Message Window .....	215
6.5.5 Download the Application .....	216
6.5.5.1 Application Status Bar .....	217
6.5.6 Device Control .....	217
6.5.6.1 Start/stop the Device .....	217
6.5.6.2 Log Window .....	217
<b>6.6 Testing and Debugging the Project .....</b>	<b>218</b>
6.6.1 Step-By-Step Debugging .....	218
6.6.2 About Breakpoints .....	220
6.6.2.1 About Breakpoints .....	220
6.6.3 Setting, Removing, Enabling, and Disabling Breakpoints .....	222
6.6.3.1 How to Set Breakpoints .....	222
6.6.3.2 How to Enable/Disable a Breakpoint .....	222
6.6.4 Printf Function .....	224
6.6.5 Soft Oscilloscope Debugging .....	225
6.6.5.1 How to Plug Motion Variables .....	225
6.6.5.2 How to Plug PLC Variables .....	227
6.6.6 Compare PLC Programs .....	229
6.6.7 Variable Animation .....	231
6.6.7.1 Variable Monitoring .....	231
6.6.7.2 IEC 61131-3 Editor Debugging .....	234
6.6.7.3 Pipe Network Editor Debugging .....	234
<b>6.7 Managing a Project .....</b>	<b>235</b>
6.7.1 Print .....	236
6.7.1.1 Printable Elements .....	236
6.7.1.2 Page Setup .....	236
6.7.1.3 Print .....	236
6.7.1.4 Print Preview .....	237
6.7.1.5 Print Project .....	237
6.7.2 Use the Reference Folder .....	238
<b>7 KAS IDE Reference .....</b>	<b>239</b>
<b>7.1 Choose a Workspace Layout .....</b>	<b>239</b>
7.1.1 Move Child Windows .....	239
7.1.2 Move Toolbox .....	239
7.1.2.1 Dock Window .....	239
<b>7.2 Menus and Toolbar Overview .....</b>	<b>241</b>
7.2.1 File Menu .....	241
7.2.2 Edit Menu .....	245

7.2.3 Tools Menu .....	246
7.2.3.1 User Options .....	246
7.2.4 Window Menu .....	248
7.2.5 Help Menu .....	249
7.2.6 Main Toolbar .....	250
7.2.7 Device Toolbar .....	251
7.2.8 EtherCAT Toolbar .....	252
7.2.9 Online Change Toolbar .....	252
7.2.10 Debug Toolbar .....	253
<b>7.3 Windows Standard Conventions .....</b>	<b>254</b>
7.3.1 Windows Manipulation .....	254
7.3.2 Mouse Manipulation .....	254
7.3.3 Table Manipulation .....	254
7.3.3.1 Sorting Items .....	254
7.3.3.2 Selecting a Cell .....	254
7.3.3.3 Selecting a Row .....	254
7.3.3.4 Resizing a Column .....	255
<b>7.4 Keyboard Shortcuts .....</b>	<b>256</b>
7.4.1 Common Keyboard Shortcuts .....	256
7.4.2 FBD Editor Keyboard Shortcuts .....	257
7.4.2.1 FBD Editor (common) .....	257
7.4.2.2 FBD Editor (when editing) .....	257
7.4.2.3 FBD Editor (during debug) .....	257
7.4.3 FFLD Editor Keyboard Shortcuts .....	258
7.4.3.1 FFLD Editor (when editing) .....	258
7.4.3.2 FFLD Editor (during debug) .....	260
7.4.4 SFC Editor Keyboard Shortcuts .....	261
7.4.5 ST Editor Keyboard Shortcuts .....	262
7.4.5.1 ST Editor (common) .....	262
7.4.5.2 ST Editor (when editing) .....	262
7.4.5.3 ST Editor (during debug) .....	262
7.4.6 Graphic Editor Keyboard Shortcuts .....	263
7.4.7 Table Keyboard Shortcuts .....	263
<b>7.5 Bookmarks .....</b>	<b>264</b>
<b>7.6 Create Programs .....</b>	<b>265</b>
7.6.1 Project Structure .....	265
7.6.2 IEC 61131-3 Editors .....	265
7.6.3 Some Tips... .....	266
7.6.3.1 About Drag-and-Drop .....	266
7.6.3.2 About Autocompletion .....	266
7.6.3.3 About tooltip on variable .....	266
7.6.3.4 About Bookmarks .....	267
7.6.4 Select Function Blocks .....	267
7.6.5 Select Variables and Instances .....	267
7.6.6 Drag and Drop Programming .....	267
7.6.7 Autocompletion When Programming .....	268
7.6.8 Tooltips on Variables .....	269

---

7.6.9	Selecting Function Blocks	269
7.6.10	Selecting Variables and Instances	270
7.6.11	Change Operators and Functions in FFLD and FBD Editors	270
7.6.12	Change Function Blocks in FFLD and FBD Editors	271
7.6.13	Sequential Function Chart (SFC) Editor	272
7.6.13.1	Using the SFC toolbar	273
7.6.13.2	Draw SFC divergences	274
7.6.13.3	View SFC charts	274
7.6.13.4	Move or copy SFC charts	275
7.6.13.5	Enter SFC macro-steps	275
7.6.13.6	Renumber steps and transitions	275
7.6.13.7	Enter actions of a step	276
7.6.13.8	Enter the condition of a transition	276
7.6.13.9	Enter notes for steps and transitions	277
7.6.14	Function Block Diagram (FBD) Editor	278
7.6.14.1	Using the FBD toolbar	279
7.6.14.2	Draw FBD connection lines	282
7.6.14.3	Select FBD variables and instances	283
7.6.14.4	View FBD diagrams	283
7.6.14.5	Move or copy FBD objects	283
7.6.14.6	Insert FBD objects on a line	284
7.6.14.7	Resize FBD objects	284
7.6.15	Structured Text (ST) / Instruction List (IL) Editor	286
7.6.15.1	ST / IL Language selection	286
7.6.15.2	ST / IL Syntax coloring	286
7.6.15.3	Intellisense	287
7.6.15.4	Auto-completion of words	287
7.6.15.5	ST / IL Drag-and-drop features	289
7.6.15.6	How to Read Output of a MC Function Block in ST	289
7.6.16	Free Form Ladder Diagram (FFLD) Editor	291
7.6.16.1	Using the FFLD Editor	294
7.6.16.2	Power rail and lines	296
7.6.16.3	Contacts and coils	296
7.6.16.4	Function blocks	296
7.6.16.5	Data In/Out	297
7.6.16.6	Jumps and RETURN	297
7.6.16.7	Selection grid	297
7.6.16.8	Move and copy items	298
7.6.16.9	View FFLD diagrams	299
<b>7.7</b>	<b>Create Variables</b>	<b>300</b>
7.7.1	Use the Dictionary	300
7.7.2	Create Variables from the Editors	300
7.7.3	Data Types	301
7.7.4	Complex Structures	302
7.7.5	Variable Editor	304
7.7.6	Editing Variables as Text	304
<b>7.8</b>	<b>Create Functions and Function Blocks</b>	<b>306</b>



7.8.1 Declare Functions or Function Blocks .....	306
7.8.2 Define Parameters and Private Variables .....	307
7.8.3 Finalize Functions or Function Blocks .....	308
7.8.4 Call Functions or Function Blocks .....	308
<b>7.9 Use the Defines List .....</b>	<b>309</b>
7.9.1 Internal Defines .....	309
7.9.2 Global Defines .....	310
7.9.3 Local definitions .....	311
<b>7.10 Map Input and Output to Variables .....</b>	<b>312</b>
7.10.1 Map I/O from the Project Explorer .....	312
7.10.2 PLC Variable Creation Wizard .....	316
7.10.3 PLC Variable Selector .....	317
7.10.3.1 Create PLC Variable Button .....	317
7.10.3.2 Advanced Button .....	317
7.10.4 Analog I/O Parameters .....	318
7.10.4.1 AKT2G Input Terminals .....	318
7.10.4.2 AKT2G Output Terminals .....	318
7.10.4.3 AKT Input Terminals .....	318
7.10.4.4 AKT Output Terminals .....	319
<b>7.11 Configuring EtherCAT .....</b>	<b>321</b>
7.11.1 EtherCAT Devices tab .....	321
7.11.1.1 Scan Devices .....	323
7.11.2 EtherCAT Master Settings Tab .....	324
7.11.3 ENI File Tab .....	326
7.11.4 ESI Files Tab .....	327
7.11.5 FSoE / PDO Connections Tab .....	328
7.11.6 ESI Files with References (MDP) .....	331
7.11.6.1 EtherCAT Device Configuration - MDP Devices .....	331
7.11.7 ESI File Management .....	332
<b>7.12 EtherCAT Devices .....</b>	<b>334</b>
7.12.1 EtherCAT Device General Properties Tab .....	334
7.12.1.1 Information .....	334
7.12.1.2 Topology .....	335
7.12.2 Configuration Tab .....	335
7.12.3 PDO Selection/Mapping Tab .....	335
7.12.3.1 Map PLC Variable to PDO Object .....	337
7.12.3.2 Select Input and Output PDOs .....	338
7.12.3.3 Viewing the contents of a PDO .....	338
7.12.3.4 How To Set Up a Variable to Monitor Safety Data .....	339
7.12.4 PDO Editor Tab .....	342
7.12.4.1 Editable PDOs .....	342
7.12.4.2 PDO Content .....	344
7.12.4.3 PDO Restrictions and Compiler Errors .....	347
7.12.4.4 Redundant PDO Entries .....	348
7.12.4.5 About Safety, PDOs, and Variables .....	349
7.12.5 Distributed Clock tab .....	350
7.12.5.1 Oversampling devices .....	351

7.12.6 CoE Init Commands tab .....	352
7.12.6.1 Adding CoE Commands .....	353
7.12.7 CoE Object Dictionary Tab .....	355
<b>8 Understanding KAS .....</b>	<b>357</b>
<b>8.1 IEC 61131-3 .....</b>	<b>357</b>
8.1.1 Introduction .....	357
8.1.2 Data Types .....	357
8.1.2.1 Structures .....	358
8.1.2.2 Arrays .....	358
8.1.3 Variables .....	359
8.1.3.1 About Retain Variables .....	359
8.1.3.2 Working with Variables .....	360
8.1.3.3 Retain Variables .....	361
8.1.4 Constant Expressions .....	364
8.1.4.1 Examples .....	365
8.1.5 Program Organization Units .....	367
8.1.5.1 Difference between Functions and Function Blocks .....	367
8.1.5.2 Functions .....	367
8.1.5.3 Function Blocks .....	367
8.1.5.4 Programs .....	369
8.1.5.5 User-Defined Function Blocks .....	370
8.1.6 Programming Languages .....	371
8.1.6.1 Sequential Function Chart (SFC) .....	371
8.1.6.2 Structured Text (ST) .....	372
8.1.6.3 Function Block Diagram (FBD) .....	372
8.1.6.4 Free Form Ladder Diagram (FFLD) .....	372
8.1.6.5 Instruction List (IL) .....	372
8.1.7 Alias Definitions .....	373
8.1.8 Handling Exceptions .....	374
8.1.8.1 Startup Exceptions .....	374
8.1.8.2 Shutdown Exceptions .....	374
8.1.8.3 Division By Zero Exceptions .....	374
<b>8.2 Motion Concepts .....</b>	<b>376</b>
8.2.1 Introducing Motion .....	376
8.2.1.1 Motion Control Main Functions .....	376
8.2.1.2 Single and Multi-Axis Motion .....	376
8.2.1.3 Hardware Organization of Motion Functions .....	376
8.2.1.4 Motion Profile .....	377
8.2.1.5 KAS Touch Probe & Fast Input Overview .....	378
8.2.2 PipeNetwork or PLCopen .....	380
8.2.2.1 Motion Engine Differences of PipeNetwork and PLCopen .....	380
8.2.3 PipeNetwork Concept .....	382
8.2.3.1 PipeNetwork .....	383
8.2.3.2 Pipe Structure .....	384
8.2.3.3 Pipe Block .....	385
8.2.3.4 Axis Pipe Block .....	388
8.2.3.5 Executing Motion .....	392

8.2.3.6	Pipe Block Lifetime .....	392
8.2.3.7	Motion State Machine .....	393
8.2.3.8	Phase Execution in the Pipe Network .....	393
8.2.3.9	Use Motion Function Block for Pipe Network .....	394
8.2.3.10	Function - General rules .....	395
8.2.3.11	Using the Generated PipeNetwork Function .....	396
8.2.4	PLCopen® .....	398
8.2.4.1	PLCopen Function Blocks .....	398
8.2.4.2	PLCopen Function Blocks - Overview .....	400
8.2.4.3	PLCopen Function Blocks - General Rules .....	426
8.2.4.4	PLCopen State Machine .....	430
8.2.4.5	Programming a Dual Core Controller .....	432
<b>8.3</b>	<b>EtherCAT Motion Bus Concepts .....</b>	<b>436</b>
8.3.1	Functional Principle .....	436
8.3.2	EtherCAT Features .....	436
8.3.2.1	Protocol .....	437
8.3.2.2	Topology .....	438
8.3.2.3	Distributed Clock (Synchronization) .....	439
8.3.2.4	Performance .....	439
8.3.2.5	Safety over EtherCAT .....	440
8.3.2.6	Gateways .....	441
8.3.2.7	Device profiles .....	441
8.3.2.8	File Access over EtherCAT (FoE) .....	442
8.3.3	EtherCAT Implementation .....	442
8.3.3.1	Master Configuration .....	442
8.3.3.2	Slave Configuration .....	444
8.3.3.3	EtherCAT State Machine .....	445
8.3.3.4	PDOs for AKD, AKD2G, AKD-N, and S300/S700 .....	446
8.3.4	CANopen .....	455
8.3.4.1	CANopen Status Machine .....	455
8.3.4.2	CANopen Control Word .....	458
8.3.4.3	CANopen Status Word .....	459
<b>8.4</b>	<b>AKD Drive .....</b>	<b>460</b>
8.4.1	AKD Drive Configuration .....	460
8.4.1.1	Connection Modes .....	460
8.4.1.2	AKD Configuration According to EtherCAT State .....	461
<b>8.5</b>	<b>Tasking Model / Scheduling .....</b>	<b>463</b>
8.5.1	Priority Between Motion and PLC .....	463
8.5.1.1	EtherCAT Processing Time .....	463
8.5.1.2	NVRAM Processing Time .....	464
8.5.1.3	What happens when a PLC Program is overrunning the Cycle Duration .....	464
8.5.2	Priority Between PLC Programs .....	465
<b>9</b>	<b>Using the KAS Simulator .....</b>	<b>466</b>
9.1	Start KAS Simulator .....	466
9.1.1	KAS Runtime Log Window .....	467
9.2	Stop KAS Simulator .....	468
9.3	KAS Simulator Axes Tab .....	469

---

<b>9.4 Describing KAS Simulator Graphical User Interface</b>	<b>470</b>
9.4.1 Windows Overview	470
9.4.1.1 Main window	470
9.4.1.2 KAS Simulator log window	471
<b>9.5 KAS Simulator Menus Description</b>	<b>471</b>
9.5.1 File Menu	471
9.5.2 Help Menu	471
<b>10 Using the AKD PDMM or PCMM Controller</b>	<b>472</b>
<b>10.1 Booting the PDMM or PCMM</b>	<b>473</b>
10.1.1 Boot Sequence	473
10.1.2 Boot Startup Script	474
10.1.3 Booting from the Recovery Image	474
<b>10.2 Working with the Hardware</b>	<b>475</b>
10.2.1 AKD PDMM or PCMM Memory	475
10.2.2 PDMM or PCMM B3 Button Menu	475
10.2.3 Display the IP Address of a PDMM or PCMM	476
10.2.4 Booting from the Recovery Image	476
10.2.5 Reset the Controller to Factory Settings	477
10.2.5.1 Resetting While the Drive is Running	477
10.2.6 SD Card Support	478
10.2.6.1 Supported SD Card Formats	478
10.2.7 Backup and Restore a PDMM or PCMM	479
10.2.7.1 About the data transfer	479
10.2.7.2 Backup Using the B3 Button	480
10.2.7.3 Restore Using the B3 Button	480
10.2.8 EtherCAT Devices Backup and Restore	481
10.2.8.1 EtherCAT Devices Backup	482
10.2.8.2 EtherCAT Devices Restore	484
10.2.8.3 AKD Backup/Restore Compatibility	485
10.2.8.4 Import / Export Backup from Webserver	486
10.2.8.5 EtherCAT Devices Backup/Restore Limitations	487
10.2.8.6 Troubleshooting EtherCAT Devices Backup/Restore	487
10.2.9 Configure Controller Onboard I/O	489
10.2.10 About Errors and Alarms	489
10.2.11 Using the KAS Web Server	489
10.2.12 Web Server Home Page	490
10.2.12.1 Security	491
10.2.12.2 User Authentication	493
10.2.13 KAS Application	493
10.2.13.1 Axis	495
10.2.13.2 Log Configuration	496
10.2.13.3 Log Data	497
10.2.13.4 User Data	498
10.2.13.5 Shared Directory	499
10.2.14 Web Server Settings	500
10.2.14.1 Firmware Tab (AKD PDMM and PCMM Only)	500
10.2.14.2 Network Tab (PDMM or PCMM Only)	503

10.2.14.3	File System Tab .....	504
10.2.14.4	SD Card Tab (PDMM or PCMM Only) .....	504
10.2.14.5	User Account .....	504
10.2.15	Backup & Restore from the Web Server .....	506
10.2.15.1	Backup Tab .....	506
10.2.15.2	Restore Tab .....	509
10.2.15.3	Import/Export .....	509
10.2.16	Diagnostic (PDMM and PCMM Only) .....	510
10.2.16.1	Errors and Alarms .....	510
10.2.16.2	Hardware Status .....	510
10.2.16.3	Crash Reports .....	511
10.2.16.4	EtherCAT Diagnostics .....	511
<b>10.3</b>	<b>Using SSH .....</b>	<b>512</b>
10.3.1	Logging In To A Controller Via SSH .....	512
10.3.2	Change Controller's Password Via SSH .....	513
10.3.3	How to Restore the Controller's Default Password and/or SSH Settings .....	513
<b>11</b>	<b>Tools .....</b>	<b>514</b>
<b>11.1</b>	<b>PipeNetwork Editor .....</b>	<b>514</b>
11.1.1	Inserting Pipe Blocks or Comments .....	514
11.1.2	Inserting Connections .....	514
11.1.3	Connect a Comment to a Pipe Block .....	515
11.1.4	Pipe Network Editor Controls .....	515
11.1.5	Plug/Unplug Channels .....	517
<b>11.2</b>	<b>Cam Profile Editor .....</b>	<b>518</b>
11.2.1	About the Cam Profile Editor .....	518
11.2.1.1	Windows Overview .....	518
11.2.2	Cam Profile Editor's Cam Table .....	519
11.2.2.1	Modifying an Element using the Cam Table .....	520
11.2.2.2	Modifying the Type of a Cam Element .....	521
11.2.2.3	Cam Table Contextual Menu .....	521
11.2.2.4	Adding a Point to the Cam Table .....	522
11.2.2.5	Remove a Point from the Cam Table .....	522
11.2.3	Cam Profile Graph .....	523
11.2.3.1	Modifying an Element .....	523
11.2.3.2	Cam Profile Graph Contextual Menu .....	524
11.2.3.3	Zoom In and Out .....	524
11.2.3.4	Panning .....	524
11.2.3.5	Restoring Zoom and Pan .....	524
11.2.4	Cam Profile Segment Overview .....	525
11.2.5	Curve Selection and Color Table .....	529
11.2.5.1	How to change color .....	529
11.2.6	Curves Graph .....	530
11.2.7	Reload, Save, Auto Fit, and Properties Buttons .....	530
11.2.8	Import Cam Profile .....	530
11.2.8.1	About the Import .....	532
11.2.8.2	When Displaying the Imported Cam Profile .....	532
11.2.8.3	About Invalid Data .....	532

<b>11.3 Soft Oscilloscope</b>	<b>532</b>
11.3.1 Access the Soft Oscilloscope	533
11.3.2 About OpenGL	533
11.3.3 The Control Panel	534
11.3.4 The Graphical Area	539
11.3.5 Traces	539
11.3.6 Plugging Probes, Mapping Variables	540
11.3.6.1 Mapping a Variable to the Oscilloscope	540
11.3.6.2 Plugging a probe from the Dictionary	541
11.3.6.3 Plugging a probe from the Pipe Network	543
11.3.7 Setting Scale	544
11.3.8 Trace Zoom Feature	544
11.3.9 How Do I Export the Oscilloscope Data?	545
11.3.10 Practical Application: Using Trace Time To Measure CPU Load	547
11.3.10.1 Collect some data by pressing the "Start" button	548
11.3.10.2 Check the peak times	549
11.3.10.3 Heavily Loaded CPU Example	550
11.3.10.4 Over Loaded CPU Example	551
<b>11.4 Human-Machine Interface Editor</b>	<b>554</b>
11.4.1 Using Kollmorgen Visualization Builder	554
11.4.1.1 Create A New KVB Controller	555
11.4.1.2 Import Variables Into The Project	556
11.4.1.3 Design Your HMI Panel with KVB	558
11.4.1.4 Download the KVB Panel	559
11.4.2 Design the Control Panel with the Internal Control Panel Editor	559
11.4.2.1 Create Control Panel	559
11.4.2.2 Use the Control Panel control library	560
11.4.2.3 Edit the Control panel	560
11.4.2.4 Mapping Variables to the Control Panel	561
11.4.2.5 Graphic Objects	562
11.4.2.6 Graphic Objects Properties	569
11.4.2.7 Operate the Control Panel	574
11.4.2.8 Exiting Simulation Mode	575
<b>12 Advanced Topics</b>	<b>576</b>
<b>12.1 Coordinated Motion</b>	<b>576</b>
12.1.1 Overview	576
12.1.1.1 Coordinated Motion Terminology	576
12.1.1.2 Group State Diagrams	578
12.1.1.3 Coordinate Systems	578
12.1.2 Design Coordinated Motion	579
12.1.2.1 Create a Linear or Circular Coordinated Motion Application	579
12.1.2.2 Blending Between Moves	589
12.1.2.3 Blending with Transitions	591
12.1.2.4 Transition Between Moves	592
12.1.2.5 What Does MC_GrpHalt Do?	599
12.1.2.6 What Does MC_GrpStop Do?	601
12.1.2.7 Differences between MC_GrpHalt and MC_GrpStop	604

12.1.2.8 Handling Axis Errors .....	604
<b>12.2 Motion Techniques .....</b>	<b>607</b>
12.2.1 PLC Online Change .....	607
12.2.1.1 What is Online Change .....	607
12.2.1.2 How to Activate Online Change .....	610
12.2.1.3 What is the Revert button .....	611
12.2.1.4 Difference between Local and Controller versions .....	612
12.2.1.5 Pulse Limitations with Online Change .....	612
12.2.2 Using PLC Online Change .....	613
12.2.2.1 Set up an application .....	613
12.2.2.2 Enable Online Change .....	613
12.2.2.3 Using Online Change .....	613
12.2.2.4 Revert Online Change .....	614
12.2.3 What Are Fast Inputs? .....	614
12.2.3.1 Fast Inputs with Pipe Network Motion .....	614
12.2.3.2 Fast Inputs with PLCopen Motion .....	615
12.2.4 Torque Feed-forward .....	615
12.2.5 PLCopen Homing .....	616
12.2.5.1 PLCopen Homing Description .....	616
12.2.5.2 PLCopen Homing Methods .....	616
12.2.5.3 AKD Capture Engine Configuration .....	619
12.2.6 Pipe Network Homing .....	621
12.2.6.1 Fast Homing Example with the Pipe Network Motion Engine Axis Pipe Block .....	621
12.2.7 PLCopen Registration .....	623
12.2.7.1 Deciding Which Function Blocks to Use for Registration .....	623
12.2.7.2 Single-Axis Registration .....	624
12.2.7.3 Master/Slave Registration .....	624
12.2.7.4 Registration Application Guide .....	625
12.2.7.5 Fast Homing Example with the PLCopen Motion Engine .....	627
12.2.8 Pipe Network Registration and Fast Homing .....	628
12.2.8.1 Fast Homing Example with the Pipe Network Motion Engine Axis Pipe Block .....	628
12.2.8.2 Registration Position Capture Example with Pipe Network Trigger Block .....	629
12.2.9 How To Interpret a Timestamp .....	631
12.2.10 Tuning Controller Performance for EtherCAT Communication Latency .....	631
12.2.11 Error Management .....	632
12.2.12 Restarting Motion with PipeNetwork .....	632
12.2.13 Superimposed Axes with PLCopen .....	633
12.2.14 Overtravel Conditions .....	634
12.2.14.1 PLCopen Behavior .....	634
12.2.14.2 Coordinated Motion Behavior .....	634
12.2.14.3 PipeNetwork Behavior .....	634
12.2.14.4 Configuring Generation 1 AKD Drives .....	634
12.2.15 How To Create a Digitizing Axis .....	636
12.2.15.1 How To Create a Digitizing Axis When EtherCAT Devices Are Defined .....	636
12.2.15.2 How To Create a Digitizing Axis When EtherCAT Devices Are Not Connected .....	637
12.2.15.3 How To Create a Digitizing Axis When Using an Imported ENI File .....	638
12.2.15.4 How To Create a Digitizing Axis From Code .....	639

12.2.16 Working With A Digitizing Axis in PLCopen .....	640
12.2.16.1 Digitizing Axis – Encoders Without a Z-Channel .....	641
12.2.17 Practical Applications - Camming .....	642
12.2.17.1 Positioning an Axis Before Starting Camming .....	642
12.2.17.2 Resuming Camming After an E-Stop .....	642
12.2.17.3 Cam On The Fly .....	642
<b>12.3 Motion Bus and Fieldbuses .....</b>	<b>644</b>
12.3.1 EtherCAT .....	644
12.3.2 Ethernet/IP .....	644
12.3.3 Modbus & TCP/IP .....	644
12.3.4 Profinet .....	644
12.3.5 Add Unsupported EtherCAT Device .....	645
12.3.5.1 How to modify the EtherCAT image in cyclic mode .....	645
12.3.5.2 How to configure EtherCAT device .....	645
12.3.5.3 How to map PLC variables .....	645
12.3.6 Set Up FSoE Master and an AKD2G with SafeMotion Monitor .....	647
12.3.6.1 Create Safety Application .....	647
12.3.6.2 KAS IDE Project Set Up .....	647
12.3.7 AKD2G Safety Parametrization Using FSoE with SCU-1-EC and PCMM/AKD PDMM .....	648
12.3.7.1 Requirements .....	649
12.3.7.2 How-To .....	649
12.3.8 Fieldbus Editor .....	650
12.3.8.1 PCMM/AKD PDMM Ethernet/IP Data Communications .....	650
12.3.8.2 Ethernet/IP IO Scanner (Client) .....	650
12.3.8.3 Ethernet/IP Adapter (Server) .....	652
12.3.8.4 Ethernet/IP Tag Client .....	654
12.3.8.5 PDMM/PCMM EDS file for EtherNet/IP .....	656
12.3.8.6 Ethernet/IP FlexIO / PointIO .....	657
12.3.8.7 Modbus Slave .....	658
12.3.8.8 Profinet IO RT Controller Configuration .....	661
12.3.8.9 Profinet IO RT Device Configuration .....	673
12.3.9 Timeout error .....	678
12.3.10 Timeout error .....	679
12.3.11 Modular EtherCAT Concept .....	684
12.3.11.1 Introduction .....	684
12.3.11.2 Modular Design Guidelines .....	684
12.3.11.3 Backing up Modular EtherCAT devices .....	685
12.3.11.4 Limitations .....	685
12.3.11.5 How to Create an Application with Modular EtherCAT .....	686
12.3.12 Performance Guidelines .....	689
12.3.12.1 EtherNet/IP .....	689
<b>12.4 Project Structure Guidelines .....</b>	<b>691</b>
12.4.1 Introduction .....	691
12.4.2 External Files .....	691
12.4.3 Application Software Structure - Definitions .....	692
12.4.3.1 Modules to build up the Structure .....	692
12.4.3.2 State and Function Definitions .....	694



12.4.4 Application Software Structure - Implementation .....	695
12.4.4.1 SFC children building up the software .....	695
12.4.4.2 Variables for the Interface .....	695
12.4.4.3 Main module implementation description .....	696
12.4.4.4 States and Errors .....	699
12.4.4.5 Functions linked to states .....	700
<b>12.5 Project Templates .....</b>	<b>702</b>
12.5.1 Pipe Network 1-Axis Template with FFLD only .....	703
12.5.2 Pipe Network 1-Axis Template with ST only .....	704
12.5.3 Pipe Network 2-Axes Template with SFC, ST, FFLD, and FBD .....	706
12.5.3.1 PLC Programs .....	706
12.5.4 Motion .....	708
12.5.5 Control Panel .....	709
12.5.5.1 HMI .....	710
12.5.6 Pipe Network 2-Axes Template with ST only .....	711
12.5.6.1 PLC Programs .....	711
12.5.7 Motion .....	711
12.5.8 Control Panel .....	712
12.5.8.1 HMI .....	712
12.5.9 Pipe Network 2-Axes Template with FFLD only .....	714
12.5.9.1 PLC Programs .....	714
12.5.10 Motion .....	714
12.5.11 Control Panel .....	715
12.5.11.1 HMI .....	715
12.5.12 PLCopen 1-Axis Template with FFLD .....	716
12.5.13 PLCopen 1-Axis Template with ST .....	717
12.5.14 PLCopen 2-Axes Template with SFC and FFLD .....	718
12.5.14.1 PLC Programs .....	718
12.5.14.2 Motion .....	719
12.5.14.3 Control Panel .....	720
12.5.14.4 HMI .....	721
12.5.15 PLCopen 2-Axes Template with ST .....	723
12.5.15.1 PLC Programs .....	723
12.5.15.2 Motion .....	723
12.5.15.3 Control Panel .....	724
12.5.15.4 HMI .....	725
12.5.16 PLCopen 2-Axes Template with FFLD .....	727
12.5.16.1 PLC Programs .....	727
12.5.16.2 Motion .....	727
12.5.16.3 Control Panel .....	728
12.5.16.4 HMI .....	729
12.5.17 Coordinated Motion 2-Axis Template .....	731
12.5.17.1 Programs .....	731
12.5.17.2 Motion .....	731
12.5.17.3 Control Panel .....	731
12.5.18 Coordinated Motion 3-Axis Template .....	732
12.5.18.1 PLC Programs .....	732

12.5.18.2	Motion .....	732
12.5.18.3	Control Panel .....	733
12.5.19	Coordinated Motion 3-Axis (3D) Template .....	734
12.5.19.1	PLC Programs .....	734
12.5.19.2	Motion .....	734
12.5.19.3	Control Panel .....	735
<b>13</b>	<b>PDMM and PCMM Errors and Alarms .....</b>	<b>736</b>
13.1	PCMM and PDMM Errors .....	736
13.2	PCMM and PDMM Alarms .....	739
13.3	CPU Overload (E23) .....	740
<b>14</b>	<b>KAS Component Manuals .....</b>	<b>742</b>
14.1	HMI .....	742
14.1.1	HMI Accessories .....	744
14.2	Controllers .....	744
14.2.1	PCMM Specifications .....	744
14.2.2	AKD PDMM Specifications .....	745
14.2.3	NVRAM .....	745
14.2.3.1	How can I check the NVRAM space is enough to store my retain variables? ..	746
14.3	BBH FSoE Master .....	747
14.4	Kollmorgen Safety Solutions .....	748
14.5	Remote Input/Output Terminals .....	749
14.6	Drives .....	751
14.6.1	About the AKD "NC" Drive Model .....	752
14.6.1.1	FAQ's regarding the "NC" drive .....	752
<b>15</b>	<b>Troubleshooting .....</b>	<b>753</b>
15.1	FAQs .....	753
15.1.1	Basic .....	753
15.1.2	Setup .....	755
15.1.3	Computer / Controller .....	757
15.1.4	Performance .....	758
15.1.5	EtherCAT .....	759
15.1.6	Programming .....	760
15.2	Compiler Errors .....	762
15.3	CPU Load Reduction Techniques .....	764
15.4	EtherCAT Diagnostics & Errors .....	765
15.4.1	EtherCAT Diagnostics .....	765
15.4.2	EtherCAT Error Messages .....	765
15.4.2.1	Other Messages Linked to EtherCAT .....	767
15.4.3	EtherCAT Communication Diagnosis Steps .....	767
15.4.3.1	Diagnosing EtherCAT Communication Errors .....	767
15.4.3.2	Code Examples for Diagnosing EtherCAT Communication Errors .....	768
15.5	EtherCAT Coupler Error Handling And Diagnosis .....	770
15.5.1	EtherCAT Diagnostic LEDs .....	770
15.5.2	EtherCAT LED Power Supply Diagnosis .....	770
15.5.3	EtherCAT LED Off Power Supply Diagnosis .....	770
15.5.4	LEDs for EtherCAT State Machine/PLC Diagnosis .....	770
15.5.5	LEDs for EtherCAT Connection Diagnosis .....	771

---

15.5.6 LEDs for EtherCAT Data Diagnosis .....	771
<b>15.6 Troubleshooting FSoE Safety Parameters .....</b>	<b>772</b>
<b>15.7 Troubleshooting KAS IDE and SafePLC2 .....</b>	<b>774</b>
<b>15.8 Connect Remotely .....</b>	<b>777</b>
<b>16 Annexes: Lists of Manuals, Content, and Support Information .....</b>	<b>778</b>
<b>16.1 List of How-Tos .....</b>	<b>778</b>
16.1.1 PLC Code How-Tos .....	778
16.1.2 EtherCAT Fieldbus How-Tos .....	778
16.1.3 Advanced Motion How-Tos .....	778
16.1.4 Run the Application How-Tos .....	778
16.1.5 Hardware How-Tos .....	778
<b>16.2 List of KAS Manuals .....</b>	<b>778</b>
16.2.1 KAS IDE .....	779
16.2.2 KVB .....	779
16.2.3 HMI .....	780
16.2.4 I/O .....	780
16.2.5 AKD2G, AKD, & S300/S700 .....	782
16.2.6 PDMM / PCMM .....	782
<b>17 Acronyms &amp; Initialisms .....</b>	<b>784</b>
<b>18 Glossary .....</b>	<b>792</b>
<b>19 Licenses .....</b>	<b>799</b>
<b>20 Index .....</b>	<b>820</b>

## 2 Document History

Edition	Software Version	Date	Description
0.2		September 2010	First release for KAS R1
A	2.5	May 2012	Initial release of KAS IDE and AKD PDMM
B	2.6	December 2012	EtherCAT devices backup & restore; PLCopen Registration; Superimposed Axes; Cam on the fly; Ethernet/IP; Profinet; Reset to factory settings; Save project on controller; Password protection
C	2.7	June 2013	Initial doc
D	2.7	October 2013	Coordinated Motion; EtherCAT device configuration; KVB 2, softscope; generic Modbus configuration; Dictionary improvements; Control panel changes
E	2.8	May 2014	Coordinated Motion (3D linear w/ blending & transition; nD w/o blending & transition), HTTP variable interface, UDP socket interface, EtherCAT MDP management
F	2.9	May 2015	AKD-C/N support, flexible PDOs, multiple E/IP assembly files, export-compliant firmware, new functions and function blocks, GUI improvements including easier EtherCAT device setup, improved IO and variable mapping, improved IO viewing.
G	2.10	January 2016	PCMM support, Code generation types: Normal and Optimized, MC_TouchProbe: Time or Position-based capture, , <a href="#">MCFB_StepLimitSwitchFastInput</a> , HTML5 help.
H	2.11	September 2016	Parabolic cam profile support, send data to/from a controller via UDP, access KDN from the menu, improved EtherCAT diagnostics, Find All command, find variable instances, access shared directories from the webserver, configurable Modbus slave data exchange rate, double-click editing of FFLD & FBD, F_Seek function, configurable circle center point tolerance.
J	2.12	June 2017	AKD2G support, AKD 1.16, KVB 2.20, AKIX2 panels, EtherCAT Diagnostics (webserver diagnostics tab, A31, ECATWCStatus, ECATMasterStatus, ECATDeviceStatus, ECATCommErrors), MC_MoveContVel, FFLD Editor Comments, PN Editor Improvements, SSH Security
K	3.0	January 2018	AKD 1.17, AKD2G integration, MC_CreatePLCAxis, MLPNAxisCreate
L	3.01	December 2018	Multiple installed versions, updated UDFBs, Backup Import/Export on webserver, UI improvements, Softscope enhancements, security enhancements
M	3.02	June 2019	Map a digitizing axis, AKD2G, multiple instances of KAS IDE, ElasticSearch, reorder programs
N	3.03	December 2019	FSoE; T-Bot / H-Bot / Gantry robotics: Axes Groups, MC_KIN_REF structure, MC_SetKinTra; AKD2G Backup & Restore, MDP support

Edition	Software Version	Date	Description
P	3.04	June 2020	TCP/IP functions, eitadapter, file management functions, SafePLC support, WorkBench improvements, support AKT2G slices, DriveParamStrRead, multi-select PDO objects for mapping
R	3.05	December 2020	Integration of BBH SCU-1-EC with KAS IDE and BBH SafePLC2 software; Integration of E-bus AKT2G I/O devices; WorkBench connection improvements; <a href="#">MLAxisReadVel</a>
T	3.06	June 2021	Integration of stepper I/O slices and support for stepper motors. Added <a href="#">ECATDevReadParam</a> , <a href="#">MCFB_DriveFault</a> , <a href="#">MLFB_DriveFault</a> , <a href="#">MCFB_ECATRstart</a> , and <a href="#">MLFB_ECATRstart</a> . <a href="#">Access to Bits of an Integer</a> increased to 64 bits
U	3.07	December 2021	Modular EtherCAT topology, AKT2G-BRC-000 Brake Chopper, saving previous versions of KAS files, link to templates on KDN





--- / ---

## 3 Preface

Welcome to the Kollmorgen Automation Suite help and documentation! This book covers the use and programming of the KAS IDE, the AKD PDMM and PCMM, information about KAS components, and much more. This chapter explains how to use the online help provided with Kollmorgen Automation Suite™.

### 3.1 System Requirements for KAS IDE and KAS Simulator

KAS IDE and KAS Simulator are compatible with any PC having the minimum following hardware:

Element	Description
<b>Operating System</b>	Microsoft® Windows® 7 SP1 (32 or 64-bit), Microsoft® Windows® 10 (32 or 64-bit). For optimal performance, please be sure your operating system is fully updated with the latest patches.
<b>Processor type</b>	Intel® Pentium® M or equivalent processor at 1.5 GHz or greater.
<b>Memory</b>	1 GB RAM (for 32-bit) or 2 GB RAM (for 64-bit) or greater (which is recommended for complex applications).
<b>Storage</b>	16 GB (for 32-bit) or 20 GB (for 64-bit) of free space on hard disk.
<b>Display</b>	WXGA+ (1440 x 900) or higher-resolution monitor with 24-bit color. See Note #1 below.
<b>Connectivity</b>	1 Ethernet port, at either 100Mbps/s or 1Gbits/s. See Note #2 below.
<b>Web Browser</b>	A modern web browser is required to access the web server and online help. We recommend Microsoft Edge  , Google Chrome  , Mozilla Firefox  , or Internet Explorer (IE9 or later, see Note #3)  .

#### NOTE

1. Better results are achieved with OpenGL and 3D cards.
2. A 100Mb network is required in order to allow the IDE to Runtime communication to work in all conditions. The AKD WorkBench AutoTuner and Scope both require 100Mb of bandwidth to function properly.
3. IE9 should be considered a minimum. Later versions of the browser are more compliant with web standards and afford better performance and compatibility.

#### TIP

See the topic [Connect Remotely](#) for information about the ports used by the KAS IDE which may need to be opened to support connecting from an external network.

Table 1-1: Minimum System Requirements for the KAS IDE

## 3.2 Learning Kollmorgen Automation Suite

To learn Kollmorgen Automation Suite, you can either:

- Navigate this online help and choose chapters depending on your experience, or
- [Read the KAS Manuals](#)

### 3.2.1 Where to Start

The KAS documentation includes information for readers from a variety of backgrounds. To get the most out of the documentation, we recommend that you start by reading the chapters that are most relevant to you. Within each chapter, read through the topics in sequence.

#### 3.2.1.1 Beginner

- Find basic information about KAS in [Introducing Kollmorgen Automation Suite](#)
- If you are not familiar with the concepts behind KAS, read the [Understanding KAS](#)
- An overview of the KAS IDE User Interface is in [Describing KAS Graphical User Interface](#)
- To get information on how to run and debug the project, read [Launch KAS Simulator](#) and [Testing and Debugging the Project](#)

#### 3.2.1.2 Advanced User



- In order to design and create a project, refer to [Getting Started](#)
- Go to [Tools](#) if you need explanations about the tools used by the KAS IDE
- For in-depth information, refer to [Advanced Topics](#)

#### 3.2.1.3 Online

- KDN: The Kollmorgen Developer Network ("KDN") has many articles, resources, and a user community where you may learn, ask questions, and contribute to our community. Visit KDN at <https://www.kollmorgen.com/en-us/developer-network/>
- YouTube: Kollmorgen has a channel dedicated to KAS. Check out the [KAS Online Learning Collection](#).

### 3.2.2 Read the KAS Manuals

If you prefer to read printed material, the following manuals (in PDF format) are available under the C:\Program Files\Kollmorgen\Kollmorgen Automation Suite\Help folder

Document Title	PDF	Description
Release Notes		The KAS version 3.07 Release Notes contain fixed limitations, known limitations, workarounds, and information on all hardware and software components that have been updated, changed or added in this release.
Getting Started		Covers the main steps to get your KAS system up and running. <i>What does it contain?</i> <ul style="list-style-type: none"> <li>• HW Installation (Connection and Wiring): Wiring &amp; hardware details, connectors, system diagrams</li> <li>• HW Configuration: Basic configuration and settings needed to start the HW components (HMI + Industrial PC + Fieldbus + I/O)</li> <li>• SW Installation: KAS software setup</li> </ul>







Document Title	PDF	Description
<b>30 Minutes to Motion</b>		<p>Covers the main topics to help you start quickly with KAS IDE. The objective is to familiarize you with the basic principles and the way the program works by creating a simple motion application project.</p> <p><i>What does it contain?</i></p> <ul style="list-style-type: none"> <li>• Key Features</li> <li>• Explore the Workspace</li> <li>• Build a motion project</li> </ul> <p>Almost every task that you perform in KAS falls under one of the following basic steps (which may not always be completed in the following order):</p> <ol style="list-style-type: none"> <li>1. Start Projects - Create a project from scratch, or modify an existing project.</li> <li>2. Add Components - Add elements to build your project necessary to control the motion part of your system.</li> <li>3. Build Output - Select a device and generate the application that you will deliver to users.</li> <li>4. Run Output - Make the output accessible to your end-users.</li> </ol>
<b>IDE User Manual</b>		Contains the content to help you with KAS IDE, except the topics included in the Reference Manuals.
<b>Reference Manual - PLC Library</b>		Contains Technical References on <b>PLC</b> Programming Languages and Library.
<b>Reference Manual - Motion Library</b>		Contains Technical References on <b>Motion</b> Library for Pipe Network and PLCopen.

Table 1-2: List of KAS Guides in PDF Format

**TIP**

The KAS IDE allows you to include [references](#) to external files such as the PDF files listed above. For more details, refer to [Use the Reference Folder](#).

Additionally, you can add in the PDF your own comments, tips and tricks, provided that you have Adobe Acrobat®.

## 4 Introducing Kollmorgen Automation Suite

This chapter introduces Kollmorgen Automation Suite (KAS) with a product **overview** that lists the features, the components, and the different implementations.

KAS is intended for engineers who want to design and build high-performance motion control and automation systems. KAS is designed to allow you to quickly and easily compose a motion application. It can be achieved with all of the re-use and flexibility of the KAS libraries in conjunction with the IEC 61131-3 programming languages.

As can be seen, KAS can cover a wide variety of applications:

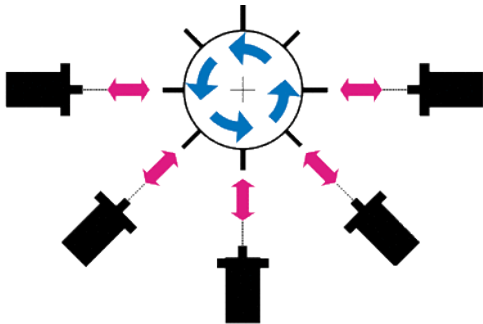


Figure 2-1: Synchronized Feeder

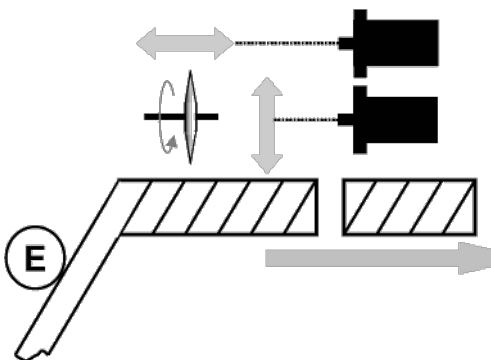


Figure 2-2: Spring Winding

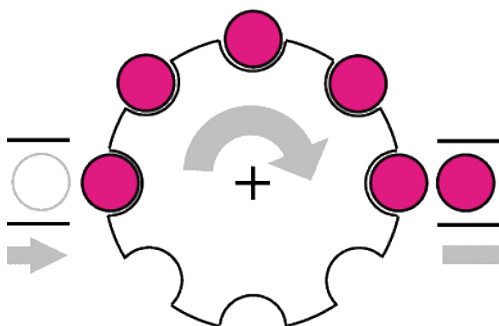


Figure 2-3: Synchronizer

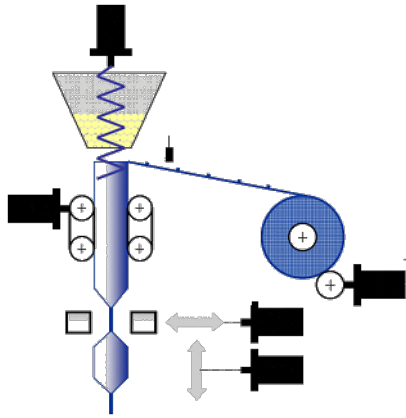


Figure 2-4: Form Fill Seal

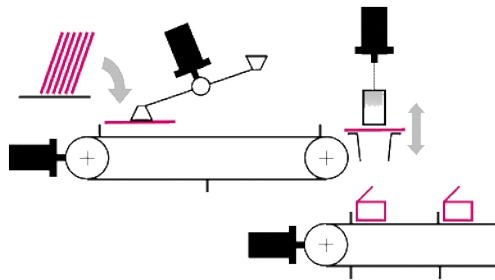
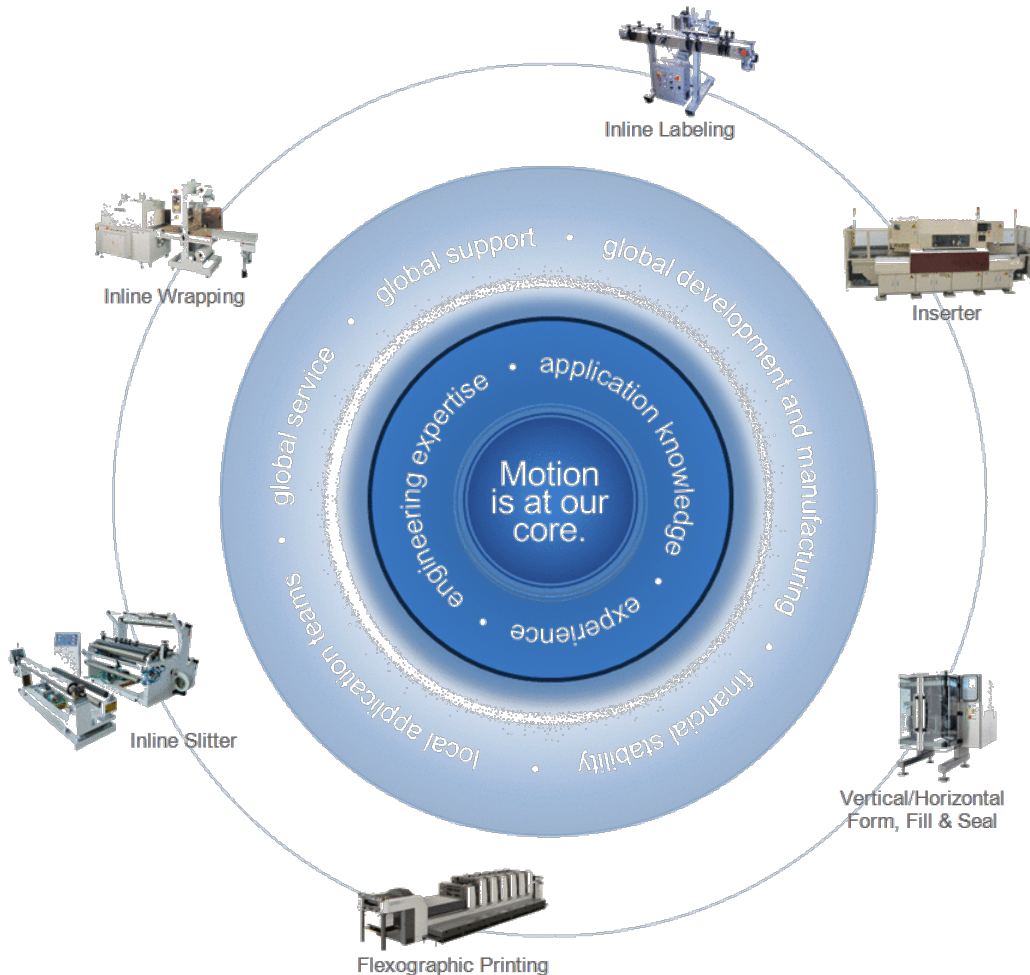


Figure 2-5: Carton Erector

## 4.1 Key Features

The purpose of KAS is to include in a single software package, all the tools you need (i.e. a soft PLC, configuration tools, and a motion controller) to create an automation system.

An overview of an investment in Kollmorgen for Motion Control and Automation Systems solution can be encapsulated as follows:



Kollmorgen Automation Suite (also known as KAS) is Kollmorgen's all-in-one solution for designing, developing and maintaining automation systems. As a solution offering, it brings many years of Motion Control experience to the market, and this is coupled with technical expertise and experience, global delivery capability, and strong financial performance.

KAS is a set of software packages designed to run and take advantage of Kollmorgen's extended set of integrated hardware products such as Programmable Automation Controllers, Programmable Drives, AKD drive family, award winning components like the AKM motor family, gear boxes, I/O terminals and Human Machine Interaction terminals (or Operator Interfaces.)

KAS provides machine builders with a **high-performance, cost-effective** and **easy to use** solution for building machines. KAS achieves this goal by integrating in a **coherent, intuitive, flexible** way the three main functionalities of a machine:

- Precise control of all moving parts (Motion control)
- Interface with machine operators (HMI)
- PLC programming of the machine (IEC 61131-3 Soft PLC)

KAS is made of two different software components:

- KAS IDE - the Integrated Development Environment allowing the development and monitoring of complex machine automation systems
- KAS Runtime - the Runtime engine offers the functionality of both a **High-Performance Motion and a PLC Engine**

### 4.1.1 Integrated Development Environment



KAS comes with a powerful Integrated Development Environment (IDE) (commonly named **KAS IDE**) which provides machine builders with all the necessary tools for designing, programming, configuring, debugging and maintaining machine applications. KAS uses the same interface, tools, and libraries to create applications for various types of KAS controllers (AKD PDMM, PCMM).

With the KAS IDE, system engineers can:

- Create new application projects using predefined or custom application **templates**
- Define the machine hardware architecture (**motion bus**, **fieldbus**, controllers (**Controllers**, PCMM), **drives** and motors) as well as the machine program (HMI panels, IEC 61131-3 **programs and function blocks**, motion block, **profiles** and axes) from a centralized **Project Explorer** which is based on a tree-structure
- Develop **PLC programs, functions and function blocks** using the five IEC 61131-3 programming languages (ST, IL, FLD, FBD and SFC), the IEC 61131-3 standard library and KASFunction Block libraries dedicated to motion, communication and monitoring
- Centrally manage all IEC 61131-3 variables with **KASvariable dictionary**
- Create and organize your own libraries of functions and function blocks
- Easily set up **HMI panels** by means of **graphical objects** that are part of the HMI control library; and map graphical objects to IEC 61131-3 variables
- Graphically design advanced multi-axis relations using **Kollmorgen's graphical motion programming** environment - also called the **Pipe Network** - with its **tool** generating code automatically
- Use ultra-fast IEC 61131-3 compiler to validate the syntactical correctness application code
- Configure hardware devices via an integrated set of configuration tools (for instance **AKD drives**, **EtherCAT I/O terminals**, etc.)
- Access controller devices to **download, start and stop the application, watch log messages and send shell commands** to the target device
- Debug the application by inserting **break points** and **stepping** into the code or by monitoring internal values (IEC 61131-3 variables, motion positions, drive's internal values) directly in the editors or with KAS advanced **softscope** tool
- Access the full online documentation

## 4.1.2 KAS Runtime



Kollmorgen Automation Suite Runtime (commonly named **the KAS Runtime**) offers, in a single software package, the functionality of both a soft PLC and a motion controller.

The KAS Runtime (virtual machine) is a high-performance deterministic environment designed to run on different hardware platforms ranging from low-cost **programmable drives** to **high-end Programmable Automation Controllers**. This gives machine builders all the flexibility when designing their machines.

KAS supports many configurations when integrating machines:

- Ranging from **single-axis** to more than **200 tightly coordinated axes**
- With a **centralized** (Programmable Automation Controllers), **distributed** (Programmable Drives) or **mixed** (Programmable Automation Controllers + programmable drives) control architecture
- Running on a **single or multiple controllers**
- Communicating via **Ethernet, EtherCAT, Profinet, EtherNet/IP, or Modbus**
- Using the high-performance **Pipe Network** or the standard **PLCopen** function blocks
- Controlling Kollmorgen's drives (**AKD**, some of the **Servostar Sxxx** drive family), **AKM** motors, and **AKT** terminals for I/Os products

The KAS Runtime can be used in the two different contexts:

- With a controller implementation (PCMM)
- With a master drive implementation (AKD PDMM)

See [Different Implementations](#) for more details.

### 4.1.3 KAS Simulator



Thanks to the **KAS Simulator** tool, development of the machine software can start before (or parallel with) bringing in production. Thus developers can already begin to develop and test their application without any hardware available. Once ready, the application can be deployed on the targeted controller.

By allowing developers to run the full application (HMI, soft PLC, motion and I/Os) totally independently from the final targeted machine, KAS helps reducing:

- **Development time**, by limiting down time waiting for other engineers (mechanical, electrical...) to test, fix and tune the real machine
- **Cost**, by limiting broken hardware due to software bugs.

See [Using the KAS Simulator](#) for more information.



## 4.2 Looking at Kollmorgen Automation Suite

### 4.2.1 Physical View

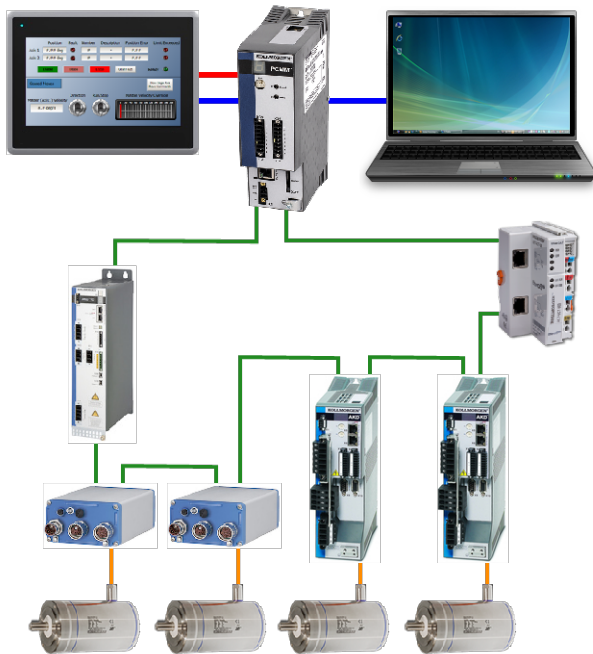


Figure 2-6: Example of Automation System

#### Key of Communication Types

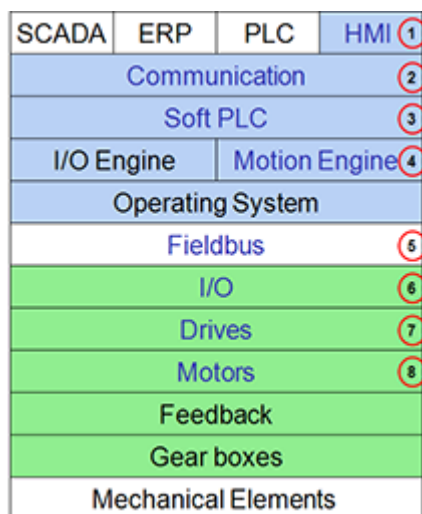
Color	Method of communication
Blue	Ethernet
Green	EtherCAT
Red	Modbus
Orange	Motor cable

### 4.2.2 Logical View

An automation system usually needs an organized hierarchy of controller systems to function and usually including the following items:

Item	Call out#	Description
HMI	1	At the end-user top level, the Human Machine Interface is where the operator can monitor or operate the system. It is usually composed of a AKI panel.
Communication	2	HMI is linked to the middle layer via a non time critical communication system (e.g. Modbus TCP protocol on Ethernet).
PLC	3	Programmable Logic Controllers is a digital computer used for automation of industrial processes, such as control of machinery on factory assembly lines. It is used to synchronize the flow of inputs from (physical) sensors and events with the flow of outputs to actuators and events.
Motion Engine	4	There are two Motion Engines available: <b>Pipe Network</b> and <b>PLCopen</b> . The Motion Engine implements different motion algorithms and <b>functions</b> to create, access and delete pipes, pipe blocks and axes. It also provides a set of Functions and Function Blocks that IEC 61131-3 applications can use to control the behavior of these algorithms.
Fieldbus	5	The fieldbus is the way to connect instruments in a plant design by linking the PLC to the external systems.
I/O	6	Input/Output refers to the communication between your automation system, and the outside world.
Drive	7	A Drive is an electronic device that provides power to a motor or servo.
Motor	8	At the bottom of the control chain is the motor which actually does the work.

Color Legend	Description
Blue Cell	Belongs to KAS
Green Cell	Kollmorgen products
White Cell	Third parties



### 4.2.3 Architectural View

The diagram shows an architectural view of what KAS is from a hardware and software perspective, or realtime and not-realtime components. Hardware is shown in blue shaded boxes and, minimally consists of a computer, a controller, one or more drives, I/O, and optionally an HMI device. Each of these components has a software aspect, many of which communicate with other components. Communication may be done over Ethernet or HTTP (seen in blue), EtherCAT (seen in green), Modbus (seen in red), or several other protocols.

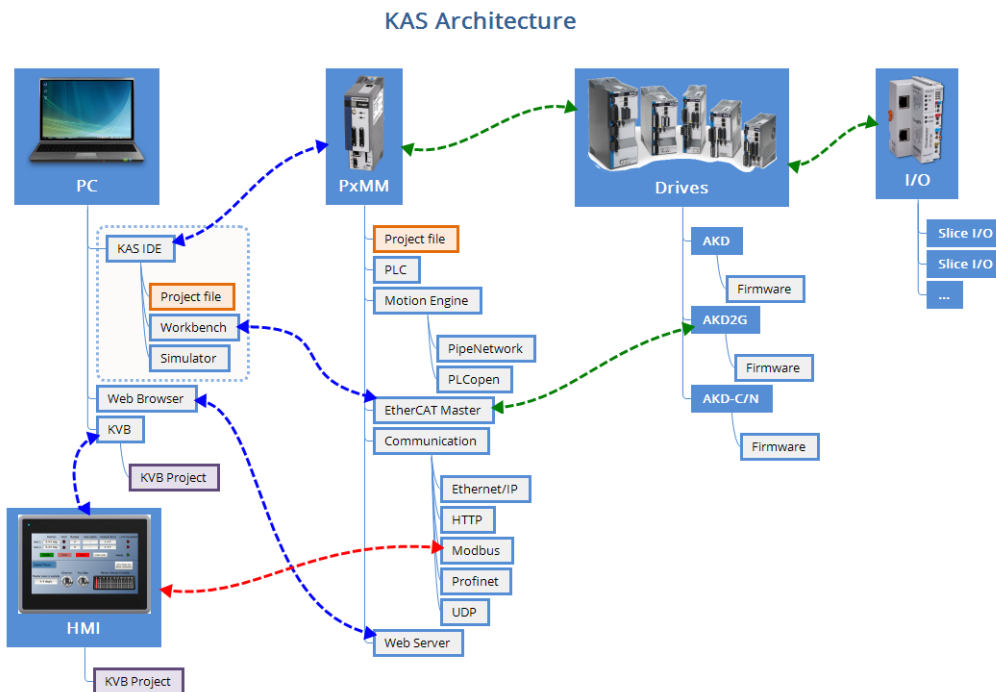


Figure 2-7: Architectural view of KAS components.

The non real-time part is composed of:

Item	Description
KAS IDE	The development tool which allows creating the project (i.e. design, create and run virtually)
KAS Runtime	...
KVB	The development tool which allows creating an HMI interface.
HMI	Available when integrated on a Programmable Automation Controller platform (not present when integrated a programmable drive)

The realtime components, including that which run the KAS Runtime engine, consists of:

Item	Description
IEC 61131-3 virtual machine	Responsible for managing an IEC 61131-3 application with its programs and variables

Item	Description
Motion manager	<p>Manages motion engines, axis objects and motion bus drivers.</p> <p>The KAS Runtime comes with two motion engines: <a href="#">Pipe Network</a> and <a href="#">PLCopen</a>. The motion engine implements different motion algorithms and <a href="#">functions</a> to create, access and delete pipes, pipe blocks and axes (e.g. <a href="#">MLPNAxisCreate</a>, <a href="#">MLGearInit</a>, <a href="#">MLPipeAct</a>).</p> <p>It also provides a set of Functions and Function Blocks that IEC 61131-3 applications can use to control the behavior of these algorithms</p>
I/O manager	Manages I/Os and I/O drivers. It works closely with the VM Manager instances to map transparently all IEC 61131-3 variables declared as input or output
Motion Bus	A plug-in giving access to the EtherCAT network
custom function blocks	A plug-in implementing <a href="#">custom function blocks</a>

Interface between the realtime and non-realtime systems is done via shared memory buffers.



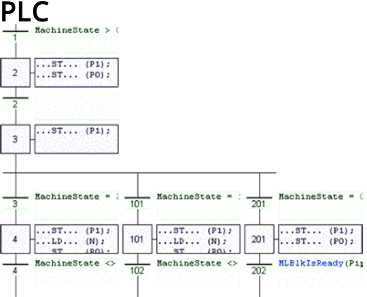
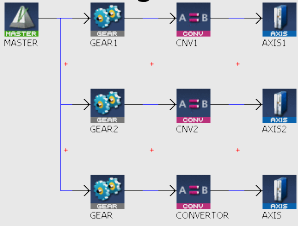


The Runtime communicates with the KAS IDE during operation to:

- Receive further instructions from the KAS IDE such as a direct motion command
- Provide status information to the KAS IDE for motion and operation of the application program
- Provide information displayed on the KAS IDE scope
- Provide Log information to the KAS IDE

#### NOTE

When the KAS Runtime is implemented with a programmable drive, the interface between the real-time and non real-time parts is done via Ethernet based on TCP/IP protocol.

### 4.3 KAS Breakdown

Domains	Concept (Technology)	Task (Tools)	Reference
<b>HMI</b> 		Using Kollmorgen Visualization Builder Add an HMI Device	HMI
<b>Controllers – PCMM &amp; PDMM</b> 	PCMM and AKD PDMM (Programmable Drive Multi-axis Master)	Add the Controller Configure the Controller	Controllers
<b>PLC</b> 	IEC 61131-3	ST editor IL editor FBD editor FFLD editor SFC editor Variable dictionary Softscope	STLanguage IL Language FBD Language FFLD Language SFC Language
<b>Motion Engines</b>  	Motion Concept <ul style="list-style-type: none"> <li>• Pipe Network</li> <li>• PLCopen</li> </ul>	Design Pipe Network Pipe Network Editor Create Cam Profile Cam Profile Editor Softscope	
<b>Fieldbus</b> 	EtherCAT	Configure EtherCAT Motion Bus	Motion Bus and Fieldbuses Cables

Domains	Concept (Technology)	Task (Tools)	Reference
<b>I/O Terminals</b> 	EtherCAT	<a href="#">Add and Configure I/O Slices</a> <a href="#">Map Input and Output to Variables</a>	<a href="#">Remote Input/Output Terminals</a>
<b>Drives</b> 	<a href="#">AKD</a> <a href="#">AKD2G</a> <a href="#">AKD-C</a> <a href="#">AKD-N</a> <a href="#">S300</a> <a href="#">S700</a>	<a href="#">Add and Configure a Servo Drive</a> <a href="#">AKD Drive Configuration</a> <a href="#">Download AKD PDMM Drive Firmware</a> <a href="#">AKD/AKD2G Firmware Update</a>	<a href="#">Drives</a>
<b>Safety</b> 	Safety over EtherCAT	<a href="#">Working with SafePLC2</a> 2. <a href="#">Projects How To Set Up a Variable to Monitor</a> 3. <a href="#">Safety Data AKD2G Safety Parametrization Using FSoE with SCU-1-EC and PCMM/AKD PDMM</a> 4. <a href="#">Set Up FSoE Master and an AKD2G with SafeMotion Monitor</a>	<a href="#">BBH FSoE Master Drives</a> <a href="#">AKT2G-SDI-004-000</a> <a href="#">AKT2G-SDO-004-000</a>
<b>Motors</b> 	Kollmorgen Servomotor		<a href="#">AKM2G</a> <a href="#">AKM</a> <a href="#">Cartridge DDR</a> <a href="#">Housed DDR</a> <a href="#">KBM</a> <a href="#">TBM</a> <a href="#">Direct Drive</a> <a href="#">Linear</a>
<b>Mechanical</b> 			<a href="#">Linear Actuators</a> <a href="#">Gearheads</a>

Table 2-1: KAS - Technologies and Tools

### 4.3.1 Human-Machine Interface



**Figure 2-8:** Hardware to Display the Human-Machine Interface

PLCs interact with people for the purpose of configuration, alarm reporting or everyday control. A Human-Machine Interface (HMI) is employed for this purpose. A simple system uses buttons and lights to interact with the end-user. Text displays are available as well as graphics on the touch panels.

Most modern PLCs can communicate over a network to some other systems, such as a computer running a SCADA system.

The communication between the HMI and the PLC is based on Modbus over TCP/IP (Modbus TCP is the Ethernet version of Modbus) by means of a standard Ethernet cable that connects the two devices.

This communication is done in the background, asynchronously, at the cycle time (20-1000 milliseconds) specified in the Controller Properties (see ["Configure the Controller" on page 119](#)). Variables defined in the HMI to describe the interface (see [Map Variables to HMI](#)) are passed to the AKD PDM/PCMM this way. This means there is no data coherency in the data exchange because the variables read by the Modbus do not come from the same PLC cycle. As this data has a rather low priority and is interpreted by human feedback, it should never be noticed by the user.

#### NOTE

Please note that Kollmorgen HMIs are limited to communicating no more frequently than every 100 milliseconds.

### 4.3.2 Controllers



Figure 2-9: High, medium and low voltage AKD PDMMs and the PCMM.

Kollmorgen Automation Suite controllers consist of the AKD PDMM (Programmable Drive Multi-Axis Master) and PCMM (Programmable Controller Multi-Axis Master).

#### 4.3.2.1 AKD PDMM and PCMM Hardware

The AKD PDMM comprises of three printed circuit boards (PCB) while the PCMM comprises of one.

- Power board (AKD PDMM only)
- AKD control card (AKD PDMM only)
- AKD PDMM and PCMM option card, available in two variants:
  - AKD PDMM**
    - NXP/Freescale QorIQ P1011 processor (800MHz)
    - NXP/Freescale QorIQ P2010 processor (1.2GHz)
  - PCMM**
    - NXP/Freescale QorIQ P2010 processor (1.2GHz)
    - NXP/Freescale QorIQ P2020 processor (Dual Core 1.2GHz)

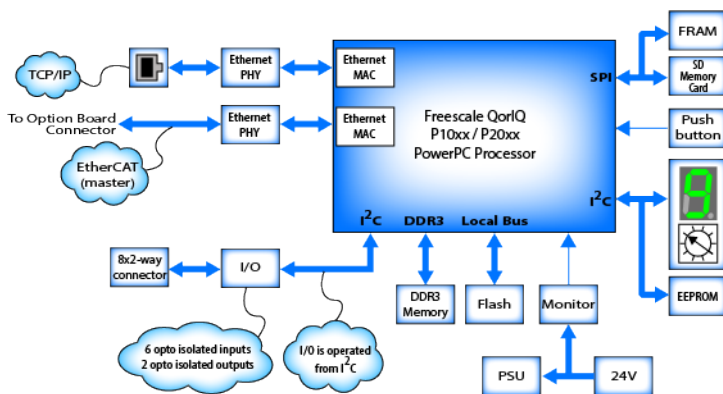


Figure 2-10: AKD PDMM and PCMM card

**NOTE**

Please note that any reference to AKD PDMM or PCMM refers to both the 800MHz and the single- and dual-core 1.2GHz variants, unless otherwise noted.



### 4.3.2.2 PDMM or PCMM Rotary Switch



The rotary switch on the AKD PDMM or PCMM can be set on a position from 0 to 9.

Switch Position	Description
<b>Position 0</b>	The drive tries to get an IP address from a DHCP server. If the DHCP fails, then the AKD PDMM or PCMM uses AutoIP to get a usable IP address.
<b>Position 1</b>	The default custom static IP address, 192.168.0.101 or a custom IP address.
<b>Position 2-9</b>	The AKD PDMM or PCMM is pre-configured with static IP addresses ranging from 192.168.0.102 (Position 2) to 192.168.0.109 (Position 9).

**TIP**

If a DHCP server is not present, the drive will assume an Automatic Private IP Address of the form 169.254.x.x

**NOTE**

The AKD PDMM or PCMM will not set (or show) an IP address without an attached network cable.

### 4.3.2.3 PDMM & PCMM Web Server

These controllers contain a web server that allows you to perform the following operations:

- Read information about the controller (model type, firmware version, version of your KAS application)
- Diagnostic your system (CPU speed and usage, total and free storage space, list the EtherCAT devices)
- Configure some parameters (change the IP address, upgrade the firmware)
- Interact with your application (Start and Stop your KAS application , see the logs)

**KOLLMORGEN**  
Because Motion Matters™

LOGIN | HOME | CONTACT US | ABOUT

KAS Application Settings Backup & Restore Diagnostics Help

**Kollmorgen Automation Suite™**

Manufacturer	_Kollmorgen
CPU Speed	_1.2GHz
CPU Cores	_1
Model Number	_AKD-M00306-M1Ec-D000
Serial Number	_B247000086
Hardware Revision	_DB
TCP/IP Mac Address	_00:23:1b:f0:08:7d
EtherCAT Mac Address	_00:23:1b:f0:08:7e

**EtherCAT**

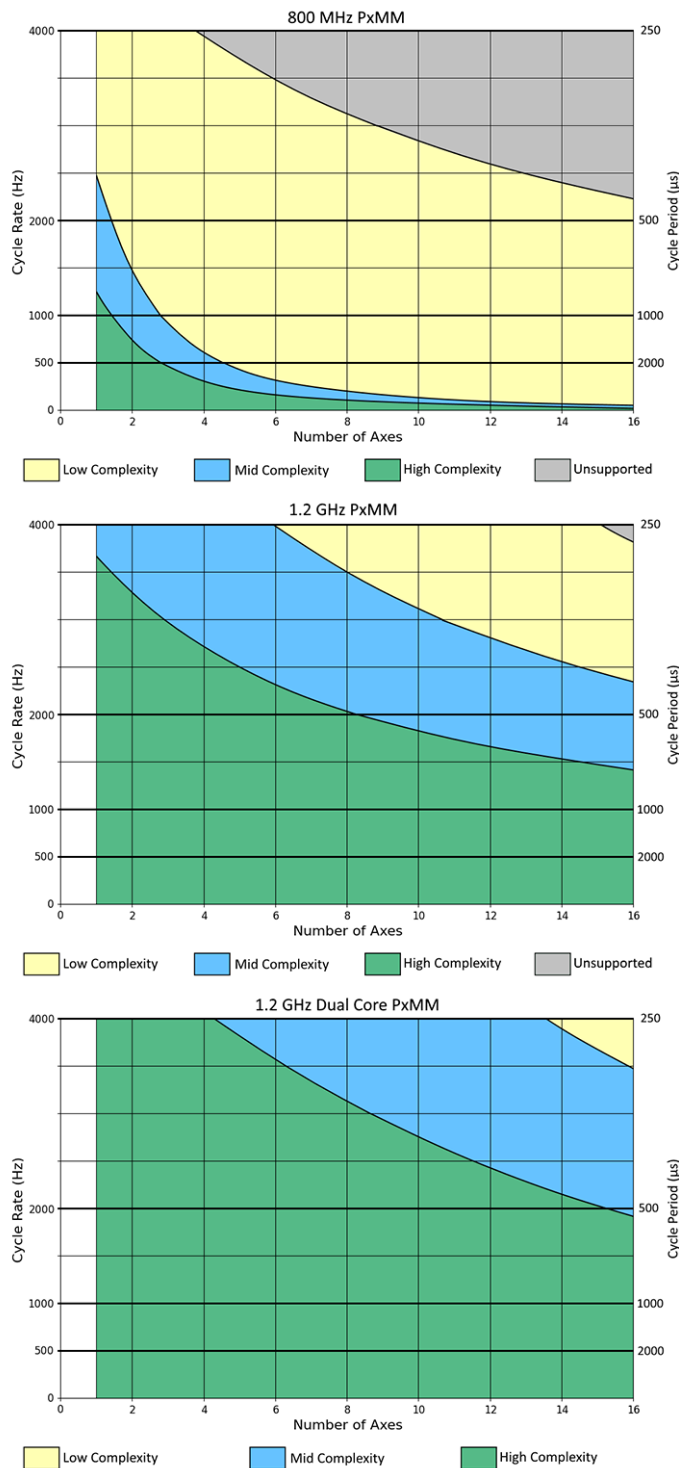
Kollmorgen  
203A West Rock Road  
Radford, VA 24141 USA

See [Using the KAS Web Server](#) for more information.

### 4.3.2.4 Controller Performance

Kollmorgen offers several controllers with varying performance. This section compares the relative performance to help you decide what is best for your needs.

#### 4.3.2.4.1 Performance Graphs



**Figure 2-11:** Graphs comparing the performance of 800 MHz single-core, 1.2 GHz single-core, and 1.2 GHz dual-core controllers.

- A *high-complexity* application is characterized by complex setups such as most axes being slave axes, and cammed or geared to a master axis, frequent moves, frequent reading and setting of external I/O points, and complex PLC logic.
- A *low-complexity* application is characterized by point-to-point and velocity moves, occasional reading of external I/O points, and simple PLC logic.
- A *middle-complexity* application lies in between and is usually characterized by most axes being point-to-point or velocity moves and some axes being cammed or geared to a master; PLC logic is split so that parts of a complex logic run at given times.

Generally speaking, Pipe Network applications use less of the controller's CPU. So for the purposes of using the above graphs, a Pipe Network application will tend to act like a slightly lower-complexity application while a PLCopen application will tend to act like a slightly higher-complexity application.

*Should I use a single 800MHz, single-core 1.2 GHz, or dual-core 1.2 GHz controller?*

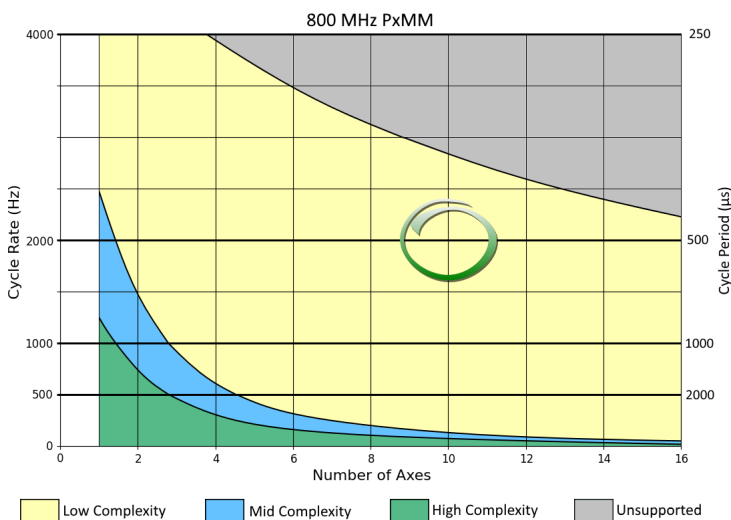
To determine which controller model to use for initial development of an application, the axis count and cycle rate of the application can be used to determine what type of application can be run on different controllers. For example, suppose you are planning a middle-performance application for 6 axes that runs at a 2 kHz (500us) cycle rate. Looking first at the 800 MHz PxMM, you can see that it can only handle a low-complexity application with these requirements. However, the single-core 1.2 GHz PxMM can handle a high-complexity application with these requirements. Since only a middle-complexity application is needed, the single-core 1.2 GHz PxMM should be sufficient for this application. If in doubt, start with a higher-performance controller model and after optimizing the application code, determine if a lower performance model will meet the present and future needs for the application.

**NOTE**

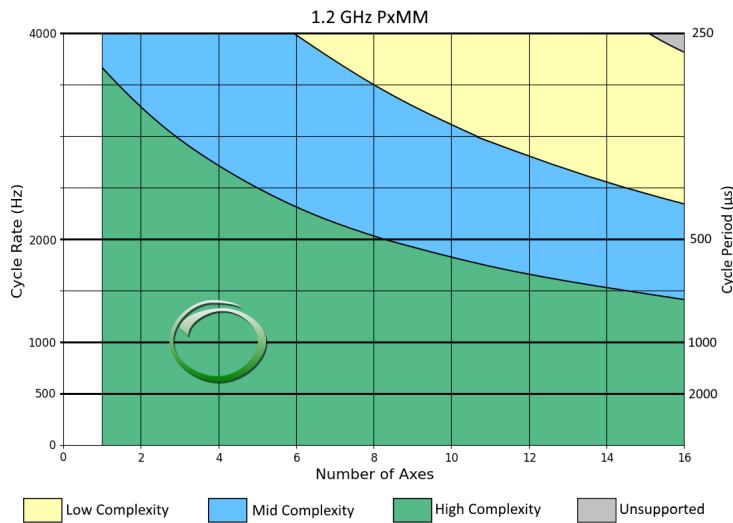
The graphs above are for guidance in initial controller selection for development. During development, the performance of an application should be measured to ensure that the selected controller will meet the performance requirements. Measurements can be done with KAS's digital oscilloscope using the trace times traces, the [GetCtrlPerf](#) function block, and by observing the CPU usage on the [Hardware Status](#) tab of the KAS controller's web page interface.

**Examples**

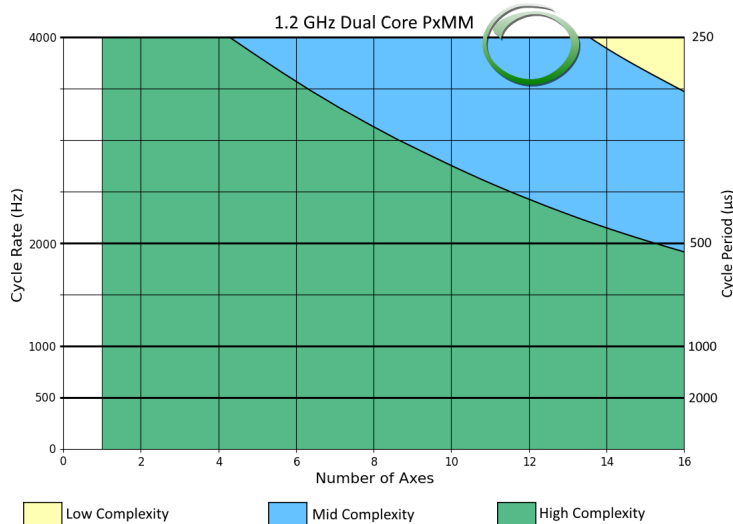
- A 10-axis low-complexity application running at 2 kHz (500us) can be handled by an 800 MHz PxMM.



- A 4-axis high-complexity application running at 1 kHz (1000us) can be handled by a single-core 1.2 GHz PxMM.



- A 12-axis middle-complexity application running at 4 kHz (250us) requires the dual-core 1.2 GHz PxMM.



How much improvement will I get in moving from a single-core 1.2 GHz PxMM to a dual-core PxMM?

The dual-core PxMM improves performance by running PLC code and motion code at the same time. Typical performance gain varies from 45% to 80% depending on the type of application being run. The greatest performance gain is obtained when PLC code execution time and motion code execution time, measurable using the trace times feature (see [Practical Application: Using Trace Time To Measure CPU Load](#)), are roughly the same. If one of these two values dominate the execution time, then the performance gain will be on the smaller side.

Can I downgrade the controller used to run my application?

- If you are using a *dual-core* PxMM and the peak CPU usage (see [Hardware Status](#)) is under 40%, then the single-core 1.2 GHz PxMM model will likely meet your needs.
- If you are using a *dual-core* PxMM and the peak CPU usage is under 25%, then the 800 MHz PxMM model will likely meet your needs.
- If you are using a *single-core* PxMM and the peak CPU usage is under 50%, then the 800 MHz PxMM model will likely meet your needs.

[KDN has an article](#) that has a tool to help you pick a controller which matches your project needs.

### 4.3.3 Communication and Fieldbus

#### 4.3.3.1 Fieldbus

Fieldbus allows a machine to be connected to other machines in an automation systems network. Typically, such a connection is referred to as a “factory automation” network connection.

#### 4.3.3.2 Motion bus

Motion requires the controller to frequently update the drive with new trajectory setpoints. The bus involved in the motion control requires to be able to handle rigid jitter and timing demands including high data throughput and low latency.

##### 4.3.3.2.1 Ethernet

Ethernet is certainly the most popular communications bus today because it is used in most computer networks. Motion control devices using Ethernet allow high-speed connections to computers without requiring special hardware. This reduces the cost and time required to make high-speed connections.

##### 4.3.3.2.2 EtherCAT

The EtherCAT technology overcomes the system limitations of other Ethernet solutions. The Ethernet packet is no longer received, then interpreted and copied as process data at every connection. Instead, the Ethernet frame is processed on the fly. Each slave node reads the data addressed to it, while the telegram is forwarded to the next device. Similarly, input data is inserted while the telegram passes through. The telegrams are only delayed by a few nanoseconds.

### 4.3.4 Input Output Bus

Input/Output refers to the communication and acquisition of data between your automation system, and the outside world (possibly a human, or another information processing system). Inputs are the signals or data received by the automation system, and outputs are the signals or data sent from it. Automation systems built with KAS are interrupt-driven. Typical interrupt uses include the following: system timers, disks I/O, power-off signals, and exceptions handling.



Figure 2-12: I/O Modules

I/O modules provide a convenient modular package which is simple to wire and add or change slice types.

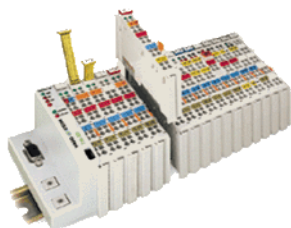


Figure 2-13: Standard I/O Couplers and Slices

### 4.3.5 Drives

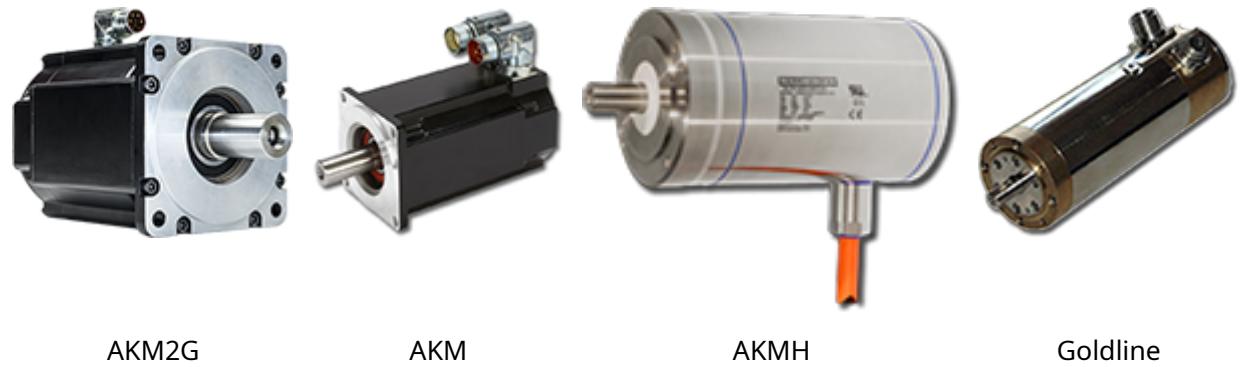


See [Drives](#) in [Hardware Devices](#) chapter for more details.

### 4.3.6 Kollmorgen Motors

**NOTE**  
 The following images are all hyperlinked to their respective product pages on [Kollmorgen.com](http://Kollmorgen.com).

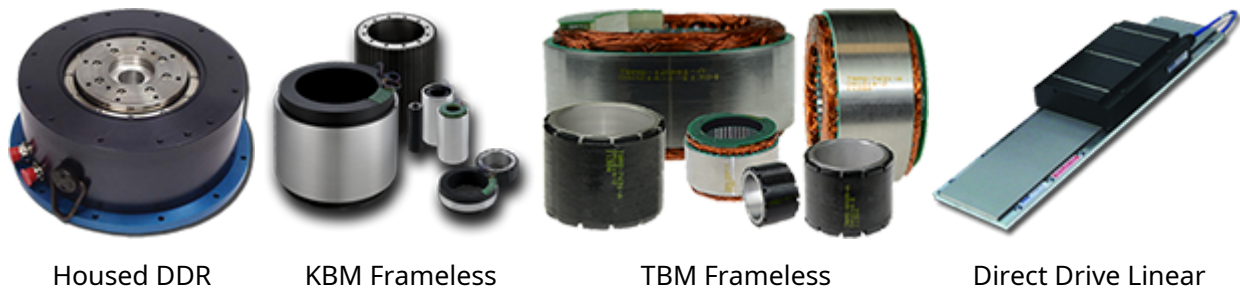
#### *Servomotors*



#### *Cartridge Motors*



#### *Cartridge DDR Direct Drives*



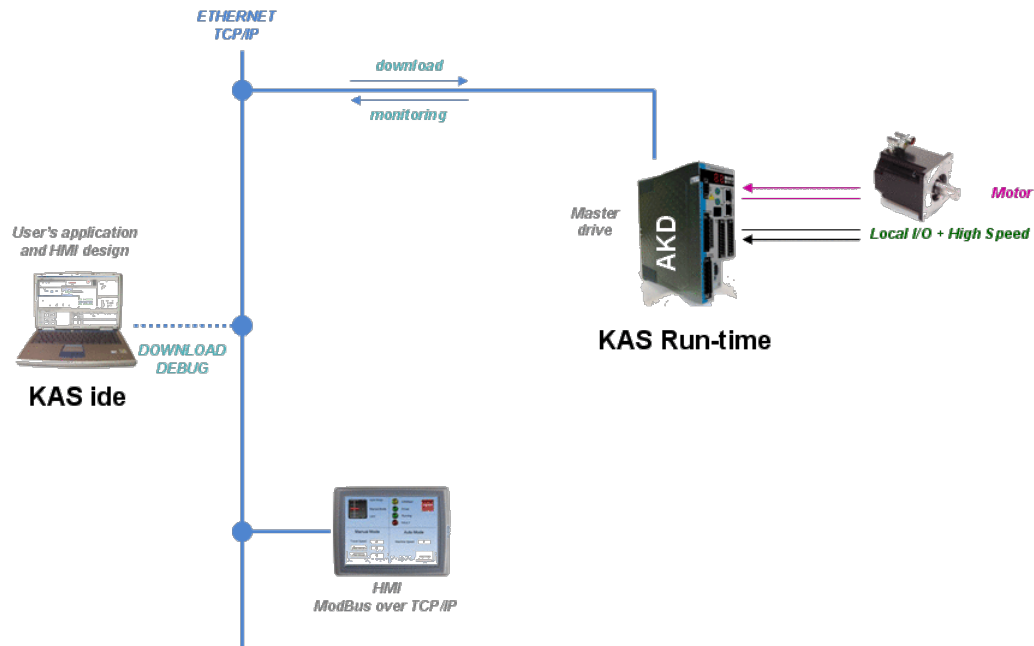
## 4.4 Different Implementations

KAS supports the following architectures:

- [Single-Axis with PDMM \(→ p. 47\)](#)
- [Multi-Axis with PDMM or PCMM \(→ p. 48\)](#)

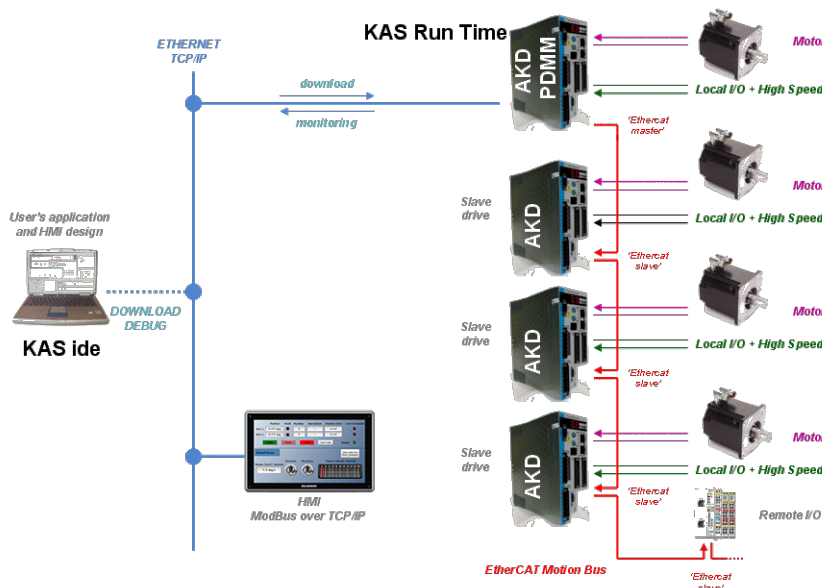
### 4.4.1 Single-Axis with PDMM

The scalable system architecture begins with an integrated programmable controller and drive package to control a single axis.



### 4.4.2 Multi-Axis with PDMM or PCMM

The AKD PDMM (integrated programmable controller and drive package) or PCMM (programmable controller) can be scaled to control additional drives w/ motors and I/O devices via EtherCAT. This configuration can manage up to 32 axes.





## 5 Describing KAS Graphical User Interface

This section provides an overview of the common elements of the KAS IDE GUI.

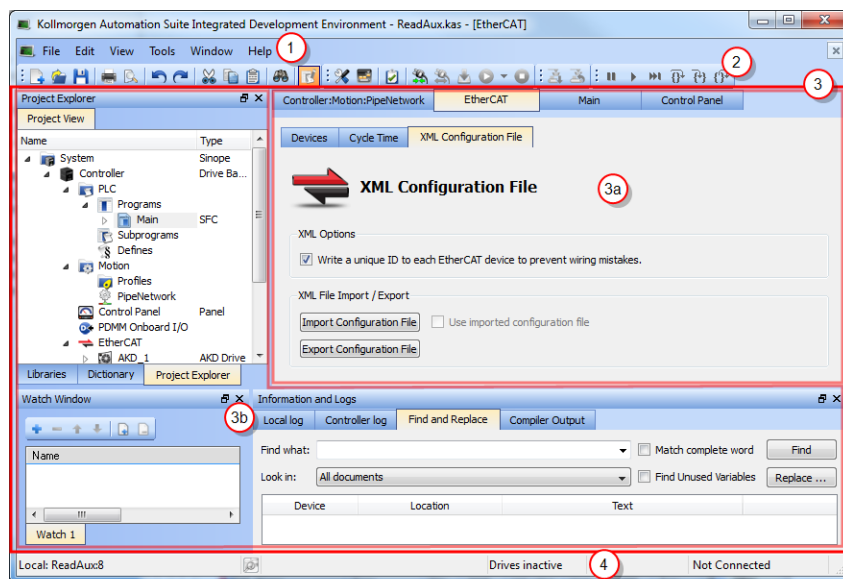
### NOTE

- For in-depth discussion of interface elements, refer to [KAS IDE Reference](#)
- For KAS Simulator GUI, refer to [Using the KAS Simulator](#)
- For AKD drive GUI View, refer to [AKD Drive](#)

### 5.1 Windows and Panels Overview

#### 5.1.1 Main Window

The KAS IDE interface provides an all-in-one-window integrated workspace.



**Figure 3-1:** KAS IDE Main Window

The main view in the Integrated Development Environment (IDE) is a Multiple Document Interface (MDI) environment. This provides an easy-to-use and customizable view; including the capability to hide, enlarge or overlap windows in order to optimize visibility.

The main view is saved when you exit the application. This ensures that your workspace remains the same each time you open and use the KAS IDE.

The KAS IDE main window contains the following items:

- Menu bar (see call out 1)
- Toolbar 2 A toolbar is a little bar with icons which is usually located under the menu bar of a window.
- Workspace 3 which contains:
  - A specific area dedicated to displaying the workspace children windows 3a
  - Several toolboxes 3b A toolbox is a child window that provides you with some functions to perform specific tasks.
- Status bar at the bottom 4 displaying the current state of the target

### 5.1.1.1 About toolboxes

The available toolboxes include:

- [5.1.2 Project Explorer](#)
- [5.1.3 Libraries](#)
- [5.1.4 Dictionary](#)
- [5.1.5 Information and Logs](#)

#### **TIP**

You can hide/show each toolbox and toolbar directly from the contextual menus in any title bar (i.e. menu, toolbar or toolboxes).

### 5.1.2 Project Explorer

The Project Explorer toolbox is a window that displays machine application information in a tree-structure representation. This window contains all the following items used to design, implement, test, and document the application.

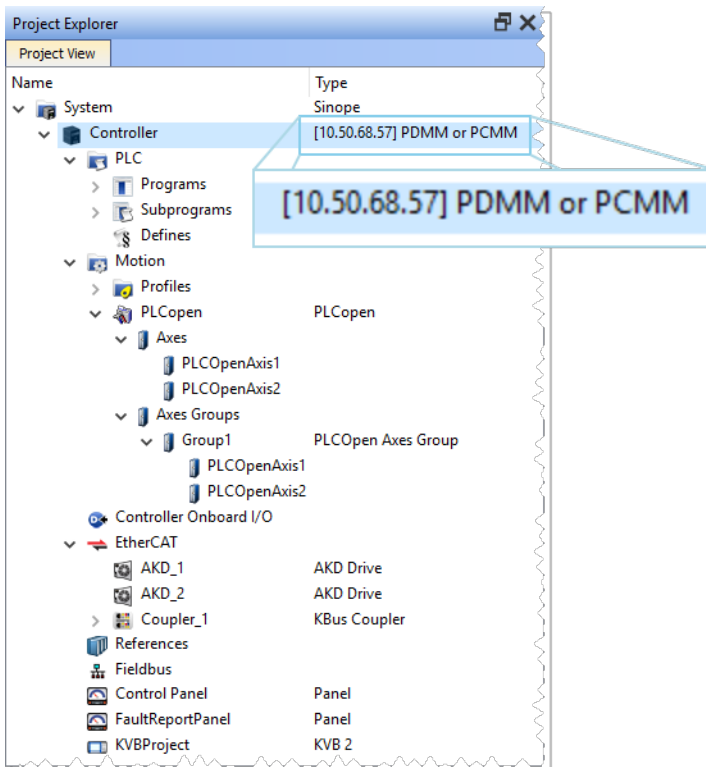


Figure 3-2: Project Explorer with Controller type and IP address.

Item	Description
<b>Hardware</b>	<ul style="list-style-type: none"> <li>Devices that make up the system such as Controllers, EtherCAT Motion Bus, servo and stepper drives, HMI devices, I/O Terminals, etc.</li> </ul>
<b>PLC (IEC 61131-3)</b>	<ul style="list-style-type: none"> <li>Programs that control the system</li> <li>User-defined Functions and Function Blocks</li> </ul>
<b>Motion</b>	<ul style="list-style-type: none"> <li>Pipe Networks or PLCopen</li> <li>Axis objects</li> <li>Cam profiles</li> </ul>

#### TIP

You can navigate in the project-tree by entering the item's initial letter, or by means of the arrow keys.

A project is made of several items that are:

- [System](#)
- [Controller](#)
  - [PLC](#)
    - [Programs](#)
    - [Subprograms](#)
    - [Defines](#)

- [Motion](#)
    - [Profiles](#)
    - [Pipe Network](#) or [PLCopen](#)
  - [Controller Onboard I/O](#)
  - [EtherCAT](#)
    - [AKD Drive](#)
    - [AKT \(K-Bus\) I/O Coupler](#)
    - [Device](#)
      - [Module](#)
  - [References](#)
  - [Fieldbus](#)
  - [Control Panel](#)
  - [KVB Project](#)
  - [SafePLC2 Project](#)
- [HMI Device](#)
    - [KVB Project](#)

**NOTE**

**Node Names:** Allowable characters for node names include A-Z, a-z, 0-9, . (period), - (hyphen), and \_ (underscore).

**5.1.2.1 System**

This item concerns the whole project. A right-click opens its menu that provides the following options:

Command	Description
<b>Add HMI Device</b>	Add a new HMI device with a KVB panel. For mode details see <a href="#">HMI Device</a> below.

Table 3-1: System Node - Contextual Menu

### 5.1.2.2 Controller

This item displays the controller's IP address and controller hardware type. It is also used to [Access the WebServer From the IDE](#). The webserver functionality may be used directly within the IDE. For more information on the webserver see [Using the KAS Web Server](#).

#### NOTE

Please note that the IP address is shown as 127.0.0.1 if the system is in simulation mode.

Command	Description
<b>Add Control Panel</b>	Add a new control panel to the controller. For more details see <a href="#">Control Panel</a> .
<b>Import Control Panel</b>	Import a pre-configured control panel for use in the project. See <a href="#">Import a Control Panel</a> for more information.
<b>Add KVB Project</b>	Add a new KVB panel which is embedded into the controller. For more details see <a href="#">KVB Project</a> .
<b>Import KVB Project</b>	Import a compressed ("zipped") KVB project, which may be created in KAS or KVB. The system will validate the compressed KVB project and add the panel.
<b>Add SafePLC2 Project</b>	Add a node to the tree that allows you to access a SafePLC2 project directly from the KAS IDE. See <a href="#">Working with SafePLC2 Projects</a> for more information.
<b>Import SafePLC2 Project</b>	Select a .SPL2 file to import.  There may only be a single instance of a SafePLC2 project in the KAS IDE. If none exists, a SafePLC2 Project will be added to the end of the Project tree. If there is a project you will be prompted to overwrite the project or cancel the import. Note that the import cannot occur if SafePLC2 is running.
<b>Add Fieldbus</b>	Add a node to access the Fieldbus Editor. See for more information.
<b>Access Web Server</b>	This command opens the web server interface in the GUI. See <a href="#">Access the WebServer From the IDE</a> and <a href="#">Using the KAS Web Server</a> for more information.

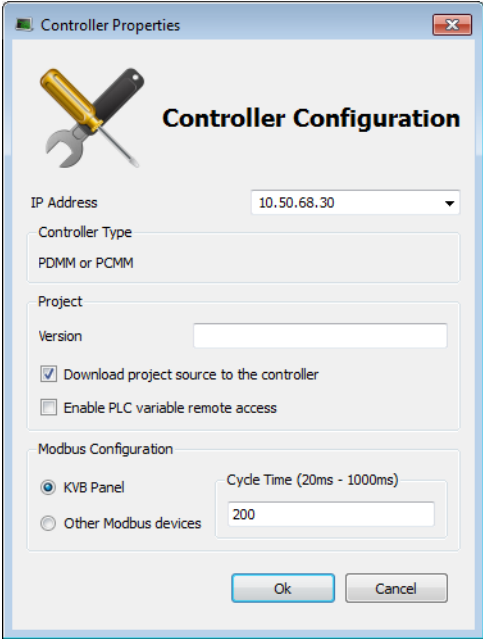

Command	Description
<p><b>Properties</b></p>	<p>Open a dialog box to configure the controller. See <a href="#">Configure the Controller</a> for a full description of this dialog box.</p>  <p><b>Figure 3-3:</b> Configure the Device</p> <p><b>Parameters Description</b></p> <ul style="list-style-type: none"> <li>• IP Address or COM: allows for connecting to and downloading your application to the controller. Changing this value modifies the IP address for the Controller.             <ul style="list-style-type: none"> <li>• The last 10 IPs entered are accessible and are stored with the project. Selecting an item in the list and pressing the 'Delete' key will clear the entry from the list.</li> <li>• When you click  to choose the <a href="#">simulation</a> mode, this address is disabled.</li> </ul> </li> </ul> <p><b>ⓘ IMPORTANT</b></p> <p>Do not use leading zeroes when entering an IP address. The connection to the controller will automatically fail if leading zeroes are used. For example, use <b>10.2.3.40</b> instead of <b>010.002.003.040</b>.</p> <ul style="list-style-type: none"> <li>• The Controller type can be a AKD PDMM or PCMM.</li> <li>• <a href="#">Different Implementations</a></li> <li>• Version number is used to ensure both versions of your application on the KAS IDE and the KAS Runtime are the same</li> <li>• Enabling "Download project source to the controller" allows for comparing source on the controller to the source on your computer.</li> <li>• The "Enable PLC variable remote access" is disabled by default. Enabling it will allow users to read/write variables using an HTTP connection.</li> <li>• "Modbus Configuration" sets what the Modbus will connect to and the Cycle Time specifies the update rate.</li> </ul>

Table 3-2: Controller Node - Contextual Menu

A controller is composed of a PLC item, a Motion item, control panels, an EtherCAT Motion Bus and some References. These items are described in the following sections.

### 5.1.2.2.1 PLC

This item contains all the PLC (Virtual Machine) part of the controller. The following items can be present in this item:

- **Program** items
- **Subprogram** items
- Some "Defines"

Command	Description
Libraries	Import new libraries

Table 3-3: PLC Node - Contextual Menu

#### 5.1.2.2.1.1 Programs

Command	Description
New Program	Add new program items (SFC, ST, FBD, IL or FFLD)
Cycle	Configure the cycle of the virtual machine For mode details on Cycle, see <a href="#">"Define the PLC Cycle" on page 204</a>
Import	Import a saved program

Table 3-4: Program Node - Contextual Menu

#### TIP

Programs and Subprograms can be reordered by drag-and-drop. Dragging an item up the list places it above the program you drop it on, while dragging down places it below.

Command	Description
Add Child SFC	Add a child program to this program. Note that this is reserved for the first SFC program only.
Import Child SFC	Import a saved SFC program to the current program. Note that this is reserved for the first SFC program only.  <i>How to import all children from one project to another?</i> <ol style="list-style-type: none"> <li>1. Export each program one at a time from the existing project</li> <li>2. Save the program (specify a location and a name, avoiding spaces in the file name)</li> <li>3. Close the project</li> <li>4. Open the project to be updated</li> <li>5. Import each saved program in the project tree</li> <li>6. Rename the program if needed</li> </ol> <p>Please note that only local variables are copied (not the global variables)</p>
Export	Save the selected program to your file server, avoiding spaces in the file name.
Rename	Rename the selected program.
Delete	Delete the selected program.
Print SFC and All Level 2	Print all PLC programs. See <a href="#">Print</a> for more details.

Table 3-5: Program Item - Contextual Menu

**TIP**

You can double-click to open the program in the workspace.

**5.1.2.2.1.2 Subprograms**

Command	Description
New Function (Subprogram)	Add a new subprogram item (ST,FBD, IL or FFLD)
New UDFB	Add a new UDFB item (ST,FBD, IL or FFLD)
Import	Import a saved program

Table 3-6: Subprogram Node - Contextual Menu

You can create your own functions as well as functional blocks that are called UDFBs (User-Defined Functional Blocks). For each of them, you can use the following commands:

Command	Description
Export	Save the selected subprogram onto your file server
Rename	Rename the selected subprogram
Create Unlocked Copy	Duplicate the selected, locked subprogram. The duplicate will not be locked.
Delete	Delete the selected subprogram
In/Out Parameters	Open the Program Properties dialog box to <a href="#">Declare Functions or Function Blocks</a> . This item is disabled for locked UDFBs.

Table 3-7: Subprogram Item - Contextual Menu

**5.1.2.2.1.3 Defines**

This item contains all the global definitions in the scope of the corresponding device.

**TIP**

You can double-click a **Define** item to show these global definitions. A file of internal defines can be found in [the installation directory of the KAS IDE] \Astrolabe\Bin\HwDef\lib.eqv.

[Use the Defines List](#)

**5.1.2.2.2 Motion**

The motion item contains the motion-specific items (i.e. the Profiles and PipeNetwork items).

Command	Description
Motion Engines	Choose the motion engine for your application between <a href="#">PLCopen</a> and <a href="#">PipeNetwork</a>

**5.1.2.2.2.1 Profiles**

This item contains all the cam profiles in the project.

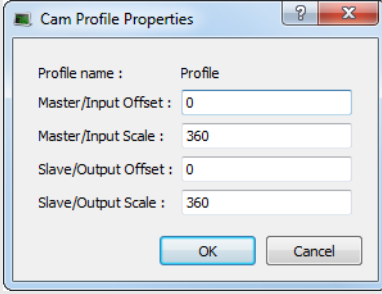
Command	Description
New Profile	Create a new cam profile and add it to this device (*.csv, *.cam) For mode details, see <a href="#">"Adding Cam Profiles" on page 198</a>
Import	Import already existing cam profiles to your project
Show compiled code	Show the code corresponding to the selected cam profile



Table 3-8: Profiles Node - Contextual Menu

Right-clicking on a cam profile provides additional commands.

Command	Description
<b>Rename</b>	Provide the cam profile a unique name
<b>Delete</b>	Remove the cam profile from the list
<b>Export</b>	Save the cam profile in CAM (.cam) format
<b>Properties</b>	Open a dialog to modify the cam profile's Master/Input Slave/Output Offset and Scale values.



For more information on cam profiles see [Adding Cam Profiles](#) and [Cam Profile Editor](#).

#### 5.1.2.2.2 Pipe Network

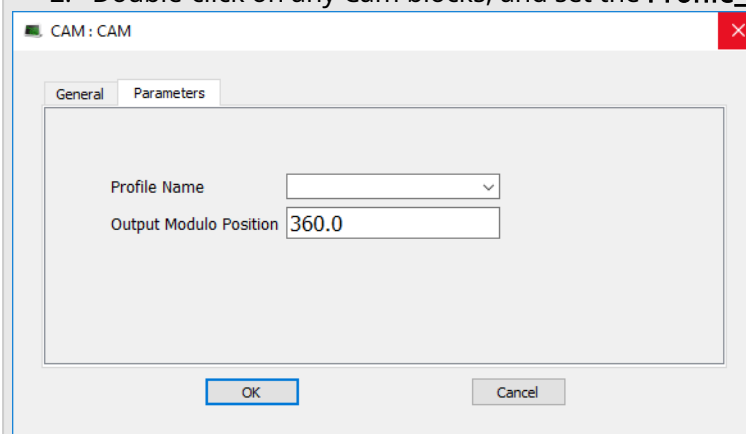
This menu applies to the Pipe Network in the project.

Command	Description
<b>Import and replace</b>	This command replaces the existing Pipe Network with a pre-saved Pipe Network. You will be presented with a dialog box to locate the pre-saved file. The Pipe Network Editor will be opened when the file is imported.
<b>Export</b>	Export the Pipe Network to a file for reuse.
<b>Show compiled code</b>	Show the code corresponding to the Pipe Network

#### ⚠ IMPORTANT

The existing EtherCAT axis mapping is lost when using **Import and replace**. Additionally, profiles assigned to Cam blocks are cleared at this time.

1. Double-click on **EtherCAT** in the Project View to open the [EtherCAT Devices tab](#) tab so you can reassign the axes.
2. Double-click on any Cam blocks, and set the **Profile\_Name** parameter.



### 5.1.2.2.2.3 PLCopen

Command	Description
<b>New Axis</b>	Add a new axis to your project For mode details, <a href="#">Create PLCopen Axis</a>
<b>Show compiled code</b>	Show the code corresponding to the PLCopen

Table 3-9: PLCopen Node - Contextual Menu

### 5.1.2.2.2.4 Axes

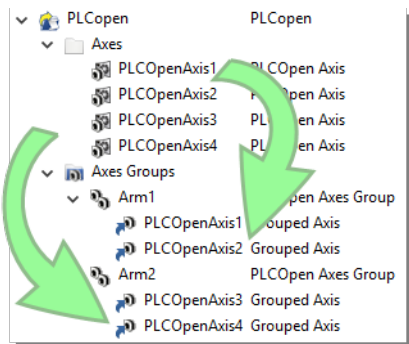
Each axis is listed here, whether it is found via a scan of the EtherCAT network, or added manually. To add an axis manually, right-click and select **New Axis**. For each PLCopen axis you can use the following commands:

Command	Description
<b>Properties</b>	Open a dialog box to configure the <a href="#">PLCopen axis data</a>
<b>Delete</b>	Delete the selected axis

Table 3-10: Axis Item - Contextual Menu

### 5.1.2.2.2.5 Axes Groups

Axes Groups are collections of related axes defined by the data type AXIS\_GROUP\_REF, typically used by coordinated motion and function blocks such as [MC\\_SetKinTra](#). Right-click on this section to create a new Axes Group. Groups are populated by dragging an existing axis from the above section into the group.

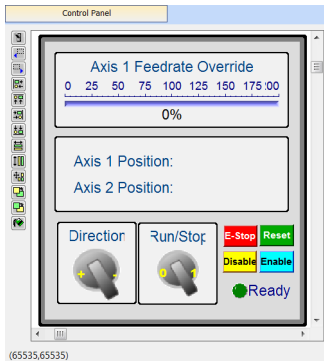


The following commands are available by right-clicking on a group.

Command	Description
<b>Delete</b>	Delete the selected Axes Group
<b>Rename</b>	Change the name of the Axes Group.

### 5.1.2.2.3 Control Panel

This item holds the Control Panel item used to provide a basic interface between you and the virtual machine.



- For more details, [Design the Control Panel with the Internal Control Panel Editor](#)
- For a more advanced tool to build HMI, [KVB Project](#)

Command	Description
<b>Rename</b>	Rename the selected Control panel
<b>Delete</b>	Delete the selected Control panel
<b>Export</b>	Export the control panel for use in other projects. See <a href="#">Export a Control Panel</a> .

Table 3-11: HMI Control Panel Node - Contextual Menu

### 5.1.2.2.4 Controller Onboard I/O

Command	Description
<b>Properties</b>	Open the Properties dialog box to configure the local I/O for the PDMM or PCMM controller.  <a href="#">Configure Controller Onboard I/O</a>

Table 3-12: Controller Onboard I/O Item - Contextual Menu

### 5.1.2.2.5 EtherCAT

This item gives access to all the devices linked to the EtherCAT Motion Bus.

Command	Description
<b>Add Device...</b>	Add a Kollmorgen drive, power supply, or coupler, pr a third-party EtherCAT device to the EtherCAT node in the Project view.
<b>Scan Devices</b>	TheKAS Runtime sends EtherCAT messages to discover the devices present in the network <a href="#">EtherCAT Devices tab</a>
<b>Enable/Disable Online Configuration Mode</b>	Toggles Online Configuration Mode on and off. See <a href="#">Online Configuration Mode</a> for more information.
<b>Properties</b>	Open the Properties dialog box. <a href="#">Configure EtherCAT Motion Bus</a>

Table 3-13: EtherCAT Node - Contextual Menu

[Add & Configure Third Party Devices](#)

#### 5.1.2.2.5.1 AKD2G Drive

You can double-click an AKD2G to set its parameters. [Configure the AKD2G/AKD Drive](#)

Command	Description
Rename	Rename the selected drive
Delete	Delete the selected drive
Configuration	Opens the Configuration tab for the AKD GUI.
Properties	Select the Properties menu to access the EtherCAT device's configuration views.

Table 3-14: AKD Drive Item - Contextual Menu

#### 5.1.2.2.5.2 AKD Drive

You can double-click an AKD to set its parameters. [Configure the AKD2G/AKD Drive](#)

Command	Description
Rename	Rename the selected drive
Delete	Delete the selected drive
Configuration	Opens the Configuration tab for the AKD GUI.
Properties	Select the Properties menu to access the EtherCAT device's configuration views.

Table 3-15: AKD Drive Item - Contextual Menu

#### 5.1.2.2.5.3 AKD-C Central Power Supply

You can double-click an AKD-C to set its parameters. [Configure the AKD2G/AKD Drive](#)

Command	Description
Rename	Rename the selected device
Delete	Delete the selected device
Properties	Select the Properties menu to access the EtherCAT device's configuration views.
Configuration	Opens the Configuration tab for the AKD GUI.

Table 3-16: AKD-C Drive Item - Contextual Menu

#### 5.1.2.2.5.4 AKD-N Drive

You can double-click an AKD-N to set its parameters. [Configure the AKD2G/AKD Drive](#)

Command	Description
Rename	Rename the selected drive
Delete	Delete the selected drive
Properties	Select the Properties menu to access the EtherCAT device's configuration views.
Configuration	Opens the Configuration tab for the AKD GUI.

Table 3-17: AKD-N Drive Item - Contextual Menu

#### 5.1.2.2.5.5 AKT2G (EtherCAT) I/O Coupler

The Coupler node acts as a gateway for E-Bus I/O slices.

Command	Description
<b>Rename</b>	Rename the selected coupler
<b>Delete</b>	Delete the selected coupler
<b>Properties</b>	Select the Properties menu to access the EtherCAT device's configuration views.

Table 3-18: AKT2G I/O Coupler Node - Contextual Menu

Note that these commands are disabled when the controller is running.

See [Remote Input/Output Terminals](#) for links to detailed documentation for each device.

#### 5.1.2.2.5.6 AKT (K-Bus) I/O Coupler

The Standard I/O Coupler node gives access to its I/O slices.

Command	Description
<b>Add I/O Slice</b>	Add a new slice (Digital or Analog Input and Output) to the selected Standard I/O Coupler
<b>Rename</b>	Rename the selected coupler
<b>Delete</b>	Delete the selected coupler
<b>Properties</b>	Select the Properties menu to access the EtherCAT device's configuration views.

Table 3-19: Standard I/O Coupler Node - Contextual Menu

Note that these commands are disabled when the controller is running.

See [EtherCAT Coupler Error Handling And Diagnosis](#) in the [Troubleshooting](#) section for information about diagnosing the coupler LEDs.

#### 5.1.2.2.5.7 I/O Slice

AKT2G (E-Bus) slices are children of the EtherCAT node while AKT (K-Bus) slices are children of the AKT I/O Coupler.

Command	Description
<b>Rename</b>	Rename the selected slice
<b>Delete</b>	Delete the selected slice
<b>Properties</b>	Open the Properties dialog box to configure the I/O slice <a href="#">Map Input and Output to Variables</a>

Table 3-20: I/O Slice - Contextual Menu

#### 5.1.2.2.5.8 Device

Double-clicking a Device accesses its EtherCAT device configuration views.

Command	Description
<b>Rename</b>	Rename the selected device
<b>Delete</b>	Delete the selected device
<b>Add Module...</b>	Add a module to an MDP device. <a href="#">Add Modules to Third Party EtherCAT Devices.</a>
<b>Properties</b>	Access the EtherCAT device's configuration views

Table 3-21: Device - Contextual Menu

### 5.1.2.2.6 Module

Command	Description
Rename	Rename the selected module.
Delete	Delete the selected module.

### 5.1.2.2.7 References

This item allows you to **insert references** into your project. Each reference is a user-defined reference that links any kind of deliverable to your project (for more details, refer to [Use the Reference Folder](#))

Command	Description
Insert Reference	Link any kind of deliverable to your current project
Delete	Delete the reference
Properties	Open the referenced file in the workspace

Table 3-22: Reference Node - Contextual Menu

### 5.1.2.2.8 Fieldbus

This item holds the Fieldbus Editor to configure the Ethernet/IP or Profinet fieldbuses. For mode details see [Fieldbus Editor](#).

### 5.1.2.3 HMI Device

This item holds the HMI (Human Machine Interface) item used to provide an advanced interface between you and the virtual machine.

Command	Description
Add KVB Project	Add a new <a href="#">KVB</a> panel to the controller. For mode details, <a href="#">Add an HMI Device</a> Note that this command is disabled when a KVB panel already exists
Import KVB Project	Import a compressed ("zipped") KVB project, which may be created in KAS or KVB. The system will validate the compressed KVB project and add the panel.
Rename	Rename the selected HMI device
Delete	Delete the selected HMI device

Table 3-23: HMI Device Node - Contextual Menu

#### 5.1.2.3.1 KVB Project

Command	Description
Open KVB	Opens the associated KVB project.
Rename	Rename the selected KVB panel
Delete	Delete the selected KVB panel
Export	Save a copy of the panel in a compressed (.zip) file.

Table 3-24: KVB Panel Node - Contextual Menu

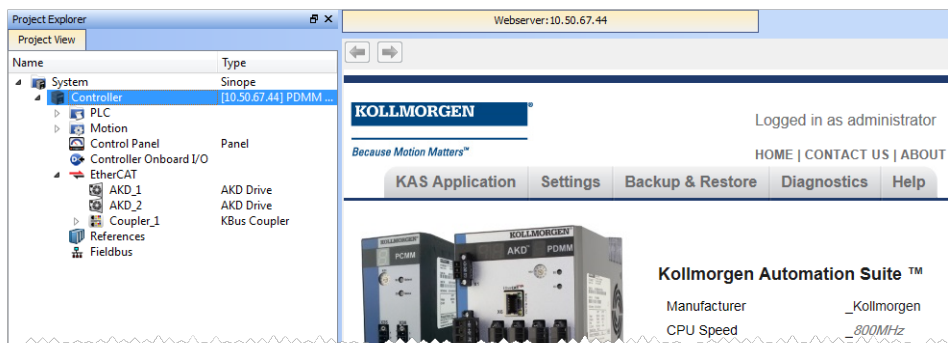
### 5.1.2.4 SafePLC2 Project

Command	Description
Delete	Delete the SafePLC2 project
Export	Select a location and name to export/create a SafePLC2 file.
Open SafePLC2	Creates or opens a synchronized SafePLC2 project. See <a href="#">Working with SafePLC2 Projects</a> for more information.
Rename	Rename the SafePLC2 project

Table 3-25: SafePLC2 Project Node - Contextual Menu

#### 5.1.2.4.1 Access the WebServer From the IDE

Double-clicking **Controller** will both expand/collapse the Controller's components as well as open the web server and automatically log into the *administrator* account. For more information on using the webserver see [Using the KAS Web Server](#).



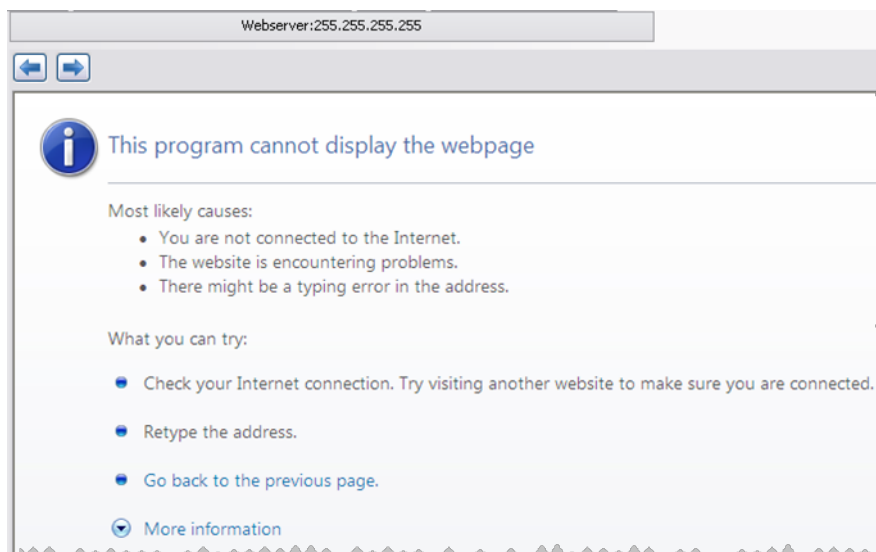
The web server can also be accessed by right-clicking the Controller node and selecting **Access webserver**.

#### TIP

IDE users are considered administrators and therefore are automatically logged into the webserver. You may change the password but there is no logout function.

By default the localhost (127.0.0.1) will be opened. To set the IP address of the controller, right click and select Properties. Enter the proper **Address** and **Controller type** then click **OK**. The page is automatically refreshed.

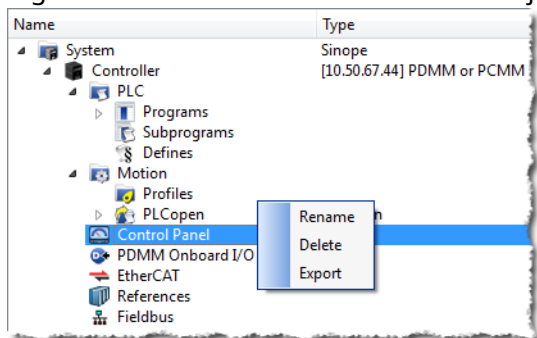
If an invalid or wrong IP address is entered, the following error will be displayed.



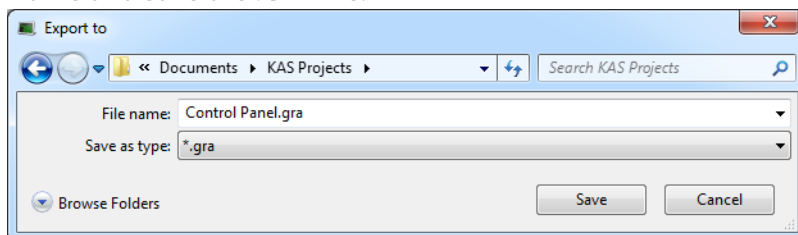
#### 5.1.2.4.1.1 Export a Control Panel

Control Panels may be exported and imported for easy reuse across projects. Any variables associated with the control panel will be preserved across projects.

1. Right-click on the Control Panel in the Project Tree and select **Export**.



2. Name and save the .GRA file.



3. Click OK to confirm the export.

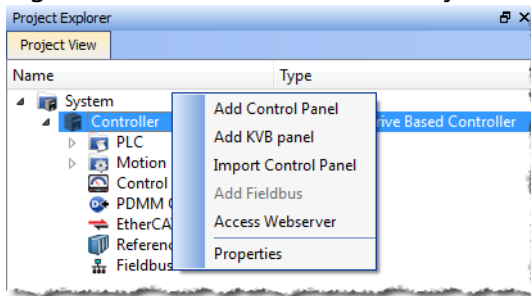
See also: [Import a Control Panel](#)



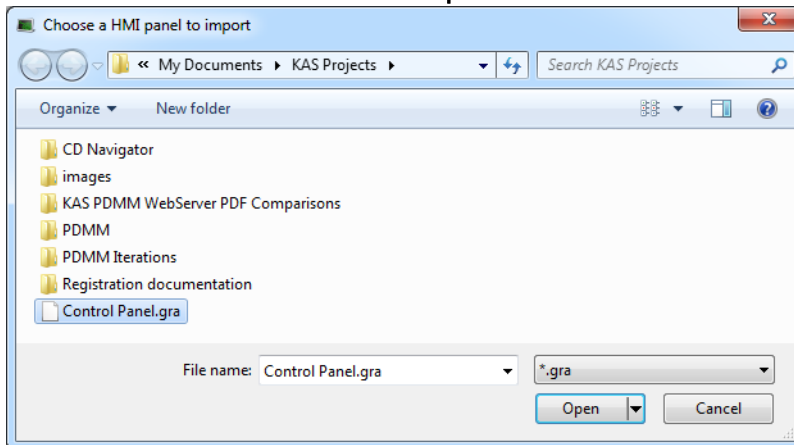
### 5.1.2.4.1.2 Import a Control Panel

Control Panels may be exported and imported for easy reuse across projects. Any variables associated with the control panel will be preserved across projects.

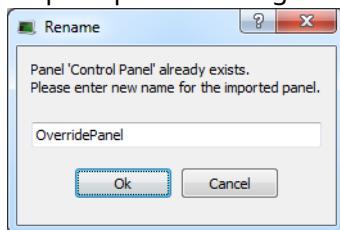
1. Right-click on **Controller** in the Project View and select **Import Control Panel**.



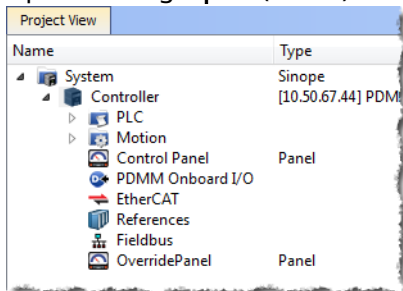
2. Locate a saved .GRA file and click **Open**.



- If the panel you are trying to import has the same name as an existing panel, you will be prompted to change the name or cancel the import.



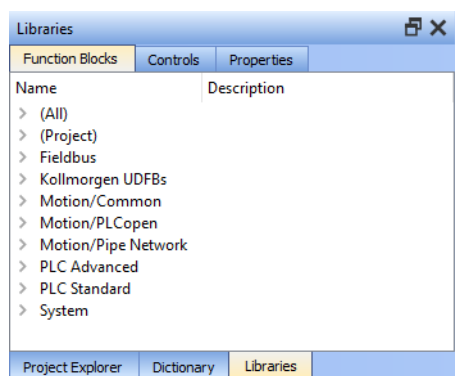
3. Upon clicking **Open** (or **OK**) the panel will be placed in the Project View.



See also: [Export a Control Panel](#)

### 5.1.3 Libraries

This toolbox contains several tabs to access all the functions of the available libraries.



**Figure 3-4:** Libraries Toolbox

#### 5.1.3.1 Function Blocks

This tab displays all the available libraries shown in a tree-structure representation and gathered by categories. You can expand a library to access all its functions. A short description of each function is also available.

The **(All)** category at the top enables you to see the full list of available functions sorted in alphabetical order.

The **(Project)** node contains all the UDFB and subprograms associated to the current project.

For more details about these libraries, refer to the following libraries description:

- PLC Standard
- PLC Advanced
- Motion/Pipe Network
- Motion/PLCopen
- Fieldbus
- System
- Kollmorgen UDFBs

#### TIP

It is possible to use the functions, UDFB or subprograms in PLC editors with a simple drag-and-drop operation.

#### NOTE

Dragging and dropping a Kollmorgen UDFB into the "Defines" editor has no effect. However, if the Kollmorgen UDFB is already imported to the project, then it's prototype will be seen in the editor.

#### 5.1.3.2 Controls

This tab displays all the controls available for the HMI design.

For more details, refer to the [Graphic Objects](#) description.

#### 5.1.3.3 Properties

This tab displays all the properties of an HMI control currently selected in the HMI editor.

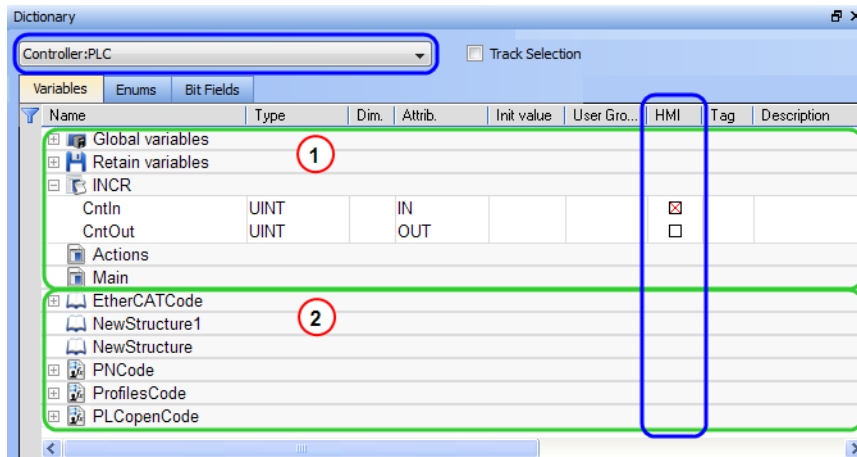
More information about setting the properties of an HMI widget can be found in [Graphic Objects Properties](#).

## 5.1.4 Dictionary

The Dictionary toolbox is used to show all the **identifiers** (variables, data types, sub-routines, etc.) defined within the project. There are three tabs within the Dictionary, the **Variables tab**, the **Enum Tab**, and the **Bit Fields Tab**.

### 5.1.4.1 Variables tab

The Variables tab is used to show all the variables defined within the project. All the variable details are displayed in order to show the variable types, dimensions, attributes, etc.



**Figure 3-5:** Dictionary Toolbox

The list of variables is split into two parts:

- All the **Variables** at the top **1**
- All the **Structures** at the bottom **2**

#### NOTE

For more information about the procedure to create an instance of a structure, see ["Call Functions or Function Blocks" on page 308](#)

#### TIP

To show all the variables of all programs, select 'PLC' in the project tree.

*About the Dictionary's contextual menu.*

Right-click in the Dictionary window to open the menu as follows:

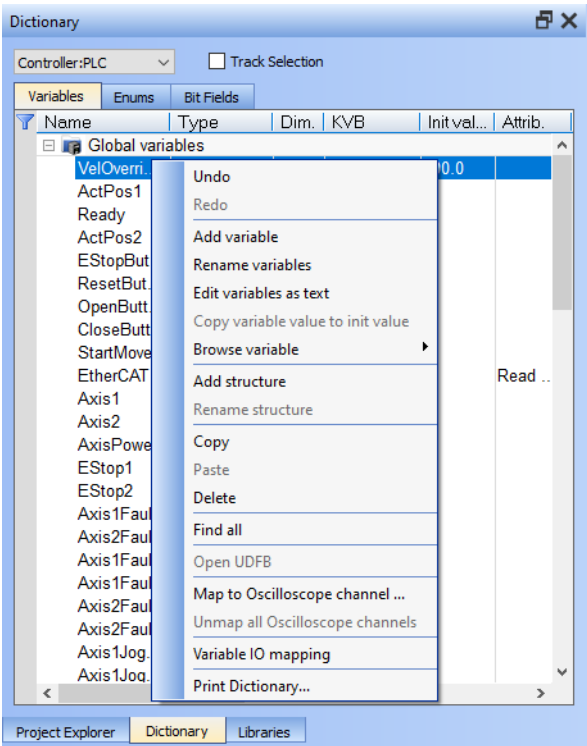


Figure 3-6: Dictionary Contextual Menu

This menu allows you to perform the following actions:

Command	Description
Undo	Undo the last action performed on the Dictionary
Redo	Redo the last undone action
Add Variable	Add a new variable in the selected level (Global, Retain, program). This automatically creates a new variable called NewVar with type BOOL For a Function or UDFB, you can specify input and output parameters (for more details, see "Define Parameters and Private Variables" on page 307)
Rename variables	This function can either replace a section of matching variables or append text to the variables' names. See <a href="#">Rename Variables</a>
Edit variables as text	This function will open a text editing dialog, allowing you to edit the variables found in the group as text using IEC 61131-3 syntax. See <a href="#">Editing Variables as Text</a> for more information.
Copy variable value to init value	This function copies the current value of a retain variable to the variable's initial value. See <a href="#">Copy a Retained Variable Value to Init Value</a> for more information.
Browse variable	This function allows you to browse all instances of a variable. See <a href="#">Browse Variable Tab</a> for more information.
Add Structure	Used to have a new complex type. A structure named NewStructure is created and variables can be dragged into it (for more details, see "Complex Structures" on page 302)
Rename Structure	Rename the selected structure
Copy	Copy a variable
Paste	Paste the copied variable to the selected level

Command	Description																																																																																																																																																																																																												
<b>Delete</b>	Delete the selected variable. A deletion can also be performed by pressing the Delete key on the keyboard																																																																																																																																																																																																												
<b>Find all</b>	This function will find all instances of the specified variable and open the results in the Find and Replace tab.																																																																																																																																																																																																												
<b>Open UDFB</b>	Open the selected UDFB instance (for more details, <a href="#">see "Monitoring UDFBs" on page 231</a> )																																																																																																																																																																																																												
<b>Map to Oscilloscope channel ...</b>	<p>Plug the selected variable on a channel. This command opens a dialog used to configure the variable plug operation.</p> <p>This command is enabled when your application is connected and running, and if the <a href="#">type</a> of variable is eligible for the softscope (i.e. BOOL, INT, SINT, DINT, LINT, UINT, USINT, UDINT, ULINT, BYTE, WORD, DWORD, LWORD, TIME and LREAL, as long as they are not in a UDFB instance).</p>																																																																																																																																																																																																												
<b>Unmap all Oscilloscope Channels</b>	Unplug all plugged probes from the softscope																																																																																																																																																																																																												
<b>Variable I/O mapping</b>	Connect a variable to an I/O.																																																																																																																																																																																																												
<b>Print Dictionary ...</b>	<p>Print all the variables displayed in the Dictionary and sorted by programs. The columns display the Name, Type, Dimension, Initial Value, and Attributes</p> <table border="1"> <thead> <tr> <th>Name</th> <th>Type</th> <th>Dim</th> <th>InitVal</th> <th>R</th> <th>E</th> </tr> </thead> <tbody> <tr> <td colspan="6"><b>(Global)</b></td> </tr> <tr> <td>TravelSpeed</td> <td>LREAL</td> <td>0</td> <td>LREAL#0</td> <td></td> <td></td> </tr> <tr> <td>MasterAbsPos</td> <td>LREAL</td> <td>0</td> <td>LREAL#0</td> <td></td> <td></td> </tr> <tr> <td>MasterDeltaPos</td> <td>LREAL</td> <td>0</td> <td>LREAL#90</td> <td></td> <td></td> </tr> <tr> <td>MachineSpeed</td> <td>LREAL</td> <td>0</td> <td>LREAL#0</td> <td></td> <td></td> </tr> <tr> <td>Axis1Status</td> <td>DINT</td> <td>0</td> <td></td> <td></td> <td></td> </tr> <tr> <td>Axis2Status</td> <td>DINT</td> <td>0</td> <td></td> <td></td> <td></td> </tr> <tr> <td>MachineState</td> <td>DINT</td> <td>0</td> <td>0</td> <td></td> <td></td> </tr> <tr> <td>bMasterAbs</td> <td>BOOL</td> <td>0</td> <td>FALSE</td> <td></td> <td></td> </tr> <tr> <td>bMasterRel</td> <td>BOOL</td> <td>0</td> <td>FALSE</td> <td></td> <td></td> </tr> <tr> <td>bEStop</td> <td>BOOL</td> <td>0</td> <td>FALSE</td> <td></td> <td></td> </tr> <tr> <td>bLedStatus</td> <td>BOOL</td> <td>4</td> <td></td> <td></td> <td></td> </tr> <tr> <td>ProFiles</td> <td>ProFilesCode</td> <td>0</td> <td></td> <td></td> <td></td> </tr> <tr> <td>EtherCAT</td> <td>EtherCATCode</td> <td>0</td> <td></td> <td></td> <td>R</td> </tr> <tr> <td>PipeNetwork</td> <td>PNCCode</td> <td>0</td> <td></td> <td></td> <td></td> </tr> <tr> <td colspan="6"><b>(Retain)</b></td> </tr> <tr> <td>LastAxisPos</td> <td>LREAL</td> <td>0</td> <td></td> <td></td> <td></td> </tr> <tr> <td colspan="6"><b>MachineLogic</b></td> </tr> <tr> <td>lastTravelSpeed</td> <td>LREAL</td> <td>0</td> <td>LREAL#0</td> <td></td> <td></td> </tr> <tr> <td>lastMachineSpeed</td> <td>LREAL</td> <td>0</td> <td>LREAL#0</td> <td></td> <td></td> </tr> <tr> <td colspan="6"><b>ProfilesCode</b></td> </tr> <tr> <td>cmdID</td> <td>DINT</td> <td>0</td> <td></td> <td></td> <td></td> </tr> <tr> <td colspan="6"><b>PNCCode</b></td> </tr> <tr> <td>cmdID</td> <td>DINT</td> <td>0</td> <td></td> <td></td> <td></td> </tr> <tr> <td>MASTER</td> <td>DINT</td> <td>0</td> <td></td> <td></td> <td></td> </tr> <tr> <td>GEAR1</td> <td>DINT</td> <td>0</td> <td></td> <td></td> <td></td> </tr> <tr> <td>CNV1</td> <td>DINT</td> <td>0</td> <td></td> <td></td> <td></td> </tr> <tr> <td>AXIS1</td> <td>DINT</td> <td>0</td> <td></td> <td></td> <td></td> </tr> <tr> <td>GEAR2</td> <td>DINT</td> <td>0</td> <td></td> <td></td> <td></td> </tr> <tr> <td>CNV2</td> <td>DINT</td> <td>0</td> <td></td> <td></td> <td></td> </tr> <tr> <td>AXIS2</td> <td>DINT</td> <td>0</td> <td></td> <td></td> <td></td> </tr> <tr> <td>PipeAXIS1</td> <td>DINT</td> <td>0</td> <td></td> <td></td> <td></td> </tr> <tr> <td>PipeAXIS2</td> <td>DINT</td> <td>0</td> <td></td> <td></td> <td></td> </tr> </tbody> </table>	Name	Type	Dim	InitVal	R	E	<b>(Global)</b>						TravelSpeed	LREAL	0	LREAL#0			MasterAbsPos	LREAL	0	LREAL#0			MasterDeltaPos	LREAL	0	LREAL#90			MachineSpeed	LREAL	0	LREAL#0			Axis1Status	DINT	0				Axis2Status	DINT	0				MachineState	DINT	0	0			bMasterAbs	BOOL	0	FALSE			bMasterRel	BOOL	0	FALSE			bEStop	BOOL	0	FALSE			bLedStatus	BOOL	4				ProFiles	ProFilesCode	0				EtherCAT	EtherCATCode	0			R	PipeNetwork	PNCCode	0				<b>(Retain)</b>						LastAxisPos	LREAL	0				<b>MachineLogic</b>						lastTravelSpeed	LREAL	0	LREAL#0			lastMachineSpeed	LREAL	0	LREAL#0			<b>ProfilesCode</b>						cmdID	DINT	0				<b>PNCCode</b>						cmdID	DINT	0				MASTER	DINT	0				GEAR1	DINT	0				CNV1	DINT	0				AXIS1	DINT	0				GEAR2	DINT	0				CNV2	DINT	0				AXIS2	DINT	0				PipeAXIS1	DINT	0				PipeAXIS2	DINT	0			
Name	Type	Dim	InitVal	R	E																																																																																																																																																																																																								
<b>(Global)</b>																																																																																																																																																																																																													
TravelSpeed	LREAL	0	LREAL#0																																																																																																																																																																																																										
MasterAbsPos	LREAL	0	LREAL#0																																																																																																																																																																																																										
MasterDeltaPos	LREAL	0	LREAL#90																																																																																																																																																																																																										
MachineSpeed	LREAL	0	LREAL#0																																																																																																																																																																																																										
Axis1Status	DINT	0																																																																																																																																																																																																											
Axis2Status	DINT	0																																																																																																																																																																																																											
MachineState	DINT	0	0																																																																																																																																																																																																										
bMasterAbs	BOOL	0	FALSE																																																																																																																																																																																																										
bMasterRel	BOOL	0	FALSE																																																																																																																																																																																																										
bEStop	BOOL	0	FALSE																																																																																																																																																																																																										
bLedStatus	BOOL	4																																																																																																																																																																																																											
ProFiles	ProFilesCode	0																																																																																																																																																																																																											
EtherCAT	EtherCATCode	0			R																																																																																																																																																																																																								
PipeNetwork	PNCCode	0																																																																																																																																																																																																											
<b>(Retain)</b>																																																																																																																																																																																																													
LastAxisPos	LREAL	0																																																																																																																																																																																																											
<b>MachineLogic</b>																																																																																																																																																																																																													
lastTravelSpeed	LREAL	0	LREAL#0																																																																																																																																																																																																										
lastMachineSpeed	LREAL	0	LREAL#0																																																																																																																																																																																																										
<b>ProfilesCode</b>																																																																																																																																																																																																													
cmdID	DINT	0																																																																																																																																																																																																											
<b>PNCCode</b>																																																																																																																																																																																																													
cmdID	DINT	0																																																																																																																																																																																																											
MASTER	DINT	0																																																																																																																																																																																																											
GEAR1	DINT	0																																																																																																																																																																																																											
CNV1	DINT	0																																																																																																																																																																																																											
AXIS1	DINT	0																																																																																																																																																																																																											
GEAR2	DINT	0																																																																																																																																																																																																											
CNV2	DINT	0																																																																																																																																																																																																											
AXIS2	DINT	0																																																																																																																																																																																																											
PipeAXIS1	DINT	0																																																																																																																																																																																																											
PipeAXIS2	DINT	0																																																																																																																																																																																																											

What is the purpose of the Track Selection check box?

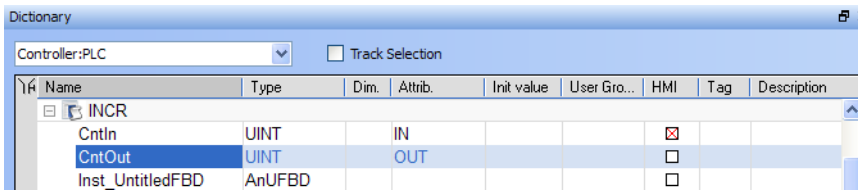
The **Track Selection** filters the displayed items in the dictionary to show only items linked to the current PLC selected program:

- **Unchecked:** All your project variables will be displayed. This is the default setting.
- **Selected:** The variables in the Dictionary are filtered to display only those that are relevant to the PLC item currently selected in the project tree. Along with the Global, retains and variables related to the selected program or UDFB, structure definitions will be displayed. The dictionary content will change accordingly if another PLC program is selected in the project tree.

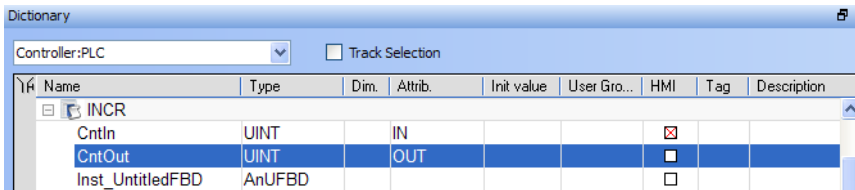
About the two editing modes for editing a variable.

There are two available modes when editing a variable in the Dictionary:

- **Cell:** only the selected cell is active



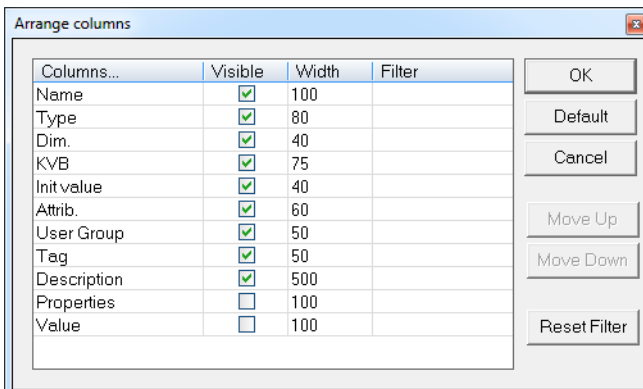
- **Row:** all the row is active



Press **Spacebar** to toggle the selection mode from cell to row ([Table Keyboard Shortcuts](#)).

How can variables be arranged and/or sorted?

- The columns in the Variables tab can be reordered, resized, and hidden by double clicking on the filter icon in top left corner of the table. This opens a dialog box which allows you to modify the table's appearance. These settings will persist until you change them. Please note that the Visible box for the Value field cannot be changed as the Value column is automatically shown in debug and online modes but hidden in edit mode.



- You can sort the list of variables in the table as follows:
- Ensure you are in **cell** editing mode (press the **Spacebar** to toggle from one mode to the other)
- Click the header of the column you want to use as the key sort order

How do I modify parameters of a variable?

(Press **Spacebar** to toggle to the relevant edition mode).

Mode	Description
One Parameter	Assuming you are in the <b>cell</b> edition mode, double-click on the parameter
All the parameters are at the same time	Assuming you are in the <b>row</b> edition mode, double-click in any parameter to open the dialog box for variable configuration as shown below. For more details on parameters, <a href="#">see "Variables" on page 71</a> .

**NOTE**  
It is not possible to modify a variable when the KAS IDE is connected to the controller.

### 5.1.4.1.1 Variables

All **variables** within the entire system project are grouped as follows:

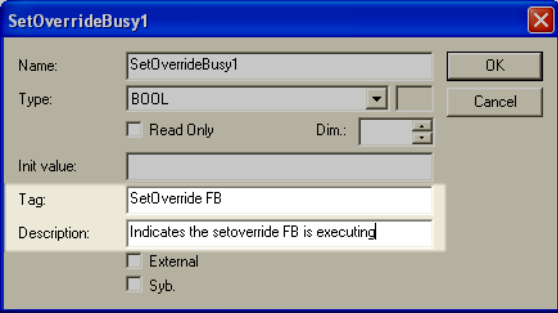
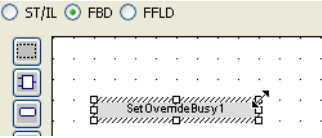
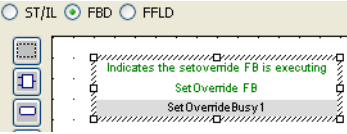
Variable	Description
<b>Global variables</b>	List all global variables that are used and accessible throughout the entire program
<b>Retain Variables</b>	List all variables that are to be retained when the system is powered down
<b>Program variables</b>	List the variables related to your specific selected program

For each variable, the Dictionary toolbox allows you to set the following parameters:

Field	Description
<b>Name</b>	The variable name
<b>Value</b>	All the variables in the Dictionary are animated with real-time values <sup>1</sup>  Note that this column is only displayed when your application is running  For more details, see <a href="#">"Variable Monitoring" on page 231</a>
<b>Type</b>	The variable <b>type</b> (which can also be UDFB or complex structure)
<b>Dim.</b>	To declare an <b>array</b> , you can specify dimension(s) for an internal variable
<b>Attrib.</b>	The variable attributes (Read Only, External, IN, OUT) as defined: <ul style="list-style-type: none"> <li>• Read Only: a variable set as Read Only is a constant (it cannot be modified in your PLC code, but it can be <b>forced</b> manually). Read Only variables can be mapped to Outputs but not to Inputs. This is because Inputs can change state and a Read Only variable would not be able to change its value to match the input state.</li> <li>• External: this attribute is not used</li> <li>• IN or OUT: Input or Output parameters of User Defined Function Blocks</li> </ul>
<b>Init value</b>	The variable initial value when you start your application (see more details <a href="#">here</a> )
<b>User Group</b>	The variable user group (used for sorting variables)
<b>HMI</b>	Select variables to be used in HMI (see <a href="#">Map Variables to HMI</a> )

---

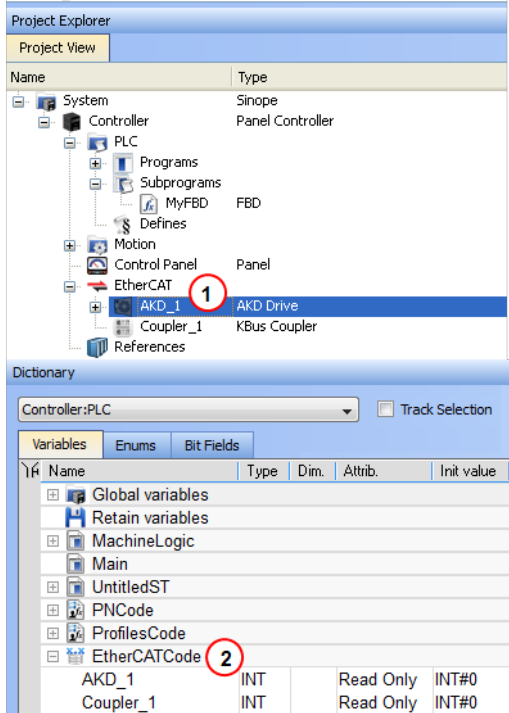
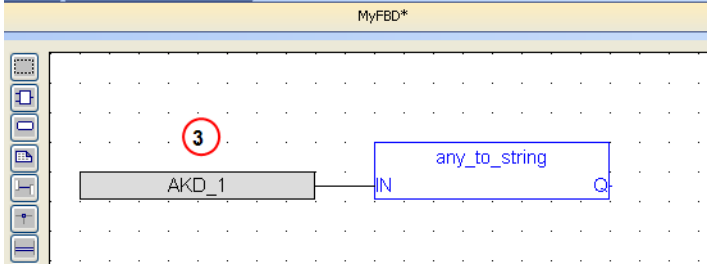
<sup>1</sup>To better track variables in Running mode, the KAS IDE dynamically computes their value along with the application execution and display the result in this column.

Field	Description
<p><b>Tag</b></p> <p>The variable tag is a short comment, that can be displayed together with the variable name in graphical editors.</p> <p>Edit the variable parameters</p>  <p>Add the variable to your FBD program</p>  <p>Resize the rectangle to make the Tag and Description visible</p> 	
<p><b>Description</b></p>	<p>The variable description is a long comment text that describes the variable</p>
<p><b>Syb.</b></p>	<p>reserved</p>



### 5.1.4.1.2 Structures

All the **structures** within the entire system project are grouped as follows:

Structure	Description
<b>EtherCATCode</b>	<p>List all the devices as PLC variables so you can use their names in FBs instance</p> <ul style="list-style-type: none"> <li>• The devices are displayed under the <b>EtherCAT</b> node in the Project Explorer (see call out <b>1</b>)</li> <li>• The same devices are also displayed as PLC variables in the Dictionary <b>2</b></li> </ul>  <p>The screenshot shows the Project Explorer on the left with the 'EtherCAT' node expanded, containing 'AKD_1' (AKD Drive) and 'Coupler_1' (KBus Coupler). The Dictionary on the right shows the 'EtherCATCode' folder containing 'AKD_1' and 'Coupler_1' as INT variables with 'Read Only' attributes.</p> <ul style="list-style-type: none"> <li>• You can use them in your PLC programs with a drag-and-drop operation <b>3</b></li> </ul>  <p>The screenshot shows a ladder logic diagram for 'MyFBD*'. A variable 'AKD_1' is connected to the 'IN' input of an 'any_to_string' function block. A callout '3' points to the 'AKD_1' variable.</p>
<b>PNCode</b>	List all the variables related to the specific Pipe Network Code
<b>ProfilesCode</b>	List all the variables related to the specific profiles
<b>PLCopenCode</b>	List all the variables related to the specific PLCopen Code
<b>UDFB variables</b>	List the variables related to a specific UDFB

### 5.1.4.1.3 Variable editor

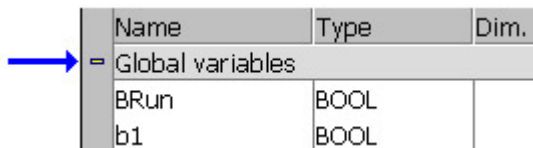
Variables are declared in the Dictionary of the KAS IDE main window.

The variable editor is a table that enables you to declare all variables of the application. Variables in the editor are sorted by groups:

- global variables
- "retain" non-volatile global variables
- I/O variables (each I/O device is a group)
- variables local to a program (including in and out parameters in case of a UDFB).

Please refer to the description of [variables](#) in the language reference for a more detailed overview.

Each group is marked with a gray header in the variable list. The "-" or "+" icon on the left of the group header can be used to expand or collapse the group:



Name	Type	Dim.
- Global variables		
BRun	BOOL	
b1	BOOL	

See how to:

- [Create New Variables](#)
- [Use the Variable Table List](#)
- [Define Structures](#)
- [Set Bookmarks](#)

#### 5.1.4.1.3.1 Create new variables

Press the INSERT key in the variable editor to create a new variable in the selected group. The variable is added at the end of the group. Variables are created with a default name. You can rename a new variable or change its attribute by using the [Variable Editor](#).

#### TIP

You cannot insert a new variable in an I/O group.

In case of a group corresponding to local variables of a UDFB, pressing the INSERT key gives you the choice between:

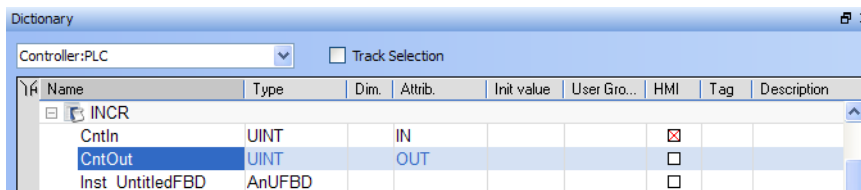
- adding an "IN" (input) parameter
- adding an "OUT" (output) parameter
- adding a private variable

IN and OUT parameters always appear at the beginning of a UDFB group.

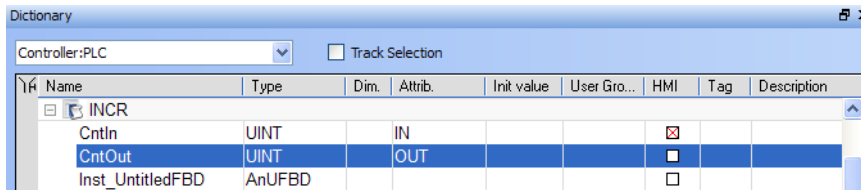
#### 5.1.4.1.4 Variable Table List

There are two available modes when editing a variable in the Dictionary:

- **Cell:** only the selected cell is active



- **Row:** all the row is active



Press **Spacebar** to toggle the selection mode from cell to row ([Table Keyboard Shortcuts](#)).

When the selection mode is on cell, the variable editor enables you to enter each piece of information directly in the cell.

Double-click or press the ENTER key to open the dialog box.

At any time you can drag with the mouse the column separators in the main grid header for resizing columns.

Press the following keys for browsing groups of variables:

Ctrl + Page Up	Move the selection to the head of the previous group
Ctrl + Page Down	Move the selection to the head of the following group

For Tables manipulation, see also [Windows Standard Conventions](#)

#### 5.1.4.1.5 Sort variables

At any moment you can sort variables of a group according to their name, type or dimension. To do this, you simply need to:

1. Move the cursor to the header of the group
2. Click on the name of the column you want to sort

The KAS IDE always keeps the original order of declared variables, to allow safe online change. Each time you insert a new variable or expand/collapse a group, the original sorting is re-applied.

#### 5.1.4.1.6 Define structures

To create a new type of data structure, use the "Add structure" command.

For more details of the full procedure, refer to [Complex Structures](#)

Each structure is represented as a group in the dictionary grid. Enter the members of the structure in its group in the same way you enter variables in another group.

New data structures are created with default names. Use the "Rename structure" command to change its name.

If a member of a structure is an instance of another structure, the nested structure must be declared BEFORE in the list.

#### 5.1.4.1.7 Name a variable

To change the name of the variable, do as follows:

1. In the Name column of the [table](#), select the cell you want to edit
2. Press ENTER (or press the first character of the new name)
3. Enter the name in the small box
4. Press ENTER to validate the name or ESCAPE to cancel the change

A variable must be identified by a unique name within its parent group. The variable name cannot be a reserved keyword of the programming languages and cannot have the same name as a standard or "C" function or function block. A variable must not have the same name as a program or a user-defined Function Block.

The name of a variable must begin by a letter or an underscore ("\_") mark, followed by letters, digits or underscore marks. It is not allowed to put two consecutive underscores within a variable name. Naming is case-insensitive. Two names with different cases are considered as the same.

##### 5.1.4.1.7.1 Naming Physical I/Os

Each I/O channel has a predefined symbol that reflects its physical location. This symbol begins with "%I" for an input and "%Q" for an output, followed by a letter identifying the physical size of the data. Refer to the description of [variables](#) for more details.

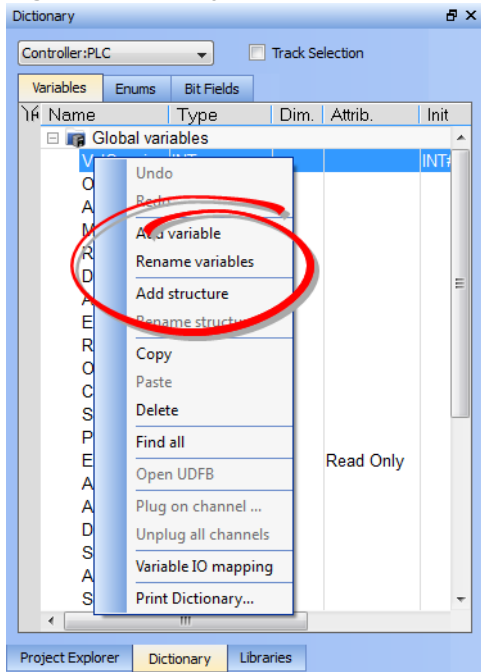
You cannot change the "%..." name of an I/O variable. This name is directly allocated according to the I/O devices defined in the I/O device list. But you can give an alias (a readable name) to each I/O channel. In that case, either the "%" name or the alias can be used in programs. The alias must fit to the same rules as a variable name.

When an alias is defined for a variable, both "%..." name and alias are displayed in the "name" column of the grid.

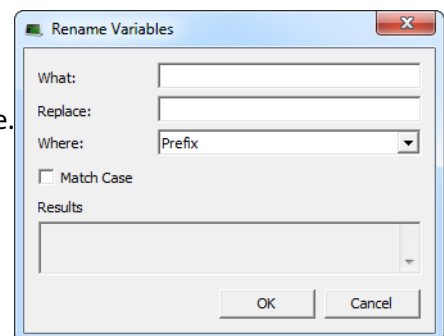
### 5.1.4.1.8 Rename Variables

Variables may be renamed from within the Dictionary. The renaming function can either replace a section of the name or append text to variable names. The search will find and replace matches first within the Dictionary and will then continue the search within programs in the current project. You will have the option to propagate the changes to programs or not.

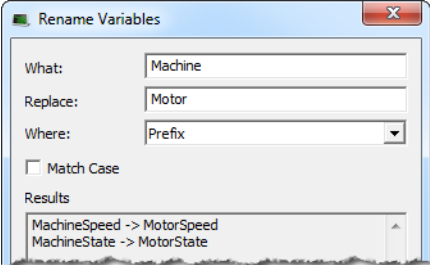
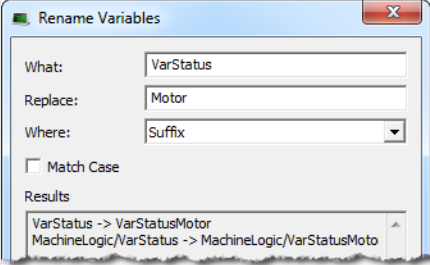
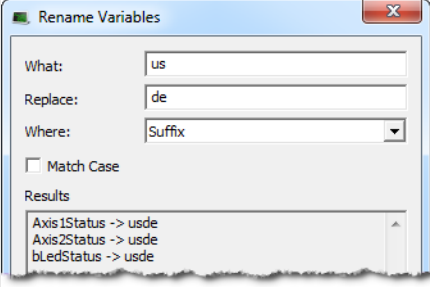
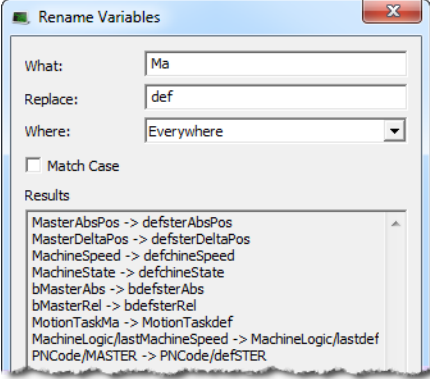
1. Right-click on any variable in the Dictionary and select **Rename Variables**.



2. Enter the phrase to be replaced in **What**.
3. Enter the new phrase in **Replace**.
4. Set the replacement method using the **Where** menu.
5. Click OK to make the changes shown in the **Results** frame.
6. The **Replace in files** dialog box opens if there are matches within programs. Select which programs to modify.



Item	What it means
<b>What</b>	Enter the variable name/text which is to be replaced
<b>Replace</b>	New text which will replace current variable name text

Item	What it means
Where	<p>How the replace will occurs</p> <div data-bbox="384 277 1367 622"> <p><b>Prefix</b> The search/replace starts from the first character in the variable name.</p>  </div> <div data-bbox="384 629 1367 974"> <p><b>Suffix</b> The "Replace" text will be appended to the "What" text, completely replacing any matches.</p>  <p>Please be aware that this search method completely replaces the search string. Use this method with caution.</p>  </div> <div data-bbox="384 1348 1367 1778"> <p><b>Everywhere</b> Searches the entire variable name.</p>  </div>
<b>Match Case</b>	The search will be case sensitive
<b>Results</b>	The search results and proposed changes are displayed.

When matches exist within programs the **Replace in Files** dialog box displays the matches and lets you choose which programs to modify. Once **OK** is clicked, the programs will be modified in the

order they appear in the list.



#### 5.1.4.1.9 Initial Value of a Variable

A variable can have an initial value. The value must be a valid [constant expression](#) that fits to the [data type](#) of the variable. The initial value is displayed in **red** if it is not a valid expression for the selected data type.

There is no initial value for arrays and instances of function blocks.

You can change the initial value of a variable by using the [Variable Editor](#).

#### NOTE

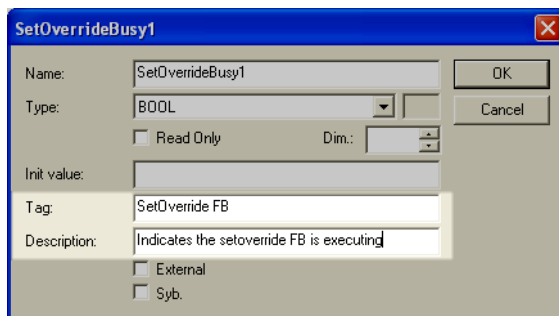
A retain variable's value may be assigned to the variable's initial value. See [Copy a Retained Variable Value to Init Value](#) for more information.

#### 5.1.4.1.10 Variable Tag and Description

For each variable, the KAS IDE enables you to enter in the dictionary two strings that describe the variable:

- The "**Tag**" is a short comment, that can be displayed together with the variable name in graphic languages.
- The "**Description**" is a long comment text that describes the variable.

To change the tag or description of a variable, enable the modification mode to [Row](#) and move the cursor to the corresponding cell. Then press ENTER to enter the new text.



#### 5.1.4.1.11 Editing variables as text using IEC 61131-3 syntax

Using IEC61131-3 syntax, variables are declared within structured blocks. Each blocks begins with "VAR", "VAR\_INPUT", "VAR\_OUTPUT" or "VAR\_EXTERNAL" keyword and ending with "END\_VAR" keyword (with no semicolon after). Below is the meaning of each keyword:

Keyword	Meaning
VAR	Memory variables. Can be global, local or <a href="#">retain</a> depending on the edited group
VAR_INPUT	Input parameters of a block. Available only when the edited group is a UDFB.
VAR_OUTPUT	Output parameters of a block. Available only when the edited group is a UDFB.
VAR_EXTERNAL	External variables. Can be global or local depending on the edited group

##### 5.1.4.1.11.1 Basic syntax for declaring a variable:

To declare a variable, simply enter its symbol, followed by ":" and its data type. If the data type is STRING, it must be followed the maximum length between parentheses. Example:

```
MyVar : BOOL;
MyString : STRING(255);
```

To indicate that a variable has the "read only" attribute, insert the "CONSTANT " keyword at the beginning of the variable declaration:

```
CONSTANT VarName : DataType;
```

To declare an array, the data type must be preceeded by "ARRAY [ dimensions ] OF". There are at most 3 dimensions, separated by commas. Each dimension is specified as "0 .. MaxBound". Below are examples:

```
Array1 : ARRAY [0 .. 99] OF DINT;
Matrix : ARRAY [0 .. 9, 0 .. 9, 0 .. 9] OF REAL;
```

Additionally, you can specify an initial value for single variables. The initial value is entered after the data type, and is preceded by ":= ". The initial value must be a valid constant expression that fits the data type. Examples:

```
MyBool : BOOL := TRUE;
MyString : STRING(80) := 'Hello';
MyLongReal : LREAL := lreal#1.0E300;
```



#### 5.1.4.1.11.2 Additional information and description texts:

As a variable may have additional properties and comment texts in the KAS IDE, we use special directives entered as IEC comments AFTER the declaration of the variable, to specify additional info. The following directives are available:

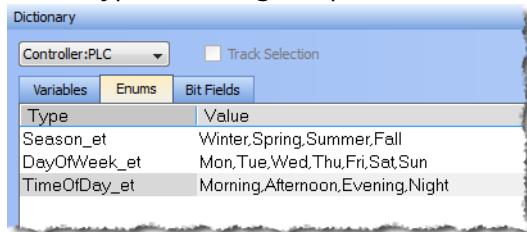
Directive	Description
<code>(*\$tag=Text*)</code>	Variable tag (short comment)
<code>(*\$desc=Text*)</code>	Variable description
<code>(*\$profile=ProfileName*)</code>	Variable embedded profile
<code>(*\$embed=Text*)</code>	Variable embedded properties (the syntax is the one shown in the variable grid, in the "Property" column)

You can also use `"/"` single line comments to enter the directives:

```
// $tag=Text
// $desc=Text
// $profile=ProfileName
// $embed=Text
```

#### 5.1.4.2 Enum Tab

This tab allows you to define enums (Enumerated Types). An enumerated type allows you to define a data type and assign a specific set of accepted values.



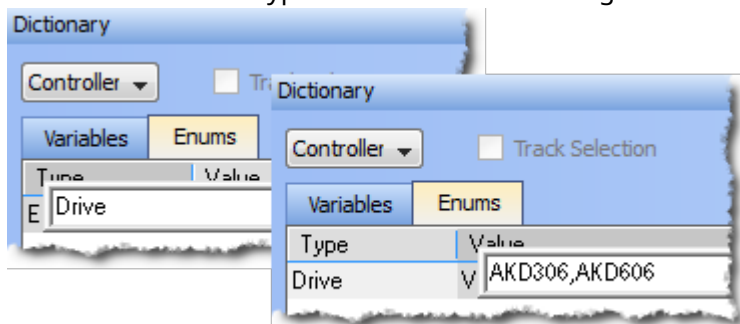
##### 5.1.4.2.1 Adding Enums

An enum may be created in one of two ways:

- Right-click within the Enum tab and select **Insert Enum**.
- Press the **Insert** key while the Enum tab is active.

This creates a default enum labelled as "EnumType*n*" with the Value of "V0, V1".

Double click on the Type or Value to make changes.

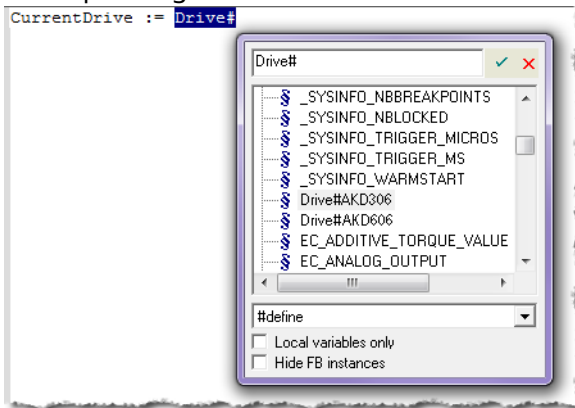


**NOTE**

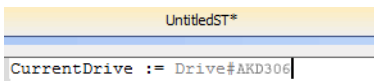
- An enum must contain two or more values. If only one value is defined, it will be not be considered and the previous value will be retained.
- An enum should not contain special characters such as #, @, etc.. If a special character is defined in the value field, it will be not be considered and the previous value will be retained.

**5.1.4.2.2 To Use Enums**

1. Enter the enum type and hash (#), the press Ctrl+Space. This opens a dialog to select the corresponding value.



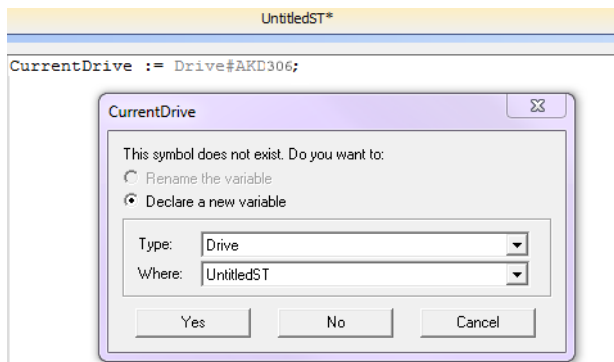
2. Select the value and click the check mark. The value is loaded into your program.



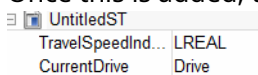
**5.1.4.2.3 Declaring Enums**

Enums may be declared in the same way as variables; at the end of the line press Enter.

As CurrentDrive is not a part of the dictionary, a dialog is opened to add the variable. Set the Types as the enum name.



Once this is added, the variable will be displayed with the selected type in the Dictionary.



### 5.1.4.3 Bit Fields Tab

A bit field packs multiple pieces of data together in one variable. Each field represents one piece of data. Each piece of data should have no dependency upon other fields. Bit Fields are used to define custom variable types and values.

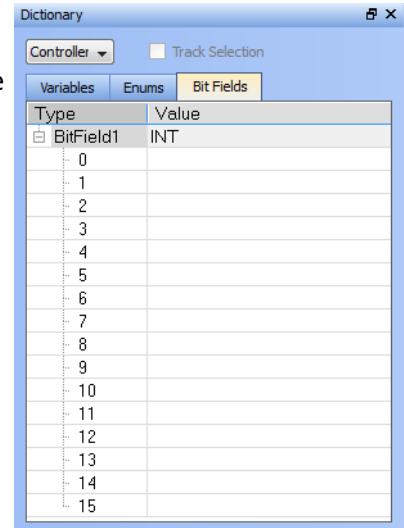
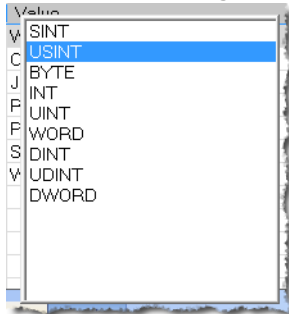
#### 5.1.4.3.1 Adding Bit Fields

A Bit Field may be created in one of three ways:

- Right-click within the Bit Fields tab and select **Insert BitFields**.
- Press the **Insert** key while the Bit Fields tab is active
- Double-click within the Bit Fields tab.

This creates a default Bit Field labelled as "BitFieldn" with a Value type "INT".

- The Bit Field type **Value** may be modified by double-clicking on value, allowing a selection from a list.



Once the Type is defined you may populate the bits.

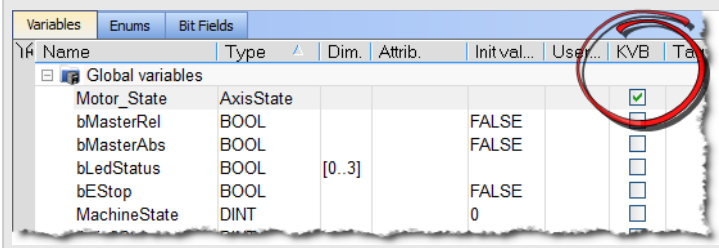
- Double click on a bit value to change the text.

**IMPORTANT**

To save your changes you must press **Enter**, **Tab**, the Up, or the Down arrow key.

**NOTE**

Bit Fields are supported over MODBUS. Below is a variable with a custom Type defined in the Bit Fields list which is flagged for KVB. It will be exported to MODBUS.

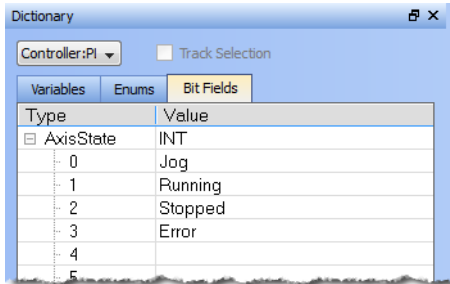


### 5.1.4.3.2 Using Bit Fields

Following is an example of setting up and testing a Bit Fields entry.

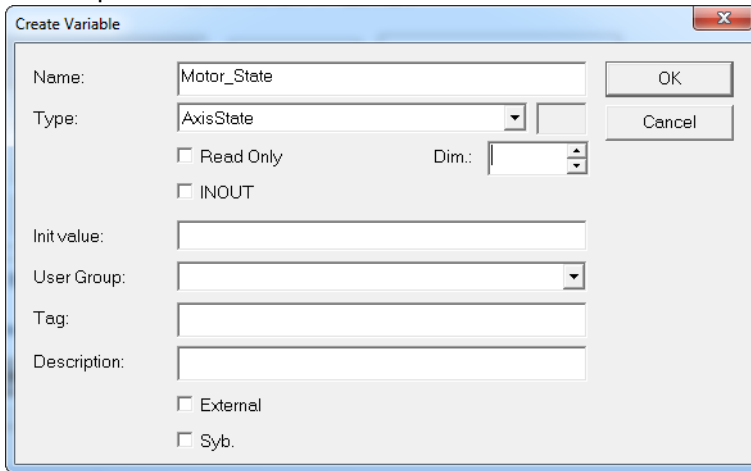
1. Define the Bit Field Type and Values.

Here we have defined a bit called **AxisState** with four values.



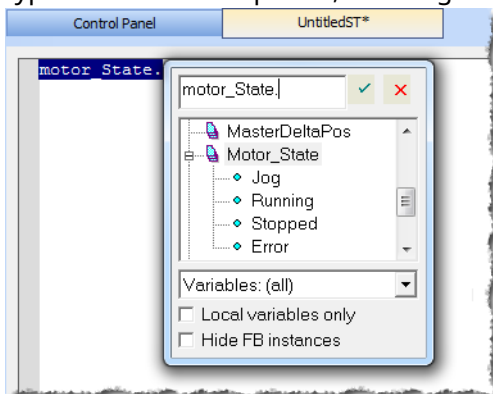
2. Create a variable whose Type that matches the Bit Field.

Here we have defined a variable called **Motor\_State**. We selected the **AxisState** type from the dropdown menu.



3. To use the bit enter `VariableName.BitName` in any editor.

Here we are using the Structured Text editor and typed "motor\_state." When the dot was typed the selector opened, showing us the available bits for the variable.



## 5.1.5 Information and Logs

The Information and Log window is used to identify current state status and can be used to identify operational errors, compilation errors, and also to quickly assist you in finding areas of the workspace or program variables.

This window contains different tabs that provide:

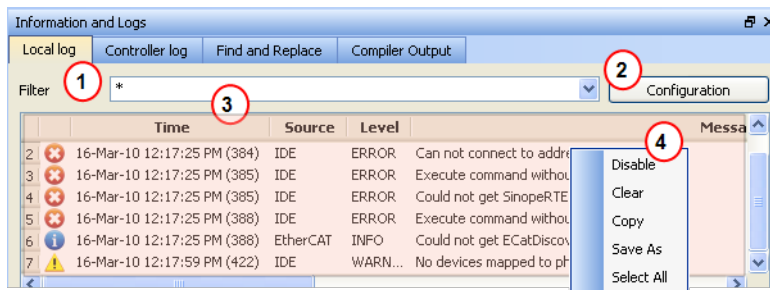
- Log messages (Local or Controller) including [Log Messages Settings](#)
- A system search function
- A list of breakpoints
- A state report on the program compiler

### 5.1.5.1 Log Messages

#### TIP

Log messages are an important source of information when you are troubleshooting with KAS IDE.

When reporting an issue to Support, copy/paste the logs in your report.



**Figure 3-7:** Log Messages

Log messages are displayed on two tabs, **Local log** and **Controller log**.

- The Local log tab shows all messages managed by the KAS IDE to explain the current state of the system and to help identify any operation errors encountered when developing your system.
- The Controller log shows all messages managed by the KAS Runtime.

Based on the configuration settings (see call out (2)), only messages that are recorded and that match the filter (1) are displayed.

#### TIP


The Configuration button is only available on the Controller log tab when the IDE is connected to a controller.

Every log message in the table widget (3) has the following information:

Field	Description
<b>Time</b>	Time when the log was recorded with the format: DD-MMMM-YY hh:mm:ss (millisecond)
<b>Source</b>	Identifies a software or hardware component issuing the messages. Each source is <b>configured</b> with a specific Level.

Field	Description
<b>Level</b>	Each message has one of the following levels with importance in ascending order: DEBUG > INFO > WARNING > ERROR > CRITICAL
<b>Message</b>	Text of the message issued from the source

Table 3-26: Log Messages - List of Fields

The table contains a contextual menu (see call out ) with the following commands:

Command	Description
<b>Disable/Enable</b>	You can stop the log recording at any time, so that no more messages are added
<b>Clear</b>	Empty the list by erasing all the messages already recorded
<b>Copy</b>	Copy the text of the selected messages to the clipboard (you can perform multi-selection with the Ctrl or Shift keys)
<b>Save As</b>	Save all the messages in a log file
<b>Select All</b>	Select all the messages that are displayed in the table

Table 3-27: Log Messages - List of Buttons

### 5.1.5.2 Log Messages Settings

The KAS IDE manages all messages according to the two following gates:

- [Configuration settings](#) define what is recorded in the database
- [Filtering](#) defines which messages are displayed in the table widget

#### 5.1.5.2.1 Configuration Settings

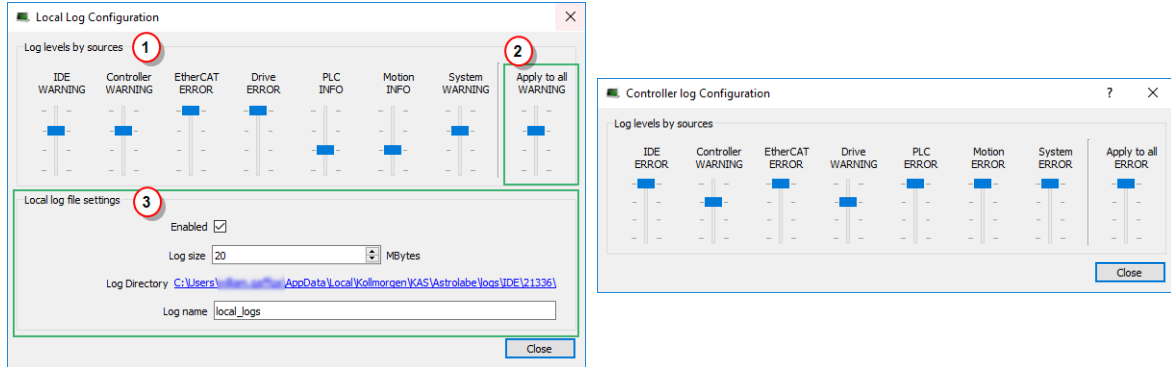







Figure 3-8: Configuration of the Local and Controller log messages

Call out#	Description
1	Each log source can be set with its own notification level. It is possible to get a maximum of log details for the selected source without getting a flood of irrelevant messages from the other sources by controlling the level of each source. See <a href="#">What are the sources of the log messages?</a> , <a href="#">What are the log message levels?</a> , and <a href="#">How do I choose the appropriate level for a log?</a>
2	You can set or reset all the sliders with the same level value
3	Logs can be recorded on the local machine as circular files. Note that on the controller, the recording of the logs is enabled by default. For more details, see <a href="#">About Local Log File Settings</a> .

What are the **sources** of the log messages?

Source	Apply to...
IDE	Win32 applications: the KAS IDE and the KAS Runtime Server (also called the KAS Runtime Front-end)
Controller	For the KAS Runtime items: Drivers, IOEngine, SinopEngine...
EtherCAT	For all kinds of EtherCAT items: Motion bus, I/Os
Drive	Messages from AKD drive
PLC	For application engineers to create custom log within the PLC programs (similar to <a href="#">printf</a> )
Motion	Messages coming from the Motion engines: PLCopen, Pipe network or VM
System	For common API and libraries. Also includes messages issued from the operating system.

What are the log message **levels**?

Level	Icon	Description
CRITICAL		Application crashes or becomes unstable. Data is corrupted. At that point, the application behavior can be unpredictable.
ERROR		The application does not behave as expected but the processes remain stable.
WARNING		System is stable but the KAS IDE warns that an unexpected event can occur. This is the default logging level. You can ignore this log.
INFO		Information status of the current process. You can ignore this log.
DEBUG		Any information logged for development purpose. You can ignore this log.

Each message has one of the following levels, with importance in ascending order:  
DEBUG > INFO > WARNING > ERROR > CRITICAL

*About Local Log File Settings*

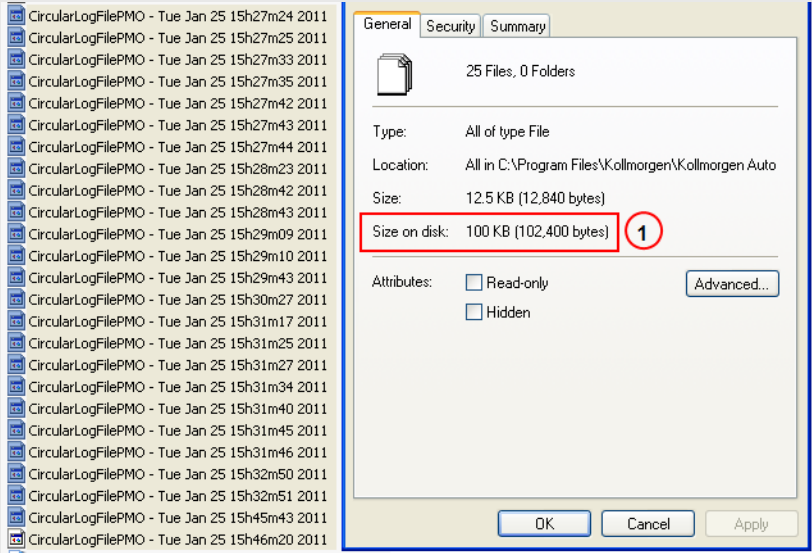
Log files are a group of small files where all the last logs are recorded. Each log is recorded as a separated line.

**TIP**

You can import the log files into Microsoft Excel using drag-and-drop.

Field	Description
Enabled	The Log File Settings has to be enabled to record all the logs.



Field	Description
Log size	<p>To prevent any overflow, you can define the maximum size on disk dedicated to the group of local log files. The minimum size is 1MB, the maximum log size is 2MB. At most, the system will store 20 files. When a new file is created, an old one is deleted, maintaining the 20 file threshold.</p> <p>In the example below, the size on disk is limited to 100 KBytes (see call out <b>1</b>).</p>  <p>The screenshot shows a file explorer window with a list of files named 'CircularLogFilePMO' followed by a timestamp. To the right, a 'Properties' dialog box is open, showing the 'General' tab. The 'Size on disk' field is highlighted with a red box and a circled '1'. The dialog also shows 'Type: All of type File', 'Location: All in C:\Program Files\Kollmorgen\Kollmorgen Auto', and 'Attributes: Read-only' and 'Hidden' checkboxes.</p>
Log Directory	<p>Note that the number of rows can vary for each file, depending on what is in the backlog when KAS creates the log files.</p> <p>The Log Files are stored in this directory. Each KAS IDE instance has a unique log directory, defined by the process identification number. You can access the local log files for this specific instance of the KAS IDE, by clicking this link to open the directory location in the file explorer. For more information see <a href="#">Where are the log files stored?</a></p>
Log name	<p>You can define the filename prefix to be used on the local machine. (on the controller, the filename prefix is: controller logs).</p> <p>The suffix to create the complete filename contains a timestamp with the following format: - day MMM DD HHMMmSS YYYY nn</p> <p><b>NOTE</b></p> <p>The Log name change is applied immediately. The name used by the next IDE instance is determined by the last IDE instance.</p>

*How do I choose the appropriate level for a log?*

When a level is set for a source, only messages with the same or higher importance are recorded. In other words, drag the level control slider **Up to reduce** the verbosity, **Down to increase** it.

When the configuration leads to lower verbosity, the treatment during the filtering is quicker.

For example, if a source is set to WARNING, then all messages with levels WARNING, ERROR and CRITICAL are recorded (DEBUG and INFO messages are discarded).

In other words, DEBUG is the most verbose, whereas ERROR is the less verbose.

**NOTE**

Critical messages are always recorded (as a consequence, the Critical level is not visible on the slider).

Where are the log files stored?

- For the local machine (IDE), the Log files are located in the following location:  
`C:\Users\(user name)\AppData\Local\Kollmorgen\KAS\Astrolabe\logs\IDE\(process ID)`
- For the controller, the Log files are located under:  
`C:\Users\(user name)\AppData\Local\Kollmorgen\KAS\Sinope Simulator\Application\logs`
- The AKD PDMM or PCMM logs are accessed via the web server page by browsing to:  
`KAS Application > Log Data.`

**TIP**

AKD PDMM/PCMM generated logs may be configured through the webpage. See [PDMM and PCMM Log Files](#) for more information on those controllers' logs.

The screenshot shows the 'Log Configuration' page in the KAS Application web interface. The page has a navigation bar with tabs for 'KAS Application', 'Settings', 'Diagnostics', and 'Help'. The main content area is divided into two sections: 'KAS Application' and 'Log Configuration'. The 'KAS Application' section displays the Kollmorgen logo and 'HIGH PERFORMANCE MOTION & PLC ENGINE' text. It also shows the version of the KAS App as '\_PN\_4axis\_HC::296' and the status as '\_stopped'. There are buttons for 'Start', 'Cold Start', and 'Stop', and an 'Auto-start' checkbox with an 'Apply' button. The 'Log Configuration' section features a table with columns for 'IDE', 'Controller', 'EtherCAT', 'Drive', 'PLC', 'Motion', 'System', and 'All'. The rows represent log levels: 'Debug', 'Info', 'Warning', and 'Error'. Each cell in the table contains a radio button. An 'Apply' button is located at the bottom left of the table.

	IDE	Controller	EtherCAT	Drive	PLC	Motion	System	All
Debug	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Info	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Warning	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Error	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

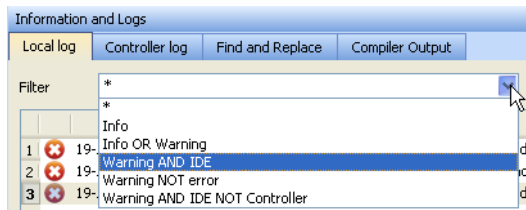
**NOTE**

It is recommended that you use either the IDE or web page method, but not both. The communication is unidirectional and the configuration is not read at runtime.

### 5.1.5.2.2 Filtering

You can narrow the list of recorded messages by specifying a filter. The filter is applied on all the strings displayed on each row of the table widget (i.e Time, Source, Level and Message).

The drop-down menu gives access to some predefined filters, which can also be edited.



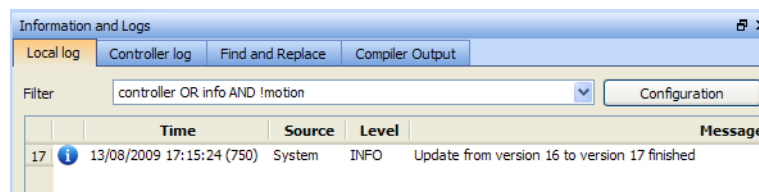
**Figure 3-9:** Filtering the Messages

For example, filtering with **Warning NOT error** means that only the lines including the word "warning" but not the word "error" are listed.

#### Filtering Rules

The following rules apply when you work with filters:

- You can combine several strings by including one of the three following Boolean operands:
- OR
- AND
- NOT (or use the exclamation mark "!")
- Several keywords separated with spaces are considered as an exact string
- Filtering is **not** case sensitive



**Figure 3-10:** Filtering the Messages - Example

#### NOTE

**Warning!** When you apply the filter, all the currently recorded messages are parsed and displayed if they match the filter. But all the upcoming recorded messages are added as new rows at the end of the table widget with **no filtering**.

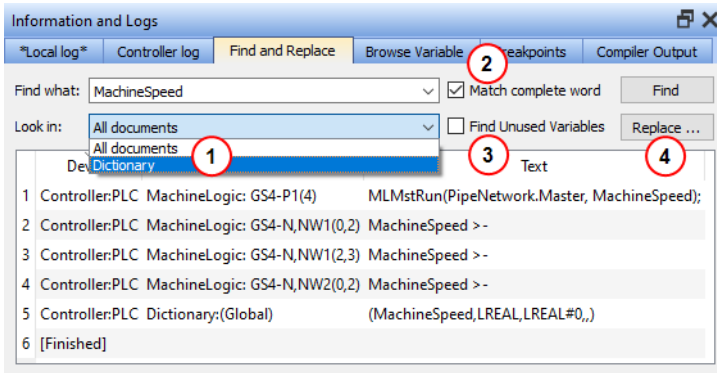
#### About Scrolling

If you select a message in the table, the scrolling is **stopped**.

All the upcoming recorded messages are added at the end of the list, but your selected message always remains in the same place (you have to scroll down to make the most recent messages visible).

If you select the last row of the table (shortcut: **Alt+Page Down**), the scrolling is **active**. The last recorded message is always selected and visible at the bottom of the table.

### 5.1.5.3 Find and Replace Tab

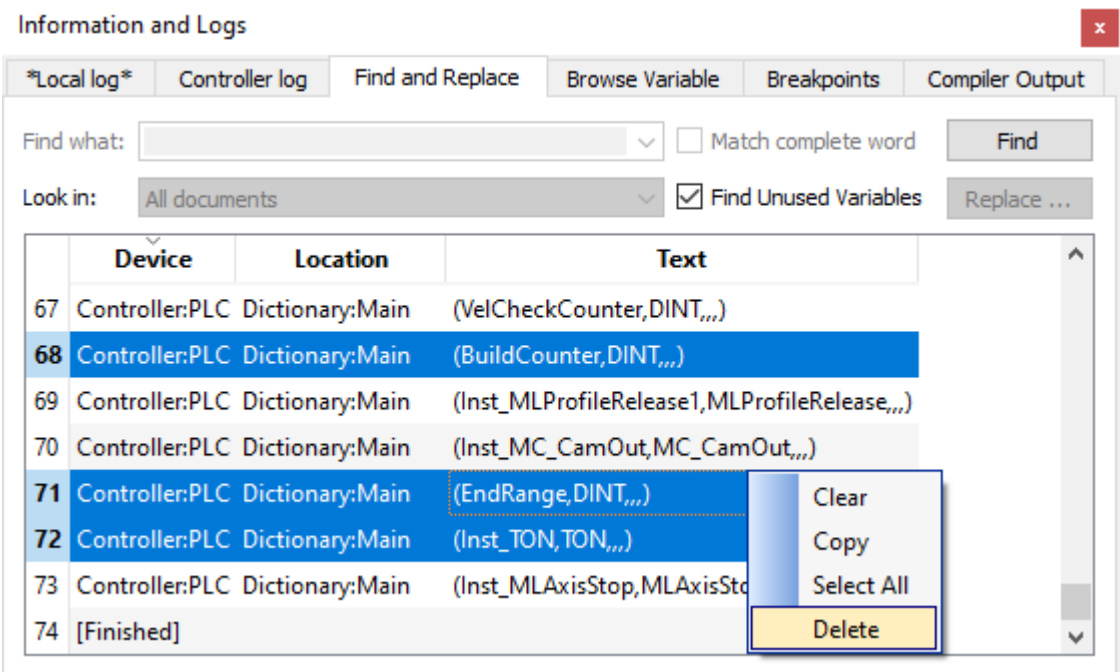


**Figure 3-11:** Find and Replace

This tab enables you to search for all the instances of a string of characters (search is **not** case sensitive) within the entire environment, and replace it if desired.

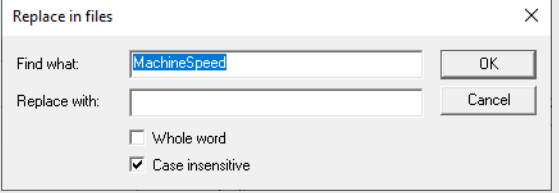
You can re-use one of the last ten entries or type a new text string.

Call out#	Description
1	The operation can be performed across: <ul style="list-style-type: none"> <li>• Dictionary: search the dictionary</li> <li>• All documents: search all files of the project and the dictionary</li> </ul>
2	The search string can be with ' <b>complete word</b> ' only. When selected, only the instances that match the complete words defined in the "Find what" field are selected (for example, a search for "MyVar" returns "MyVar" but not "MyVariable"). When this option is selected, the search is case sensitive.

Call out#	Description																																				
3	 <p>The screenshot shows the 'Information and Logs' window with the 'Find and Replace' tab active. The search criteria are: 'Find what:' (empty), 'Match complete word' (unchecked), 'Look in:' (All documents), and 'Find Unused Variables' (checked). The results table is as follows:</p> <table border="1"> <thead> <tr> <th></th> <th>Device</th> <th>Location</th> <th>Text</th> </tr> </thead> <tbody> <tr> <td>67</td> <td>Controller:PLC</td> <td>Dictionary:Main</td> <td>(VelCheckCounter,DINT,,)</td> </tr> <tr> <td>68</td> <td>Controller:PLC</td> <td>Dictionary:Main</td> <td>(BuildCounter,DINT,,)</td> </tr> <tr> <td>69</td> <td>Controller:PLC</td> <td>Dictionary:Main</td> <td>(Inst_MLProfileRelease1,MLProfileRelease,,)</td> </tr> <tr> <td>70</td> <td>Controller:PLC</td> <td>Dictionary:Main</td> <td>(Inst_MC_CamOut,MC_CamOut,,)</td> </tr> <tr> <td>71</td> <td>Controller:PLC</td> <td>Dictionary:Main</td> <td>(EndRange,DINT,,)</td> </tr> <tr> <td>72</td> <td>Controller:PLC</td> <td>Dictionary:Main</td> <td>(Inst_TON,TON,,)</td> </tr> <tr> <td>73</td> <td>Controller:PLC</td> <td>Dictionary:Main</td> <td>(Inst_MLAxisStop,MLAxisSto</td> </tr> <tr> <td>74</td> <td>[Finished]</td> <td></td> <td></td> </tr> </tbody> </table>		Device	Location	Text	67	Controller:PLC	Dictionary:Main	(VelCheckCounter,DINT,,)	68	Controller:PLC	Dictionary:Main	(BuildCounter,DINT,,)	69	Controller:PLC	Dictionary:Main	(Inst_MLProfileRelease1,MLProfileRelease,,)	70	Controller:PLC	Dictionary:Main	(Inst_MC_CamOut,MC_CamOut,,)	71	Controller:PLC	Dictionary:Main	(EndRange,DINT,,)	72	Controller:PLC	Dictionary:Main	(Inst_TON,TON,,)	73	Controller:PLC	Dictionary:Main	(Inst_MLAxisStop,MLAxisSto	74	[Finished]		
	Device	Location	Text																																		
67	Controller:PLC	Dictionary:Main	(VelCheckCounter,DINT,,)																																		
68	Controller:PLC	Dictionary:Main	(BuildCounter,DINT,,)																																		
69	Controller:PLC	Dictionary:Main	(Inst_MLProfileRelease1,MLProfileRelease,,)																																		
70	Controller:PLC	Dictionary:Main	(Inst_MC_CamOut,MC_CamOut,,)																																		
71	Controller:PLC	Dictionary:Main	(EndRange,DINT,,)																																		
72	Controller:PLC	Dictionary:Main	(Inst_TON,TON,,)																																		
73	Controller:PLC	Dictionary:Main	(Inst_MLAxisStop,MLAxisSto																																		
74	[Finished]																																				

T

o optimize your project, the **Find Unused Variables** option allows you to perform a search in the dictionary to locate variables not used <sup>1</sup> in any program. Selecting this option and clicking **Find** generates a list of the unused variables. You can select one or more variables (by Shift- or Ctrl-clicking) and right-click to open the context menu. The menu includes a **Delete** option. Double-click a variable in the list within the table widget to open it in the Dictionary.

4	<p>With the <b>Replace...</b> button, you can replace in all documents a string with another one (or re-use one of the last ten entries). The search may be refined by forcing the search to a <b>Whole word</b>, and may be case sensitive or <b>Case Insensitive</b>.</p> <p>The replace function is limited: it is not supported in the Dictionary, Pipe Network and HMI. For those, you have to make the change manually.</p>	
---	---	--

Once the search is done, the results appear in the table widget at the bottom of this tab. If a replace has been performed, the Text column provides more information about the replacement.

Double-click the item you want in the list in order to open it in its relevant location (it can be a PLC editor or the Dictionary).

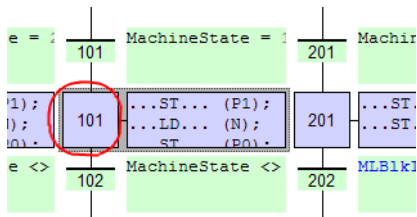
*How to Understand the Location Details – SFC Programs*

SFC Location details

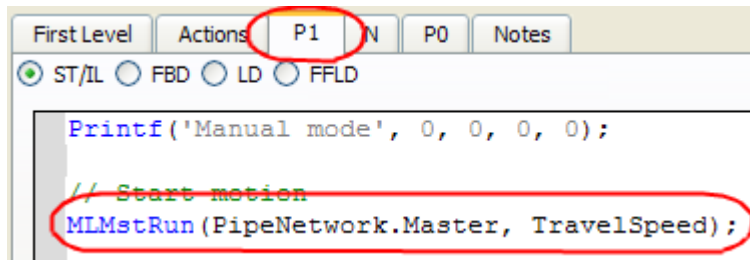
2 controller:PLC MachineLogic: GS101-P1(4) MLMstRun(PipeNetwork.Master, TravelSpeed);

<sup>1</sup>A variable is **not used** when there is no effective usage of it in your entire project. It can still be the case even when a value is assigned to a variable (e.g. MyVar := 100. \* Axis1.Velocity ;). The variable MyVar becomes **used** when it is affected as an input argument (e.g. Velocity := MyVar ;).

- Controller : PLC and MachineLogic refer to the program in the Project Explorer
- GS stands for Graphical and Step (T is for Transition)
- 101 is the reference in the editor



- -P1(4) refers to the P1 tab and the 4th line in the source code



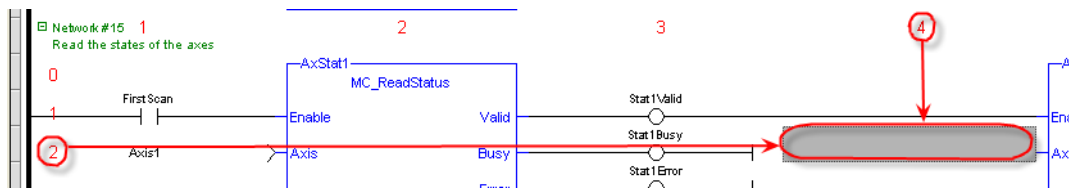
*How to Understand the Location Details – FFLD Programs*

**FFLD Location details**

```

Controller:PLC:>> Complex variables stored in a separate segment
Controller:PLC>Loading application symbols...
Controller:PLC:Main
Controller:PLC:Main: G55-N,NW15(4,2): Object not connected on the left
Controller:PLC:Main: G55-N,NW15(4,0): Bad value on box input(s)
Controller:PLC:Error(s) detected
Controller: ----- PLC Failed -----
    
```

- Controller : PLC and Main refer to the program in the Project Explorer
- GS stands for Graphical and Step (T is for Transition)
- 5 is the reference in the editor
- -N refers to the N tab
- NW15 stands for Network number 15
- (4,2) correspond to the X,Y coordinates of the cell relative to the current network



*How to Understand the Location Details – ST Programs*

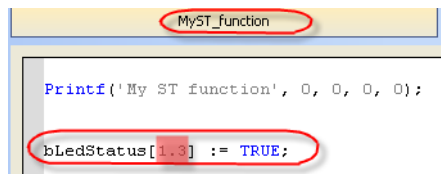
**ST Location details**

```

Controller:PLC:MyST_function
Controller:PLC:MyST_function: (5): Array index expected - must be a DINT expression
Controller:PLC:Error(s) detected
    
```

- Controller : PLC and MyST\_function refer to the program in the Project Explorer

- (5) refers to the 5th line in the source code



For more details, see ["Find and Replace Operations"](#) on page 95.

#### 5.1.5.4 Find and Replace Operations

The Find and Replace command enables you to search for a specified string of characters within your project.

You can use any of the following methods to access this functionality:

- From the **Information and Logs** toolbox
- In the **Dictionary** panel
- From an **editor** (ST/IL, FBD, FFLD)

##### 5.1.5.4.1 Information and Logs

For more details, refer to the [Information and Logs](#) toolbox.

##### 5.1.5.4.2 Dictionary

Right-click on the variable name and select the **Find all** command in the menu.

This command starts a search of all documents for the selected variable and displays the results in the table widget within the [Information and Logs](#) toolbox.

#### NOTE

This operation selects only the instances that match the complete words (for example, a search for "MyVar" returns "MyVar" but not "MyVar1").

##### 5.1.5.4.3 Editor

It is possible to perform a search and replace from a PLC editor (ST/IL, FBD, FFLD) by selecting the *Find*, *Find next*, and *Find All* commands in the contextual menu.

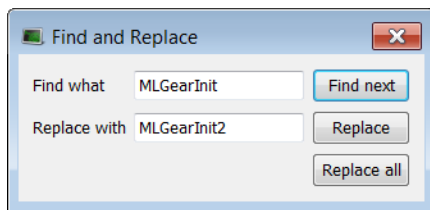


Figure 3-12: Find and Replace from an Editor

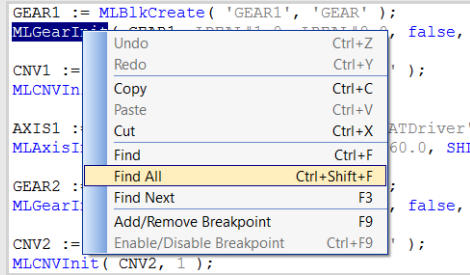
Function	Description
<b>Match Whole Word</b>	When selected, only the instances that match the complete words defined in the "Find what" field are selected (for example, a search for <code>MyVar</code> returns "MyVar" but not "MyVar1").
<b>Find next</b>	Allows you to select in the current editor the next instance of the matched string.
<b>Replace next</b>	Allows you to replace the next instance of the matched string.
<b>Replace all</b>	Allows you to replace in the current editor all instances of the matched string.

**NOTE**

The *Find*, *Replace* and *Replace all* operations work only for *variable symbol* property of the *Control*.

**TIP**

When the Find All command is applied to a variable, the system will return a list of all instances of the variable, not just those in the current program.

A screenshot of the KAS IDE code editor. The code is written in a dark theme. A context menu is open over the code, listing various editing and development actions. The 'Find All' option, with the keyboard shortcut Ctrl+Shift+F, is highlighted in yellow. The code in the background includes several lines of text with variable names like GEAR1, MLGearI, CNV1, MLCNVIn, AXIS1, MLAxisI, GEAR2, and CNV2, along with function calls like MLBlkCreate and MLCNVinit.

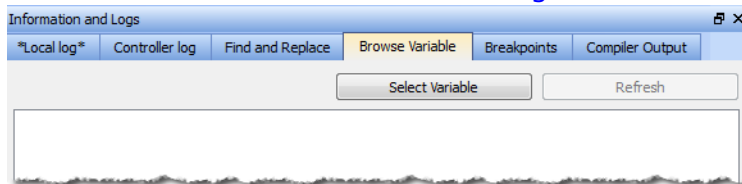
```
GEAR1 := MLBlkCreate( 'GEAR1', 'GEAR' );  
MLGearI := GEAR1;  
CNV1 := MLCNVIn( 'CNV1', 'CNV' );  
AXIS1 := MLAxisI( 'AXIS1', 'ATDriver', 60.0, SHI );  
GEAR2 := MLGearI;  
CNV2 := MLCNVinit( 'CNV2', 1 );
```



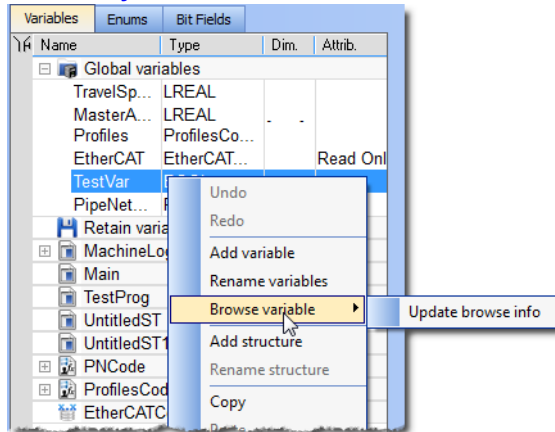
### 5.1.5.5 Browse Variable Tab

This tab is used to browse all instances of a variable. It will show the locations and usage of each instance of a single variable. There are two ways to populate the tab:

- Click the **Select Variable** button. See [Using the Browse Variable tab](#) for more information.

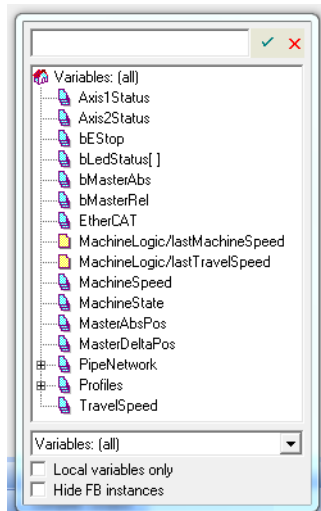


- Right-click on a variable in the Dictionary and select **Browse Variable**. See [Using the Dictionary's Browse Variable menu item](#) for more information.



#### 5.1.5.5.1 Using the Browse Variable tab

1. Click the **Select Variable** button to open a variable selection pop-up window.
2. Select a variable from the pop-up list.



3. The Browse Variable tab is updated with a two-column table detailing the variable's usage
  - **Location** shows the file name and line number where the variable is used.
  - **Usage** shows whether the variable is being read (**use**) or written to (**set**).
4. Double-click on an entry to be taken to the corresponding location in the editor.

**NOTE**

- The process of generating the Browse Variable content after clicking Select Variable can take a significant amount of time, depending upon the number of files in the project. If there are no modifications to the project, further browse operations will not take any time.
- The browse information may not be current if changes are made to the project.
- The browser information needs to be refreshed after saving and compiling a project. A message is shown in the tab to alert you if the information may not be accurate.

**5.1.5.5.2 Using the Dictionary's Browse Variable menu item**

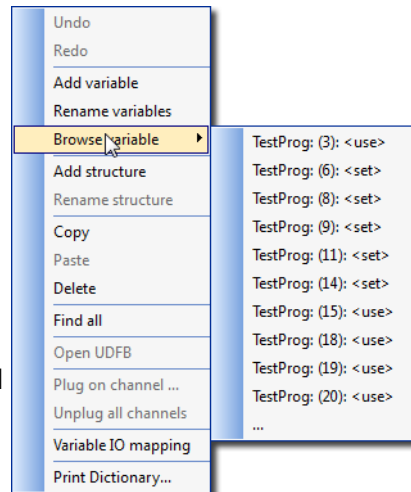
The first time this menu is accessed (per project instance) you are presented with **Update browse info**. This updates the browse data. Further selections of any variable result in the location and usage being shown directly in the menu.

- Selecting an entry will take you to the corresponding location in the editor.

When the menu has more than ten entries an ellipsis (...) is added to the bottom of the menu.

- Selecting the ellipsis (...) brings you to the Browse Variable tab, which will be populated with the variable locations and usages.

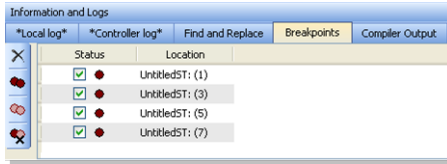
	Location	Usage
1	TestProg: (3)	use
2	TestProg: (6)	set
3	TestProg: (8)	set
4	TestProg: (9)	set



### 5.1.5.6 Breakpoints tab

The Breakpoints tab lists all of the breakpoints in the PLC program, including their position and status. Double-clicking on an entry will take you to that location in the editor.

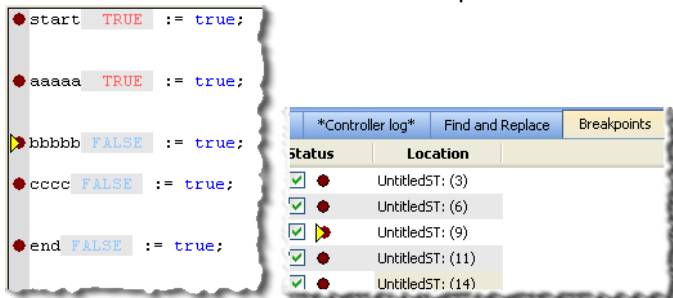
Breakpoints may be enabled and disabled singly by clicking the **Status** checkboxes. Buttons on the left of the tab provide the ability to remove single breakpoints, enable and disable all breakpoints, and remove all breakpoints.



Button	Action Performed
	Delete selected breakpoint from the list and editor.
	Enable all breakpoints. This will make all breakpoints "active"
	Disable all breakpoints. This will make all breakpoints "inactive"
	Remove all breakpoints.

Right-clicking on a breakpoint entry in the list provides for enabling, disabling, deleting the entry, and going to that location in the source code.

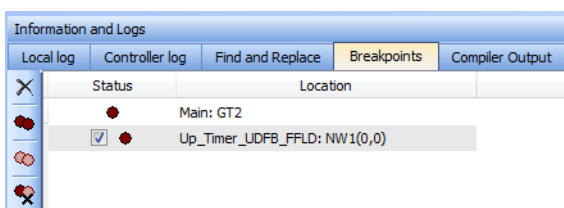
Breakpoints (both active and inactive) which have been "hit" or reached in the code are flagged with a yellow triangle. This provides a quick and easy way to identify the breakpoint. This can be seen in both the code and the Breakpoints tab.



**TIP**

Any program (except for an SFC program) that contains a breakpoint that gets "hit" during debugging will be automatically opened for your convenience.

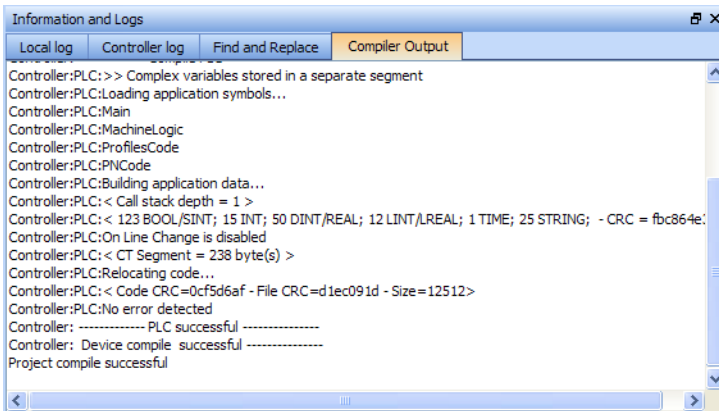
As breakpoints set in SFC programs cannot be enabled or disabled, entries in the Breakpoints widget do not have a checkbox to perform these actions.



**Figure 3-13:** Example of a breakpoint (*Main: GT2*) set in an SFC program.

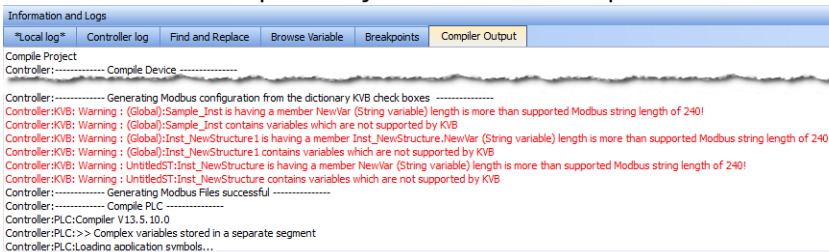
For more information on breakpoints, see [About Breakpoints](#) and [Setting, Removing, Enabling, and Disabling Breakpoints](#).

### 5.1.5.7 Compiler Output



**Figure 3-14:** Compiler Output

This tab displays information about the last project compilation. It shows information messages as well as Warnings and Errors (highlighted in red). Successful and unsuccessful output is reported within this tab to help identify and troubleshoot problem areas of the program development.



#### **TIP**

Double-click an error to open the program in the workspace and jump directly to the relevant location in the editor. This lets you rework the program and fix the error.

In the image above, the first warning is for `Sample_Inst`. Double clicking that item opens that item's location in the Dictionary.

Name	Type	Dim.	Attrib.	Init value	User ...	KVB
Global variables						
TravelSpeed	LREAL			LREAL...		<input type="checkbox"/>
MasterAbsPos	LREAL			LREAL...		<input type="checkbox"/>
MasterDeltaPos	LREAL			LREAL...		<input type="checkbox"/>
PipeNetwork	PNCode					<input type="checkbox"/>
Sample_Inst	NewStructure					<input checked="" type="checkbox"/>
Inst_NewStructure1	NewStructure					<input checked="" type="checkbox"/>

When there is a long list of statements, only the bottom part is displayed. Do not forget to scroll up.

### How to Clean-up the Code?

To clean-up your application, do as follows:

1. Scroll up to start from the top and locate the first error message
2. Fix the error

**NOTE**

Because fixing **one** piece of code can eliminate **multiple** compiler output error, it is recommended to recompile each time you correct an error.

When no more errors exist, the following messages are displayed:

- PLC successful (the IEC 61131-3 code is correct)
- Device compile successful (is related to the Motion part (e.g. CAM profiles), EtherCAT XML file...)
- Generating Modbus files (related to the variables mapped with the HMI)
- Project compile successful (the complete project is ready to be downloaded to the target)

**Text displayed:**

Operands of "\*" or "/" must be numbers and have the same type

**Meaning:**

This error appears in a ST instruction when a [constant](#) does not have the expected type in a multiplication or division operation. Typically, REAL is the default precision for floating points, so you have to explicitly declare your long real constants with the [LREAL#](#) prefix when required.

**5.1.5.7.1 How to Understand the Details of Location?**

Same explanations contained in previous section **Find and Replace** are also applicable here.

### 5.1.6 Watch Window

This toolbox enables you to add variables to a dedicated watch window to display its value in real time.

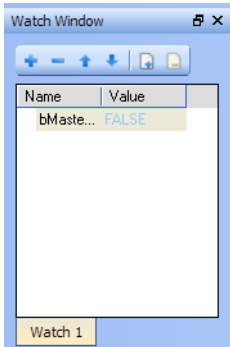


Figure 3-15: Watch Window

#### NOTE

- PLC variables viewed in the Watch Window are saved as a part of the project. This means that the next time you open the project, those variables will be pre-loaded in the watch window. This does not apply to AKD variables, which are not saved with the project.
- While variables may be changed or locked/unlocked (see [Force a Variable](#)), parameters may not be. AKD parameters in the Watch Window are read only.

#### 5.1.6.1 Multiple Watch Windows

The KAS IDE allows you to group several variables in a single watch window, and to have up to 10 different watch windows. Each of them is displayed as a tab with its own label.

Explanation for each icon:

Icon	Description
	Add a variable with the <a href="#">PLC Variable Selector</a>
	Remove a variable
	Move up the selected variable
	Move down the selected variable
	Add a new watch window. Each window is displayed as a tab with its own label. (Note that there is a limit of 10 tabs)
	Remove the selected watch window

Table 3-28: Watch Window - List of Icons

#### TIP

You can also add an existing variable in the watch window directly by using **drag-and-drop** from the Dictionary or the PLC editors.

Each variable in the table widget has the following information:

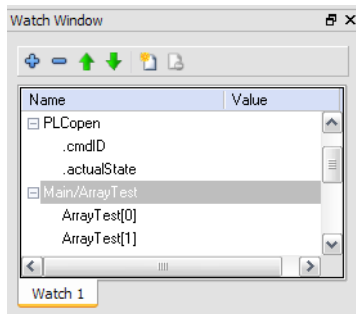
Field	Description
<b>Name</b>	Lists the variables as well as <a href="#">structure</a> , <a href="#">arrays</a> and expressions. You can double-click a variable (or press <b>F2</b> key when it is selected) to edit its name (except for structure and array members)

Field	Description
Value	When the application is running, displays the variable or expression's value. You can double-click a value to <b>force</b> modification of the selected variable

The contextual menu allows you to:

- Add a variable
- Remove a variable
- Remove all variables

### 5.1.6.2 Access Structure and Arrays



**Figure 3-16:** Watch Window - Accessing Arrays

When a structure or an array is in the watch window, you can expand its node to display all its members.

Note that structure or array members cannot be deleted, edited or moved up/down in the list.

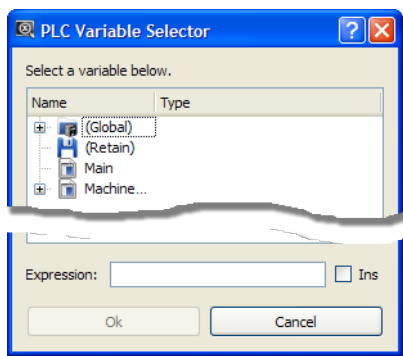
### 5.1.6.3 Add Variable

- Double-click the nodes ((Global), Main...) to expand their related variables

**TIP**

Expand AKD node if you want to add AKD parameters to the Watch Window

- Select one from the list
- Click **OK**



**Figure 3-17:** Watch Window - Selecting PLC Variable

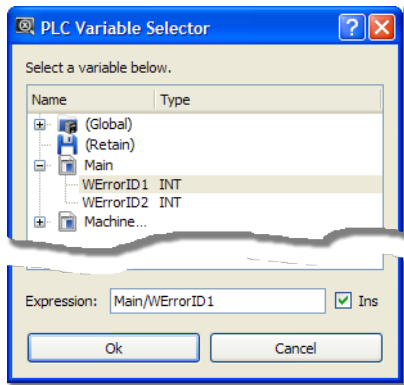
This variable is then added to the current watch window tab.

### 5.1.6.4 Add an Expression

You can enter variable strings as an expression.

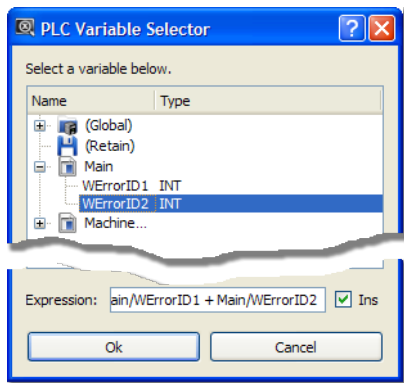
For example, if you want to add together two integer variables, follow these steps:

- Click the Add symbol **+** to open the PLC Variable Selector
- Choose a variable, but do not click OK yet  
(the variable is added to the expression field where you can do any required editing)
- Select the **Ins** option  
(this option allows you to insert the next selected variable at the current cursor position in the expression edit field)

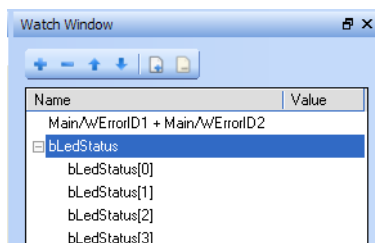


**Figure 3-18:** Watch Window - Creating Expression

- Press the PLUS SIGN (+) in the expression field
- Select another variable
- Click the **OK** button



- Then the expression is displayed into the watch window



**Figure 3-19:** Watch Window - Displaying Expression

What you can include in a complex expression:



- Index of array
- Comparison ">", "<", "<>", "="
- Operator "+", "\*", "-", "/"

Please note that the DIVIDE SIGN (/) is not interpreted as an operator when used with prefixed variables (e.g. MachineLogic/lastTravelSpeed)

### 5.1.6.5 Force a Variable

At run-time, all variables in the table widget are animated <sup>1</sup> with real-time values.

You can double-click on the value of a variable (or press the **ENTER** key when it is selected) to open a pop-up window that allows you to:

- **Force:** change the value of the selected variable. Depending on the variable type, you have the possibility to define its value either in the text field or with the check boxes.

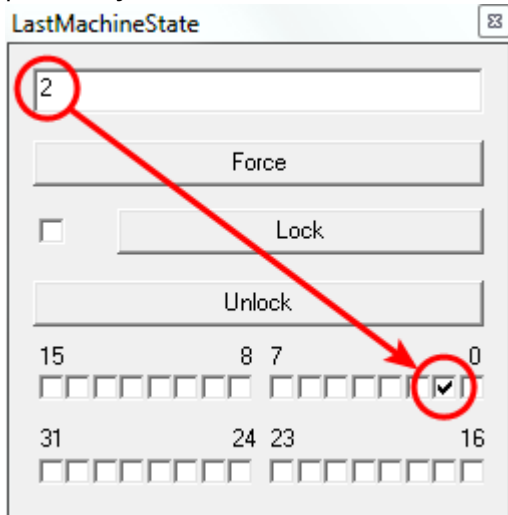


Figure 3-20: Forcing a variable

- **Lock:** Variables, member variables of a structure and variables in an array may be locked. When a variable is locked, its value is no longer changed by the runtime. You can then force its value from the debugger independently from the runtime operations.

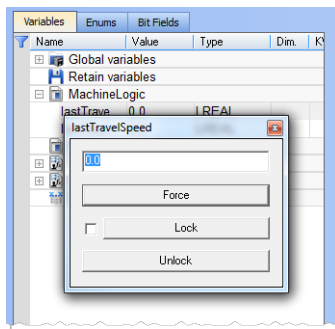
**NOTE**

The value of a locked variable is displayed with square brackets.

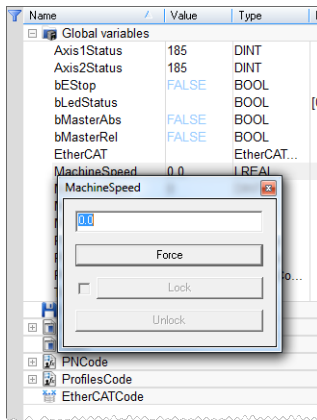
```
ActualMachineState [[ -1 ]] := -1;
```

The variable locking feature can be enabled or disabled via the PLC Options device toolbar button. If enabled, the Lock and Unlock buttons are accessible:

<sup>1</sup>To better track variables and expressions of the PLC programs in test mode, the KAS IDE dynamically computes their value along with the execution and displays the result



If disabled (default), the Lock and Unlock buttons are not accessible:



### ⓘ IMPORTANT

If PLC variable locking is enabled, the controller Runtime requires an additional 3% to 10% CPU processing power to manage the PLC variable locking. For the best controller performance, disable PLC variable locking.

- **Unlock:**  
Remove the lock on a variable so that it can be changed again by the runtime.

### 5.1.7 AKD Drive

First generation AKD drives support one axis per drive, while AKD2G drives support two axes per drive. When configuring drives in the [EtherCAT Devices tab](#), you will need to designate which motion axis applies to each "drive axis", which is the actual axis the drive controls.

In addition to the different views, the WorkBench provides a [toolbar](#) and a [status bar](#) to display some extra information.

#### 5.1.7.1 WorkBench Toolbar

The toolbar provides access to the following:

- Enable / Disable the drive (software enable)
- Start / Stop the Service Motion
- Mode:position / torque / velocity
- Disable & Clear Faults: Click this button to clear the fault, then click the Enable button to enable the drive again

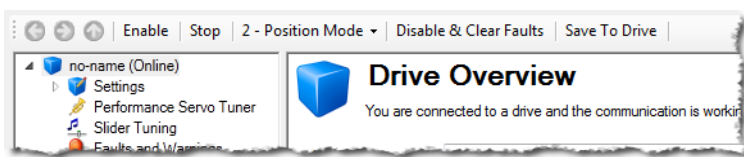



Figure 3-21: AKD Toolbar

#### TIP

To stop all the AKD drives at the same time, click on the Stop button  in the [Device Toolbar](#).

#### 5.1.7.2 WorkBench Status Bar

The status bar provides the following information on the drive:

- A fault indicator (No Faults / x Faults) that becomes red when any AKD gets a fault  
You can also set the [Log message](#) to get more details on the drive messages
- The drive status: active / inactive
- The software (SW) enable status
- The hardware (HW) enable status



Figure 3-22: AKD Status Bar

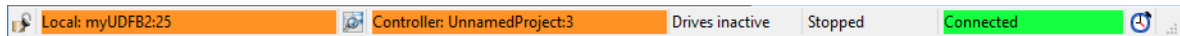
For the SW and HW enable status indicators, the color code is:

- **Green** when it is OK (i.e. everything is ready to do motion)
- **Red** in case of errors / faults
- **Grey** for all other cases (for example when SW or HW is not enabled: status is not green because a motion could not happen, and not red because it is not an error)


### 5.1.8 Status Bar

A status bar located at the bottom of the KAS IDE main window displays the five following labels from left to right:

- [Local version](#)
- [Controller version](#) (application version located in the controller)
- [Drives state](#)
- [Controller state](#) (stopped/running)
- [Connection state](#)



**Figure 3-23:** Status Bar Labels

An icon  between the Local and Controller versions allows to show any differences (for more details, [Compare PLC Programs](#)).

The space on the left of the status bar is reserved for messages.

#### 5.1.8.1 Local Version

This label provides information about the version locally present in the KAS IDE. There are three different states:

- *Nothing displayed* (for instance when no project is loaded)
- Version information (when available)
- Compilation error (background in **red** )

#### TIP

You can position the mouse over the text field to display a tooltip with the detailed version information.

#### 5.1.8.2 Controller Version

This label provides information about the version present in the controller. There are three different states:

- *Nothing displayed* (when not connected)
- No Application in the controller
- Version information (when available)  
Syntax of the version label is: <project\_name>:<version>

#### TIP

When an application is active in the target, you can hold the mouse over the text field to display a tooltip with the detailed version information, including a timestamp of the compilation.

#### 5.1.8.3 Drives State

There are three different states:

- Drives inactive (drives are disabled or your application is not connected to the target)
- Drives active (at least one drive is active)
- Drives error ( at least one drive is in error)

### 5.1.8.4 Controller State

The Controller state label lets you know if the Controller is running or stopped. There are three different states:

- *Nothing displayed* (the label is empty when the KAS IDE is not connected to the target)
- Controller is stopped
- Controller is running

### 5.1.8.5 Connection State

The Connection label displays the Connection state between the KAS IDE and the Controller. There are five different states:

- Not connected
- Connecting
- Connected (background in **green**)
- Connection Error (background in **red**)
- Unexpected Disconnection (background in **red**)

**TIP**

You can hold the mouse over the text field to display a tooltip with some detailed information about the Error, and the Controller address when connected.

### 5.1.8.6 Color Codes

The Local and Controller version labels has an **orange** background in case of version mismatch between the IDE and the Controller. This warns you that you have to download the new version of the application.

The Local version label has a **red** background if the compilation fails.

List of use cases for the labels of the status bar

The following table summarizes all cases for the labels of the status bar.

Connection state	Local version	Controller version	Controller status	Connection status
Disconnected				Not Connected
Disconnected	Version A			Not Connected
Connecting	Version A			Connecting...
Connected	Version A	No Application	Stopped	Connected
Connected	Version B	Version B	Stopped or running	Connected
Connected	Version B	Version A	Stopped or running	Connected
Disconnected	Compile error			Not Connected
Connected	Compile error	Version A	Stopped or running	Connected
Comm. error	Version A			Connection Error
Disconnected	Version A			Unexpected disconnection

Table 3-29: Connection Status



## 6 Getting Started



This chapter provides explanations and procedures to accomplish common **tasks** with the KAS IDE.

### 6.1 KAS IDE to Runtime Compatibility

The KAS software includes two main software components, the IDE and the Runtime. The IDE resides on your PC and the Runtime resides on your programmable automation controller or programmable drive. For optimum operation, the IDE you install and run on your PC must be compatible with the Runtime you install on your controller or programmable drive.

The KAS installer and AKD PDMM or PCMM Runtime contain the version information in the filename:

```
KAS-Setup-3.07.0.60001.exe
KAS-PCMM-M-M1EC-3.07.0.60001.img
KAS-PCMM-M-MCEC-3.07.0.60001.img
KAS-AKD PDMM-M-M1EC-3.07.0.60001.img
KAS-AKD PDMM-M-MCEC-3.07.0.60001.img
```

#### NOTE

The version in the text above is only an example of the syntax. The firmware file will have a different version number.

The version can be determined by the following:

Version Source	Found From	Example
IDE, Controller or Simulator	Help menu > About	
PDMM	Web server Settings Firmware tab	

The compatibility between the IDE and the Runtime is defined by matching the digits in the versions. The IDE and Runtime use the same version scheme:



```
major . minor . micro . revision
```

- If the IDE to Runtime major.minor is NOT equal, they are NOT compatible.
- If the IDE to Runtime major.minor is equal, but the micro is NOT equal, they are not 100% compatible.
- If the IDE to Runtime major.minor.micro is equal, they are compatible.

### Examples

IDE	Runtime	Compatible?	IDE to Runtime Connection
3.07 .xx.xxxx	1.2.xx.xxxx	NO, No connection.	The IDE displays an error message.
3.07 .0.xxxx	3.07.1.xxxx	YES*	Connection possible. The IDE displays warning and requires user to press "ok" to continue.
3.07 .0.xxxx	3.07.0.xxxx	YES	Normal connection.
3.07 .0.30540	3.07 .0.30540	YES	Normal connection.

\* Not a recommended configuration. The IDE will operate, but there can be different features available between the IDE and the Runtime. It is recommended to upgrade either the IDE or the Runtime to matching versions.

## 6.2 KAS PLC Code to Runtime Compatibility

It is possible that after a Controller Runtime upgrade that the Runtime and the compiled project versions `major.minor.micro` may be different. The Runtime will not start if there is an incompatibility between the versions.

<b>AKD PDMM or PCMM</b>	An E24 error will be displayed and an error will be logged if compiled project version is incompatible with the Runtime version.
<b>PAC</b>	An error will be logged if the compiled project version is incompatible with the Runtime version.

To correct the incompatible versions, either re-compile and download the KAS project file using the same IDE version as the Runtime version OR change the Controller's Runtime version to match the compiled project's version.

### 6.3 KAS IDE to Project File Compatibility

This section helps describe forwards and backward compatibility between the KAS IDE and project files.

#### *About Backward Compatibility*

The KAS IDE is backward compatible, meaning the KAS IDE can open project files made with older versions of the software. There are a few things to note.

- An "older" project file will be converted to take advantage of features in the newer version.
- As there may be multiple conversions needed to update the file, if any step is not backward compatible you will be warned with a pop-up message and the process will continue with the next conversion step.
- You can exit without saving the project conversion.
- You may use the "Save As" function to make a copy of the newly converted project file, preserving the original.

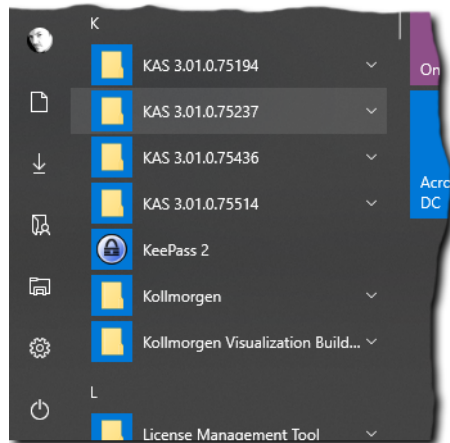
#### *About Forward Compatibility*

The KAS IDE strives to be forward compatible, meaning it can open project files made with newer versions of the software. If there are no compatibility issues, the KAS IDE will simply open the file.

## 6.4 Starting the KAS IDE

There are multiple ways to start the KAS IDE.

- Double-click on a desktop shortcut.
- Double-click on a project file to open it with the most recently installed version of the KAS IDE..
- *Windows 10:* From the Start menu, under "K".
  - All installed versions may be found here, including one legacy version (in the Kollmorgen folder), and all installed versions of v3.01 and later. See [Can I install more than one version of the KAS IDE?](#) for more information.



- *Windows 7:* Open **All Programs** and start the **KAS IDE** application located under the **Kollmorgen** folder.

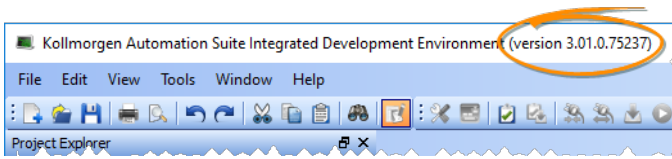
### **TIP**

The KAS IDE supports simultaneously running multiple instances. Double-clicking a shortcut or project file, or selecting the application from the Start menu will open another instance, Double-clicking on a project file more than once will open a copy of the project in another instance of the KAS IDE.

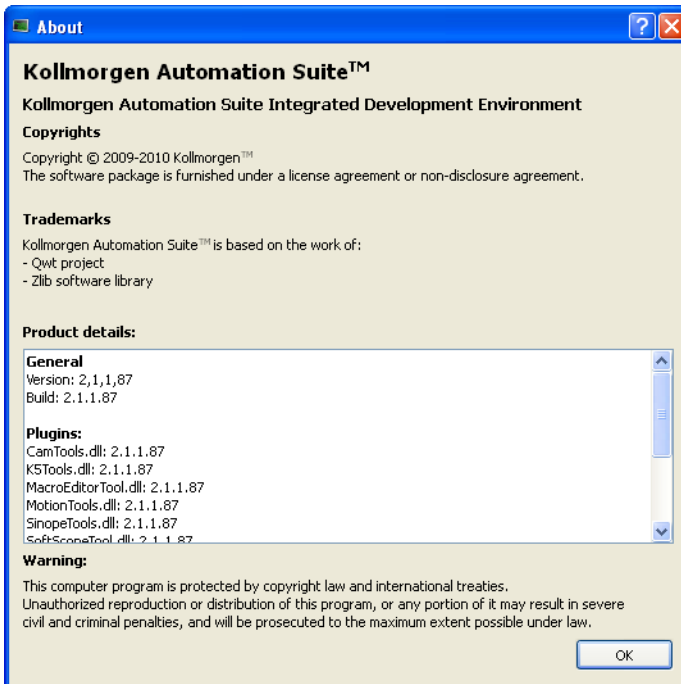
See the FAQs [How do I launch my project with a particular version of the KAS IDE?](#) and [How do I choose the default version of the KAS IDE to use?](#) for more options.

### 6.4.1 Find KAS IDE Version

The version of the KAS IDE or KAS Simulator being used is visible in the title bar of the application.



For more detailed information you may use the **About** command in the **Help** menu.



**Figure 4-1: About Window**

This window displays the application versions as well as all the plug-in versions included in the KAS IDE and loaded during start up.

### 6.4.2 Access Help System

You can access the online help using the **Documentation** command in the **Help** menu.

See also "Use the Context-Sensitive Help provided in "

### 6.4.3 KAS GUI

For a better understanding of KAS menus, toolbar and workspace items (description and manipulation), refer to [Describing KAS Graphical User Interface](#)

### 6.4.4 KAS IDE Log Window

#### 6.4.4.1 Log Information

The KAS log window (see [Log Messages](#)) provides a running display of activity related to the execution of the application. Items displayed include application startup and initialization information.

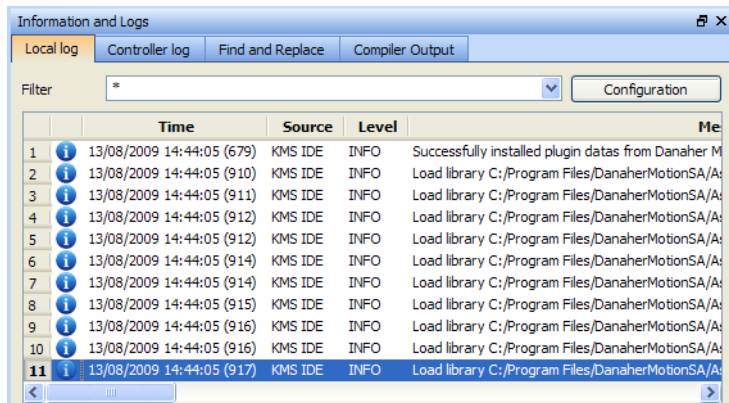


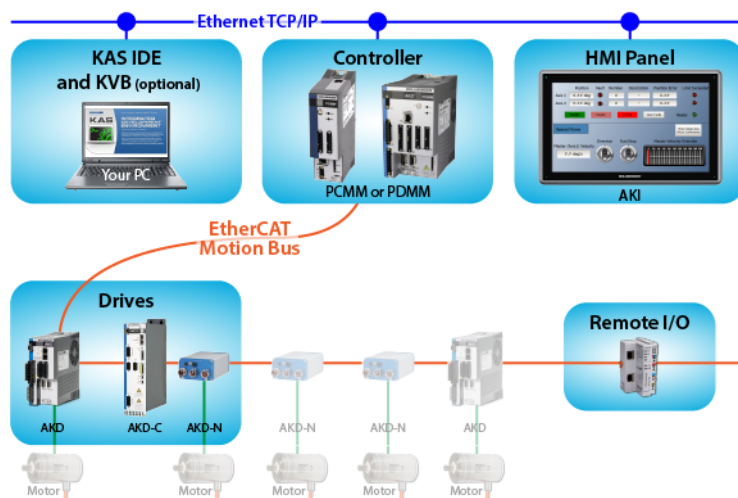
Figure 4-2: Log Messages

#### Information and Logs

### 6.4.5 Create a Project

There are six basic aspects of setting up a KAS project.

1. Setting up the controller
2. Setting up the drives
3. Defining the remote I/O
4. Setting up the EtherCAT Motion Bus
5. Creating the program in the KAS IDE
6. Setting up the HMI panel

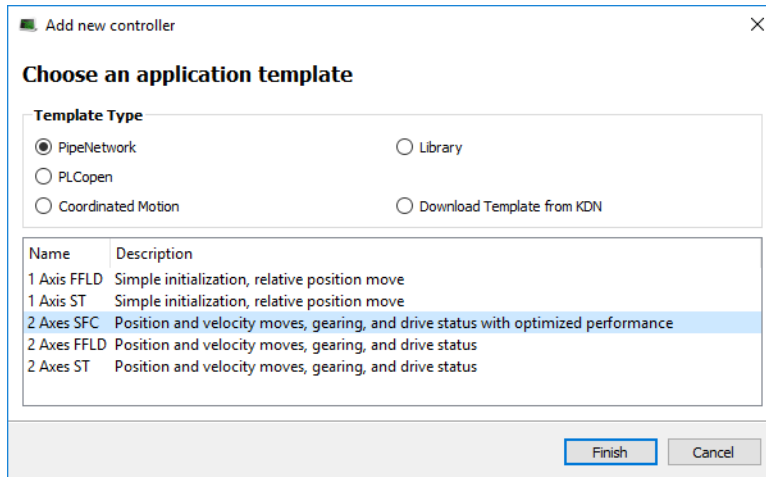


## 6.4.6 Add and Configure a Controller

### 6.4.6.1 Add the Controller

To add a controller to your project:

- Click the **New** command in the **File** menu to start the Controller Creation Wizard
- Choose the motion engine option (Pipe Network or PLCopen) and select the application template (see [list](#) below)



**Figure 4-3:** Select an Application Template

- Click the **Finish** button
- Click the **Save As** command in the **File** menu
- Define the Project Name and its Location
- Click **OK**

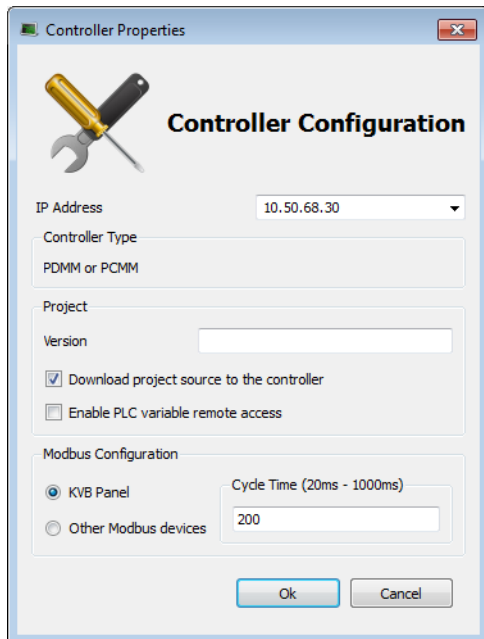
*List of available application templates*

Template Type	Template name	Description
PipeNetwork	1 Axis FFLD	Simple initialization, relative position move
	1 Axis ST	Simple initialization, relative position move
	2 Axes FFLD	Position and velocity moves, gearing, camming, and drive status
	2 Axes ST	Position and velocity moves, gearing, camming, and drive status
	2 Axes SFC	Position and velocity moves, gearing, camming, and drive status with optimized performance.
PLCopen	1 Axis FFLD	Simple initialization, relative position move
	1 Axis ST	Simple initialization, relative position move
	2 Axes FFLD	Position and velocity moves, gearing, camming, and drive status
	2 Axes ST	Position and velocity moves, gearing, camming, and drive status
	2 Axes SFC	Position and velocity moves, gearing, camming, and drive status with optimized performance.

Template Type	Template name	Description
Coordinated Motion	2 Axes - Linear / Circular	Raster Scan Motion Path, 2 PLCopen axes
	3 Axes - Linear / Circular	Raster Scan Motion Path, 2 PLCopen axes and 1 PipeNetwork axis
	3 Axes - Linear (3D)	Diamond/Square Motion Path, 3 PLCopen axes
KAS Runtime	Library	Allows you to create a custom library ( <a href="#">Create and Use Custom Libraries</a> )
Download Template from KDN	KDN	Upon selecting Finish, the dialog closes and a web browser opens a page on KDN which contains custom-built template files, which may have enhanced, or specific functionality. Download and save a template file, then open it from within the KAS IDE.

### 6.4.6.2 Configure the Controller

The controller is configured using the Controller Properties dialog box.



**Figure 4-4:** Configure the Controller Properties

To set-up the controller:

1. In the Project Explorer, right-click on the new controller to open the contextual menu
2. Select the **Properties** command
3. Define the IP Address

*A note about addressing*

- For the KAS Runtime Simulator, enter the localhost IP address: 127.0.0.1
- For the AKD PDMM or PCMM enter the IP address of the controller (e.g. 10.155.100.150). The default is 192.168.0.1, which is the same as the default IP address on the PCMM or AKD PDMM device.

**NOTE**

You must ensure that controller is accessible by the KAS IDE machine (see [FAQ section](#) for IT issues)

**IMPORTANT**

Do not use leading zeroes when entering an IP address. The connection to the controller will automatically fail if leading zeroes are used. For example, use **10.2.3.40** instead of **010.002.003.040**.

- (Optional) Specify a version number (the string can be composed of any character)

**TIP**

Versioning can be useful when you make improvements to your application and need a version control system (Use a [Version Control System](#)). The version is saved in your project file.

- (Optional) Choose whether the project's source code should be downloaded to the controller. This is enabled by default and your preference is saved with the project. Disabling this option means a comparison of source on the controller and in the IDE will not be available.
- (Optional) Choose whether PLC variables may be read/written via an HTTP connection. this requires that the source be downloaded to the controller. By default this option is disabled so unauthorized changes may not be made.

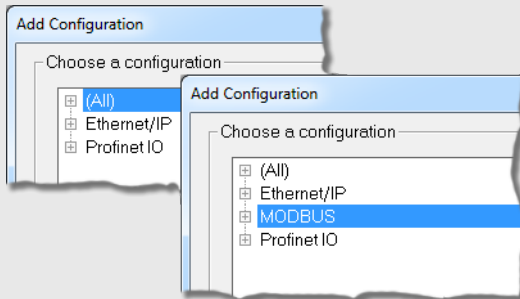
**TIP**

Be sure to recompile the application before downloading it to the controller.

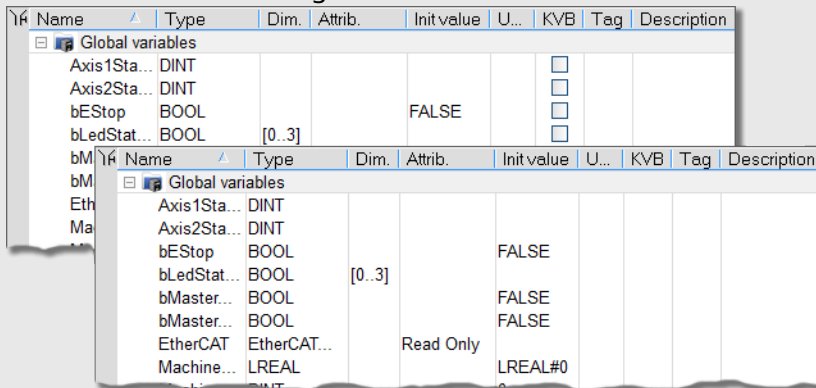
- Select whether Modbus will go to a KVB Panel or be handled by another Modbus device, and set the **Cycle Time** (the communication update frequency, see "[Modbus Slave](#)" on page 658 for more information).

**NOTE**

- If KVB Panel is selected there will be no node to add a Modbus configuration in the Fieldbus Editor.



- If Other Modbus devices is selected then the KVB checkbox for Dictionary items will be hidden.. Motion configuration will need to be added from the Fieldbus Editor.





8. Click **OK**

### 6.4.7 Configure EtherCAT Motion Bus

**TIP**

Before configuring your EtherCAT settings, you may wish to add third party EtherCAT devices. See [Add & Configure Third Party Devices](#) for more information.

Double-click the **EtherCAT** node in the Project Explorer to open the EtherCAT properties dialog in the workspace. This window is composed of following tabs:

Tab	Description
<a href="#">EtherCAT Devices tab</a>	Displays all the E-Bus devices present in the project tree
<a href="#">EtherCAT Master Settings Tab</a>	Allows you to configure the EtherCAT bus master
<a href="#">ENI File Tab</a>	Allows you to use an external configuration file
<a href="#">ESI Files Tab</a>	Display, add, and remove available ESI files

KAS includes an integrated tool to configure the EtherCAT **master** and start up the fieldbus operation.

The configuration tool enables you to:

- Describe your motion **topology** as a configuration tree (see procedure in [EtherCAT Devices tab](#))
- Associate variables to the I/O channels of devices (see procedure in [Map Input and Output to Variables](#))

#### About Slave devices

Slave devices can support several PDOs (for the list, [PDOs for AKD, AKD2G, AKD-N, and S300/S700](#)). Some of them are mandatory; others are optional.

One of the main tasks of the EtherCAT configuration is to select the PDOs used by each slave (see also [EtherCAT Devices Summary Form](#)) and group them all in the EtherCAT image.

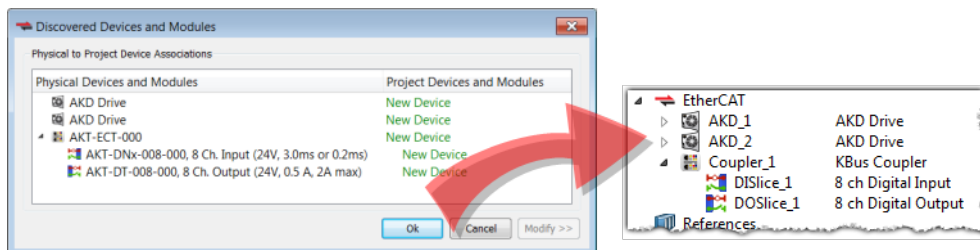
**NOTE**

PDOs contain real-time cyclic data which is deterministic. Non-cyclic data is not deterministic and is defined by Service Data Objects ([SDO](#)).

As explained in the [introduction](#), input and output parameters are grouped in predefined blocks called PDOs.

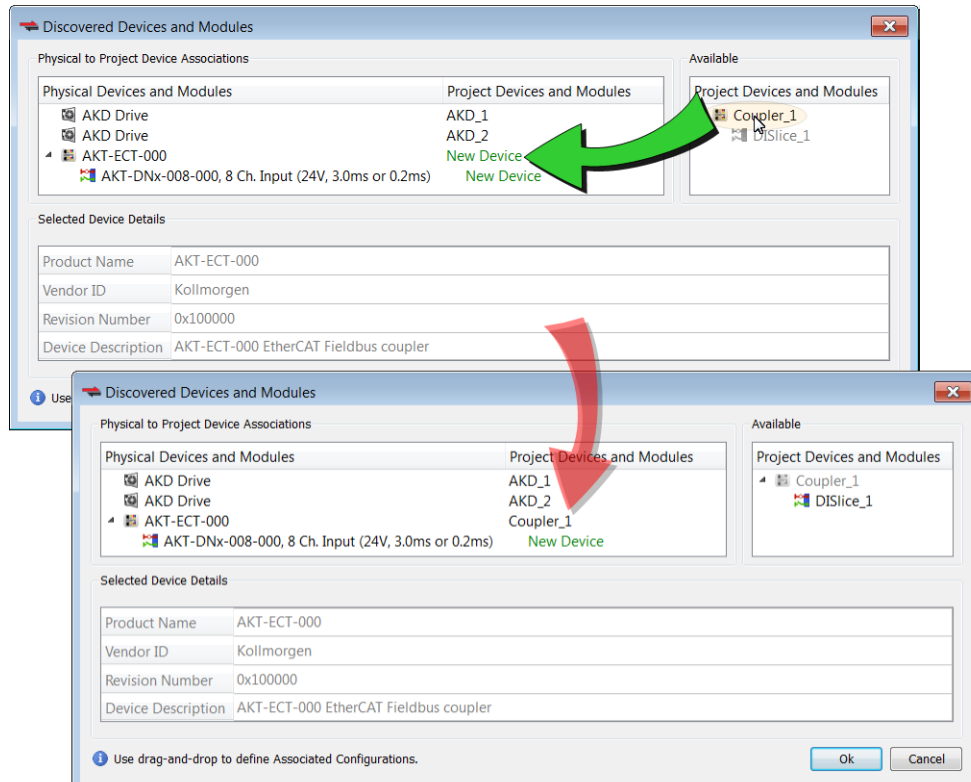
#### 6.4.7.1 EtherCAT Scan and Association

Physical EtherCAT devices, modules, couplers, and slices are discovered during the **Scan Devices** routine and are automatically set as project devices in the Project Tree. These device associations will persist through subsequent scans unless you change them. The **Scan Devices** is accessed by double-clicking the EtherCAT node in the project explorer, then pressing the **Scan Devices** button in the **Devices** tab.



Changing the network's physical topology (by adding moving, or removing devices) or changing the associations between physical devices and project devices will require that **Scan Devices** be

run again. The **Discovered Devices and Modules** window will display any changes and the **Modify** button will be enabled. When **Modify** is clicked the dialog box expands to show Available Project Devices and Modules. Click on a device and drag it to the corresponding **New Device** to create an association. Repeat as necessary for slices and modules.



Once the associations are complete, click **Ok**. See the following procedures for more details.

- New projects with no project devices: [Scan and Associate Network Devices](#). This creates default configurations for the discovered physical devices/modules and couplers/slices.
- Adding, removing, or moving devices: [Re-Scan and Change Device Associations](#).
- Adding, removing, or moving couplers with slices: [Re-Scan and Change Coupler/Slice Associations](#).
- Adding, removing, or moving devices with modules: [Re-Scan and Change Device/Module Associations](#).

The physical network topology will be compared to the project's expected network topology after the EtherCAT devices are associated and the PLC application is started. An error will be reported if the topologies do not match and the PLC program will not run.

#### NOTE

The physical topology must match the project's expected topology for the PLC application to operate the devices properly.

### 6.4.7.1.1 Scan and Associate Network Devices

To discover the physical EtherCAT devices, follow these steps:

1. In the Project Explorer, double-click the **EtherCAT** node to open the **Devices** tab. If there are no devices, the EtherCAT network topology has not yet been scanned and discovered.
2. In the **Devices** tab, click the **Scan Devices** button.

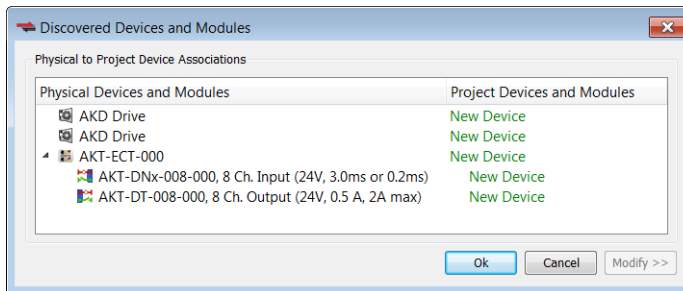
**NOTE**

The topology discovery is only enabled when the controller is not running an application.

- If the scan process fails, refer to the [EtherCAT Error Messages](#).



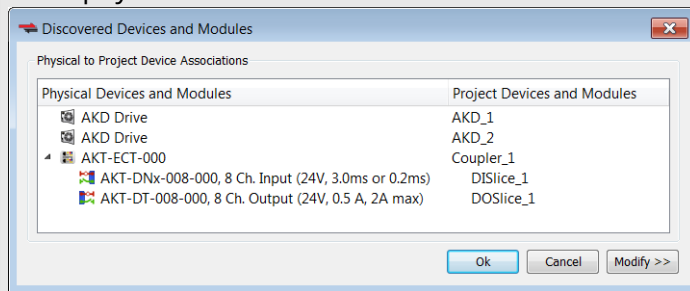
- If physical devices are discovered during the scan, they will appear under the Physical Devices column.



3. Press **OK**. The devices will appear in the **Devices** tab and in the **Project Explorer**. Their associated project devices are set to the defaults.
4. If you have previously compiled your project, you will be asked if you wish to recompile the project.

**TIP**

If you click **Scan Devices** again, the existing project devices will be automatically matched to the same physical devices. The association will not be lost from rescanning.

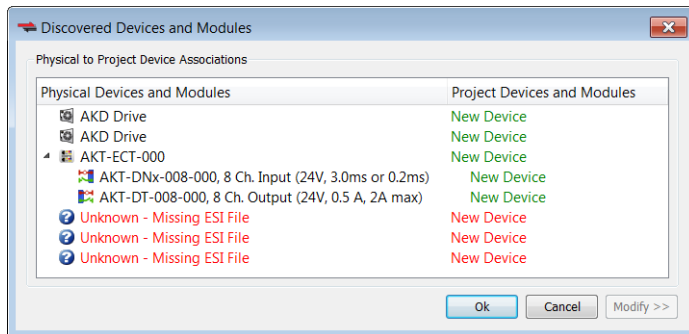


To change the device associations see [Re-Scan and Change Device Associations](#).

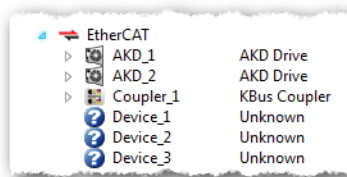
To troubleshoot an error due to a missing ESI file, please see [Unknown – Missing ESI File](#).

### 6.4.7.1.1.1 Unknown - Missing ESI File

The **Discovered Devices and Modules** view will identify a device as "Unknown" if a corresponding ESI file is not found during the Scan for physical EtherCAT devices. For example, the last three devices are missing ESI files:

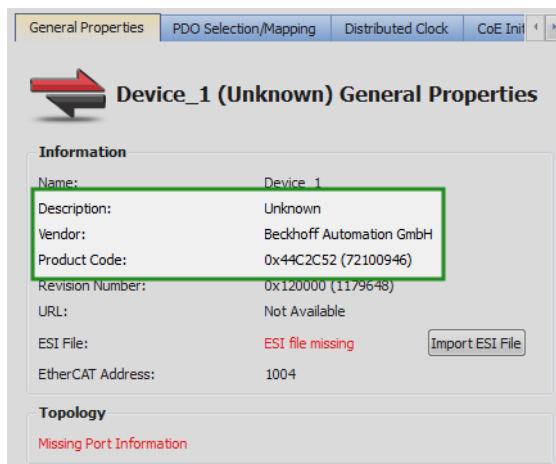


Click on the **Ok** button and the Unknown devices will be added to the Project View:



To correct the problem:

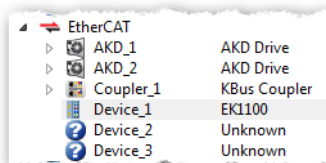
1. Double-click on the "Unknown" device in the Project View or right-click and select **Properties**.
2. From the General Properties view, identify the **Vendor**, **Product Code**, and **Revision Number**.



3. Contact the vendor for the EtherCAT ESI file for this device.

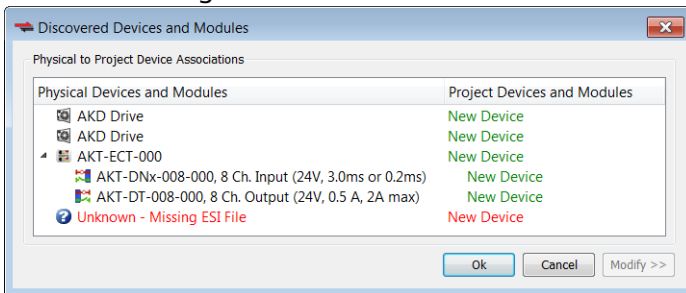
The vendor may provide an ESI file that supports multiple devices or they may provide a separate ESI file for each device.

4. Import the ESI file using the **Import ESI File** button and the device specific information will be automatically updated in the Project View and General Properties.

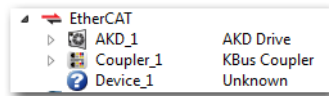


### 6.4.7.1.1.2 Missing ESI File - MDP Device

During the Scan for physical EtherCAT devices, if a MDP device is discovered that does not have a corresponding ESI file, then the **Discovered Devices and Modules** view will identify the device as “Unknown” and will not be able to discover any modules under the device. For example, the last device is missing an ESI file:

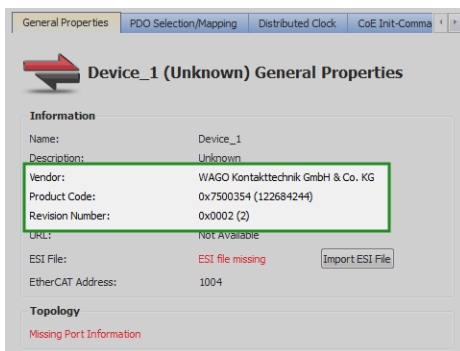


Click on the **Ok** button and the **Unknown** device will be added to the Project View.

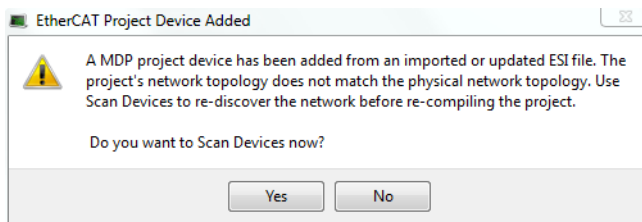


To correct the problem:

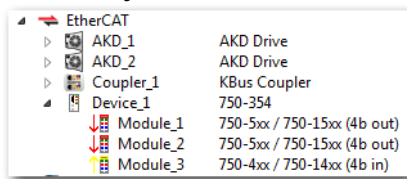
1. Double-click on the “Unknown” device in the Project View or right-click and select **Properties**.
2. From the General Properties view, identify the **Vendor**, **Product Code**, and **Revision Number**.



3. Contact the vendor for the EtherCAT ESI file for this device. The vendor may provide an ESI file that supports multiple devices or they may provide a separate ESI file for each device.
4. Import the ESI file using the **Import ESI File** button. A pop-up message will indicate that this a MDP device and the EtherCAT network must be scanned to re-discover the modules:



After the scan and association of devices/modules, the MDP device and its modules will be added to the Project View:



### 6.4.7.1.2 Re-Scan and Change Device Associations

Device Association must be updated after the following actions.

- Changing the physical network topology by adding, moving, or removing EtherCAT devices.

**⚠ CAUTION**

Changing the EtherCAT network topology (plugging and unplugging EtherCAT cables) is supported while power is supplied. Personal injury or damage to devices may occur if EtherCAT devices are installed, wiring is changed, or the terminal configuration is changed while power is supplied ("hot swapping").

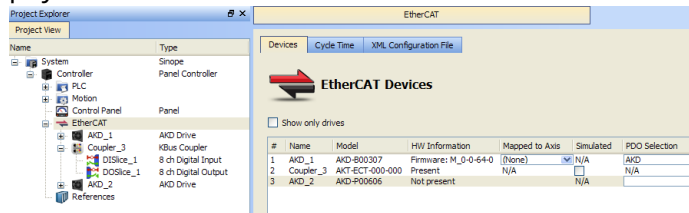
- If you want to change the associations between devices and configurations.
- Changing the FSoE address via physical switch settings (DIP switches, rotary switches, etc.) for Safety devices.

To update the associations:

1. Re-scan and re-discover the physical network topology.
2. Click the **Modify** button to expand the Discovered Devices and Modules view.
3. Drag-and-drop the project coupler and/or slice(s) to associate them with the physical couplers/slices.

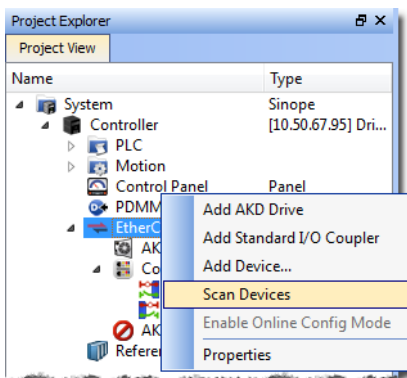
For example, a project has the following configuration:

- The first AKD Drive and a I/O Coupler were created during a previous scan
- The second AKD was added to the project manually, and is not yet associated with any physical drive



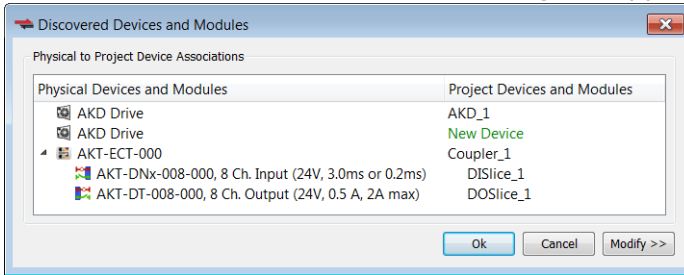
To update the network topology:

1. In the Project Explorer, double-click the EtherCAT node to open its **Properties** and click the **Scan Devices** button in the Devices tab. Alternatively, right-click on the EtherCAT node in the Project Explorer and select **Scan Devices**.

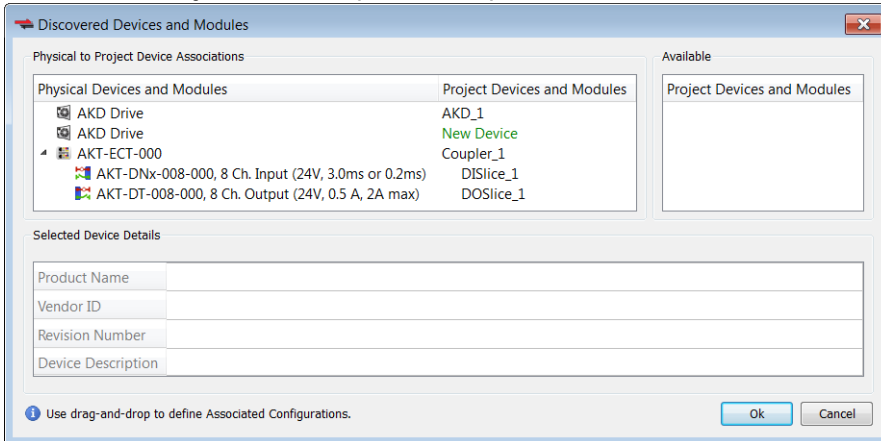




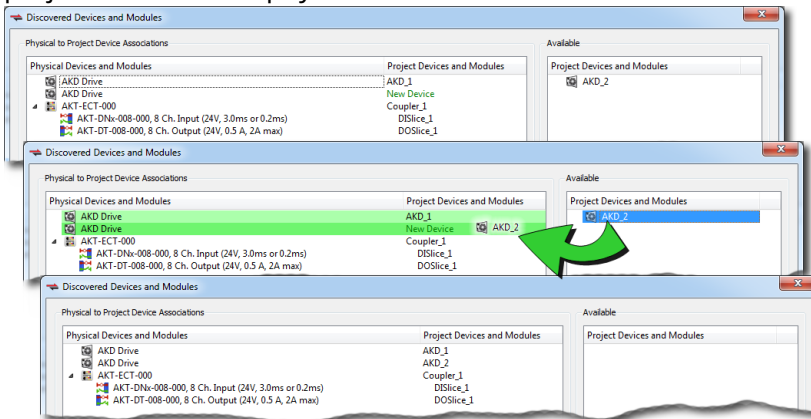
- The Discovered Devices and Modules dialog box appears after a successful scan.



- Click the **Modify** button to open the expanded view.

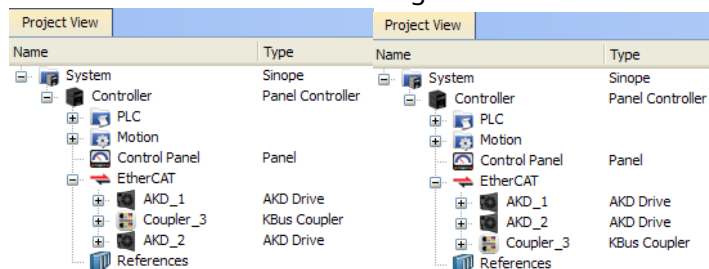


- Drag-and-drop **AKD\_2** from **Available** to the second physical AKD drive. This associates the project device to the physical device.



- Click **OK** to confirm the modified association.
- If you have previously compiled your project, you will be asked if you wish to recompile the project.

The KAS IDE automatically reorders the EtherCAT nodes and the I/O terminals in the Project Explorer to match the physical device order on the network. The following figures show the Project View before and after associating devices.



### 6.4.7.1.3 Re-Scan and Change Coupler/Slice Associations

Coupler or slice associations must be updated after the following actions.

- Changing the physical network topology by adding, moving, or removing I/O slices attached to an EtherCAT coupler
- Changing the associations between physical and project I/O slices.

To update the associations:

1. Re-scan and re-discover the physical network topology.
2. Click the **Modify** button to expand the Discovered Devices and Modules view.
3. Drag-and-drop the project coupler and/or slice(s) to associate them with the physical couplers/slices.

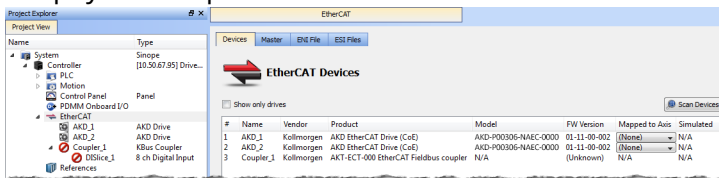
#### ! IMPORTANT

After changing the association of an EtherCAT device, you have to recompile the project and download this new version to save your modifications on the target.

#### 6.4.7.1.3.1 EtherCAT Scan & Association Example 1 - Associating Couplers and I/O Slices

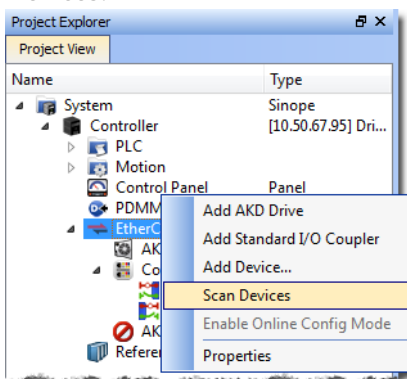
In this example the project has the following configuration:

- Two AKD drives which were created during a previous scan
- A coupler and I/O slice were manually added to the project but are not yet associated with the physical coupler and I/O slice.



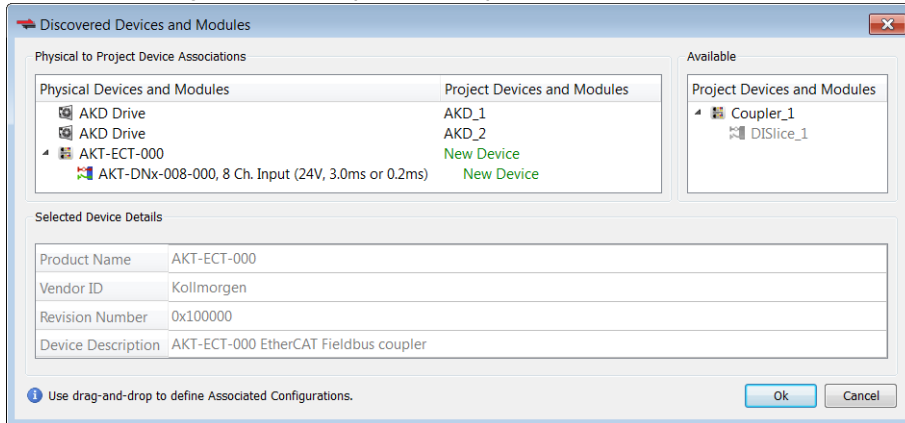
To update the network topology:

1. In the Project Explorer, double-click the EtherCAT node to open its Properties and click the Scan Devices button in the Devices tab. Alternatively, right-click on the EtherCAT node in the Project Explorer, and select Scan Devices.

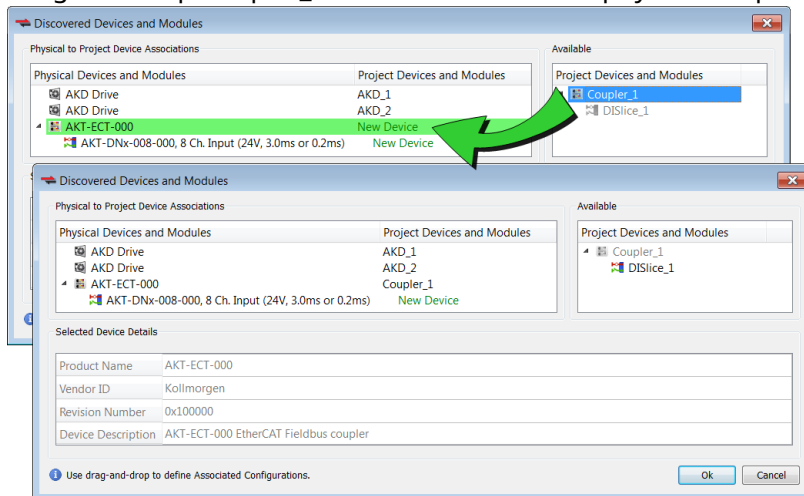


2. The Discovered Devices and Modules dialog box appears after a successful scan.

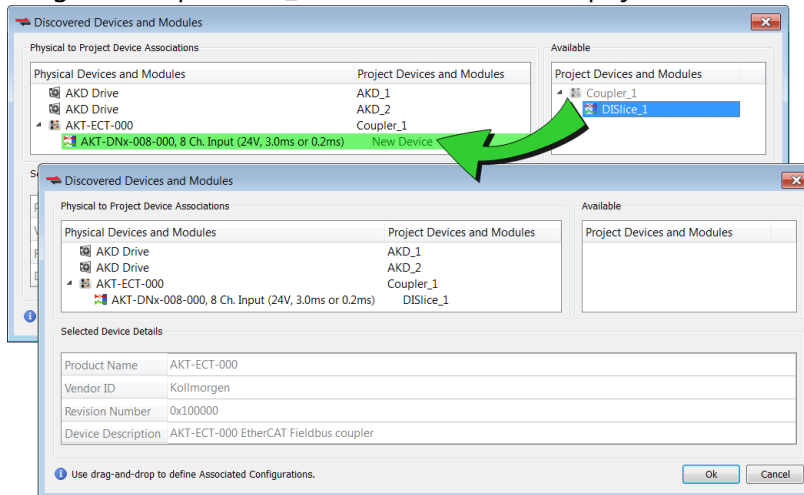
3. Click the Modify button to open the expanded view.



4. To associate the project coupler and I/O slice to the physical coupler and I/O slice:  
 1. Drag-and-drop Coupler\_1 from Available to the physical coupler (AKT-ECT-000).

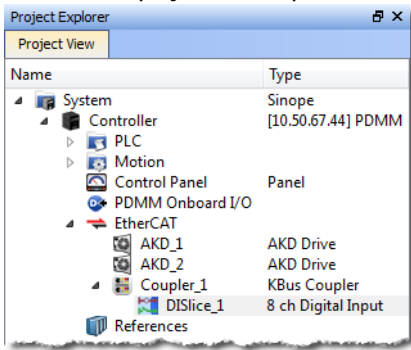


2. Drag-and-drop DISlice\_1 from Available to the physical I/O slice (AKT-DNx-008-000).



5. Click OK to confirm the modified associations.

The KAS IDE updates the EtherCAT nodes for the coupler and I/O slice in the Project Explorer to match the physical coupler and I/O slice.

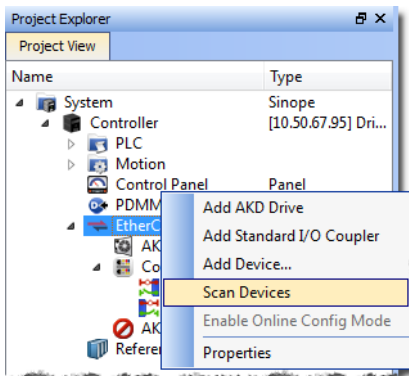


### 6.4.7.1.3.2 EtherCAT Scan & Association Example 2 - Adding Physical I/O Slices

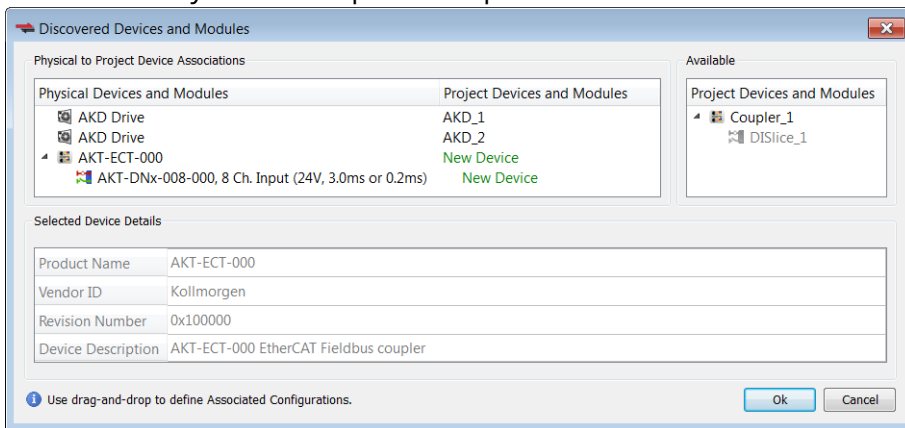
In this example another physical I/O slice is added to the coupler.

To update the network topology:

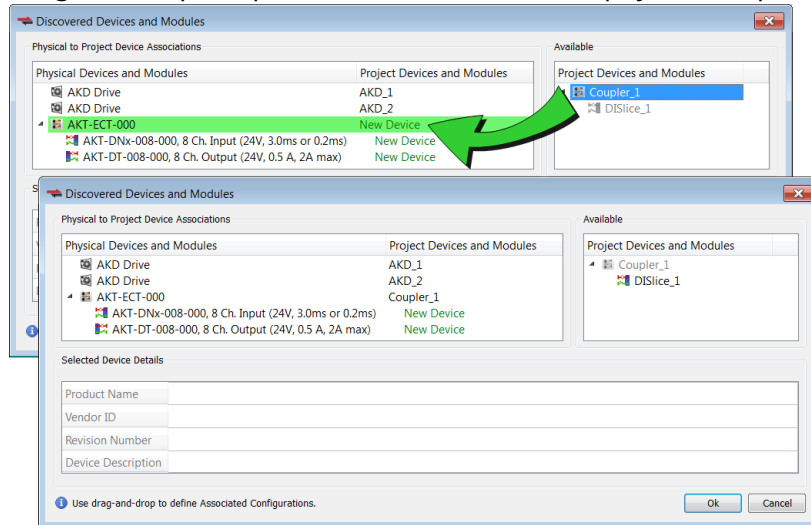
1. In the Project Explorer, double-click the EtherCAT node to open its Properties and click the Scan Devices button in the Devices tab. Alternatively, right-click on the EtherCAT node in the Project Explorer, and select Scan Devices.



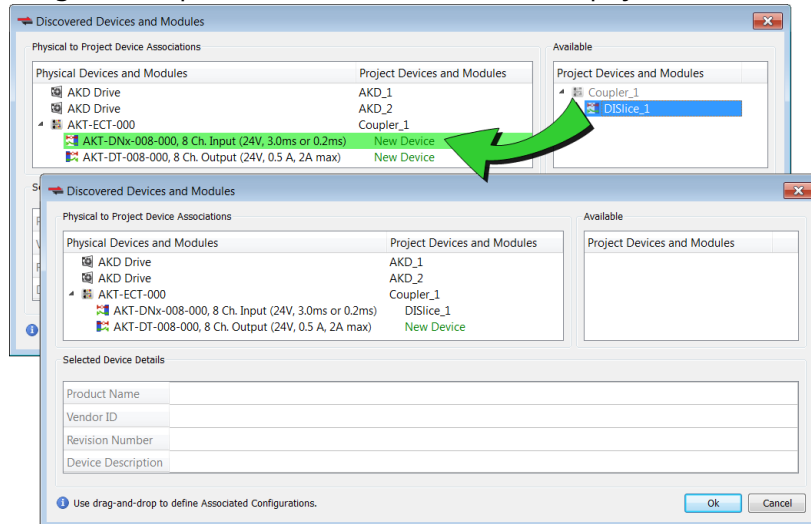
2. The Discovered Devices and Modules dialog box appears after a successful scan.
3. Click the Modify button to open the expanded view.



4. To associate the coupler and I/O slice configurations with the physical coupler and I/O slice:
  1. Drag-and-drop Coupler\_1 from Available to the physical coupler (AKT-ECT-000).

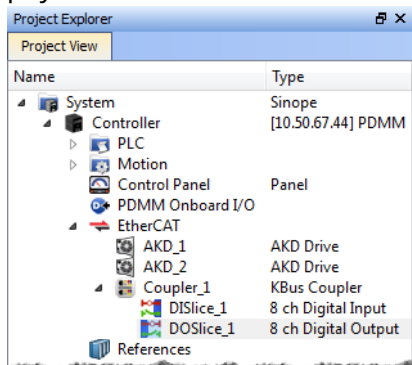


2. Drag-and-drop DISlice\_1 from Available to the physical I/O slice (AKT-DNx-008-000).



5. Click OK to confirm the modified association.

The KAS IDE creates the EtherCAT node for the added I/O slice in the Project Explorer to match the physical I/O slice:



#### 6.4.7.1.4 Re-Scan and Change Device/Module Associations

Device or module associations must be updated after the following actions.

- Changing the physical network topology by adding, moving, or removing modules attached to an EtherCAT modular device.
- Changing the associations between a physical module and a project module.

To update the associations:

1. Re-scan and re-discover the physical network topology.
2. Click the **Modify** button to expand the Discovered Devices and Modules view.
3. Drag-and-drop the project coupler and/or slice(s) to associate them with the physical couplers/slices.

#### ! IMPORTANT

After changing the association of an EtherCAT device, you have to recompile the project and download this new version to save your modifications on the target.

#### NOTE

A MDP device's configurable properties (PDO selections/content, PLC variable maps, and CoE Init-Commands) are determined by the modules under the device. The default values for the device's configuration are determined from the ESI file.

The device's configuration can be modified from the KAS IDE EtherCAT device configuration tabs. Although the project modules cannot be configured directly, they can be associated with Physical modules by dragging-and-dropping them in the Discovered Devices and Modules view.

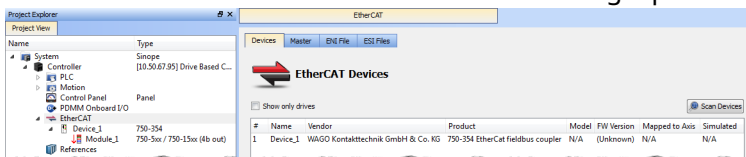
When a project module is associated with a Physical module, its configuration (PDO selections/content, PLC variable maps, and CoE Init-Commands) is automatically updated at the device level. The order of the modules under the device, is determined by order the Physical modules are discovered. The order of CoE Init-Commands also follows the order of the Physical modules. This includes any user specified CoE Init-Commands.

It is recommended to check the CoE Init-Command ordering after changing module associations, to verify it still meets any user-specific ordering requirements.

#### 6.4.7.1.4.1 EtherCAT Scan & Association Example 3 - Add a Module to a Device

In this example a module is added to the device on the physical network:

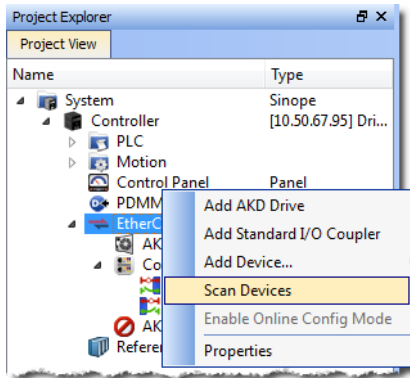
- A device with a module which was created during a previous scan.



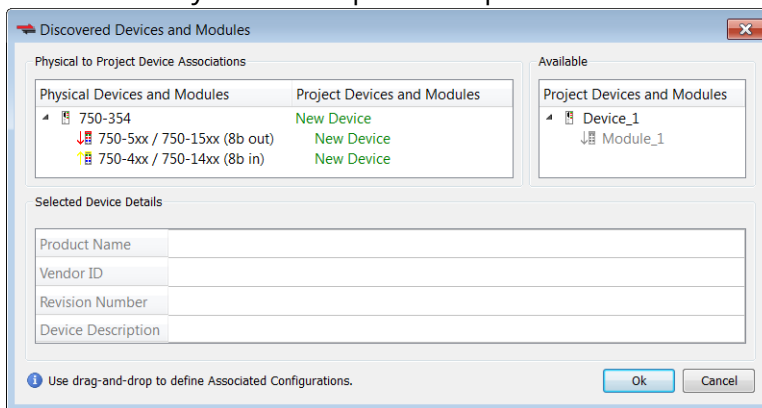
- Another module needs to be added to match the device on the physical network.

To update the network topology:

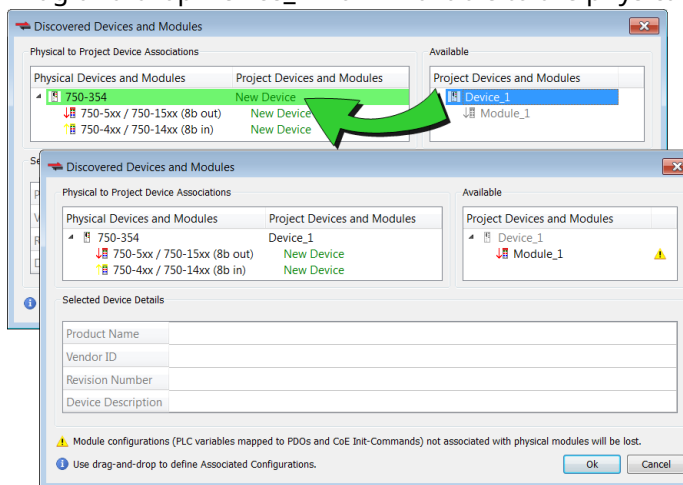
1. In the Project Explorer, double-click the EtherCAT node to open its Properties and click the Scan Devices button in the Devices tab.  
Alternatively, right-click on the EtherCAT node in the Project Explorer, and select Scan Devices.



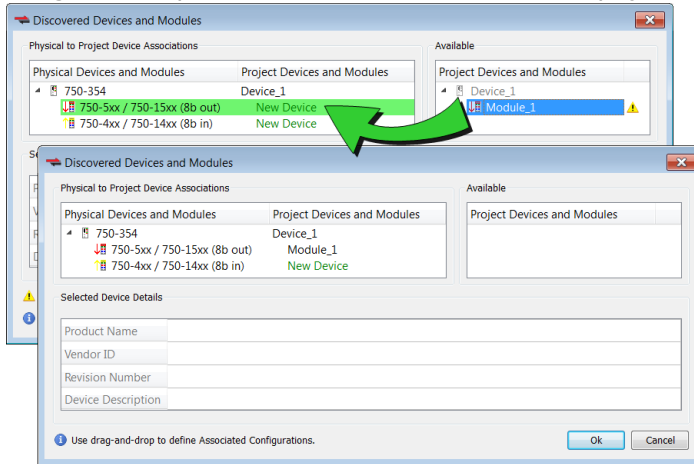
2. The Discovered Devices and Modules dialog box appears after a successful scan.
3. Click the Modify button to open the expanded view.



4. To associate the project device and module to the physical device and module:
  1. Drag-and-drop Device\_1 from Available to the physical device (750-354).

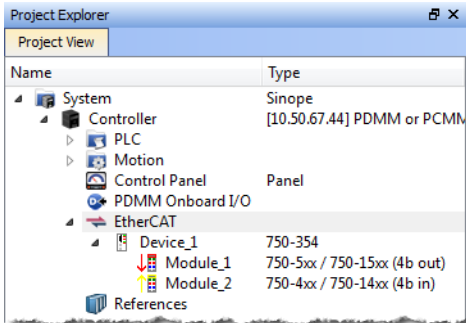


2. Drag-and-drop Module\_1 from Available to the physical module (750-5xx).



5. Click OK to confirm the modified associations.

The KAS IDE updates the EtherCAT nodes for the device and modules in the Project Explorer to match the physical device and modules.

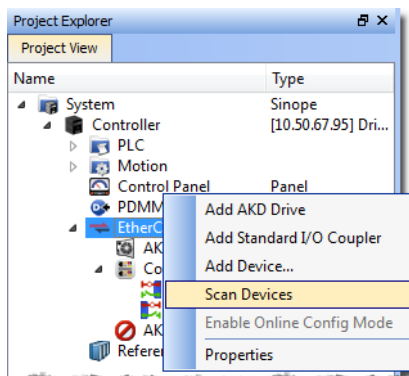


#### 6.4.7.1.4.2 EtherCAT Scan & Association Example 4 - Remove a Module From a Device

In this example a module is removed from the device on the physical network.

To update the network topology:

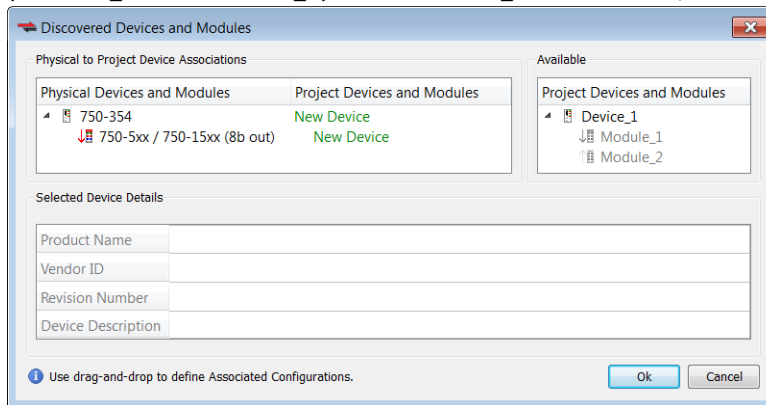
1. In the Project Explorer, double-click the EtherCAT node to open its Properties and click the Scan Devices button in the Devices tab. Alternatively, right-click on the EtherCAT node in the Project Explorer, and select Scan Devices.



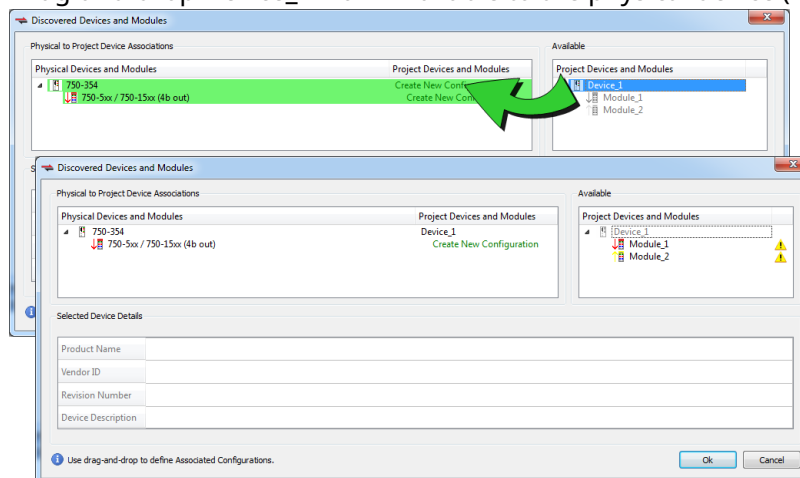
2. The Discovered Devices and Modules dialog box appears after a successful scan.



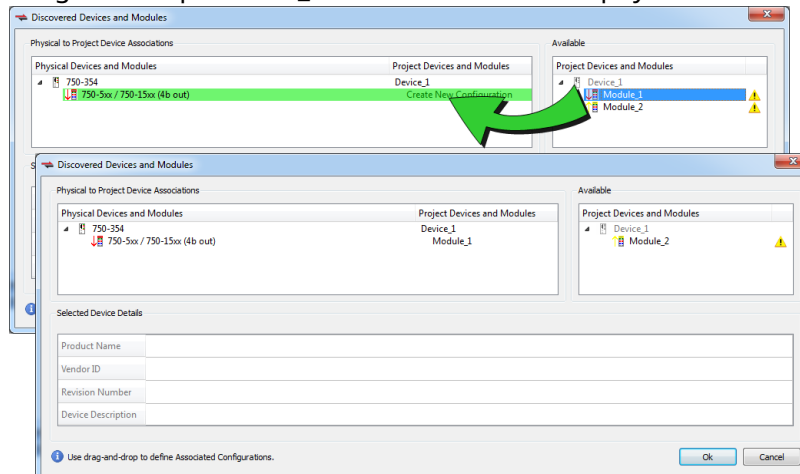
- Click the Modify button to open the expanded view. Note that there are two modules (Module\_1 and Module\_2) under Device\_1 in Available, but there is only one physical module.



- To associate the project device and module to the physical device and module:
  - Drag-and-drop Device\_1 from Available to the physical device (750-354).



- Drag-and-drop Module\_1 from Available to the physical module (750-5xx).

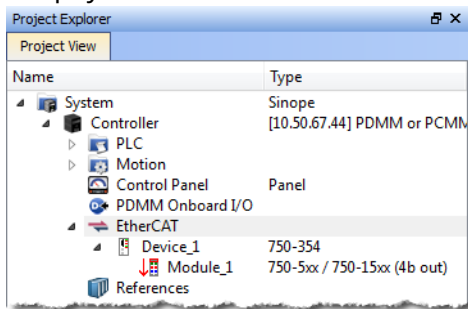


#### NOTE

Module\_2 cannot be associated with a physical module. This is because there are no compatible Physical modules available. Module\_2 and any of its user configuration settings will be lost when the OK is clicked.

- Click OK to confirm the modified configuration. Module\_2 is automatically deleted.

The KAS IDE creates the EtherCAT node for the device and modules in the Project Explorer to match the physical device and modules. Module\_2 is no longer available.



## 6.4.8 Overview: Create Programs

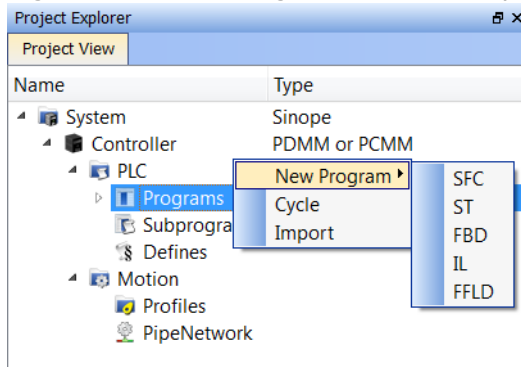
Once the project has been created and the EtherCAT devices have been defined, the program may be created.

### **TIP**

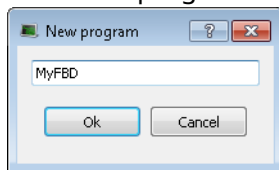
The KAS IDE has several program languages you may select from.

- [Sequential Function Chart \(SFC\)](#)
- [Function Block Diagram \(FBD\)](#)
- [Free Form Ladder Diagram \(FFLD\)](#)
- [Structure Text \(ST\) and Instruction List \(IL\)](#)

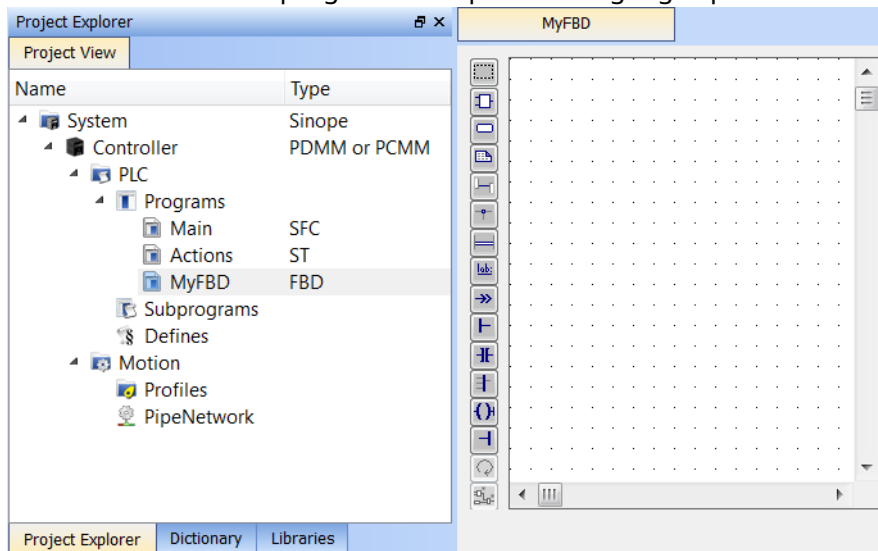
1. Right-click on the Programs item in the project tree and select the language you wish to use.



2. Name the program.



3. Click Ok to create the program and open the language-specific editor.



The empty program is ready for you to add functions and function blocks. Other information you may need to complete your program includes:

- [Overview: Create Variables](#)
- [Overview: Create and/or Call Functions & Function Blocks](#)

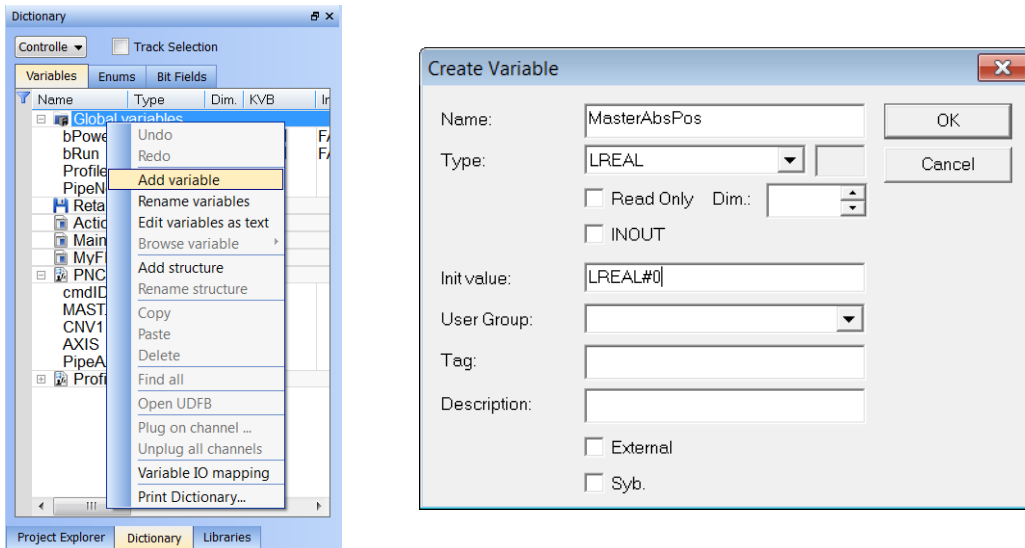
- [Overview: Using Constants](#)
- [Overview: Map Input & Output to Variables](#)

For more in-depth information on the programming languages, see [Create Programs](#) in the [KAS IDE Reference](#) section.

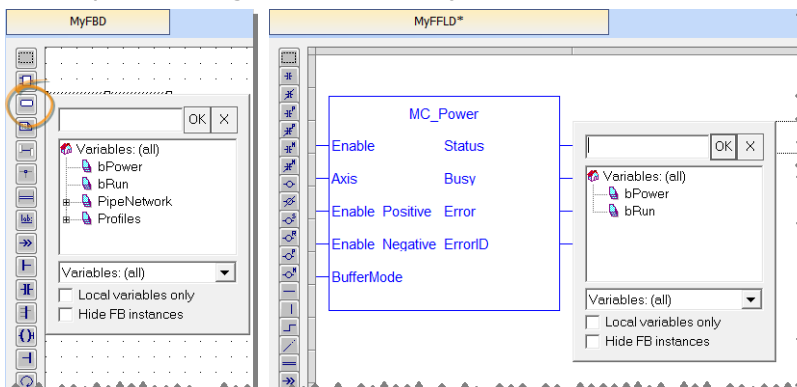
### 6.4.8.1 Overview: Create Variables

Variables may be created in either of two ways:

- From the [Variables tab](#) of the Dictionary. Clicking **Add Variable** opens the **Create Variable** dialog box.



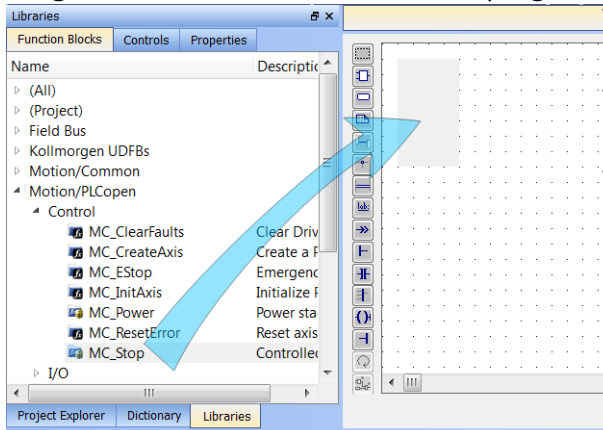
- From the program [Editor](#). Below we see examples of the FBD (click on the Variable button) and FFLD (double-click on the in or out pins of the function) Editors. Enter a name and click OK to open a dialog box that allows you to define the variable.



### 6.4.8.2 Overview: Create and/or Call Functions & Function Blocks

To call a function or function block:

1. Drag an item from the [Libraries](#) into a program

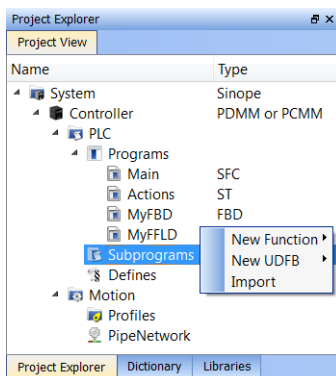


2. Name the new instance of the function or function block

See [Call Functions or Function Blocks](#) for more in-depth information.

To create a function or function block:

1. Declare the function or function block by right-clicking on Subprograms in the Project Tree.



2. Name the function or function block
3. Define the inputs and outputs, and any private variables
4. Call the new function or function block

See [Create Functions and Function Blocks](#) for more in-depth information.

See Also: [Differences Between Functions and Function Blocks](#)

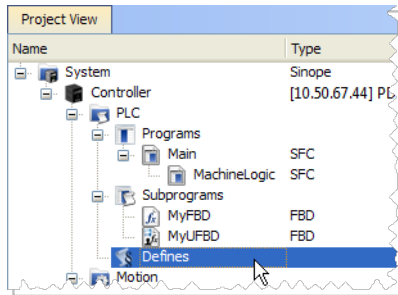
### 6.4.8.3 Overview: Using Constants

Constants, or Defines, exist at Internal, Global, and a Local level.

Level	Scope
<a href="#">Internal Defines</a>	All the projects present on your machine
<a href="#">Global Defines</a>	All the programs within your project. These are user-defined.
<a href="#">Local definitions</a>	Only the current program currently open

How to define constants

1. Double click on the Defines item in the Project Tree.



2. Use the text editor to create the defines using the following syntax.

```
#define Identifier Equivalence (* comments *)
```

See [Use the Defines List](#) for more in-depth information.

### 6.4.8.4 Overview: Map Input & Output to Variables

I/O from the EtherCAT motion bus or the controller is easily mapped to PLC variables through the Project Explorer. The exact process will vary slightly depending on the device.

1. Expand the Controller, and EtherCAT nodes in the Project tree.
2. Double-click on the device or I/O slice entry.
3. Map the variables in any of three ways:
  - Drag and drop variables from the Dictionary.
  - Use the [PLC Variable Creation Wizard](#) for Kollmorgen devices.
  - Directly map/unmap the inputs and outputs using the [PLC Variable Selector](#).

See [Map Input and Output to Variables](#) for more in-depth information.

## 6.4.9 Adding Motion

There are two ways to create motion, depending on the motion engine:

- For Pipe Network, refer to [Design Motion with Pipe Network](#)
- For PLCopen, refer to [Design Motion with PLCopen Axis](#)
- If you are not sure which engine to use, refer to [PipeNetwork](#) or [PLCopen](#)

For high-level discussions about motion, refer to the sections within [Motion Concepts](#).

### 6.4.9.1 Design Motion with Pipe Network

The contents of this section detail how to create and modify a Pipe Network.

#### 6.4.9.1.1 Create the PipeNetwork

To create the PipeNetwork, do as follows:

1. In the Project Explorer, double-click the **PipeNetwork** button to open the graphical [PipeNetwork Editor](#).

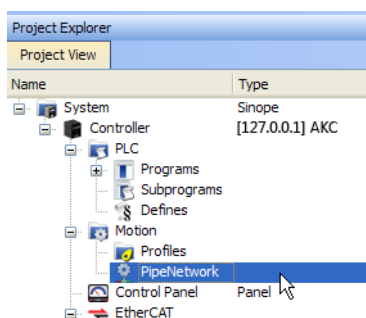
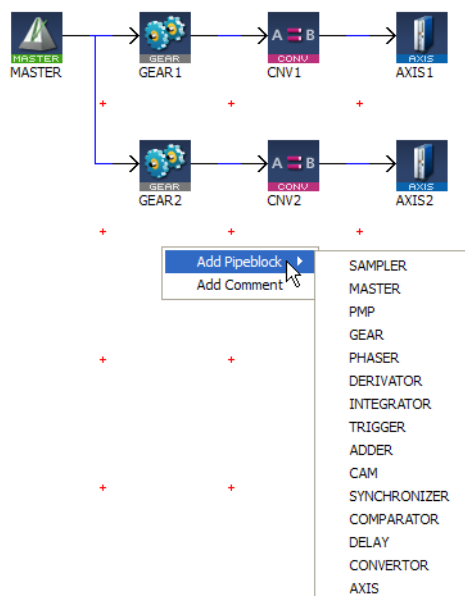


Figure 4-5: PipeNetwork - Open Editor

#### NOTE

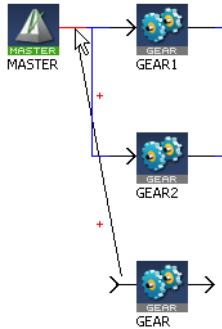
If you have created a project from a template (for instance the standard two-axis template) there is already a PipeNetwork in the editor.

2. To add a new Pipe Block, right-click on the editor's background and select the **Add Pipeblock** command in the menu
3. Choose in the drop-down menu the type of Pipe Block you want to add



**Figure 4-6:** PipeNetwork - Add Pipeblock

- To link the newly created Pipe Block, move the arrow to the corresponding Pipe Block with a drag-and-drop operation



**Figure 4-7:** PipeNetwork - Create a Link

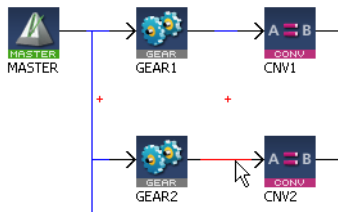
*How to delete a Pipe Block?*

There are two methods.

- Right-click on the Pipe Block and select the **Delete** command in the contextual menu.
- Select the Pipe Block by clicking on it and then press the **Delete** key.

*How to change a link?*

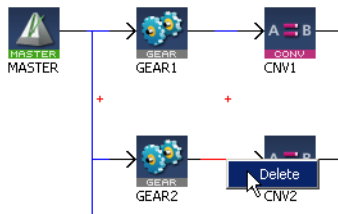
- Select the link so that it becomes Red



**Figure 4-8:** PipeNetwork - Edit a Link

You can either:

- Right-click and select the **Delete** command if you want to remove the link



**Figure 4-9:** PipeNetwork - Delete a Link



- Move the arrow to another Pipe Block with a drag-and-drop operation

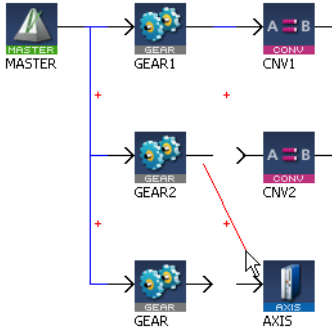


Figure 4-10: PipeNetwork - Move a Link

#### 6.4.9.1.2 Edit Properties of Pipe Blocks

Initial values for Pipe Network blocks are entered in the parameter screen for each block. To get to the parameter screen, right-click on a Pipe Block and select the **Properties** command in the contextual menu.

- Right-click on the Pipe Block and select the **Properties** command in the menu

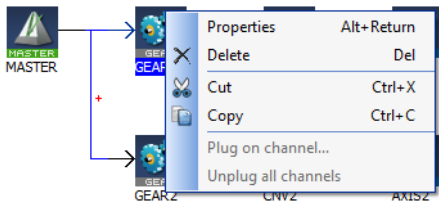


Figure 4-11: Pipe Network - Pipe Block Properties

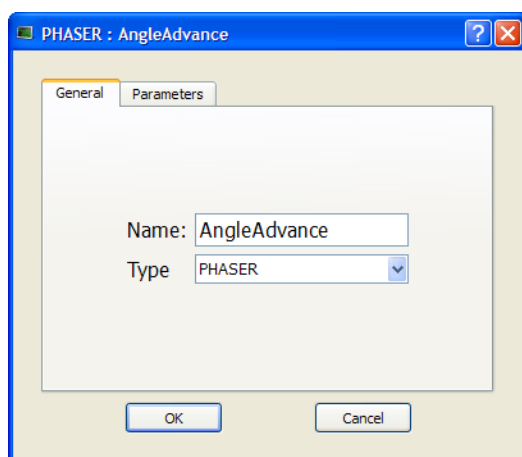
- The **Properties** window can also be opened by selecting the block and pressing **Alt+Enter**.

You can change the name (or even the type of Pipe Block) in the **General** tab.

The **Parameters** tab gives access to properties related to the type of Pipe Block .

See example

In this example, the selected name "AngleAdvance" would be used in the PLC application program for this Pipe Network block.



#### 6.4.9.1.3 Map the Axis to the Drive

To link the axis to an EtherCAT drive, you have to do the mapping as described in [Mapped to Axis](#).

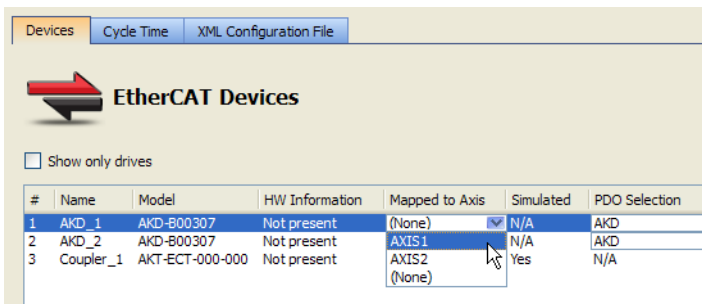


Figure 4-12: Pipe Network - Mapping Axis to Drive

#### 6.4.9.1.4 Add Comments to Pipe Network

To add a comment:

1. Right-click on the Pipe Network editor's background and select the **Add Comment** command in the menu
2. Right-click on the comment opens the contextual menu to let you edit (**Properties** command) or delete the comment

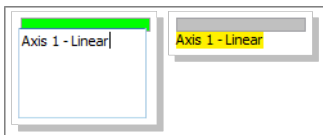


Figure 4-13: Pipe Network comments: editing, and deselected.

Click and drag the gray bar to move the comment.

#### 6.4.9.1.5 Set the Axis Block Position Units

You can set up the position units in the parameter screen of the Axis block.

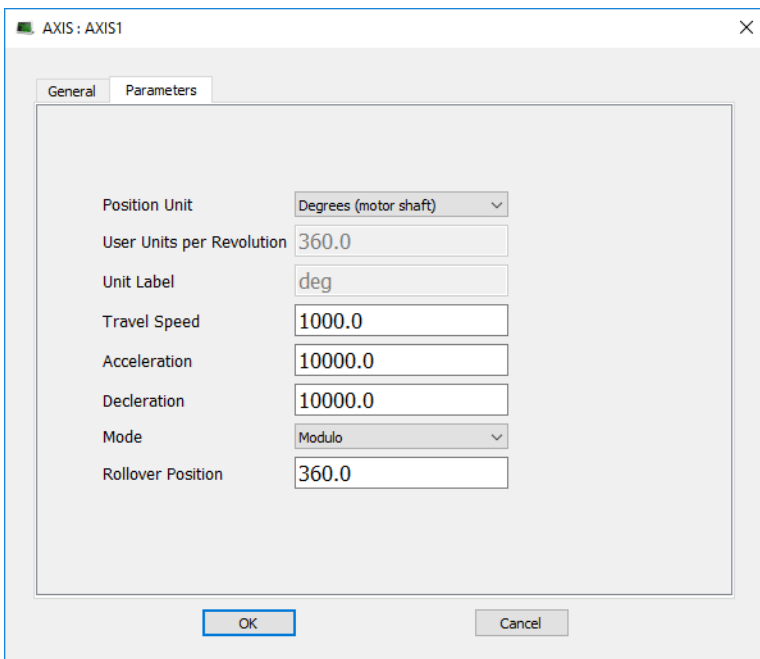


Figure 4-14: Setting Axis Units

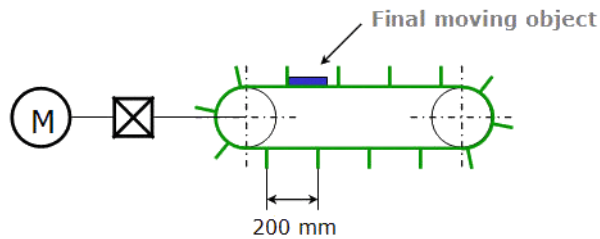
Some guidelines for suitable settings are given below:

- The user unit is adapted for the machine
- The user unit should be meaningful for the user

- The same user unit should be used for all related axes, for simplicity
- The user unit should refer to the position of the final moving object, instead of any intermediate part (e.g. the belt position rather than the position of the axis shaft)

**Speed** is defined in User Units for position / second, e.g. cm/s

**Acceleration** in User Units for position / second<sup>2</sup>, e.g. cm/s<sup>2</sup>

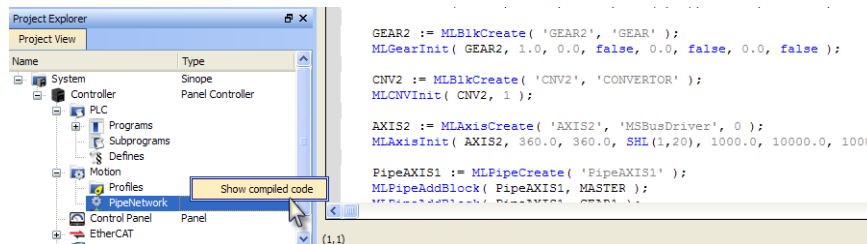


**Figure 4-15: Setting the Units - Example**

In the example above, the belt flaps are about 60 degrees away from each other on the wheel, so 6 flaps would pass on a rotation of the axis. This would correspond to 1200mm/revolution. Because the diagram specifies millimeters, one should select "Custom" from the Position Unit drop down box, enter "1200.0" into the User Units Per Revolution text box, and "mm" box into the Unit label text box.

#### 6.4.9.1.6 Show Pipe Network and Profiles-Generated Code

You can access the code equivalent to the graphical representation with the contextual menu of the Pipe Network item in the Project Explorer as follows:



**Figure 4-16: Display Source Code of the Pipe Network**

The KAS IDE provides a set of Functions and function blocks for each of the Pipe Blocks. These function blocks allow the logic part of the application to control and interact with the motion engine.

#### 6.4.9.1.7 Pipe Network Functions for the PLC

After creating the Pipe Network, the complete project has to be compiled before you can use the Pipe Network in your PLC Programs. Compiling creates a list of Functions that can be used in the PLC Program. These Functions simplify programming by combining the same function block for all axes in the Pipe Network:

Pipe Network Function	Function Blocks included (for 2 axis system)
MLPN_ACTIVATE :	MLPipeAct( PipeAXIS1 ); MLPipeAct( PipeAXIS2 );
MLPN_CONNECT :	MLCNVConnect( CNV1, AXIS1 ); MLCNVConnect( CNV2, AXIS2 );
MLPN_POWER_ON :	MLAxisPower( AXIS1 ); MLAxisPower( AXIS2 );

Pipe Network Function	Function Blocks included (for 2 axis system)
MLPN_POWER_OFF :	MLAxisPowerOFF( AXIS1 ); MLAxisPowerOFF( AXIS2 );
MLPN_DEACTIVATE :	MLPipeDeact( PipeAXIS1 ); MLPipeDeact( PipeAXIS2 );

For more details on all constant definitions related to Pipe Network, [Use the Defines List](#)

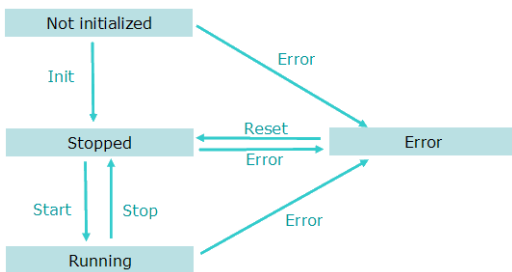
**TIP**

To see how these functions are used, open a project, go to the [Project Explorer](#), right-click on PipeNetwork and select the **Show compiled Code** command

**6.4.9.1.8 Initialize and Start up a Pipe Network**

See *Motion State Machine*

The Motion State Machine is driven by the IEC 61131-3 application with the help of dedicated function blocks.



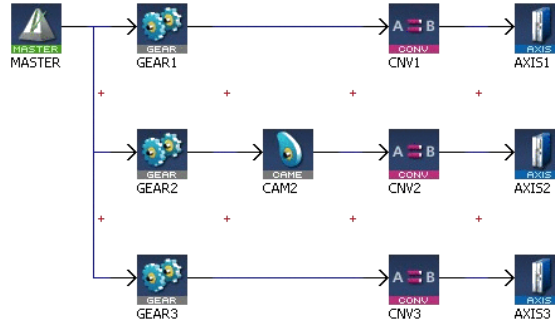
**Figure 4-17: Motion State Machine**

Each arrow represents a transition from one State to another one.

To start-up a Pipe Network in your IEC 61131-3 application program, you have to perform the following steps with their respective functions:

Step	ML function blocks	Description
<b>Motion Init</b>	MLMotionInit	Initialization of the Motion is done with this dedicated function Set the Motion engine update rate. Wait for acknowledgement: MLMotionStatus() = MLSTATUS_INITIALISED to continue program operation
<b>Create Cam Profiles</b>	Profiles(MLPR_CREATE_PROFILES);	Create Cam Profiles from cam files
<b>Create Pipe Network</b>	PipeNetwork(MLPN_CREATE_OBJECTS);	
<b>Motion Start</b>	MLMotionStart	Starts the motion engine, motion bus driver, and initializes EtherCAT network to operational mode, then waits for acknowledgement: MLMotionStatus() = MLSTATUS_RUNNING to continue program operation
<b>Power on all axes</b>	PipeNetwork(MLPN_POWER_ON);	

Step	ML function blocks	Description
Activate the pipes	PipeNetwork (MLPN_ACTIVATE);	
Connect the axes to the pipes	PipeNetwork (MLPN_CONNECT);	For example: in the following Pipe Network this function connects the Converter blocks (CNV1, CNV2 and CNV3) to the Axis blocks



#### 6.4.9.1.9 How the Pipe Network interacts with PLC programs

Each Pipe Block is supported by several ML function blocks in the function block Library. As soon as you add a Pipe Block, it is included as well in the Variable Editor.

- Add the FB into your program (see procedure [here](#))
- Select the variable to update

```

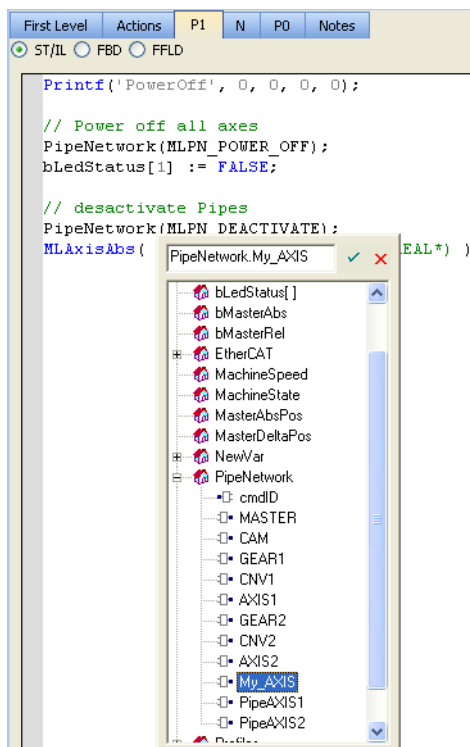
First Level  Actions  P1  N  PD  Notes
ST/IL  FBD  FFLD
Printf('PowerOff', 0, 0, 0, 0);

// Power off all axes
PipeNetwork(MLPN_POWER_OFF);
bLedStatus[1] := FALSE;

// deactivate Pipes
PipeNetwork(MLPN_DEACTIVATE);
MLAxisAbs( ID(*DINT*), Position(*LREAL*) )

```

- Press **CTRL+SPACE** to open the [Variable Editor](#)
- Expand the PipeNetwork node and select the name of the Pipe Block in the list (all the Pipe Blocks created in the Pipe Network are listed)



Then your ST instruction is updated

```
// deactivate Pipes
PipeNetwork(MLPN_DEACTIVATE);
MLAxisAbs( PipeNetwork.My_AXIS, Position(*LREAL*) )
```

**NOTE**

When you add a new Block in the Pipe Network, you first need to compile your project to make the block visible in the list of items.

- Click the  icon to update your code

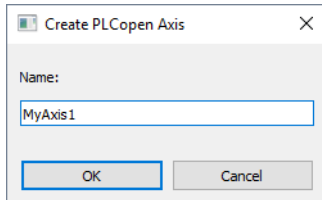
### 6.4.9.2 Design Motion with PLCopen Axis

This section explains how to modify an existing PLCopen Axis, and how to create a new one.

#### 6.4.9.2.1 Create PLCopen Axis

To create a new PLCopen axis, follow these steps:

1. In the Project Explorer, right-click on the **Axes** folder under **PLCopen** and select the **New Axis** command in the menu
2. Give the new axis a name the Create PLCopen Axis dialog.



3. Fill in the axis properties by right-clicking on the new axis. See [Modify PLCopen Axis](#) for details.
4. Compile the project to generate the code for the PLCopen axis reference(s).

Each axis will show in the variable type **PLCopenCode**, which is used by the PLCopen Global variable.

5. The **PLCopen.XXXXXX\_ref** (MyAxis1 in the example below) is an instance of an that can be used in your PLC programs.

```
IF bMotionEngineStarted THEN //When motion engine is running, mon
  Inst_MC_ReadActPos( TRUE, PLCopen.
```

```
IF bMotionEngineStarted THEN //When motion engine is running, mon
  Inst_MC_ReadActPos( TRUE, PLCopen.MyAxis1_ref );
```

Name	Type	Dim.
CloseButton	BOOL	
EStopButton	BOOL	
OpenButton	BOOL	
Ready	BOOL	
ResetButton	BOOL	
StartMove	BOOL	
ActPos1	LREAL	
ActPos2	LREAL	
PLCopen	PLCopenCode	
Profiles	ProfilesCode	

#### **TIP**

The PLCopen axis Name and Number in the PLCopen Axis Data dialog, are automatically applied to the PLCopen instance under Global variables in the Dictionary when the project is compiled.

#### 6.4.9.2.2 Modify PLCopen Axis

A PLCopen axis can be modified by using the PLCopen Axis Data dialog. To display this dialog you can:

- Double-click on a PLCopen axis in the [Project Explorer](#)
- Right-click the PLCopen axis in the project manager and select **Properties** in the menu as shown below

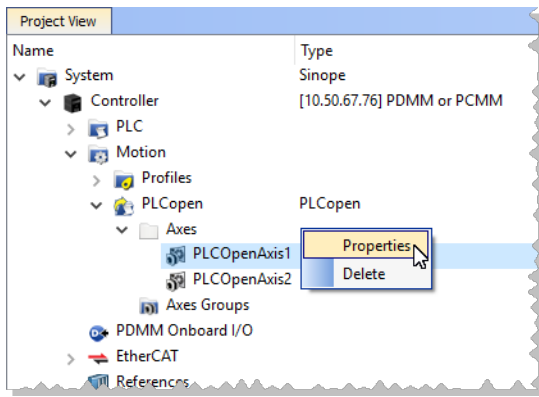


Figure 4-18: PLCopen Axis Context Menu

The PLCopen Axis Data dialog is displayed as follows:

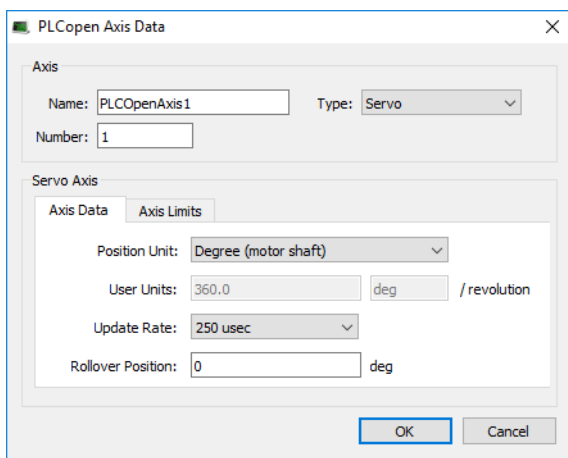
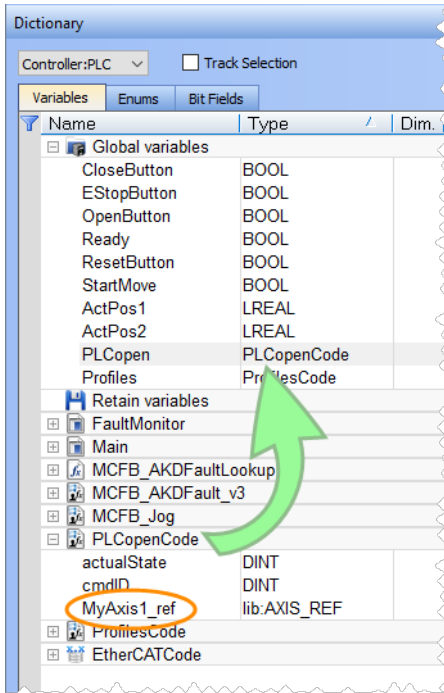


Figure 4-19: PLCopen Axis Data Dialog

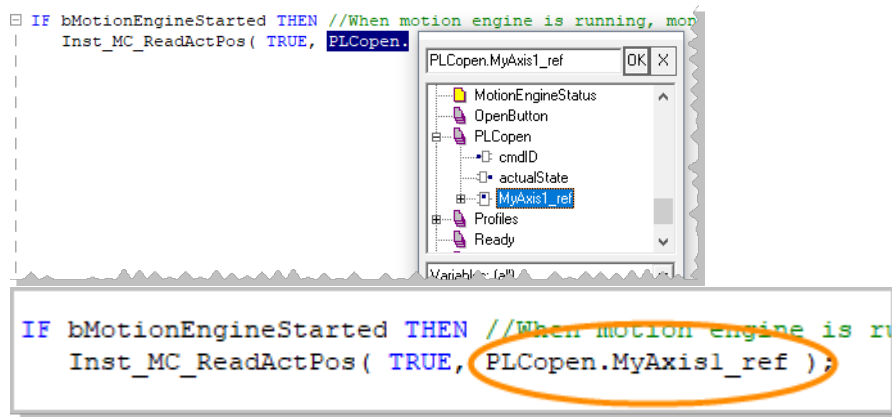


### 6.4.9.2.2.1 About Axis Name and Number

The PLCopen Axis Name and Number in the PLCopen Axis Data dialog, are automatically applied to the PLCopen instance of an AXIS\_REF structure under Global variables in the Dictionary when the project is compiled.



The PLCopen.XXXXXX\_ref (MyAxis1 in the example below) instance can be used in your PLC programs.



#### 6.4.9.2.2.2 Common Axis Parameters

Three types of axes are available: *Servo/Stepper*, *Digitizing* and *Virtual*. All types have common parameters related to an axis.

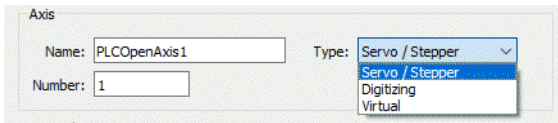


Figure 4-20: PLCOpen Axis Parameters

Parameter	Description
<b>Name</b>	The user-defined name of the axis. The name can consist of 1-16 alphanumeric characters. Spaces are not allowed in the name. The Axis Name identifies the axis displayed in the Project Tree and in the Simulator.
<b>Type</b>	<p>With a <i>Servo/Stepper</i> axis, commands are sent to the axis and feedback is read from the axis. After selecting this option you will need to define the <a href="#">Axis Data Parameters</a> and <a href="#">Axis Limits Parameters</a>.</p> <p>A <i>Digitizing</i> axis contains position feedback data only, no motion can be commanded. Digitizing axes can use extra or secondary feedback from the Drive or from a third-party EtherCAT feedback device. Selecting a third-party EtherCAT feedback device will require you to enter EtherCAT Bus data (see <a href="#">Digitizing Axis Parameters</a>) which will vary depending upon your EtherCAT devices. Also see <a href="#">How To Create a Digitizing Axis</a> for specifics on different methods.</p> <p>A <i>Virtual</i> axis is an axis with no feedback or drive hardware. The feedback for a virtual axis is automatically generated from the command position. There is no limit to the number of virtual axes that may be used in an application.</p>
<b>Number</b>	The axis number (1-256) specifies the axis for PLCOpen motion function blocks.

### 6.4.9.2.2.3 Digitizing Axis Parameters

When creating a digitizing axis you must set the [Common Axis Parameters](#) (name and number), the [EtherCAT Bus parameters](#), and the [Axis Data Parameters](#). The Digitizing axis type has some additional parameters to define for third-party EtherCAT feedback devices.

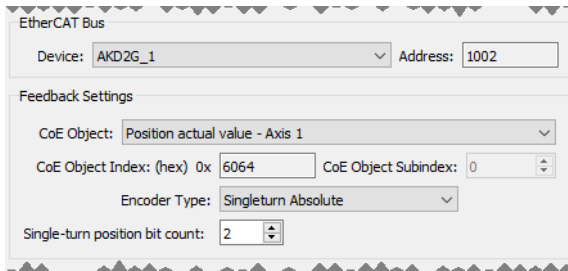



Figure 4-21: PLCopen Axis - Bus Parameters

The EtherCAT device parameters are:

Parameter	Description
<b>Device</b>	Select the device that will provide the digitizing axis feedback. The list is populated by the devices defined in the EtherCAT section of the Project View. 
<b>Address</b>	The digitizing <b>Address</b> is the EtherCAT address of the device in the network. It is the 4-digit node address of the servo drive on the bus. This address is required to assign a digitizing axis to an EtherCAT node that already has a servo axis assignment. The address for the EtherCAT slaves are defined as follows. <ul style="list-style-type: none"> <li>• first device: 1001</li> <li>• second: 1002</li> <li>• third: 1003</li> <li>• etc.</li> </ul>
<b>CoE Object</b>	Select the item that will be used to read the feedback.
<b>CoE Object Index</b>	The CoE object index of the feedback device. This is a hexadecimal value.
<b>CoE Object Sub-Index</b>	The sub-index of the feedback device. This is a decimal value,
<b>Encoder Type</b>	Select <b>Singleturn Absolute</b> or <b>Multiturn Absolute</b> .
<b>Single-turn position bit count</b>	Typically 12, 16, 20, or 32-bits. A value of "0" indicates that rollover is not applicable.

#### NOTE

**AKD2G:** Digitizing axes mapped to an AKD2G device are particularly easy to work with.

- The PDO 0x1A01 is automatically populated with available feedbacks.

**PDO Content**

0x1A01 Contents: Size : 8 bytes

Index	Subindex	Object Name	Size [bit]
0x60E4	1	1st additional position act...	32
0x60E4	2	2nd additional position ac...	32

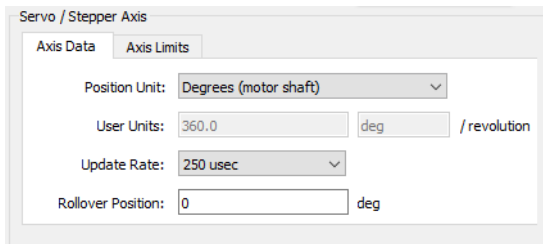
- When a feedback object 0x60E4:1 to 0x60E4:2 is selected, the **Encoder Resolution** is automatically set to 31 bits and the **Feedback Units** field is initialized to the 2<sup>31</sup> value.
- The **Feedback Units** text box is automatically adjusted to 2(Encoder Resolution) when the **Encoder Resolution** is changed.
- Feedback units will be configured on the AKD2G device based on the resolution specified for the digitizing axis. This is done using init commands during project compilation.

 **TIP**

For more information see [Working With A Digitizing Axis in PLCopen](#).

#### 6.4.9.2.2.4 Axis Data Parameters

If a *Servo/Stepper* axis is selected, two tabs are available: Axis Data and Axis Limits. If a Digitizing axis is selected, only the Axis Data tab is available. By default, new axes will be defined using degrees as the measurement unit.



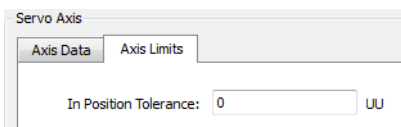
**Figure 4-22:** Servo/Stepper Axis - Axis Data

The Servo/Stepper Axis - Axis Data parameters are:

Parameter	Description
<b>Position Unit</b>	<p>This field sets the units (see UNIT Parameters ) used by the drive axis. When the axis is defined as an AKD2G the controller applies the user units value and label to the drive. These user units will be applied for both Op-mode (running application) and Pre-Op mode (online config) when the EtherCAT network is started up.</p> <ul style="list-style-type: none"> <li>• Counts 16 bit</li> <li>• Degree</li> <li>• Radian</li> <li>• Custom - Selecting Custom allows you to specify the User Units value.</li> </ul>
<b>User Units / Revolution</b>	<p>The value used to define an axis position or distance for an axis. The default values are:</p> <p>Counts: 65,536 / revolution</p> <p>Radian: 6.28318530717 (or <math>2\pi</math>) / revolution</p> <p>Degree: 360 / revolution</p> <p>When the Unit is set to Custom you may enter your own value and units name, such as <i>gradians, centimeters, or inches</i>.</p>

Parameter	Description
<b>Update Rate</b>	<p>The rate at which the axis's feedback is read and a new command position is generated.</p> <p>The choices are:</p> <ul style="list-style-type: none"> <li>• 250 µsec</li> <li>• 500 µsec</li> <li>• 1 msec</li> <li>• 2 msec</li> <li>• 4 msec</li> <li>• 8 msec</li> </ul> <p>This rate can be slower or equal to the <a href="#">EtherCAT Cycle Time</a>. The EtherCAT Cycle Time specifies the rate at which data is transferred between the control and the drives. The axis Update Rate is the rate at which the PLCopen code reads the feedback, runs its interpolation, and generates a new command position. By allowing some axes to run at a slower rate and staggering the updates on which these axes are interpolated, more axes and/or quicker execution times can be achieved since every axis does not have to be interpolated every update.</p> <p>If you select an axis Update Rate which is faster than the EtherCAT Cycle Time, the axis is set to run at the EtherCAT Cycle Time.</p>
<b>Rollover Position</b>	<p>The value at which the axis position rollovers to zero. Rollover Position is specified in User Units.</p> <p>For example:</p> <p>If the rollover position is 1000, the axis position counts up from 0 to 999 and then rollover back to 0. In the reverse direction, the axis position counts down to 0 and then rollover to 999.</p> <p>If Rollover Position is 0, no rollover occurs. Axis positions become negative values when counting down below 0.</p>

#### 6.4.9.2.2.5 Axis Limits Parameters



**Figure 4-23:** Servo Axis - Axis Limits

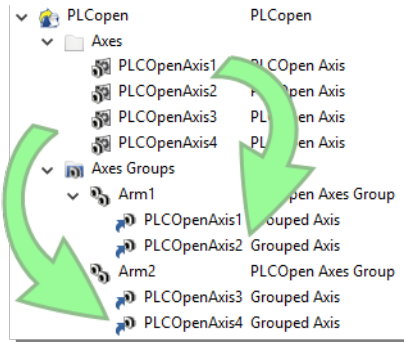
The Servo Axis - Axis Limits parameters are:

Parameter	Description
<b>In Position Tolerance</b>	The maximum distance between the axis's actual position and its commanded endpoint for the axis to be considered "in position". The In-Position Tolerance is specified in User Units.

### 6.4.9.2.3 Create PLCOpen Axes Groups

If your project utilizes coordinated motion and/or robotic transformations, you will need to create one or more Axes Groups from within the Project Explorer.

Axes Groups are collections of related axes defined by the data type `AXIS_GROUP_REF`, typically used by coordinated motion and function blocks such as [MC\\_SetKinTra](#). Right-click on this section to create a new Axes Group. Groups are populated by dragging an existing axis from the above section into the group.



Axes Groups are easily modified by dragging the axes between or within groups. Right-clicking on a group lets you **Rename** or **Delete** the group.

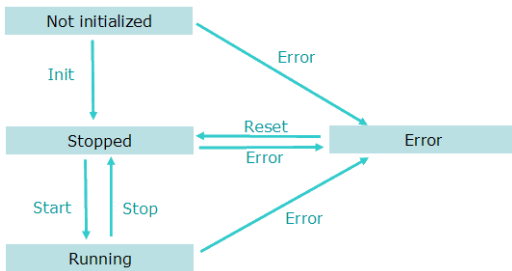
#### See Also:

- [MC\\_KIN\\_REF Structure](#)
- [MC\\_SetKinTra](#)
- [What are Axes Groups?](#)

### 6.4.9.2.4 Initialize and Start PLCopen

See *Motion State Machine*

The Motion State Machine is driven by the IEC 61131-3 application with the help of dedicated function blocks.



**Figure 4-24:** Motion State Machine

Each arrow represents a transition from one State to another one.

To start the PLCopen engine in your IEC 61131-3 application program, you have to perform the following steps with their respective functions:

Step	Function Calls	Description
<b>Motion Init</b>	MLMotionInit	Initialization of the Motion is done with this dedicated function Set the Motion engine update rate. Wait for acknowledgement: MLMotionStatus() = MLSTATUS_INITIALISED to continue program operation
<b>Create Cam Profiles</b>	Profiles(MLPR_CREATE_PROFILES);	Create Cam Profiles from cam files
<b>Create/Initialize PLCopen Axes</b>	PLCopen (0);	Create and initialize PLCopen axes from the PLCopenCode attached to the PLCopen node in the Project tree. To view the PLCopenCode, right click on the <b>PLCopen</b> node and select <b>Show Compiled Code</b> .
<b>Motion Start</b>	MLMotionStart	Starts the motion engine, motion bus driver, and initializes EtherCAT network to operational mode, then waits for acknowledgement: MLMotionStatus() = MLSTATUS_RUNNING to continue program operation

### 6.4.9.3 Design Coordinated Motion

This section discusses how to create a coordinated motion application, including adding coordinated motion to existing applications.

For more information on Coordinate Motion an the associated functions and function blocks see:

- [Coordinated Motion](#) in the Advanced Topics section
- [Coordinated Motion Function Blocks](#)

#### 6.4.9.3.1 Create a Linear or Circular Coordinated Motion Application

A Coordinated Motion application can be created in one of two ways:

- Use a Coordinated Motion template to create a new application. Two Coordinated Motion templates are currently available.



- The first template controls two PLCopen axes in coordinated motion.
- The second template controls two PLCopen axes in coordinated motion plus a third independent Pipe Network axis.
- Modify an existing application to include coordinated motion functions. When modifying an existing application, axes need to be grouped to define the axes that will be active when performing coordinated motion on that group. More information about Axes Groups can be found in the section [What are Axes Groups?](#)

**NOTE**

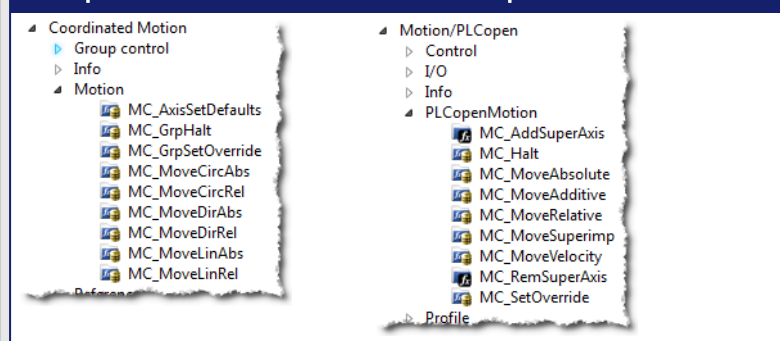
Coordinated motion can only be performed with PLCopen axes. Pipe Network axes do not support this feature, although Pipe Network axes can be moved independently from coordinated motion groups. Any synchronization between coordinated motion and Pipe Network axes must be performed by the PLC application.

Related axes are "grouped" in an axes group. Coordinated motion is then performed on an axes group. For more information see [What are Axes Groups?](#)

**TIP**

There are two vital concepts to remember when making interpolated motion.

- Interpolated motion requires creating a motion group that results in a second group coordinate system.
- Group coordinate system positions are only affected by group motion, and non-group coordinate system positions are only affected by non-group motion.

**Group motion commands Non-Group motion commands****NOTE**

Typically axes that become part of a motion group are first homed using non-group function blocks to establish a home or starting position for the group motion.

Typically, the following set of function blocks should be called before executing coordinated motion.

1. Call **MLMotionInit** (BasePeriod) to initialize the motion engine. Base period is specified in microseconds.

```
MLMotionInit(1000.0); // 1000 µSec -> 1 mSec
```

2. Call **MC\_CreateAxesGrp** (Enable, GroupName, UpdateRate, MaxNumberOfAxes, AxesGroupRef) to create a Coordinated Motion Axes Group

**NOTE**

MC\_CreateAxesGrp needs to be called between MLMotionInit() and MLMotionStart().

```
Inst_MC_CreateAxesGrp(TRUE, 'GROUP1', 6, 2, Group1_ref);
```

In the example above, the axes group name is 'GROUP1', the update rate is 1 mSec (specified by '6') and the maximum number of axes that can be added to the group is 2. The group reference variable 'Group1\_ref' will be used in future coordinated motion function block calls to reference this newly created group.

3. Call [MC\\_InitAxesGrp](#) (Enable, AxesGroup, VelLimit, AccLimit, DecLimit, JerkLimit) to initialize the path limits for velocity, acceleration, deceleration, and jerk.

```
Inst_MC_InitAxesGrp(TRUE, Group1_ref, 100.0, 300.0, 300.0, 1000.0);
```

In the example above, the kinematic limits for axes group 'Group1\_ref' will be set. The velocity limit will be set to 100.0 user units/second, acceleration and deceleration limits will be set to 300.0 user units/second<sup>2</sup> and jerk will be set to 1000.0 user units per second<sup>3</sup> (Jerk will be supported in a future release).

4. Call [MC\\_CreatePLCAxis](#) (AxisName, BusInterface, BusAddress, AxisNumber, AxisType, UserUnits, FeedbackUnits, Rollover, UpdateRate) to create a Coordinated Motion Axis. This function needs to be called for each Coordinated Motion Axis wanted in the application.

#### NOTE

MC\_CreatePLCAxis needs to be called between MLMotionInit() and MLMotionStart().

```
Inst_MC_CreateAxis(TRUE, 'CoordAxis1', 'EtherCATDriver', 1001,
CoordAxis1_AxisNum, 0, 360, 1048576, 0, 6);
Inst_MC_CreateAxis(TRUE, 'CoordAxis2', 'EtherCATDriver', 1002,
CoordAxis2_AxisNum, 0, 360, 1048576, 0, 6);
```

In the example above:

- Two axes are created and are named 'CoordAxis1' and 'CoordAxis2'.
  - The bus interface for both is 'EtherCATDriver'.
  - The address of the drive on the bus is 1001 and 1002.
  - The axis numbers are set with variables CoordAxis1\_AxisNum and CoordAxis2\_AxisNum which is set to an integer value between 1 and 256. Each axis number is unique.
  - The axis type for both, '0', indicates a servo axis.
  - The user units are 360, which is the 'user unit' portion of the 'user unit/feedback' ratio.
  - The feedback units are 1048576, which is the 'feedback' portion of the 'user unit/feedback' ratio.
  - The rollover position for both, '0' indicates no rollover.
  - The update rate for both, '6', indicates a 1mSec update rate.
5. Call [MLMotionStart](#) () to start the Motion and the motion bus driver. This also initializes the EtherCAT network to operational mode.

```
MLMotionStart();
```

6. Call [MC\\_AddAxisToGrp](#) (Execute, AxesGroup, Axis, IdentInGroup) for each axis to be added to the group.

```
Inst_MC_AddAxisToGrp(TRUE, Group1_ref, CoordAxis1_ref, 0);
Inst_MC_AddAxisToGrp(TRUE, Group1_ref, CoordAxis2_ref, 1);
```

In the example above, we are adding two axes, `CoordAxis1` and `CoordAxis2`, to the group referenced by `'Group1_ref'`. The axes are stored in the `IdentInGroup` positions 0 and 1. Note that when the group was created, it was specified that no more than 2 axes will be part of this group. Therefore, valid `IdentInGroup` locations are 0 and 1.

7. Call [MC\\_Power](#) (Enable, Axis, EnablePositive, EnableNegative, BufferMode) for each Coordinated Motion Axis to enable the drive and close the servo loop.

```
Inst_MC_Power1(TRUE, CoordAxis1_ref, TRUE, TRUE, 0);
Inst_MC_Power2(TRUE, CoordAxis2_ref, TRUE, TRUE, 0);
```

In the example above, drives `CoordAxis1_ref` and `CoordAxis2_ref` will be enabled and the position loop will be closed. Note that parameters `'TRUE, TRUE, 0'` are place holders for future use and are not currently used.

8. Call [MC\\_GrpEnable](#) (Execute, AxesGroup) to change the state of the Coordinated Motion Axis Group from `GroupDisabled` to `GroupStandby` and allow motion to be performed on the group.

```
Inst_MC_GrpEnable(TRUE, Group1_ref);
```

In the example above, `'Group1_ref'` state will be changed from `GroupDisabled` to `GroupStandby`. The group must be in `GroupStandby` in order to perform motion.

9. For the examples that follow, we want to set the current location of the axes in the group to 0, 0. This can be done by calling [MC\\_GrpSetPos](#) (Execute, AxesGroup, Position[], Relative, CoordSystem, BufferMode)

```
PosAbs[1] := 0;
PosAbs[2] := 0;
Inst_MC_GrpSetPos(TRUE, Group1_ref, PosAbs, 0, MC_COORDINATE_SYSTEM_ACS, 0);
```

In the example above, the axis positions of `'Group1_ref'` will be set to 0, 0. `'PosAbs'` specifies the position for each axis in the group. `'Relative'` input, `'0'`, uses `'PosAbs'` to set the absolute position. The coordinate system is set to `ACS`. The buffer mode, `'0'`, is a placeholder for future use and is not currently used.

#### NOTE

No motion will be performed when this function block is executed.

10. *Optional:* To Add more axes to the group, modify the above code in the following way:
  - In Step 2: Update the `MaxNumberOfAxes` input argument so that the group can handle the desired number of axes.
  - In Step 4: Create the additional axes that will added to the group.
  - In Step 6: Add the additional axes to the group.
  - In Step 7: Power on the additional axes.
  - In Step 9: You will need to increase the size of the `PosAbs` array so it matches the number you used in step 2, and set the position of the additional axes to zero.

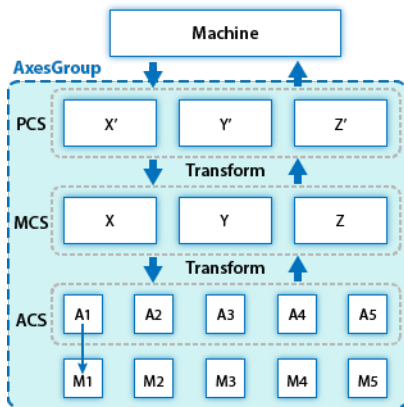
After the above function calls have been made, we can start coordinated motion moves.

[Performing a Linear Move](#)

[Performing a Circular Move](#)

#### 6.4.9.3.1.1 What are Axes Groups?

Related axes are grouped in an Axes Group to support interpolation. Axes Groups are accessed via the type AXES\_GROUP\_REF. The following image shows the relationships between the different CSs and groups.



**Figure 4-25:** Overview of AxesGroup

The Axes Group, shown in blue above, provides the interface to the user of the group of axes. To access the relevant coordinate system, the relevant function blocks have an input Coordinate System which supports the three levels ACS, MCS, and PCS.

Parameters in the AxesGroupRef can include remaining time and remaining distance before target position (or velocity or equal) is reached.

#### 6.4.9.3.1.2 Performing a Linear Move

Linear moves can be programmed using absolute or relative positions using the following function blocks:

- **MC\_MoveLinAbs** which commands interpolated linear movement on an axes group to the specified absolute positions.
- **MC\_MoveLinRel** which commands interpolated linear movement on an axes group to the specified relative positions.

Prior to performing any coordinated moves, some setup is needed (see "[Create a Linear or Circular Coordinated Motion Application](#)" on page 579). Once these steps have been performed, a linear move can be performed.

In the following examples, two linear moves will be performed. The first move is an absolute linear move that goes from (0, 0) to (100, 200). The second move is a relative linear move that goes a distance of (-75, 50) from the end of the first move. The BufferMode input is set to 'Buffered', meaning this move will wait for the first move to complete before it begins executing.

- **To Perform an Absolute Linear Move**

Call **MC\_MoveLinAbs** (Execute, AxesGroup, PositionArray, Velocity, Acceleration, Deceleration, Jerk, CoordSystem, BufferMode, TransitionMode, TransitionParameter). PositionArray is an array of absolute end positions containing one position for each axis in the group. The inputs velocity, acceleration, deceleration, and jerk establish the maximum values for the move.

In this example, PosArrayAbs[0] represent the x-axis and PosArrayAbs[1] represent the y-axis.

```
PosArrayAbs[0] := 100;
PosArrayAbs[1] := 200;
TransParam[0]  := 0;
TransParam[1]  := 0;
```

```
Inst_MC_MoveLinRel(TRUE, Group1_ref, PosArrayAbs, MaxVel, MaxAcc,
MaxDec, 0, MC_COORDINATE_SYSTEM_ACS, 1, 0, TransParam);
```

In the example a linear move will be performed on axis group 'Group1\_ref'.

- PosArrayAbs contains the absolute end points of the axes in the group. The axis stored in position 0 (IdentInGroup) of the group will be moved to 100.0. The axis stored in position 1 of the group will be moved to 200.0.
  - The maximum velocity is specified by variable MaxVel and is specified in 'user units/sec'.
  - The maximum acceleration and deceleration are specified by variables MaxAcc and MaxDec and are specified in 'user units/sec<sup>2</sup>'.
  - The maximum jerk is currently not supported and can be set to a value of 0.
  - The coordinate system is ACS
  - The BufferMode is set to 1, indicating the move is buffered. For more information about buffer modes, see the [Buffer Modes](#) overview.
  - The TransitionMode is set to 0, indicating no transition mode will be used. For more information about transition modes, see the [Transition Between Moves](#) section.
  - The TransParam array is required and the contents can be set to 0 since the transition mode is not being used. There has to be one array entry for each axis in the group.
- **To Perform a Relative Linear Move**

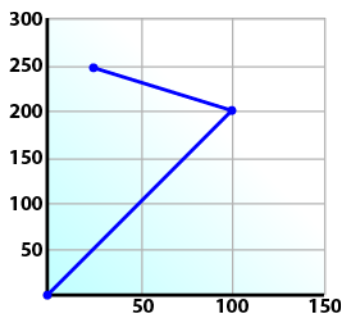
Call `MC_MoveLinRel` (Execute, AxesGroup, Distance, Velocity, Acceleration, Deceleration, Jerk, CoordSystem, BufferMode, TransitionMode, TransitionParameter). The Distance input is an array of distances, one distance for each axis in the group. The inputs velocity, acceleration, deceleration, and jerk establish the maximum values for the move.

In this example, DistArrayRel[0] represent the x-axis and DistArrayRel[1] represent the y-axis.

```
DistArrayRel[0] := -75.0; // Start pt 100 - rel 75 -> 25 absolute
end pt
DistArrayRel[1] := 50.0; // Start pt 200 + rel 50 -> 250 absolute
end pt
TransParam[0] := 0;
TransParam[1] := 0;

Inst_MC_MoveLinRel(TRUE, Group1_ref, DistArrayRel, MaxVel, MaxAcc,
MaxDec, 0, MC_COORDINATE_SYSTEM_ACS, 1, 0, TransParam);
```

In the example above, all the variables have the same meaning as the absolute linear example except DistArrayRel. DistArrayRel contains the relative distance to move for each axis in the group. The axis stored in position 0 (IdentInGroup) of the group will be moved a distance of -75.0. The axis stored in position 1 of the group will be moved a distance of 50.0.



- **To Perform a Linear Move With More Than Two Axes**

**NOTE**

The dimensionality of the move is determined by the number of axes mapped to the group. This implies that a group which could hold a maximum of three or more axes will do two dimensional moves if it only has two valid axes mapped to it.

In order to perform higher dimensional moves, additional axes must be added to the group. The steps to do this are detailed in [Create a Linear or Circular Coordinated Motion Application](#).

After the additional axes are added perform the following steps.

1. From within the Dictionary, update the array size of the variable being passed (PosArrayAbs and DistArrayRel in the examples above) to the Position input so that its length matches the maximum number of axes allowed in the group.
2. Set the desired values for the additional axes in the now larger position arrays.

#### 6.4.9.3.1.3 Performing a Circular Move

Circular moves can be programmed using absolute or relative positions using the following function blocks:

- [MC\\_MoveCircAbs](#) which commands interpolated circular movement on an axes group to the specified absolute positions.
- [MC\\_MoveCircRel](#) which commands interpolated circular movement on an axes group to the specified relative positions.

Prior to performing any coordinated moves, some setup is needed (see ["Create a Linear or Circular Coordinated Motion Application" on page 579](#)). Once these steps have been performed, a circular move can be performed.

In the following examples, two circular moves will be performed. The first move is an absolute circular move that goes from (0, 0) to (90, 90). CircMode specifies that the aux point (0, 180) will be crossed during the paths start to end. The second move is a relative circular move whose end point is (90, 90) from the end of the first move. In this move, CircMode specifies that the aux point (0, 90) is the relative center of the circle. The BufferMode input is set to 'Buffered', meaning this move will wait for the first move to complete before it begins executing.

- **To perform an Absolute Circular Move:**

Call [MC\\_MoveCircAbs](#) (Execute, AxesGroup, CircMode, AuxPoint[], EndPoint[], PathChoice, Velocity, Acceleration, Deceleration, Jerk, CoordSystem, BufferMode, TransitionMode, TransitionParameter).

In this example, PosCircAuxAbs[0] and PosCircEndAbs[0] represent the x-axis. PosCircAuxAbs[1] and PosCircEndAbs[1] represent the y-axis.

```
PosCircAuxAbs[0] := 0; // A point on the circle that is crossed
on the
PosCircAuxAbs[1] := 180; // path from start to end point.
PosCircEndAbs[0] := 90; // Absolute end point.
PosCircEndAbs[1] := 90;

Inst_MC_MoveCircAbs(TRUE, Group1_ref, MC_CIRC_MODE_BORDER,
PosCircAuxAbs, PosCircEndAbs, MC_CIRC_PATHCHOICE_CLOCKWISE, MaxVel,
MaxAcc, MaxDec, 0, MC_COORDINATE_SYSTEM_ACS, MC_BUFFER_MODE_
BUFFERED, MC_TRANSITION_MODE_NONE, TransParam);
```

In the example a circular move will be performed on axis group 'Group1\_ref'.

- CircMode is defined as MC\_CIRC\_MODE\_BORDER. This mode indicates that the AuxPoint array input will indicate a point on the circle which is crossed on the path from the starting point to the end point. See [Circular Moves Diagrams](#) for more information on CircMode movement options.
  - The AuxPoint array, 'PosCircAuxAbs', defines an absolute point on the circle which is crossed on the path from the starting point to the end point. The contents of this array are determined by the CircMode variable, MC\_CIRC\_MODE\_BORDER.
  - The EndPoint array, 'PosCircEndAbs', contains the absolute end point for each axis in the group. The absolute end point of the axis stored in position 0 (IdentInGroup) of the group will be 90.0. The absolute end point of the axis stored in position 1 of the group will be 90.0.
  - PathChoice is only relevant when CircMode is set to MC\_CIRC\_MODE\_CENTER. In this case, this parameter is not used.
  - The maximum velocity is specified by variable MaxVel and is specified in 'user units/sec'.
  - The maximum acceleration and deceleration are specified by variables MaxAcc and MaxDec and are specified in 'user units/sec2'.
  - The maximum jerk is currently not supported and can be set to a value of 0.
  - The coordinate system is ACS
  - The BufferMode is set to MC\_BUFFER\_MODE\_BUFFERED, indicating the move is buffered. For more information about buffer modes, see the [Buffer Modes](#) overview.
  - The TransitionMode is set to MC\_TRANSITION\_MODE\_NONE, indicating no transition mode will be used. For more information about transition modes, see the [Transition Between Moves](#) section.
  - The TransParam array is required. The TransParam array is a 2-element array containing the corner distance and velocity for the transition. Transitions are not used in this example and therefore the contents can be set to 0.
- **To perform a Relative Circular Move:**

Call `MC_MoveCircRel` (Execute, AxesGroup, CircMode, AuxPoint[], EndPoint[], PathChoice, Velocity, Acceleration, Deceleration, Jerk, CoordSystem, BufferMode, TransitionMode, TransitionParameter).

In this example, PosCircAuxRel[0] and PosCircEndRel[0] represent the x-axis. PosCircAuxRel[1] and PosCircEndRel[1] represent the y-axis.

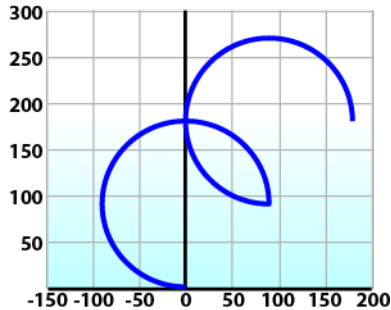
```
PosCircAuxRel[0] := 0; // Relative center of the circle.
PosCircAuxRel[1] := 90;
PosCircEndRel[0] := 90; // Relative end point.
PosCircEndRel[1] := 90; // Start pt 90,90 + rel 90,90 -> 180,180
absolute end pt

Inst_MC_MoveCircRel(TRUE, Group1_ref, MC_CIRC_MODE_CENTER,
PosCircAuxRel, PosCircEndRel, MC_CIRC_PATHCHOICE_CLOCKWISE, MaxVel,
MaxAcc, MaxDec, 0, MC_COORDINATE_SYSTEM_ACS, MC_BUFFER_MODE_
BUFFERED, MC_TRANSITION_MODE_NONE, TransParam);
```

In the example all the variables have the same meaning as the circular absolute example except:

- CircMode is defined as MC\_CIRC\_MODE\_CENTER. This mode indicates that the AuxPoint array input will indicate the center point of the circle. See [Circular Moves Diagrams](#) for more information on CircMode movement options.
- The AuxPoint array, 'PosCircAuxRel', defines the relative center point of the circle. The contents of this array are determined by the CircMode variable, MC\_CIRC\_MODE\_CENTER.

- The EndPoint array, 'PosCircEndRel', contains the relative end point for each axis in the group. The relative end point of the axis stored in position 0 (IdentInGroup) of the group will be 90.0. The relative end point of the axis stored in position 1 of the group will be 90.0.
- PathChoice is relevant when CircMode is set to MC\_CIRC\_MODE\_CENTER. In this case, PathChoice is MC\_CIRC\_PATHCHOICE\_CLOCKWISE which specifies the direction of the path.

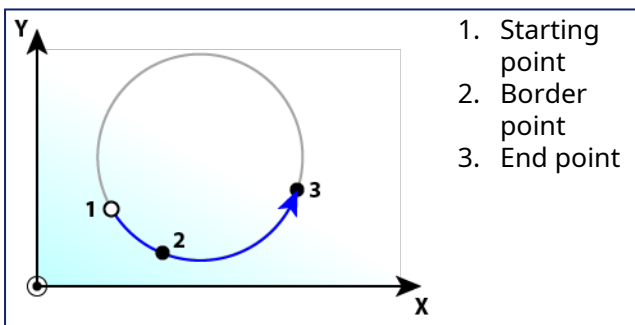


#### 6.4.9.3.1.4 Circular Moves Diagrams

##### CircMode = BORDER

The user defines the end point and a border point (= input 'AuxPoint') on the sector of the circle which the machine will traverse. For Relative mode, both points are defined relative to the starting point.

<b>Advantages</b>	<ul style="list-style-type: none"> <li>• The border point can usually be reached by the machine, i.e. it can be taught.</li> </ul>
<b>Disadvantages</b>	<ul style="list-style-type: none"> <li>• Restricted to angles &lt; 360° in one single command.</li> </ul>

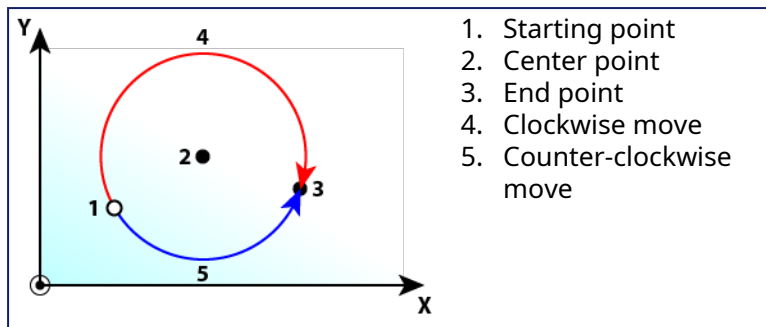


##### CircMode = CENTER

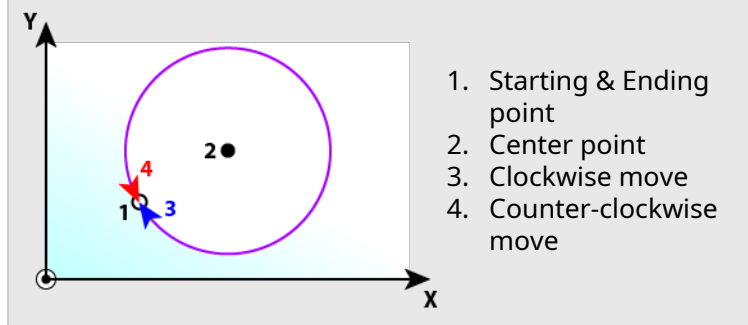
The user defines the end point and center point (= input 'AuxPoint') of the circle. The input 'PathChoice' defines clockwise or counter-clockwise motion. For Relative mode, both points are defined relative to the starting point.

<b>Advantages</b>	<ul style="list-style-type: none"> <li>• Full 360° moves are possible.</li> </ul>
<b>Disadvantages</b>	<ul style="list-style-type: none"> <li>• Cannot perform zero-distance moves.</li> <li>• Over-determination of the circle equation.</li> </ul>



**NOTE**

A 360° move will be performed if the end point is the same as the start point.



### 6.4.10 Add an HMI Device

To control your application, HMI panels can be downloaded to a dedicated [HMI device](#) (as described in the following procedure), but it can also be embedded into a targeted [controller](#).

Modbus TCP is used for communication from an HMI panel's ethernet port to the X32 connector on the top of a AKD PDMM or PCMM. Modbus communication may be active while programming the controller through the KAS IDE while the AKD PDMM or PCMM is connected to a network, switch, or hub. The KAS IDE is automatically linked to the KVB HMI development software, and the Modbus addresses are automatically set. See [KDN](#) for more information.

When running the KAS Simulator, an internal HMI editor is also available to debug your application (for more details, [Design the Control Panel with the Internal Control Panel Editor](#))

#### 6.4.10.1 Create KVB Project

KVB projects are managed in the Project Explorer and can be created as follows:

1. In the Project Explorer, right-click on the **System** item to open the contextual menu
2. Select the **Add HMI device** command
3. Select the device name within the list and Click OK

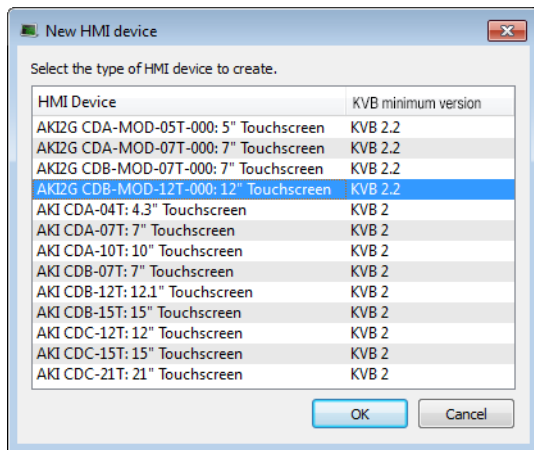


Figure 4-26: Select an AKI to add.

#### **TIP**

The content will vary depending upon what version of KVB is installed on your system. If you have KVB 1.1 installed you may open projects that have a KVB 2.1 panel, but you will not be able to open the panel in KVB.

4. Right-click on the newly created item and select the **Rename** command to change its name
5. Right-click and select the **Add KVB Project** command

#### **NOTE**

Note that this command is disabled when a KVB panel is already created for the current HMI device

### 6.4.10.2 Map Variables to HMI

For HMI, the variable mapping is done in two phases.

- Phase 1 - Tag all of the variables to be exported in from the HMI project (see procedure below)
- Phase 2 - Use this mapping file when [designing the HMI](#)

The tag operation directly takes place in the Dictionary, as follows:

1. Open the Variable list editor available in the **Dictionary** toolbox

#### **TIP**

Double-click the Dictionary header to display the widget as a popup window in order to have more space.

2. In the KVB column, select the variables you want to map

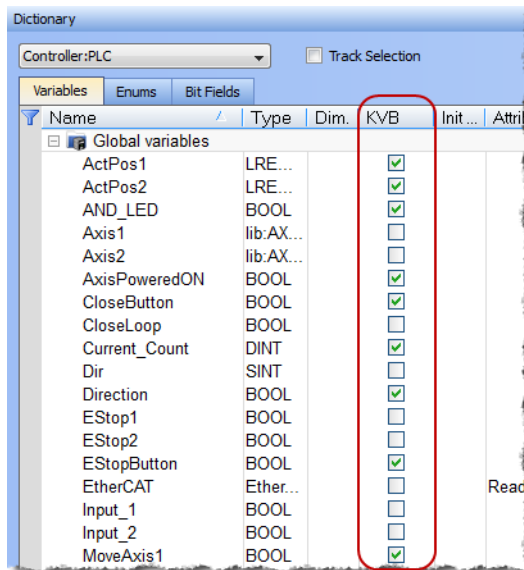


Figure 4-27: Variable Mapping to KVB.

#### **TIP**

Selecting the variable alone does not guarantee it will be exported. The POU must be set to compile as well. See [Ensuring Variables are Exported](#) for more information.

#### **IMPORTANT**

Being based on Modbus, the communication is limited to 32 bits. As a consequence:

- Data type conversion can lead to a loss in accuracy:
  - LREAL variables are saved as REAL
  - LINT variables are converted to DINT
  - ULINT variables are saved as UDINT
- LWORD data types cannot be used within the HMI when using KVB. This is because variables of this type are not exported, even when selected.
- STRING variable data types are allowed in the HMI when using KVB 2.0 or later (they are not supported in KVB 1.2). String variables allow up to 240 characters.
- Variables of types "PNCode", "ProfilesCode", or instances of UDFB cannot be used within the HMI.
- String variables whose size exceed the maximum string length of 240 characters generate a warning message in the Compiler Output window when being exported

over Modbus. Only the first 240 characters will be exported. See [Compiler Output](#) for more information.

```

Information and Logs
*Local log* Controller log Find and Replace Browse Variable Breakpoints Compiler Output
Compile Project
Controller:----- Compile Device -----
Controller:----- Generating Modbus configuration from the dictionary KVB check boxes -----
Controller:KVB: Warning : (Global):Sample_Inst is having a member NewVar (String variable) length is more than supported Modbus string length of 240!
Controller:KVB: Warning : (Global):Sample_Inst contains variables which are not supported by KVB
Controller:KVB: Warning : (Global):Inst_NewStructure.1 is having a member Inst_NewStructure.NewVar (String variable) length is more than supported Modbus string length of 240!
Controller:KVB: Warning : (Global):Inst_NewStructure.1 contains variables which are not supported by KVB
Controller:KVB: Warning : UntitledST:Inst_NewStructure is having a member NewVar (String variable) length is more than supported Modbus string length of 240!
Controller:KVB: Warning : UntitledST:Inst_NewStructure contains variables which are not supported by KVB
Controller:----- Generating Modbus Files successful -----
Controller:----- Compile PLC -----
Controller:PLC:Compiler V13.5.10.0
Controller:PLC:>> Complex variables stored in a separate segment
Controller:PLC:Loading application symbols...

```

#### TIP

Non-primitive data types which are not exported over Modbus generate a warning message in the Compiler Output window. See [Compiler Output](#) for more information.

3. Compile the application to create the Modbus mapping file.

#### TIP

Generally you do not need to access this, but if you do the file used for manually importing variables into KVB (named **HMI Variable Import File.txt**) is located in the following location.

- Windows 7: C:\Users\{user}\AppData\Local\Kollmorgen\KAS\Project
- Windows XP: C:\Documents and Settings\{user}\Local Settings\Application Data\Kollmorgen\KAS\Project

#### NOTE

- If you modify the set of tagged variables in the dictionary, you have to update the text file by recompiling the project.
- The Modbus variables defined in KAS IDE are imported in Kollmorgen Visualization Builder only when you start KVB (there is no update in real-time between the two applications).

4. Open the **KVBProject** from the Project Tree. This will automatically transfer the tagged variables..

### List of variables that you can export

The following types of variables can be exported to the HMI:

- The fundamental data types: BOOL, SINT, INT, USINT, UINT, BYTE, WORD, DINT, UDINT, DWORD, TIME, REAL, LREAL, STRING.
- Arrays of supported data types
- Structures that include members of supported data types

### Examples of structures that you can export

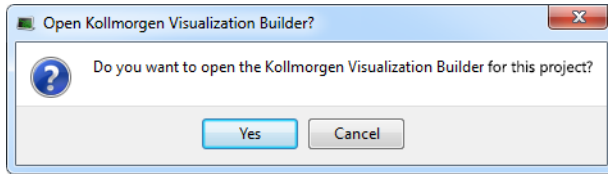
- Arrays of BOOLS such as a structure that includes a BOOL array member and instances of UDFB members.
- An array of structures that include INT and LREAL members
- A structure that includes both instances of UDFB member as well as an embedded structure which includes an INT member and instances of a UDFB member. This is due to the embedded structure has an INT member and therefore the outer structure can be exported too.

### Examples of structures that you cannot export

- A structure that includes instances of UDFB or LWORD members only.

### 6.4.10.3 Design KVB Panel with Kollmorgen Visualization Builder

- Double-click the new KVB panel to open the builder (for more details, refer to [Using Kollmorgen Visualization Builder](#).)



**Figure 4-28:** Open the Kollmorgen Visualization Builder Builder

#### **!** IMPORTANT

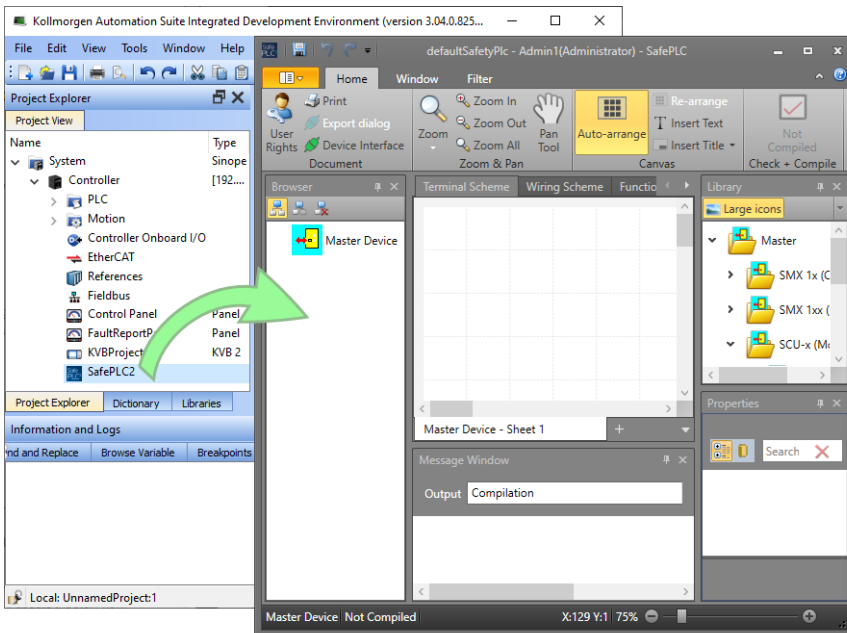
You must have the specific application already installed on your machine.

#### **↖** TIP

Be sure to close the Kollmorgen Visualization Builder before deleting the KVB Panel from the IDE.

### 6.4.11 Working with SafePLC2 Projects

A node may be added to the Project tree which connects to a [SafePLC2](#) project. This allows you to synchronize a SafePLC2 project with a KAS project. The KAS IDE automatically generates the PDOs for the connected devices and will establish the Black Channel with the EtherCAT Safety network.



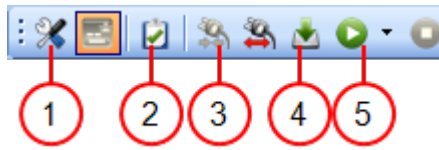
There are two methods for creating the synchronization between KAS Runtime and SafePLC2.

- Direct - this is used when both the KAS IDE and SafePLC2 are installed on the same computer.
- Import - this is used when KAS IDE and SafePLC2 are installed on different computers.

Please see the online help or the AKT2G I/O Manual for detailed directions.

### 6.4.12 Overview: Compile, Connect, Download, and Run

The last part of building a project is to compile the code, connect to the controller, download, and then run the application.



1. [Set the Compilation Options](#)
2. [Compile the Application](#)
3. [Connect to the Controller](#)
4. [Download the Application](#)
5. [Start/stop the Device](#)

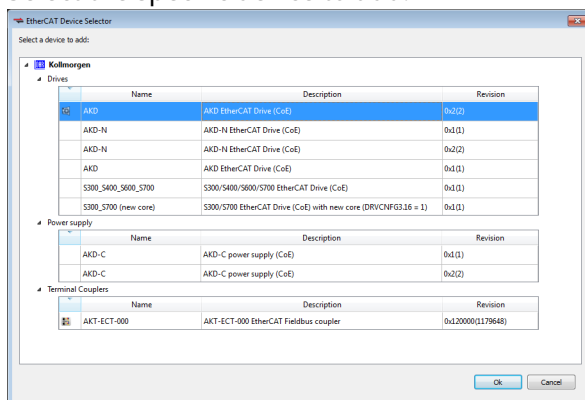
### 6.4.13 Advanced Project Creation

This section provides reference for setting up hardware, performing tasks which may not apply to every user, and other more advanced topics.

### 6.4.14 Add and Configure a Servo Drive

#### 6.4.14.1 Add the Drive

1. In the Project Explorer, right-click the **EtherCAT** node to open the menu
2. Select the **Add Kollmorgen Device** command (this option is only enabled when you are **not** connected to the controller)
3. Select the specific device to add.

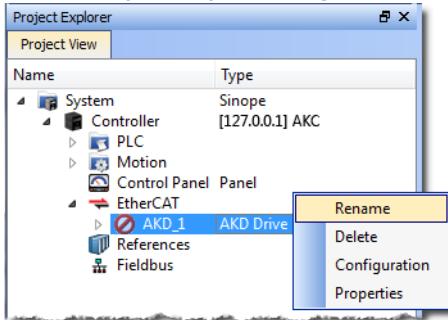


#### NOTE

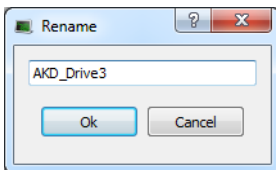
When choosing an AKD be sure to select the newest revision. Older revisions may show in the list of options, but they may not be feature complete. If one of these drives is selected, only the General Properties tab will be available.

4. Click **Ok** when you are done (for more details about the AKD drive GUI, click [here](#))

- In the Project Explorer, right-click the device node and select the **Rename** command



- Define the name for the new device  
Note that the name is limited to 10 characters and can only include letters, numbers, and underscores.



- Click **OK**

**TIP**

An alternative method to add a drive is to rely on the [auto scan](#) feature.

When an AKD drive is added to the project tree, it must be mapped to a physical drive. This step is explained in [Add & Configure Third Party Devices](#).

**6.4.14.2 Configure the AKD2G/AKD Drive**

The Configuration tab under the EtherCAT : AKD tab allows you to configure drives once the following conditions have been met:

- The [scan](#) has been performed
- Your project is compliant with the physical devices on the EtherCAT network
- You have activated the [Online Configuration](#) mode

You then have access to all of the AKD configurations and parameters.

- In the Project Explorer, double-click the new AKD Drive or select Configuration from the context menu to open all the parameters linked to it.

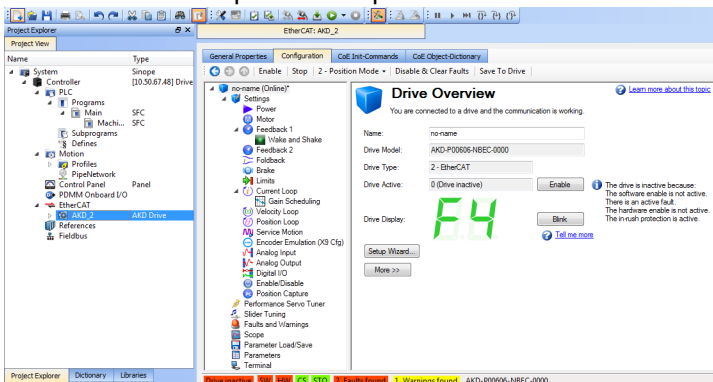


Figure 4-29: AKD Configuration



- Define the motion parameters in the **Limits** tab to configure the limited motor torque, speed, and acceleration

**Limits**  
This page shows all the drive limits all in one place.

**Current Limits**

Positive Peak Current:	18.000	Ams
Negative Peak Current:	-18.000	Ams

**Velocity Limits**

Positive Speed Limit:	3,000.000	rpm
Negative Speed Limit:	-3,000.000	rpm
Over-Speed Limit:	3,600.000	rpm

**Position Limits**

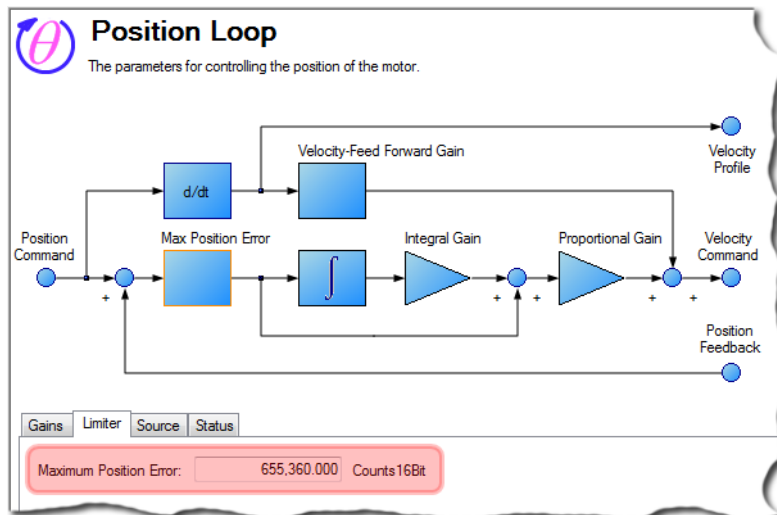
Maximum Position Error:	655,360.000	Counts16Bit
Position Limit 0 <input type="checkbox"/>	0.000	Counts16Bit
Position Limit 1 <input type="checkbox"/>	1,048,576.000	Counts16Bit

**Acceleration Limits**

Acceleration:	10,000.170	rpm/s
Deceleration:	10,000.170	rpm/s

**Motor Limits**  
Motor limits are set through the Motor Foldback Screen: [Goto Foldback](#)

- Define the motion parameters in the **Position Loop** tab to configure the limiting following error.



- Define the resolution of the feedback position in the **Feedback** tab  
Note that for all feedback types, the motor position feedback sent from the AKD drive to the controller through EtherCAT is normalized to 20 bits/rev or 1048576 counts/rev
- Then, you must define the units to be used for the motion 1 :
  - For Pipe Network, refer to [Set the Axis Block Position Units](#)
  - For PLCopen, refer to [Axis Data Parameters](#)

#### NOTE

User units in the PLC language editors are:

- Position : User unit
- Velocity User unit/sec
- Acceleration: User unit/sec<sup>2</sup>

Several AKD tabs contain units that follow the standard AKD format:

<sup>1</sup>The normal units screen in the AKD Work bench GUI is not included in the IDE

- Position: 16 bits/rev
- Velocity: RPM
- Acceleration: RPM/ Sec

- To ensure high performance, define the load for your servo system.. KAS IDE provides several options for performing the drive tuning:
  - Slider Tuning - Allows adjustment to the desired bandwidth using the slider (pre-calculated tuning)
  - Performance Servo Tuner - Takes the drive through an automatic tuning sequence
  - Manual Tuning - Allows you to set gains individually for Current Loop, Velocity Loop, and Position Loop

For more details on AKD configuration, [AKD Drive Configuration](#)

For more details on AKD Firmware Download, refer to the [FAQ](#) section.

**NOTE**

After your application is downloaded to the controller you can activate the [Online Configuration Mode](#) to configure your drives with the [Setup Wizard...](#)  
 For more details, [AKD Setup Wizard...](#)

### 6.4.14.3 Save Parameters to Non-Volatile Memory

The AKD parameters can be saved to non volatile memory in the drive. For more details, refer to [Save parameters to NV memory](#)

The screenshot shows the 'EtherCAT Devices' window with a table listing 18 devices. The table columns are: #, Name, Vendor, Product, Model, FW Version, Drive Axis Number, Mapped to Axis, and Simulated. Row 11 is highlighted in blue. Below the table, there are two buttons: 'Save parameters to NV memory' and 'Upgrade Firmware'.

#	Name	Vendor	Product	Model	FW Version	Drive Axis Number	Mapped to Axis	Simulated
1	AKD_1	Kollmorgen	AKD EtherCAT Drive (CoE)	AKD-M00306-MCEC-0000	01-20-00-001	1	(None)	N/A
2	AKD_2	Kollmorgen	AKD EtherCAT Drive (CoE)	AKD-P00306-NAEC-0000	01-20-00-001	1	(None)	N/A
3	Coupler_1	Kollmorgen	AKT-ECT-000 EtherCAT Fieldbus coupler	N/A	(Unknown)	N/A	N/A	N/A
4	Coupler_2	Kollmorgen	AKT2G-ECT-000-000 EtherCAT Gateway	N/A	(Unknown)	N/A	N/A	N/A
5	AI_Slice_1	Kollmorgen	AKT2G-AN-430-000 4Ch. Ana. Input +/-10V Diff., +/-20mA SingleEnded, 16 Bit	AKT2G-	05	N/A	N/A	N/A
6	AO_Slice_1	Kollmorgen	AKT2G-AT-425-000 4Ch. Ana. Output +/-10V, 12bit	AKT2G-AT-42	04	N/A	N/A	N/A
7	AO_Slice_2	Kollmorgen	AKT2G-AT-410-000 4Ch. Ana. Output 0-10V, 12bit	AKT2G-AT-41	04	N/A	N/A	N/A
8	AI_Slice_2	Kollmorgen	AKT2G-AN-240-000 2Ch. Ana. Input PT100 (RTD)	AKT2G-AN-24	07	N/A	N/A	N/A
9	AI_Slice_3	Kollmorgen	AKT2G-AN-400-000 4Ch. Ana. Input Thermocouple (TC)	AKT2G-AN-40	07	N/A	N/A	N/A
10	DI_Slice_1	Kollmorgen	AKT2G-DN-002-000 2Ch. +/- Counter 24V, 100kHz	DN-002	04	N/A	N/A	N/A
11	DI_Slice_2	Kollmorgen	AKT2G-DNH-008-000 8Ch. Dig. Input 24V, 10µs	N/A	(Unknown)	N/A	N/A	N/A
12	DI_Slice_3	Kollmorgen	AKT2G-DN-008-000 8Ch. Dig. Input 24V, 3ms	N/A	(Unknown)	N/A	N/A	N/A
13	DO_Slice_1	Kollmorgen	AKT2G-DT-008-000 8-channel digital output terminal 24 V DC, 0.5 A	N/A	(Unknown)	N/A	N/A	N/A
14	DI_SafeSlice_1	Kollmorgen	AKT2G-SDI-004-000, 4 Ch. Safety Input 24V	EL1904	06	N/A	N/A	N/A
15	DO_SafeSlice_1	Kollmorgen	AKT2G-SDO-004-000, 4 Ch. Safety Output 24V, 0.5A	EL2904	04	N/A	N/A	N/A
16	AKD2G_1	Kollmorgen	AKD2G Dual Axis FS1 EtherCAT Drive (CoE)	AKD2G-SPE-6V06D-A1F3-0000	M_02-05-00-000	1	PLCAxis2	N/A
17	Device_1	BBH Products GmbH	SCU	SCU-	03.00.00.61	2	PLCAxis2	N/A
18	AKD2G_2	Kollmorgen	AKD2G Dual Axis FS2 EtherCAT Drive (CoE)	AKD2G-SPE-6V06D-A2F3-0000	M_02-05-01-000	N/A	N/A	N/A

### 6.4.14.4 AKD Setup Wizard...

The wizard allows you to configure drives once the following conditions have been met:

- The [scan](#) has been performed
- Your project is compliant with the physical devices on the EtherCAT network
- You have activated the [Online Configuration](#) mode

You then have access to the AKD parameters that are used when the drive is running.

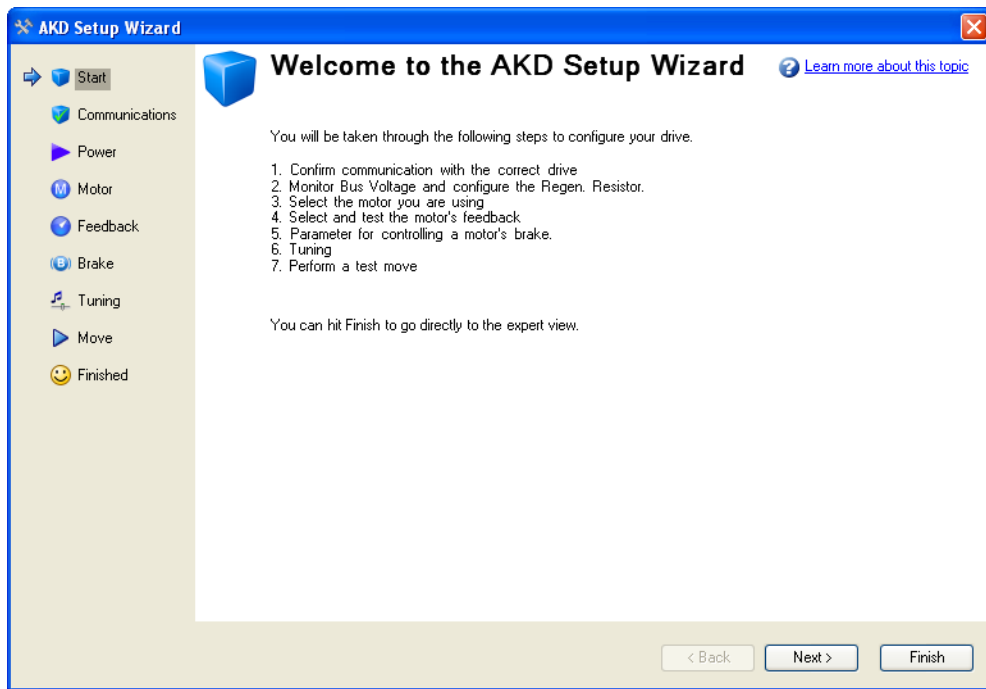


Figure 4-30: AKD Setup Wizard

#### 6.4.14.5 Configure Onboard I/O

The procedure to define the local I/Os of the AKD drive is very similar to the one for I/O slices, with the following exceptions:

- Channel column also contains in brackets the connector and pin number
- [PLC variable selection](#) applies to digital inputs as well as analog inputs and outputs.

#### 6.4.14.6 Digital Input Mode

##### NOTE

The way to set the Digital Input mode on an AKD drive has changed with KAS 2.6. Previously, this setting was done in the Onboard I/O view (seen below) where the IO was mapped to PLC variables.

Properties Analog Input Configuration

### AKD Onboard I/O Properties

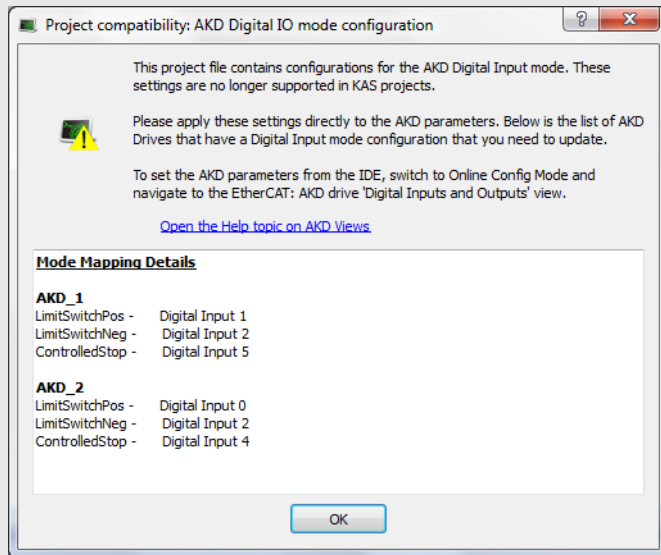
AKD Onboard I/O Connectors (X7 and X8)

1) Select the desired channels  
2) Right-click and select 'map'

Channel	Variable Name	Mode
Digital Input 1 (X7 pin 10)	akdIO_1	Positive Limit Switch
Digital Input 2 (X7 pin 9)	akdIO_2	Negative Limit Switch
Digital Input 3 (X7 pin 4)	akdIO_3	Controlled Stop
Digital Input 4 (X7 pin 3)	akdIO_4	{none}
Digital Input 5 (X8 pin 6)	akdIO_5	{none}
Digital Input 6 (X8 pin 5)	akdIO_6	{none}
Digital Input 7 (X7 pin 2)	akdIO_7	{none}
Digital Output 1 (X7 pin 7 & 8)		
Digital Output 2 (X7 pin 5 & 6)		
Analog Input		

The 'Digital Inputs and Outputs' are now accessed using the new AKD GUI integration inside KAS. This will allow you to save this setting inside the drive instead of pushing it every time the application is started.

Files created with KAS 2.5 or earlier, with specific mode settings for any Digital Input of an AKD drive, will automatically be converted to the new project format and these settings will be removed. A dialog box will pop up prior to removal to inform you which settings will be removed from the project.

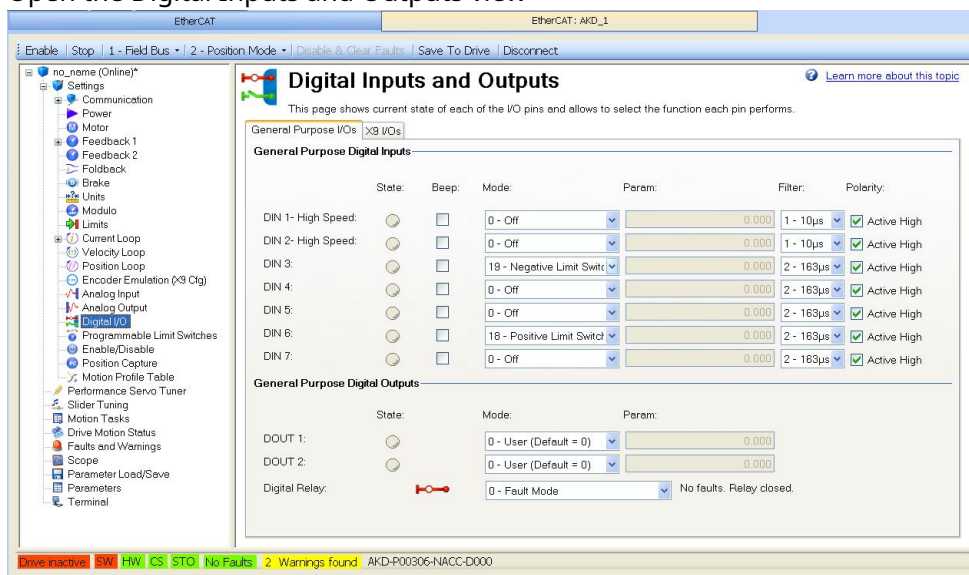


For each drive you will find the list of digital inputs with their specific mode settings. You can copy/paste the content of this text area to a text editor to keep track of them before closing the dialog box.

#### 6.4.14.6.1 How to configure Digital Input mode setting inside WorkBench

To set the mode of a Digital Input

1. Connect to the controller
2. Go to [Online Configuration Mode](#)
3. Open the desired drive's WorkBench by double-clicking on the AKD drive node in the project tree view.
4. Open the Digital Inputs and Outputs view



On this screen, you can set the mode of each Digital IO on the drive using the dropdown list in the Mode column. For more information about this view see .

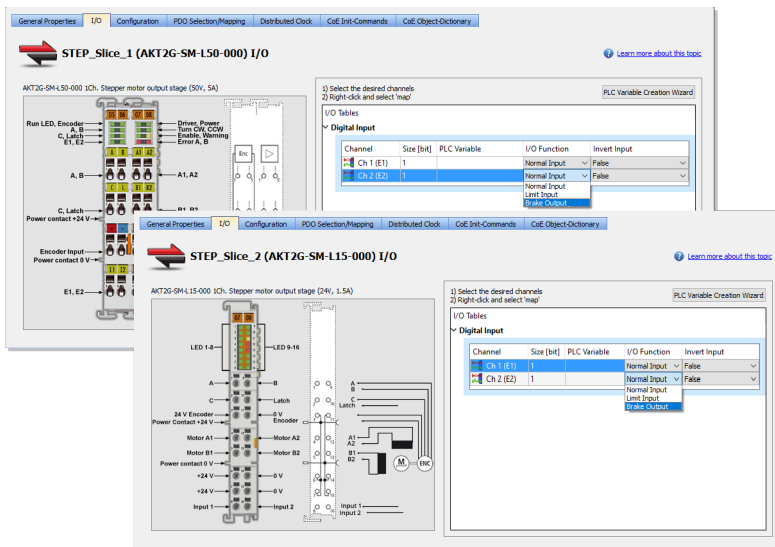
### 6.4.15 Add and Configure a Stepper Drive

Following are the steps for adding an [AKT2G-SM-L15-000](#) or [AKT2G-SM-L50-000](#) stepper drive to your project.

1. Add an AKT2G stepper I/O slice to your project. See [Add and Configure I/O Slices](#) for details.
2. Configure the device. See [AKT2G-SM-Lxx Stepper Drive Configuration Tab](#)
3. Assign variables to the I/O. See [AKT2G-SM-Lxx Stepper Drive I/O Tab](#)

#### 6.4.15.1 AKT2G-SM-Lxx Stepper Drive I/O Tab

This tab is used to assign PLC variables to device I/O.



Element	Description
PLC Variable	A PLC variable may be assigned to each channel. Double-click (or right-click and select <b>Map</b> ) to open the PLC Variable Selector to choose or create a variable. Alternatively you may use the <a href="#">PLC Variable Creation Wizard</a> to automatically create a list of variables for mapping.
I/O Function	The AKT2G-SM-Lxx stepper slices provide two digital inputs. <ul style="list-style-type: none"> <li>• <b>Normal Input:</b> This selection configures the I/O channel as a normal input.</li> <li>• <b>Limit Input:</b> This selection configures the I/O channel as the hardware limit switch. The drive is disabled when a digital input acting as a source for the hardware limit is triggered.</li> <li>• <b>Brake Output:</b> This selection configures the I/O channel as the <b>Brake Output</b>. This option is supported only for the second I/O channel on the terminal.</li> </ul>
Invert Input	This option inverts the state of the input when set to True. This is set to False by default. This setting is applicable only when the I/O channel is configured as either <b>Normal Input</b> or <b>Limit Input</b> .

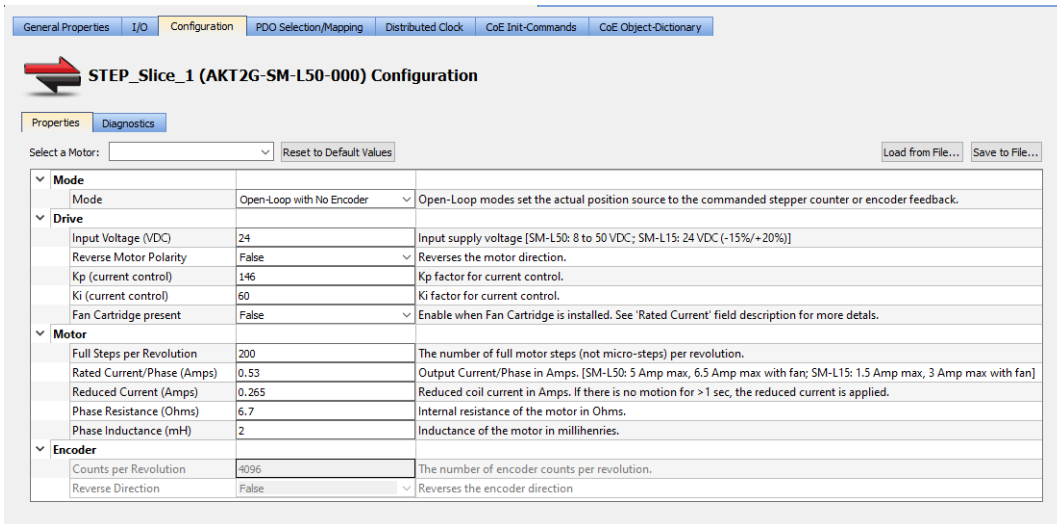
**TIP**

- See [Add and Configure I/O Slices](#) for adding a stepper drive to your project.
- For information on the devices see [AKT2G-SM-L15-000](#) or [AKT2G-SM-L50-000](#).
- See [AKT2G-SM-Lx Object Description](#) for stepper drive-specific object descriptions.

### 6.4.15.2 AKT2G-SM-Lxx Stepper Drive Configuration Tab

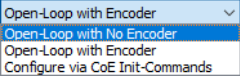
This tab is used to configure the device, based on the parameters of the stepper motor being used, and to diagnose any problems with the stepper terminal. To make configuration simple, a Kollmorgen stepper motor can be selected from the [Properties Tab](#), which will automatically populate the stepper motor parameter fields. The stepper motor parameters can also be easily replicated to a file, and transferred between devices. Once the project is compiled the [CoE Init Commands tab](#) will be populated based on these values. Troubleshooting the stepper terminal can begin by gathering information from the [Diagnostics Tab](#)

#### 6.4.15.2.1 Properties Tab



- Select a Motor:** The drop-down list contains nearly 500 Kollmorgen stepper motor models and their specific parameters. Selecting a model automatically populates the appropriate motor-specific fields. See the [Kollmorgen Stepper Optimizer](#) if you need to match a motor series name to the model number.
- Reset to Default Values:** Clicking this button will change all of the stepper motor parameters to the default values.
- Load from File...:** This function prompts you to select an XML file to populate all of the stepper parameters, not just those specific to the motor.
- Save to File...:** This function creates an XML file containing all of the current stepper parameters, which is easily stored or transferred between devices.

## Stepper Motor Parameters

Section	Element	Description
Mode	Mode	<p>Select the CoE object used for the actual position.</p> <ul style="list-style-type: none"> <li> <b>Open Loop with No Encoder:</b> This option should be selected when there is no position feedback device for the stepper motor. In this mode, the command position increments sent to the stepper drive will be accumulated in an internal counter and returned to the controller as the actual position. The pLCopen and PipeNetwork axis motion function blocks such as <a href="#">MC_ReadActPos</a>, <a href="#">MLAxisReadActPos</a> return actual position, which is the stepper drive's internal counter. With this mode, the controller commands position without considering position feedback. There is no assurance that the motor will get to its destination. </li> <li> <b>Open Loop with Encoder:</b> With this mode the controller is commanding position without considering feedback. There is no assurance that the motor will get to its destination as there is no ability to self-correct. </li> <li> <b>Configure via CoE Init-Commands:</b> With this selection stepper motor configuration parameters such as motor full steps per revolution, feedback device type and resolution, etc. must be configured manually in the <a href="#">CoE Init Commands tab</a>. </li> </ul> 
Drive	Input Voltage (VDC)	<p>Specify the input supply voltage. Range is:</p> <ul style="list-style-type: none"> <li>SM-L15 = 24 VDC (-15%/+20%)</li> <li>SM-L50 = 8 - 50 VDC</li> </ul>
	Reverse Motor Polarity	Select <b>True</b> or <b>False</b> from the drop-down menu to reverse the motor direction.
	Speed Range	Select the maximum output full-step frequency from the drop-down list.
	Kp (current control)	Specify the proportional gain factor for current control.
	Ki (current control)	Specify the integral gain factor for current control.
	Fan Cartridge Present	Set this to "True" if a <a href="#">AKT2G-AC-FAN-001</a> fan cartridge is installed. Having a fan present allows you to set the <a href="#">Rated Current / Phase (Amps)</a> to a higher value.

Section	Element	Description
Motor	Full Steps per Revolution	Specify the number of full motor steps per revolution.
	Rated Current / Phase (Amps)	<p>Specify the motor's rated current per phase in Amps. This is also known as the RMS (root mean square) current. This value will be converted from RMS to Peak output current, by multiplying the value by 1.414 (square root of 2) to specify the maximum output current for the stepper drive.</p> <p>The 64x microstep resolution causes a 30 percent current loss compared to full-step resolution. This is due to the output wave form being closer to a sine wave than a square wave. The rated current/phase value is converted to the peak output current to compensate for the current loss. Since the peak output current may exceed the motor's rated current per phase at standstill, it is very important to configure the <a href="#">Reduced Current/Phase (Amps)</a>, to prevent the motor from overheating.</p> <p>Maximum Motor Current/Phase (RMS) values are:</p> <ul style="list-style-type: none"> <li>SM-L15 = 1.06 Amps without a fan, and 1.12 Amps with a fan</li> <li>SM-L50 = 3.53 Amps without a fan, and 4.59 Amps with a fan</li> </ul> <p>Maximum converted Peak output values are:</p> <ul style="list-style-type: none"> <li>SM-L15 = 1.5 Amps without a fan, and 3 Amps with a fan</li> <li>SM-L50 = 5 Amps without a fan, and 6.5 Amps with a fan</li> </ul>
	Reduced Current/Phase (Amps)	<p>Specify the motor's current per phase in Amps when at standstill. This is also known as the RMS (root mean square) current. This value will be converted from RMS to Peak output current, by multiplying the value by 1.414 (square root of 2) to specify the maximum output current for the stepper drive when the motor is at standstill.</p> <p>The Reduced Peak Current is applied when there is no motion for &gt; 1 sec. At a minimum, the Reduced Current/Phase can be set to 70% of the <a href="#">Rated Current / Phase (Amps)</a>. Setting the Reduced Current/Phase to 50% (or lower) of the Rated Current/Phase is recommended.</p>
	Phase Resistance (Ohms)	Specify the internal resistance of the motor in Ohms.
	Phase Inductance (mH)	Specify the motor's inductance in millihenries.
Encoder	Counts per Revolution	Specify the number of encoder counts per revolution.
	Reverse Direction	Select <b>True</b> or <b>False</b> from the drop-down menu to reverse the encoder direction.

#### TIP

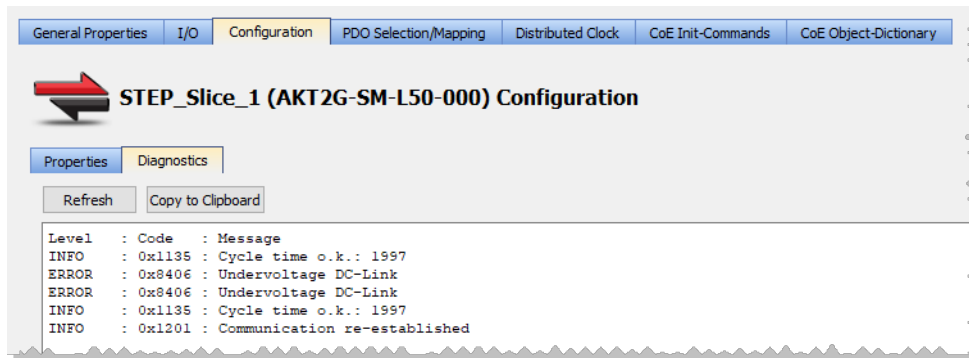
- See [Add and Configure I/O Slices](#) for adding a stepper drive to your project.
- For information on the devices see [AKT2G-SM-L15-000](#) or [AKT2G-SM-L50-000](#).
- See [AKT2G-SM-Lx Object Description](#) for stepper drive-specific object descriptions.

#### 6.4.15.2.2 Diagnostics Tab

This tab gathers the diagnostic messages available on the connected stepper terminal. This information may be used for troubleshooting during commissioning or operation. The PLC



application must be running on a controller or Online Configuration mode must be enabled to acquire diagnostics data.







**Refresh:** Clicking this button reads and displays the stepper terminal's messages. A comprehensive list of messages may be found in [AKT2G-SM-Lxx Diagnostic Messages](#). This button is disabled under the following conditions.

- The PLC application is neither running nor in online configuration mode.
- A stepper I/O slice is not present on the EtherCAT network e.g. offline device
- The PLC application is running on the KAS Simulator.

**Copy To Clipboard:** Clicking this button copies the message data.

*If your application is not running perform the following steps.*

1. Connect to your controller. 
2. Compile and download the application. 
3. Enable [Online Configuration Mode](#).   
- OR -  
Start the application. 

### 6.4.16 Add and Configure I/O Slices

This topic discusses adding AKT (K-Bus) and AKT2G (EtherCAT/E-Bus) I/O devices. All devices are children of the EtherCAT node in the Project tree.

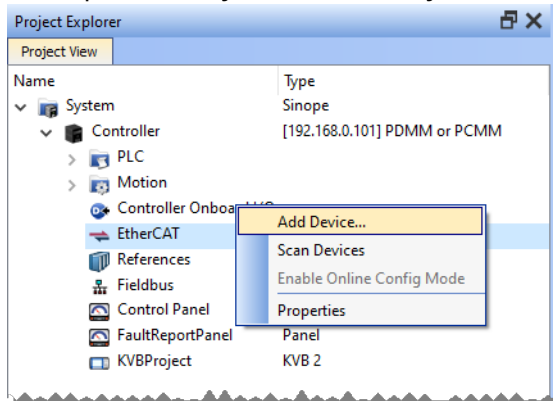
- AKT2G (E-Bus) devices are each directly under the EtherCAT node.
- AKT (K-Bus) slices are children of the coupler, which is a child of the EtherCAT node.

AKT2G example		AKT example	
EtherCAT		EtherCAT	
⊗ Coupler_2	AKT2G-ECT-000-000	⊗ Coupler_1	KBus Coupler
⊗ DI_Slice_1	AKT2G-DNH-008-000	⊗ DISlice_1	8 ch Digital Input
⊗ DO_Slice_1	AKT2G-DT-008-000	⊗ DOSlice_1	8 ch Digital Output
⊗ AKD2G_1	AKD2G Dual Axis SIL2	⊗ AKD2G_1	AKD2G Dual Axis SIL2

For local I/O, refer to [Configure Onboard I/O](#) (→ p. 179).

#### 6.4.16.1 Add an AKT (K-Bus) I/O Coupler

1. In the Project Explorer, right-click the EtherCAT node to open the menu
2. Select the **Add Device** command  
(this option is only enabled when you are **not** connected to the controller)



3. Select the coupler and click **Ok**
4. In the Project Explorer, right-click the coupler node and select the **Rename** command to change the name
5. Click **Ok**
6. The coupler is added to the EtherCAT node.

### 6.4.16.2 Add an AKT (K-Bus) I/O Slice

1. In the Project Explorer, right-click the Coupler node to open the menu
2. Select the **Add I/O Slice** command
3. Choose the I/O slice from the list

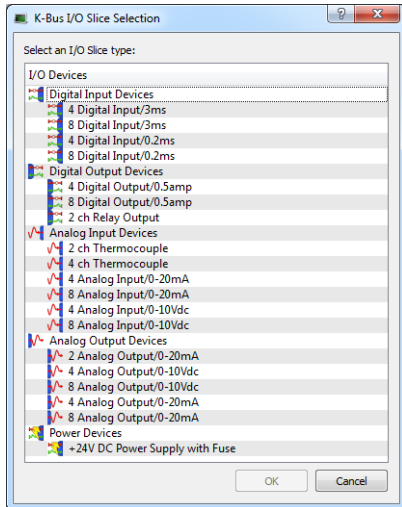
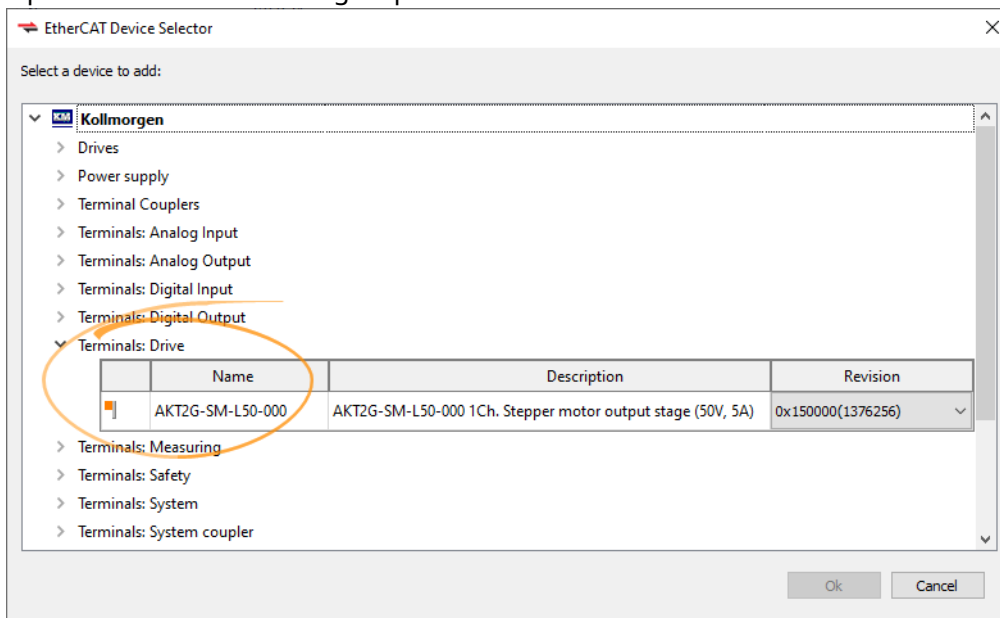


Figure 4-31: Add I/O Slice

4. Click **OK**
5. The device is added below the coupler.

### 6.4.17 Add an AKT2G Stepper Drive

1. In the Project Explorer, right-click the EtherCAT node to open the menu
2. Select the **Add Device** command
3. Open the Terminals:Drive group and choose a device from the list.



4. Click **OK**
5. The device is added to the EtherCAT node.

For more information see:

- [AKT2G-SM-Lxx Stepper Drive I/O Tab](#)
- [AKT2G-SM-L15-000](#) for product information
- [AKT2G-SM-L50-000](#) for product information

#### **6.4.17.1 Configure the I/O Slice**

For more details, refer to [Map Input and Output to Variables](#)

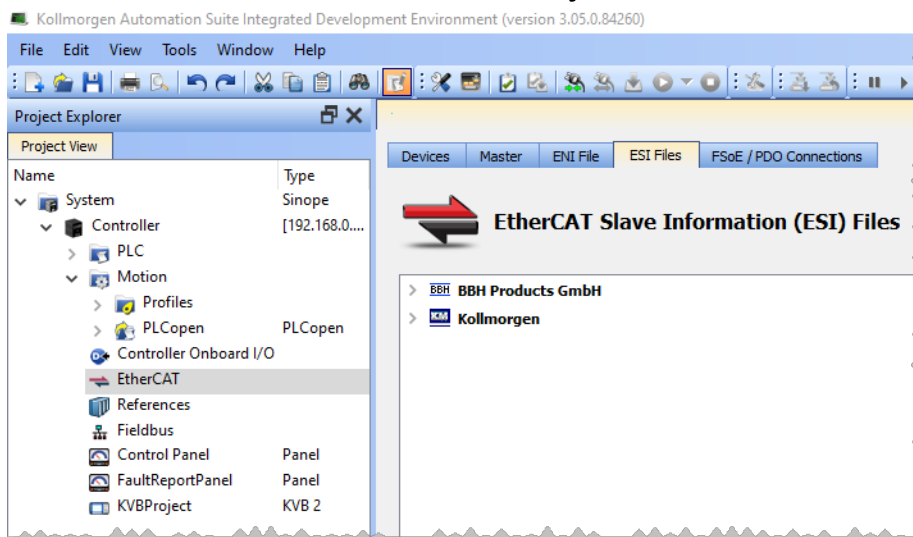
### 6.4.18 Add & Configure Third Party Devices

EtherCAT devices which are physically connected to the network can be scanned and discovered. EtherCAT devices may also be manually added to the EtherCAT node in the Project view. Manually added devices allow the project to be configured before connecting.

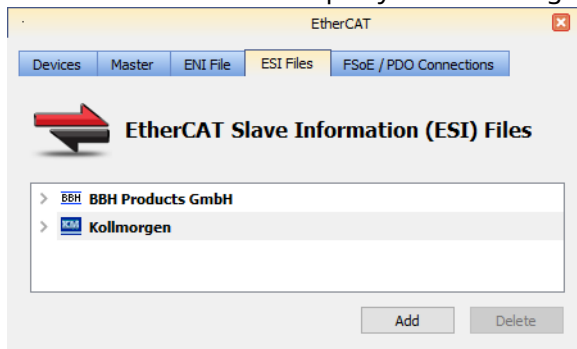
#### NOTE

- KAS IDE does not support third party drives.
- The KAS IDE supports third-party FSoE Masters. BBH SCU-1-EC support is integrated with the KAS IDE and is recommended. Please contact BBH for the latest ESI file that matches with the SCU-1-EC firmware version. For more details, see the [BBH web-site for SCU-1-EC](#). Other third-party FSoE Masters are possible, but require the FSoE/Connected PDOs to be configured manually by the user.

1. Double-click on the EtherCAT node in the **Project View** and select the **ESI Files** tab.

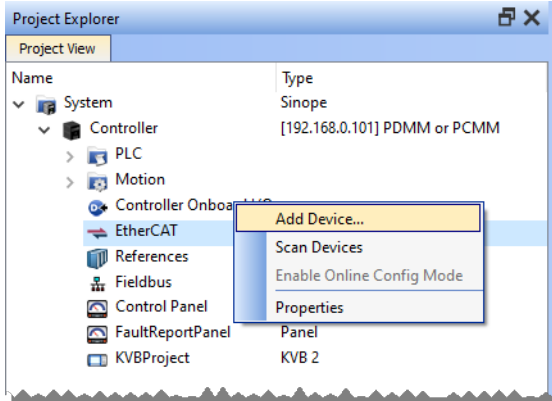


2. Add ESI files from a third-party vendor using the **Add** button.

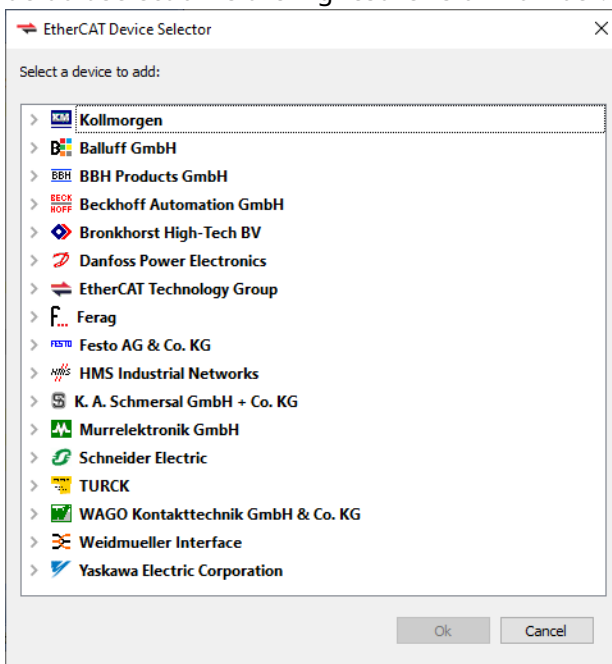


3. Navigate to where you have the third-party ESI file saved on your computer and click **Open**. This will add the manufacturer and device to the list of available devices.

- Right-click on the EtherCAT node and select **Add Device...** from the menu.



- Select the device from the **EtherCAT Device Selector** dialog box. If a specific revision number is required, it may be selected from the drop-down list in the revision column. The default selection is the highest revision number.



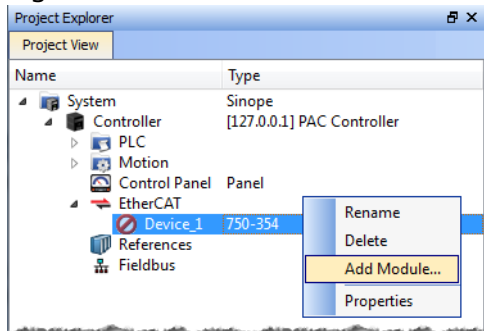
- Click **Ok**.

The device is added to the project tree in the EtherCAT node.

### 6.4.18.1 Add Modules to Third Party EtherCAT Devices

Modules may be added to third party MDP (Modular Device Profile) EtherCAT devices in the Project view. This helps to preconfigure the EtherCAT network in the project before connecting to the controller.

1. Right-click on a MDP EtherCAT device and select **Add Module...** from the menu.



2. Select a module from the EtherCAT Module Selector dialog box.
3. Click OK.

The module is added to the project under the EtherCAT device.

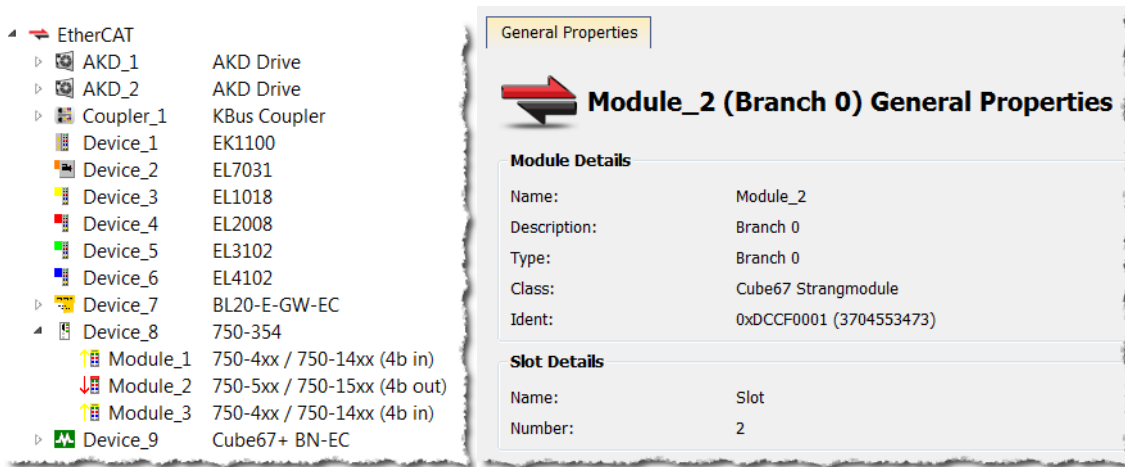
#### 6.4.18.1.1 Limitations

When adding modules from the Project View, there are several limitations to be aware of.

- Modules cannot be manually added to devices that are mapped to physical devices. Modules may only be added to an unmapped, manually added MDP device.
- Manually adding a MDP device will not automatically add mandatory modules. Either consult the device documentation to identify the mandatory modules and add them manually or discover the mandatory modules under the physical MDP devices, using the EtherCAT network scan.

### 6.4.18.2 Modules – General Properties Tab

A device may have no modules, or it may have several. Double-clicking or right-clicking on the module allows access to the General Properties tab. This tab provides information about the selected module.



**Figure 4-32:** Devices in the EtherCAT list and the General Properties tab of a Device's Module.

The tab is divided into two sections, Module Details and Slot Details. Module Details provides the module's basic information. Slot Details provides the module's slot information.



### 6.4.19 Use Pre-defined Libraries

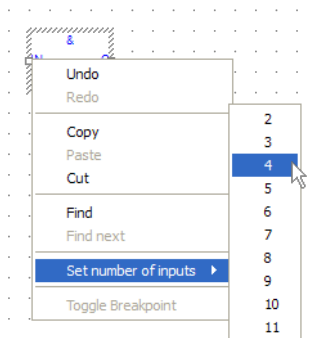
The [Libraries](#) toolbox allows you to select the functions.

#### **TIP**

- The **(All)** category at the top enables you to see the full list of available blocks.
- You can access a specific function by entering its initial letters on the keyboard (if the elapsed time between two strikes is greater than 1 second, the KAS IDE considers the last letter as the new initial).

#### Drag-and-drop into the editors

1. When the function is selected, move it with a drag-and-drop operation in the program editor
2. In the editor, right-click on a function to set the number of input pins if the block allows an extension.



**Figure 4-33:** Set the Pins Number of the Block

#### Drag-and-drop into the dictionary

If you have selected a function block, you can drag-and-drop it in the program declaration within the Dictionary toolbox, to create an instance of that object.

## 6.4.20 Create and Use Custom Libraries

You first need to create a custom library before you can use it to define a new item: function, function block or variable (for more details on library usage, refer to [Use the Custom Library](#)).

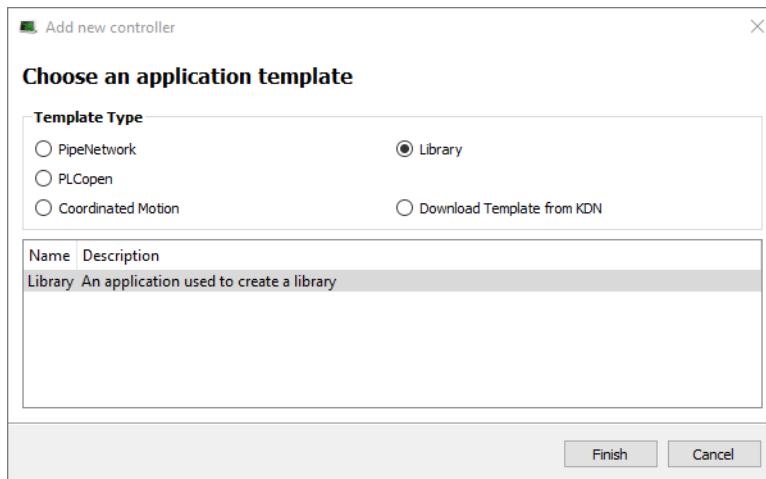
### NOTE

There is a difference between **Libraries'** usage and the **Import / Export** commands related to PLC programs.

- **Import/export** is equivalent to a copy and paste operation of programs: when you update the source of your UDFB, the other programs are not updated because the code has been duplicated.
- **Library** is a unique source that can be shared between different projects (like a dll in C): when you modify the library, all the linked projects are impacted.

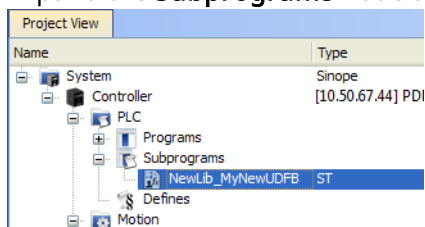
### 6.4.20.1 Create the Custom Library

1. In the **File** menu, click the **New** command (save your current opened project if necessary)
2. Select the **Library** application template



**Figure 4-34:** Create a Custom Library - Select the Library Template

3. Click the **Finish** button
4. Click the **Save As** command in the **File** menu
5. Define the Library Name (extension \*.kal) and its Location
6. Click **OK**
7. In the Project Explorer, expand the **Controller** and **PLC** nodes
8. Right-click on **Subprograms** and choose **New UDFB** in the contextual menu, then select the type of programming language
9. Expand the **Subprograms** node and rename the new UDFB



### NOTE

It is the name of the variable type which is displayed in the dictionary if you use this library in another project.

**IMPORTANT**

Duplicate UDFB names in a library are not possible. Only the first instance found is kept when importing the library definitions in a project.

To avoid this situation when designing your libraries, use a prefix to identify the library for all UDFBs and functions in the libraries (in the current procedure, the prefix is: NewLib\_).

10. Create the UDFB program (for more details, refer to [Create Programs](#))
11. In the **File** menu, click the **Save** command

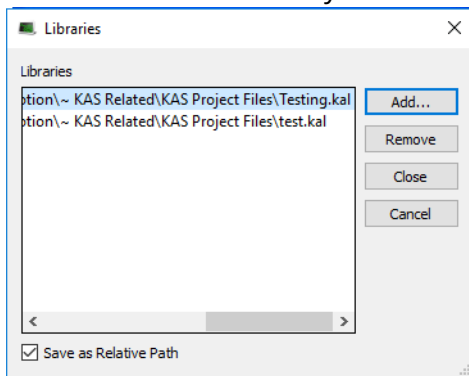
**6.4.20.2 Use the Custom Library**

1. Open the project where you want to use a library
2. In the Project Explorer, expand the **Controller** node
3. Right-click on **PLC** and choose **Libraries** in the menu
4. Click **Add**

**NOTE**

You can add as many external libraries as you want

5. Select the \*.kal file already created before and click **Open**



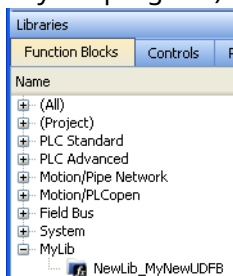
**Figure 4-35:** Use a Custom Library - Select the Library

**TIP**

You can use the **relative** path to specify the path relative to the working directory where your project is saved. This ensures consistency when you move your project and your library.

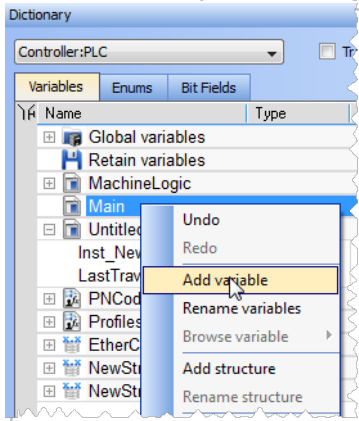
Conversely, the **absolute** path points to the same location on your file system regardless of your project directory.

6. Click **Close**
7. The library is displayed in the Library widget and you can now drag-and-drop the UDFB (or any subprogram) of this library in any editor



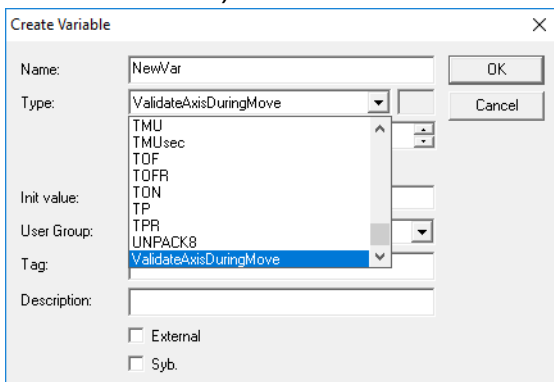
**Figure 4-36:** Use a Custom Library - Display the Library

- In the Dictionary toolbox, right-click on the program and choose **Add variable** in the menu



**Figure 4-37:** Use a Custom Library - Add a Variable

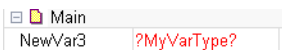
- In the Type drop-down menu, select the type defined in the external library (it can be at the bottom of the list)



**Figure 4-38:** Use a Custom Library - Select the Type

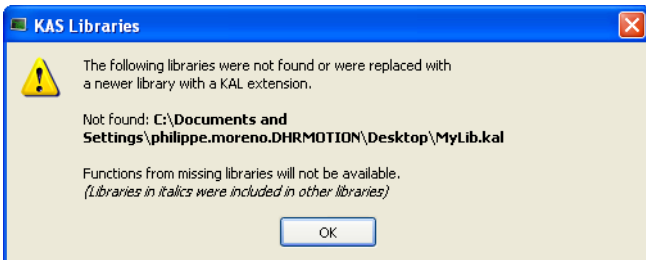
**What happens when you remove a library from your project?**

If you remove a library from your project, all its types are removed from your project and all variables based on the library are displayed in the dictionary in red with question marks



**What happens when a library no longer exists?**

If you open a project containing a link on a library which is no longer available, a warning is displayed:



To recover the libraries, you have two options:

- Enter the new path to this library (assuming it still exists on your machine) using the library dialog (see [Use the Custom Library](#)).
- Find the missing library and copy the library back to the path originally specified. Note that the project has to be closed and re-opened for the library to be read again.

*Broken link displayed in **Italics***

If a library references another library which is no longer available, a dialog with the library link that causes the problem is displayed in italics.

It can happen for example if your project has referenced LIB-4, which in turn references LIB-1-ND, but LIB-1-ND does not exist.

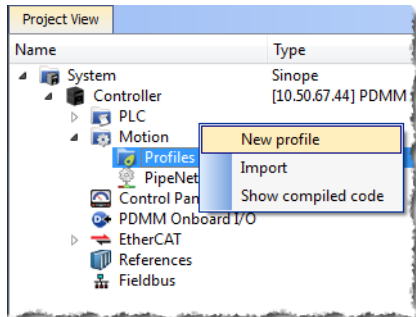
To recover your project, you have to open LIB-4 and fix the issue (i.e. LIB-1-ND broken link), then re-open this project again.

## 6.4.21 Adding Cam Profiles

### 6.4.21.1 Create Cam Profiles

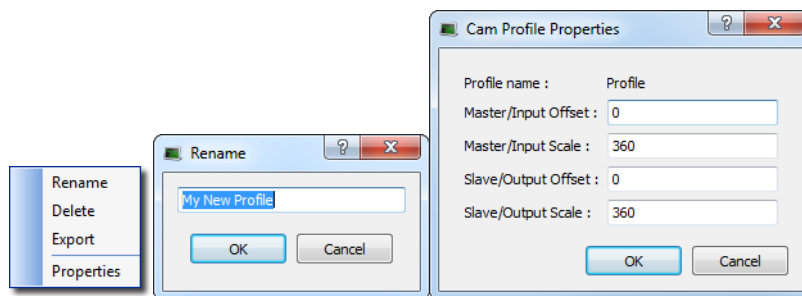
To create a cam profile, do as follows:

1. In the Project Explorer, right-click the **Profiles** item and select the **New profile** command in the contextual menu



**Figure 4-39:** Cam - New Profile

2. A new profile named "Profile" is created with default parameters.
3. Right-click on the new profile to rename it or change its properties.



**Figure 4-40:** Cam - Define Profile Filename

4. Click on the new profile to edit it.

Field	Description
<b>Profile name</b>	<p>The name of the Profile which is:</p> <ul style="list-style-type: none"> <li>• displayed in the Project Explorer</li> <li>• used in the <b>Properties</b> of the cam Pipe Block. <a href="#">see "Use Cam Profiles" on page 201</a>.</li> </ul>

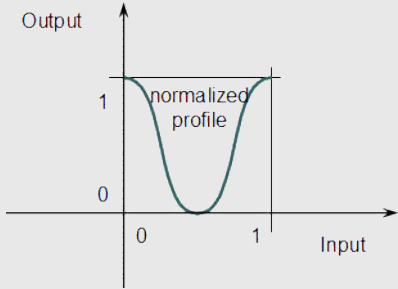
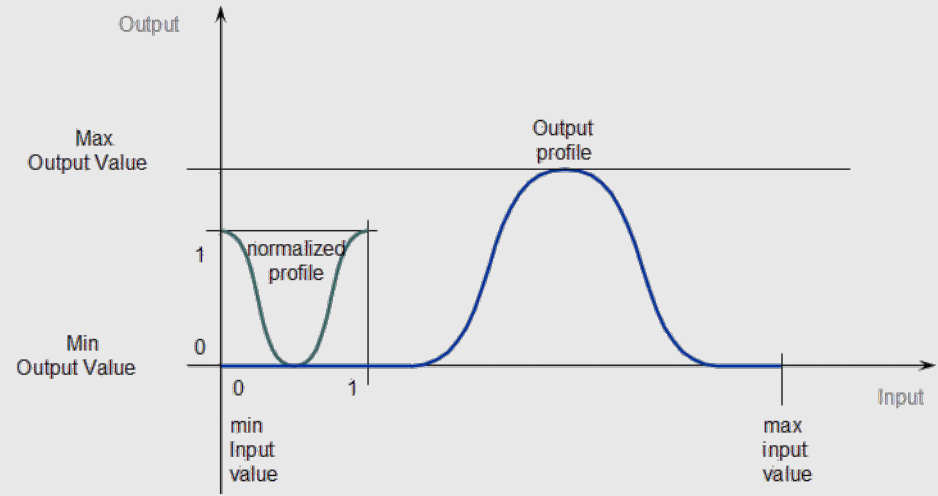
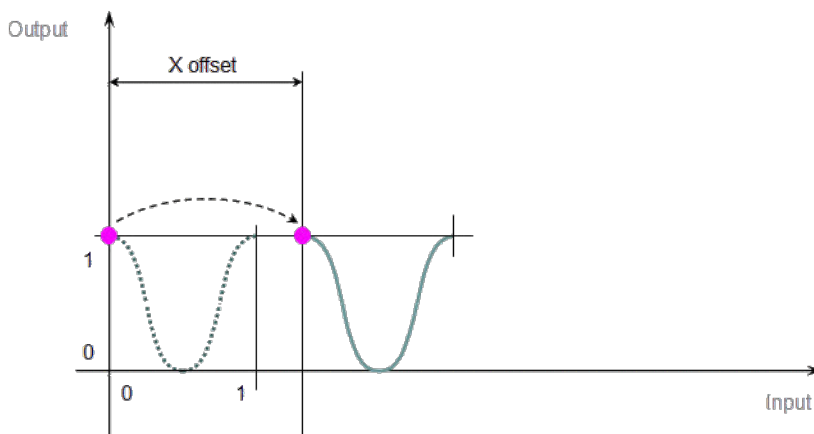
Field	Description
Master/Input Offset	<p>Offsets and scales on the X and Y axes transform the normalized profile into the output profile as shown on the two figures below:</p>  <p><b>Figure 4-41: Cam - Normalized Profile</b></p>  <p><b>Figure 4-42: Cam - Output Profile</b></p> <p>For more details, refer to the paragraph below: <a href="#">Four Parameters Transforming the Cam Profile</a></p>
Master/Input Scale	
Slave/Output Offset	
Slave/Output Scale	

Table 4-1: Cam Profile Parameters

*Four Parameters Transforming the Cam Profile*

**Master/Input offset**



**Figure 4-43: Cam Profile Transformation - Step 1**

### Master/Input scale

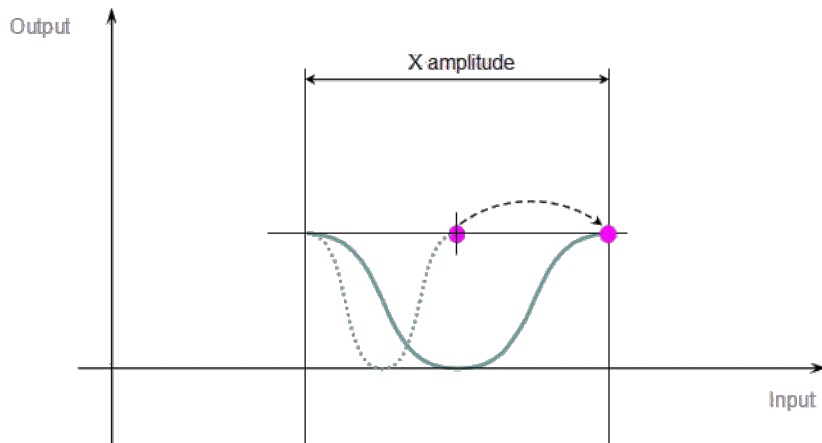


Figure 4-44: Cam Profile Transformation - Step 2

### Slave/Output scale

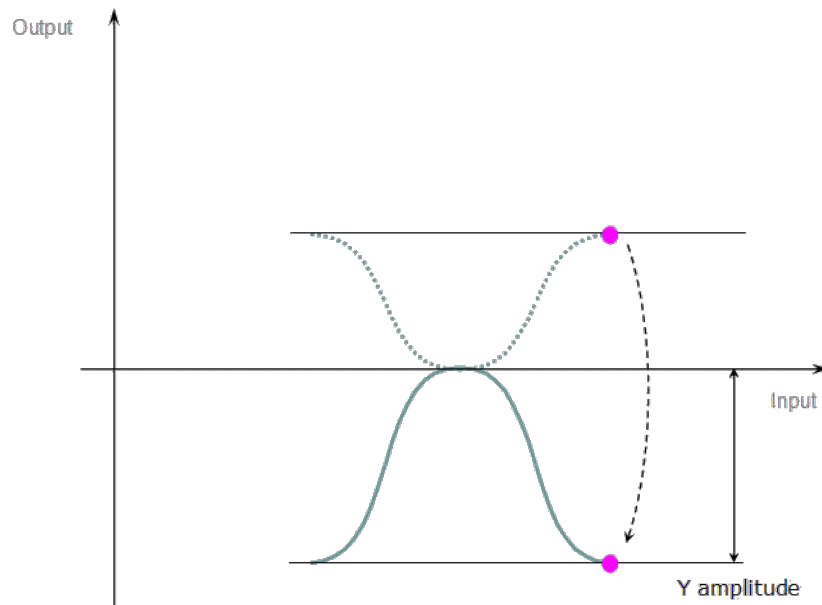
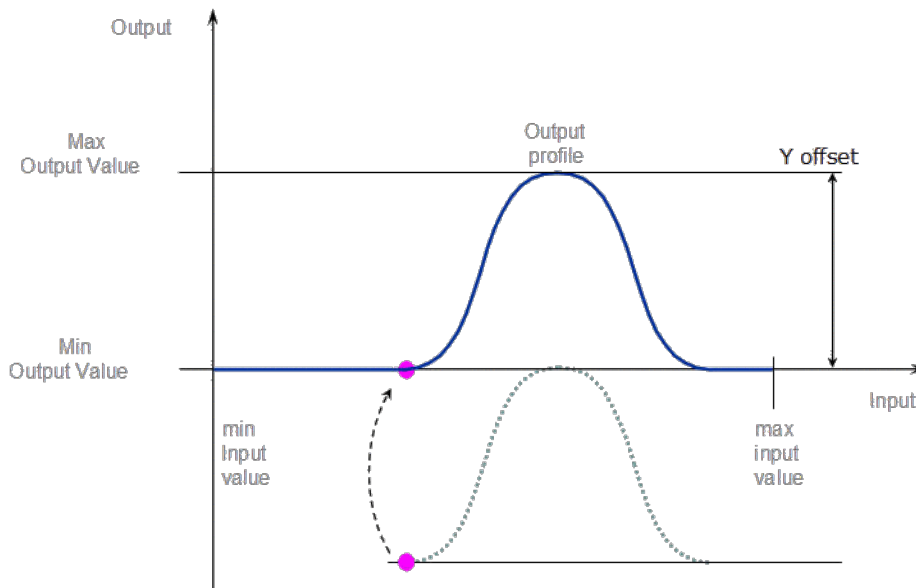


Figure 4-45: Cam Profile Transformation - Step 3

### Slave/Output offset





**Figure 4-46:** Cam Profile Transformation - Step 4

#### NOTE

When you change a Cam Profile property, a dialog box indicates the progression of the operation.

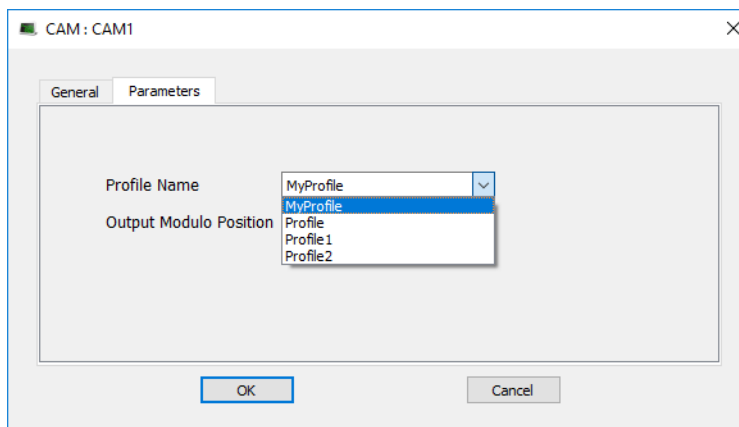
For more details about editing the profile, refer to [Cam Profile Editor](#).

See also: [Use Cam Profiles](#)

#### 6.4.21.2 Use Cam Profiles

Once defined (see ["Create Cam Profiles" on page 198](#)), you can associate a cam profile to a cam Pipe Block in the Pipe Network as follows:

1. Right-click on the cam Pipe Block and select **Properties** in the menu
2. In the **Parameters** tab, enter the profile's name



**Figure 4-47:** Cam - Associate Profile to a Pipeblock

#### NOTE

Separating the declaration of the cam Pipe Block from the cam profile provides the capability to prepare several different cam profiles and then apply one of them to the cam Pipe Block.

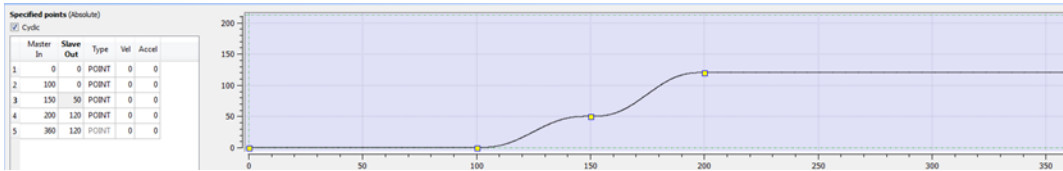
#### NOTE

If you change the profile's filename, do not forget to update the cam Pipe Block accordingly.

### 6.4.21.3 Cam Profile Switching

At pipe network start-up or when using the [MLCamSwitch](#) function block, a Cam block's output position is set to the output position of the initialized profile.

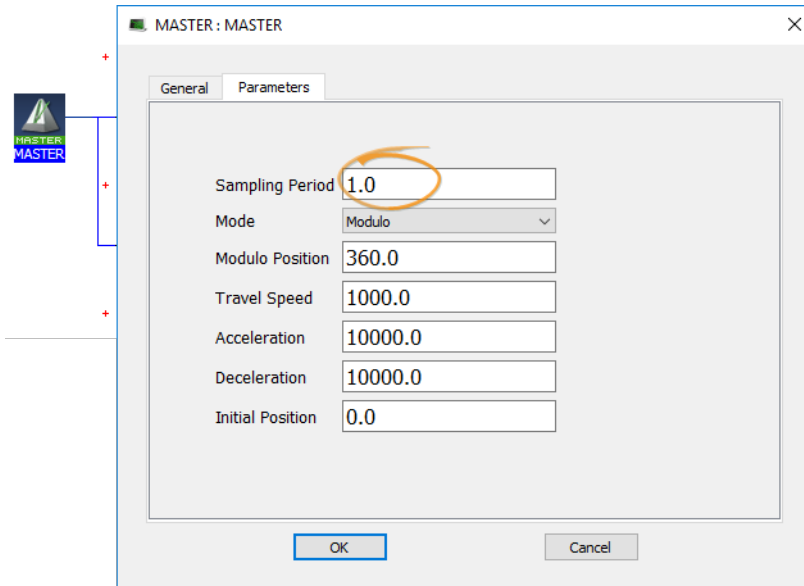
For example if the following profile is switched in and the present pipe network position at the input of the cam block is 150 the position at the output of the cam block will be 50. Then as the input to the Cam block increases or decreases the Cam output value will increase or decrease respectively. And it will go through the modulo without a position jump.



## 6.4.22 Define Scheduling

### 6.4.22.1 Periodicity

The period of execution of a pipe is the time spent between two successive computations of set values for the same pipe. The period of execution of a pipe is specified by the PERIOD parameter of the input Pipe Block.

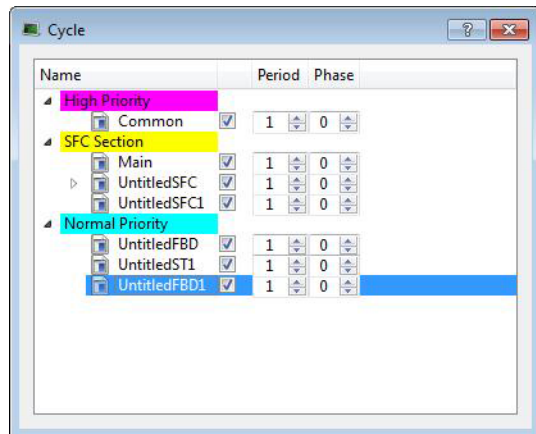


**Figure 4-48:** Set the Period of Execution

All the pipe values are computed independently of events and sequences execution.

### 6.4.22.2 Order of Execution

The order of execution of programs within a project is determined from the Cycle window.



The order of execution is always

1. High Priority
2. SFC Section
3. Normal Priority

The order within each grouping is determined by the vertical ordering. Using the image above, the order of execution for Normal Priority programs is:

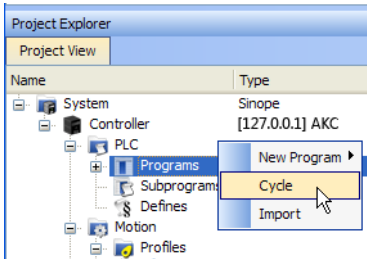
1. UntitledFBD
2. UntitledST1
3. UntitledFBD1

For more information see [Define the PLC Cycle](#).

### 6.4.22.3 Define the PLC Cycle

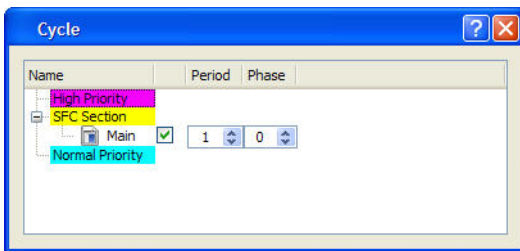
The cycle specification defines the number of cycles between successive executions of the programs.

1. In the Project Explorer, expand the PLC node and right-click on the Programs item to open the contextual menu and select the **Cycle** command



**Figure 4-49:** Edit the Cycle

The Cycle window allows the regulation of the following parameters: Period and Phase.



**Figure 4-50:** Define the Cycle

The cycle configuration dialog box is used to configure the programs priority into the Virtual Machine.

Column	Description
<b>Name</b>	List of PLC programs grouped together by priority level. The priority defines the order of execution. The SFC programs have a specific section as they must be grouped together.  <i>High Priority</i> means "executed before SFC", and <i>Low Priority</i> means "executed after SFC".
<b>Check box</b>	Enables or disables the execution of the corresponding program.
<b>Period</b>	Defines how many cycles are set between two executions of the program. You can define various sampling periods for programs of the application. Default period is "1" (the program is executed on each cycle). Giving a slower period to some programs is an easy way to give higher priority to some other programs.

Column	Description
<b>Phase</b>	<p>Defines an offset that enables you to dispatch slow programs among few cycles.</p> <p>The goal of postponing the program execution is to reduce execution peak loads.</p> <p>Example:            a program with period=2 and Phase=1 is executed each <b>even</b> cycle            a program with period=2 and Phase=0 is executed each <b>odd</b> cycle</p>

Table 4-2: Cycle Parameters

In the **High** and **Normal** Priority sections, you can adjust the order of the programs with a drag-and-drop operation according to the expected sequence. In each section, the program on the top is executed first.

Select the program you want to set with a higher priority, then drag and drop it to the relevant priority level.

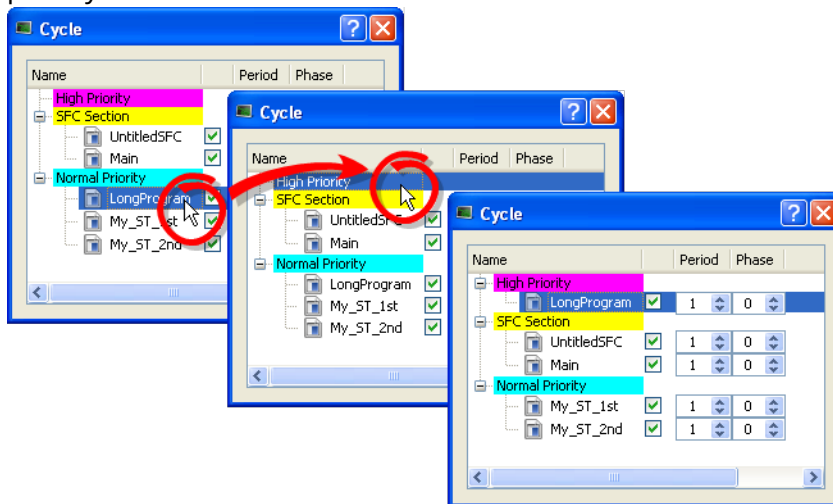


Figure 4-51: Change Priorities by Defining the Cycle

If all programs are with a Period set to 1, the KAS IDE is more loaded. The choice of the Period for the programs gives you the possibility to distribute the load of the application.

[Tasking Model / Scheduling and Order of Execution.](#)

#### 6.4.22.3.1 How to specify the duration of a cycle

This parameter is defined in [EtherCAT Master Settings Tab](#).

#### 6.4.22.3.2 Ensuring Variables are Exported

Program Organization Units (POUs) which contain variables (see ["Map Variables to HMI" on page 171](#)) must be compiled in order for the variable to be exported. For example, in the following set of images we see a POU (*UntitledST*) with two variables, *NewVar* and *NewVar1* and only *NewVar1* is set to be exported (1). The POU, however, is not set to be executed in the Cycle dialog box (2). This will cause a compile error (3).

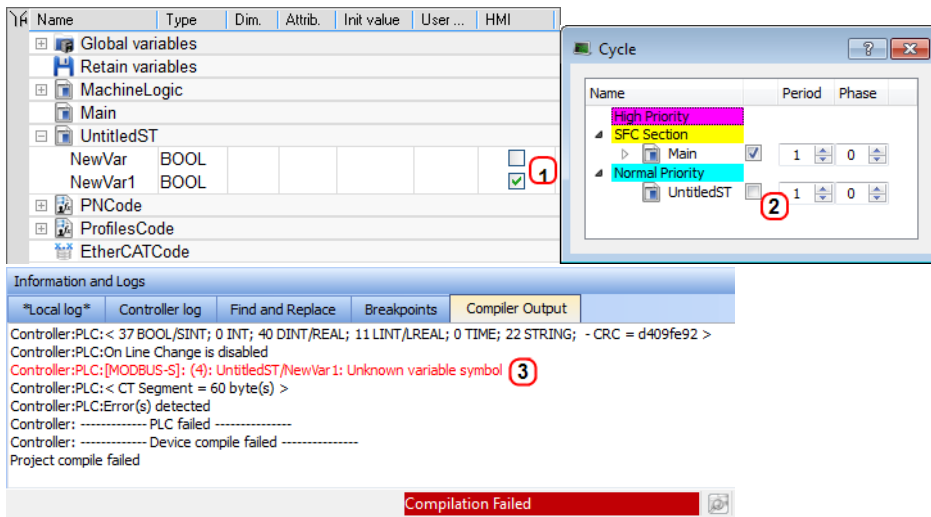


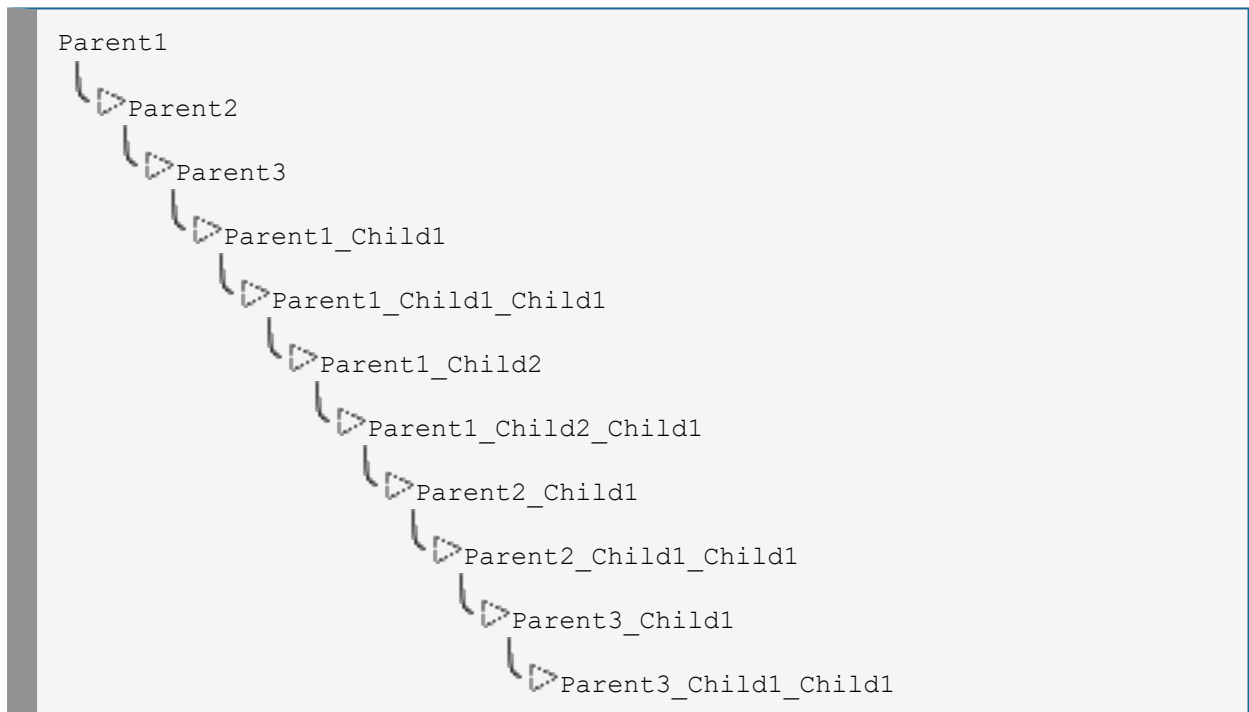
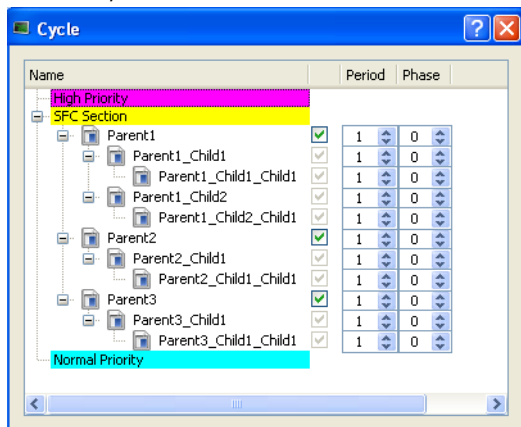
Figure 4-52: Example of a variable not being exported and the resulting compile error.

#### 6.4.22.4 About Parent-Child relationships and execution order

The SFC Section allows for editing the period and phase values of parent and child programs. Parent and child programs follow certain rules:

- A parent program can be enabled or disabled. If a parent is disabled the child will also be disabled. A child program cannot be disabled.
- Parent programs are allowed to move across priorities. Child programs will follow the movement of a parent. Child programs are not allowed to move independently.
- When a child program is created or imported, it will inherit the enabled/disabled state of the parent program.
- The SFCs are executed at the set cycle period and phase. All parent programs will be executed first and then the children programs will be executed in order.

To understand the last rule, consider the following Cycle example. There are three parent programs, each with a number of child programs. All parents are executed, followed by the children, in order. The actual flow is illustrated below the image.



**⚠ IMPORTANT**

*Parent SFCs should run faster than their children.* If this is not the case, the stop condition can be vague. When a child runs slower than its parent it does not stop when the parent stops, but at the child's next execution. This means the parent could execute more, while the child is still running.

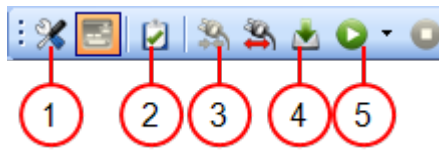
**👉 TIP**

A child program is initiated at Phase 0 in respect to its parent.



## 6.5 Running the Project

This chapter explains how to build, download and run your project.



Step	Description
1	Set the compilation options to run your project in Debug or Release modes, and choose if you want to activate the Online Changes
2	Compile the application and see all the remaining warnings and errors
<div style="background-color: #f0f0f0; padding: 5px;"> <p><b>NOTE</b></p> <p>Before step 3, you need to start the KAS software (KAS Simulator or KAS Runtime) on the target device where you want to run your project.</p> </div>	
3	Connect the KAS IDE to the target device
4	Download the Application compiled on the KAS IDE to the target device
5	Start / stop the device, and control your application with the script commands

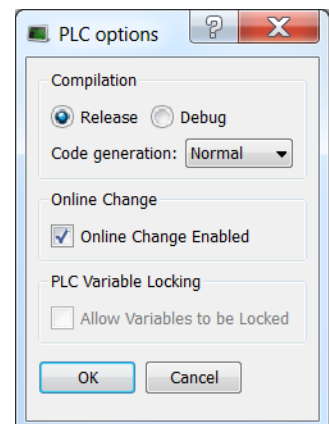
### 6.5.1 Set the Compilation Options

You can open the **PLC options** with the  icon.

If you want [step-by-step debugging](#) to be available during simulation or online testing, you need to select the **Debug** compiling mode. If step-by-step debugging is no longer required, select the **Release** compiling mode in order to give highest performance to your application.

When you incorporate additional statements (such as trace outputs) in your code, you must select the **Debug** compiling mode so that they are taken into consideration (in RELEASE mode, those statements are not included).

- For Conditional Compiling, [Conditional Compiling](#)
- For Online Changes, [PLC Online Change](#)



#### Code generation

There are two code compilation options, **Normal** and **Optimized**.

- **Normal** code allows for use of all PLC features in the KAS IDE and Runtime.
- **Optimized** code allows for faster PLC execution with the following limitations:
  - [Debug](#) features are disabled, including breakpoints and step-by-step debugging.
  - [PLC Online Change](#) is unavailable.
  - [WAIT / WAIT\\_TIME](#) are not supported.
  - Supported by AKD PDMM and PCMM controllers.

*Why select Normal?*

Normal code generation creates P-code and executes inside a virtual machine.

**Size constraints:** The P-code is smaller than the same program translated to "C-code".

**For development and debugging:** Normal code allows you to set breakpoints, step through PLC code, and use online change. Additionally, as normal code is interpreted (which means that

the code is read by the KAS Runtime engine that then determines the instructions to run), the interpreter can apply many additional runtime checks that would be harder to implement with optimized code.

*Why select Optimized?*

Optimized code generation creates C code (as in the C programming language), which is compiled into native machine code, and executes directly.

**Execution speed:** Optimized code executes significantly faster, allowing the controller to run more complex PLC applications.

#### TIP

Using **Normal** code is recommended in order to take advantage of all KAS PLC features, unless the controller is unable to meet performance requirements.

### 6.5.1.1 Conditional Compiling

The compiler supports conditional compiling directives in ST, IL, FLD, and FBD languages. Conditional compiling directives condition the inclusion of a part of the program in the generated code based on pragma. Conditional compiling is an easy way to manage several various machine configurations and options in one unique application project.

Conditional compiling uses [definitions](#) as conditions. Below is the main syntax:

```
#ifdef CONDITION
    statementsYES...
#else
    statementsNO...
#endif
```


If CONDITION has been defined using [#define](#) syntax, then the "statementsYES" part is included in the code, else the "statementsNO" part is included. The "#else" statement is optional.

#### TIP

[Intellisense](#) facilitates the reading by coloring in gray the part of the program which is not active.

**How to define conditional compiling directives?**

Languages	Description
ST and IL	Directives must be entered alone on one line of text <pre>#ifdef DEF_A1_PeriodicAxis MLPhaSetPhase(PipeNetwork.PHASE1,DEF_A1_PosPeriod-A1_RefPos4); #else MLPhaSetPhase(PipeNetwork.PHASE1,DEF_A1_LinearPeriod -A1_RefPos4); #endif</pre>
FBD	Directives must be entered as the text of <a href="#">network breaks</a>

Languages	Description
FFLD	<p>Directives must be entered as a network <b>pragma</b> with the  icon.</p> <p>In the example below, if <b>CONDITION</b> has been defined using <code>#define</code> syntax, then the networks <b>2 to 4</b> are included in the code, else the networks <b>5 to 12</b> are included.</p> <pre data-bbox="411 394 1010 1126"> # Network #1 E-stop both axes when the E-stop button is pressed # ifndef CONDITION Network #2 Reset the axis errors of both axes when the Reset button is pressed  Network #3 Close the servo loop and enable the drive when the Enable button is pressed. Open the servo loop and disable the drive when the Disable button or the E-stop button is pressed. Network #4 Close the servo loop and enable the drive when CloseLoop is high. Open the servo loop and disable the drive when CloseLoop is low.  # else Network #5 Get the Axis 1 actual position for the Control Panel to display  Network #6 Get the Axis 2 actual position for the Control Panel to display Network #7 Raw feedback positions Network #8 Read the states of the axes Network #9 If both axes have no errors and are enabled, turn on the Ready indicator Network #10 Monitor Velocity of each Axis Network #11 EtherCAT status word of each Axis # endif Network #12 Drive Fault observer  End of Module </pre>

**NOTE**

Conditional compilation do not apply to actions in an SFC step.

The condition "`__DEBUG`" is automatically defined when the application is compiled in **DEBUG mode**. This allows you to incorporate some additional statements (such as trace outputs) in your code that are not included in **RELEASE mode**.

```


#ifdef __DEBUG
    Printf('In debug mode', 0, 0, 0, 0);
#endif

```

### Running the Project

### 6.5.2 Compile the Application

After creating all the elements of your project, you are ready to compile it. The project must be compiled before it is simulated or downloaded to the target.

You can compile your project with the compile icon  in the toolbar (**Ctrl+B** shortcut). The compiler reports messages in the Information and Logs toolbox (see [Compiler Output](#) tab). No other actions are possible when the compilation is in progress.

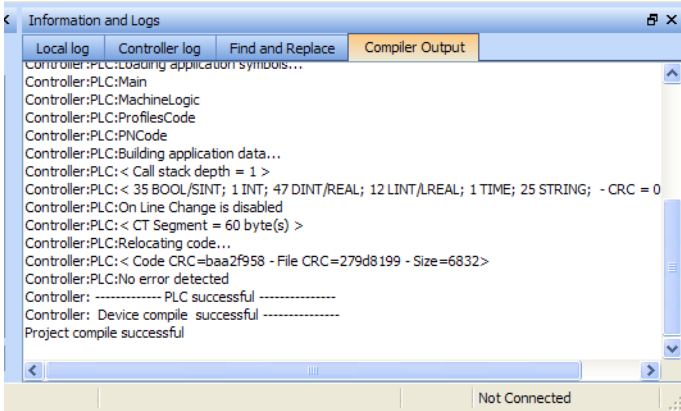


Figure 4-53: Compiler Output

**TIP**

Errors are easily located using the information and logs window as shown below.

Double-click on an error in the list to open the program and jump directly to the relevant location in the editor.

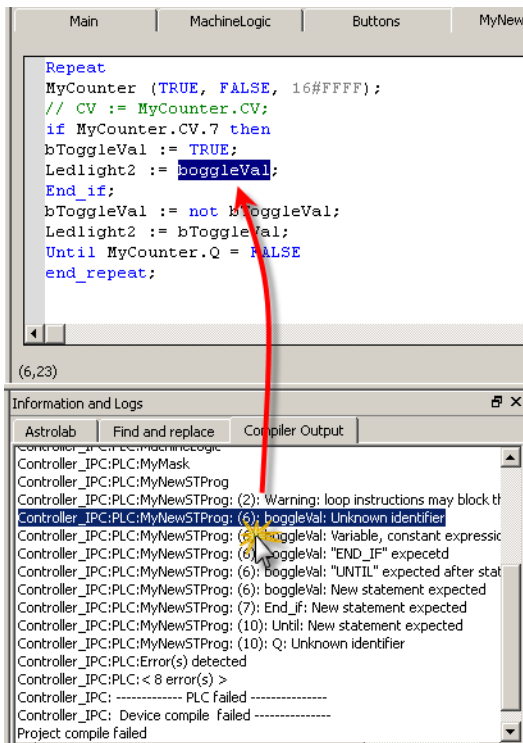


Figure 4-54: Error Location when Compiling

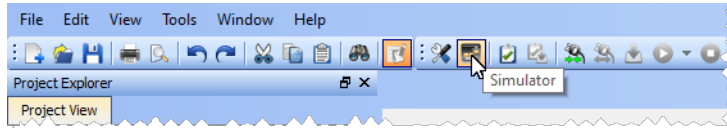
To locate source code, you can also use the **Find and replace** feature (for more information, refer to [Find and Replace Tab](#))

#### NOTE

In FFLD, when a function, function block or UDFB is not connected on the left, then it is ignored (removed at compiling time).  
This case only applies for functions - **not** for function blocks.

### 6.5.3 Launch KAS Simulator

If you want to simulate your application, click on the Simulator button in the toolbar.



Once the program opens, you can start the simulation using the Control Panel.

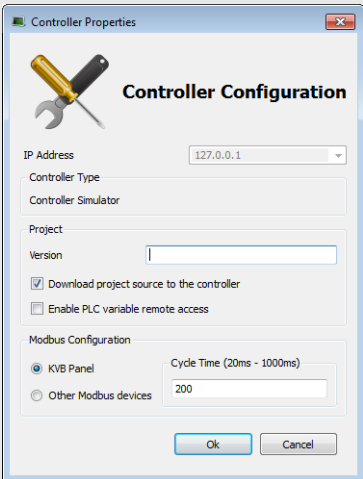
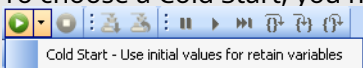
[Using the KAS Simulator](#)

### 6.5.4 Connect to the Controller

KAS provides all the commands for controlling the target in the **Device** toolbar:



Figure 4-55: The Device Toolbar


Icon	Description
	Show the controller communication properties dialog (PLC options) (see details below)
	Change the controller IP address to connect with KAS Simulator. In this simulated mode, the controller properties change as shown below and the Controller Type includes the IP address 127.0.0.1. 
	For more details, refer to <a href="#">Using the KAS Simulator</a>
	Compile project
	Compile and download the project. This downloads the application and Project Source files.
	Establish connection with the target controller (see possible statuses <a href="#">here</a> )
	Close connection with the target controller
	Download the application to the targeted controller (note that the application must <b>not</b> be running). For more details, refer to <a href="#">Download the Application</a>
	Start the application. It can be either: <b>Warm Start</b> (default mode): <a href="#">Retain Variables</a> are loaded at the application startup. They are <b>Not</b> re-initialized; whereas other variables are started with their initial values. <b>Cold Start</b> : use <a href="#">retain variables</a> with their default values. Such starts occurs from time to time but are few. To choose a Cold Start, you have to drop-down the Start button as follows: 
	Stop the application


**NOTE**

If the previous download failed (due to events such as the Ethernet being disconnected or the controller being turned off), you will not be able start your application. Additionally:

- no version information is shown in the web server or status bar
- the Start, Cold Start and Download buttons are not available on the web server
- if the AKD PDMM or PCMM tries to auto-start, an E24 error is flashed on the 7-segment display.

Review the Controller logs for more information.

Ensure the Simulated device mode is active (the icon  must be selected)

To establish the connection with the target controller, click the Connect Device icon .

**NOTE**

You need to configure the device before connecting (see [Configure the Controller](#))

**6.5.4.1 Actions to Prevent Compatibility Issues**

The software versions of the KAS IDE and the KAS Runtime have to match to avoid compatibility problems.

See [KAS IDE to Runtime Compatibility](#) for more information.

**TIP**

The software versions of the KAS IDE and the KAS Runtime are also available in the local [log](#) messages (the level for this message is INFO).

When another KAS IDE is already connected to the controller, a warning is displayed and the connection is discarded to prevent any conflict.

**6.5.4.2 Application Status Bar**

The status bar provides global information about the target and the name of the running application currently stored in the device.

Text displayed with **orange** background means that the version of the application is different between the KAS IDE and the target.

For more details, [The following table summarizes all cases for the labels of the status bar.](#)

**6.5.4.3 Message Window**

Every log message has the following information:

- Timestamp
- ID
- Message

**NOTE**



Once connected to the device, it is no longer possible to edit the PLC programs, unless Online Change is active (see [Compile the Application](#))

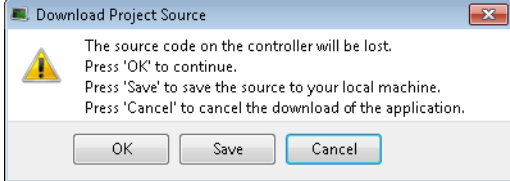
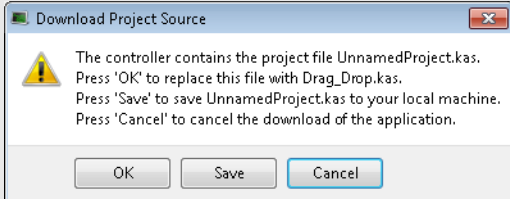
**IMPORTANT**

Depending on the number of AKD drives physically present in the EtherCAT network, the KAS IDE might slow down when getting data.

The KAS Runtime is **not concerned** with this limitation.

## 6.5.5 Download the Application

- Click the *Compile and Download*  button (**Ctrl + F7**) or the *Download*  button. The application is compiled and downloaded or just downloaded to the controller.
  - If you enabled the "Download Project Source in the controller" option in the Controller Properties dialog box (see [Configure the Controller](#)) then the source code is also copied to the controller.
  - If you disabled the "Download Project Source in the controller" option and the controller has project source loaded you will be prompted with a message. The message presented will vary, based on whether your current project and the project on the controller have the same name or not.

Project Names	Message	Actions
Same name		<p><b>OK:</b> Source on controller is deleted.</p> <p><b>Save:</b> Source on controller is overwritten and is also saved to "My Documents". If that source file already exists in My Documents you will be prompted to rename the source file, overwrite that file or not. If you do not rename or overwrite the file then the download to the controller will be canceled.</p> <p><b>Cancel:</b> Download is stopped.</p>
Different name		<p><b>OK:</b> Source on the controller will be replaced.</p> <p><b>Save:</b> Source on the controller will be saved to "My Documents" as above.</p> <p><b>Cancel:</b> Download is stopped.</p>

- You will be prompted with an alert to connect to a controller if you try to Compile and Download without being connected.
- You will be prompted with an alert if there is not sufficient space on the controller for the application. Clearing the [User Data](#) is one method to create more space on a AKD PDMM or PCMM.

### TIP

The versions between the KAS IDE and the KAS Runtime must be the same if you want to be able to debug your application (for example to display the animated values in the editors).

### IMPORTANT

If the IDE version differs from the runtime after compiling an application, the function blocks defined in the IDE and those implemented in the virtual machine of the runtime can possibly be different. To prevent this potential mismatch, you must compile and download your application again.

### NOTE

In addition to downloading the application to the controller's flash memory, you can download the project's source code, allowing you to store the project. See [File Menu](#) for information on retrieving saved projects.



### 6.5.5.1 Application Status Bar

The tooltip of the application [status bar](#) gives more information about the application stored in the target: name of the project, name of the device, version of the application, its build number and date of compilation.

To view the tooltip, hold the mouse over the application status bar and wait for 1 or 2 seconds without moving the mouse.

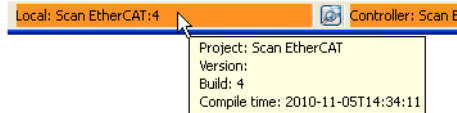




Figure 4-56: Device Tooltip displays Version

## 6.5.6 Device Control

### 6.5.6.1 Start/stop the Device

With the KAS IDE

You can start / stop the device with the buttons  and .

With the KAS Runtime

In the KAS Runtime menu you can click the **start / stop** command.

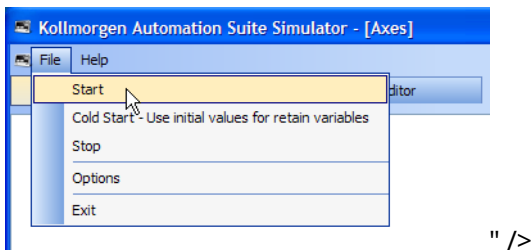


Figure 4-57: Start Device with the KAS Runtime

### 6.5.6.2 Log Window

The Log window displays all run-time messages issued by the device or by the KAS Simulator when testing the application.

The log area of the KAS IDE and the KAS Runtime Simulator are the same. It contains the log messages as described in [Information and Logs](#)

## 6.6 Testing and Debugging the Project

During system validation it is essential that the KAS IDE allows you to monitor the application program execution and to capture critical events and their data when they occur.


A Control Panel (designed with an [internal editor](#)) can be used to provide a basic interface.

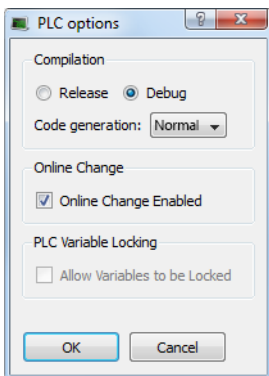
### 6.6.1 Step-By-Step Debugging

To minimize risk, the KAS IDE in conjunction with the KAS Simulator allows checking and validating the application program prior to deployment of the machine/system in production. This is achieved by capturing critical events in a step-by-step mode.

In addition to the cycle-by-cycle execution mode, the debugger has a rich collection of powerful features for making step-by-step debugging in the source code of your application.

#### NOTE

Step-by-step debugging is available only if the project has been compiled with the **DEBUG** option. This option can be selected from the project compiling PLC options dialog box, accessible with the  icon.



**Figure 4-58:** PLC Options - Debug Compiling Mode

- An application compiled in **Debug** mode includes additional information for stepping. This leads to bigger code size and reduced performance.
- When debugging is finished, it is recommended to compile your application in **Release** mode to give highest performance to your application.

Step-by-step debugging is available:

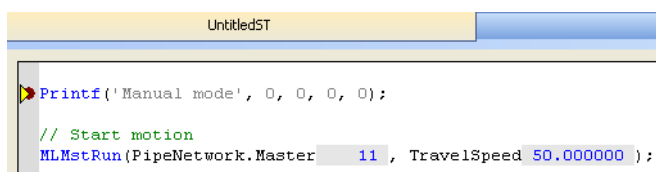
- In ST and IL text programs (a step is considered as a statement)
- In FFLD programs (a step is considered as a rung)
- In FBD (a step is considered as a graphic symbol corresponding to an action)

#### IMPORTANT

Step-by-step debugging is **not possible in SFC** programs, see [About SFC](#) for more information.

There are two possibilities for entering the step-by-step debugging mode:

- Set a breakpoint in a program (for more details, see ["About Breakpoints" on page 220](#))






When you start your application and the breakpoint is reached, the execution stops at the specified location and you can run one step further in the program with the stepping commands.

- When the target is in cycle stepping mode (STOP), you can step to the beginning of the first program.



#### TIP

Pausing a program will not interrupt the current VM cycle. The current cycle will finish and execution will be paused before the beginning of the next VM cycle.

The following commands are available from the [Debug Toolbar](#) for stepping:

Icon	Description
	Step Over the next instruction: If the next instruction is a call of a function block or a sub-program, the execution passes over to the following instruction.
	Step Into the next instruction: The next step will be at the beginning of the called block (if the next instruction is <b>not</b> a call of a function block or a sub-program, then the Step Into behaves like the Step Over)
	Step Out the current block: If the current stepping position is in a called function block or a sub-program, the execution continues up to the end of the current block. Otherwise, the Step out behaves like the Step Over.

In addition to these commands, you can click at any time:

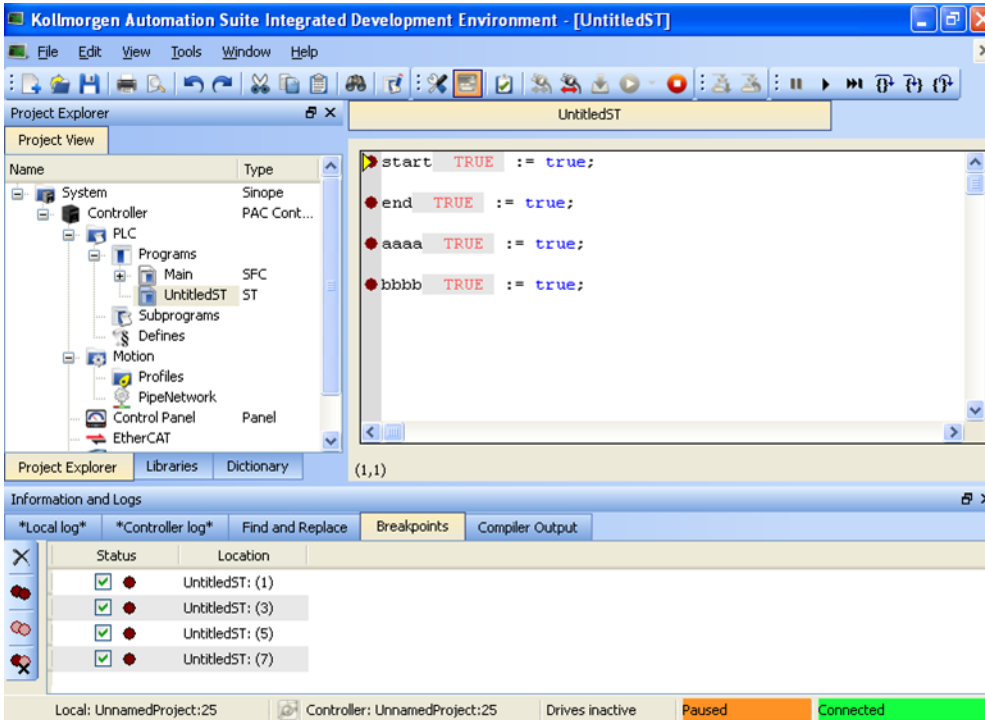
Icon	Description
	Execute the cycle (from the current position up to the end of the last program)
	Restart the target in "normal" execution mode (RUN)

## 6.6.2 About Breakpoints

The step-by-step debugging feature is enabled by setting breakpoints in the source code of the application.

### NOTE

This feature is only available when you have chosen the **DEBUG** mode (for more details, [Set the Compilation Options](#)).



### 6.6.2.1 About Breakpoints

- Breakpoints are a marker that is set in code which, when reached, stops the code's execution at that location. This lets you run one step further in the program with stepping commands.

#### TIP

Pausing a program will not interrupt the current VM cycle. The current cycle will finish and execution will be paused before the beginning of the next VM cycle.

- Breakpoints are shown as a red circle (dark or light) in the left margin.
- Breakpoints may be active (●) or inactive (○).
- Breakpoints are active only when the IDE is connected to a target running an application that is compiled from the exact code displayed in the editor.
- Breakpoints are inactive if:
  - the IDE is not connected to a target
  - the IDE is connected but not running
  - the IDE is connected to a different version of the code
  - the IDE is connected to the code but a modification has been made in Edit mode.
- Breakpoints will always be applied to the target, based on their position in the editor. If a breakpoint is moved in the editor, then you reconnect to a target, the breakpoint in the target will be moved to the new position.
- A Breakpoint that has been "hit" has a yellow triangle (▶ and ▶) to indicate it has been reached in the code.

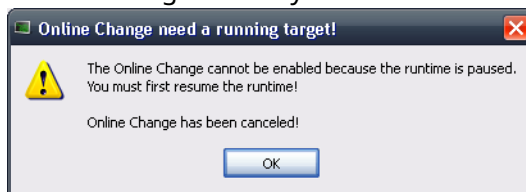
- Breakpoints are saved when saving the KAS application and are reloaded when loading a KAS application.
- See [Setting, Removing, Enabling, and Disabling Breakpoints](#) for information on working with breakpoints.
- See [Breakpoints tab](#) for information on the **Breakpoints** tab in the **Information and Logs** widget.
- Projects support a maximum of 16 breakpoints. This includes both enabled and disabled breakpoints.

**NOTE**

Breakpoints can significantly increase the PLC cycle time execution. This is due to the fact that the VM must evaluate the breakpoint condition at every cycle.

**6.6.2.1.1 About Online Change**

- Online Change cannot be enabled when the KAS Runtime is paused due to a breakpoint. Online Change can only be activated when the target is running.



- Every breakpoint is activated if an Online Change is performed successfully.

**NOTE**

The breakpoints are not activated synchronously but in a reasonable time.

- All breakpoints become inactive when an Online Change is reverted.

### 6.6.3 Setting, Removing, Enabling, and Disabling Breakpoints

This section discusses working with breakpoints within the editor. See [Breakpoints tab](#) for information on the **Breakpoints** tab in the **Information and Logs** widget, including modifying breakpoints in bulk. For background information see [About Breakpoints](#).

#### 6.6.3.1 How to Set Breakpoints

1. Open your program in the IEC 61131-3 Editor.
2. Click on the line (for ST/ IL) or diagram (for SFC 1 , FBD or FFLD) where you want to set the breakpoint.
3. Press **F9** or right-click and select **Add/Remove Breakpoint** from the menu.

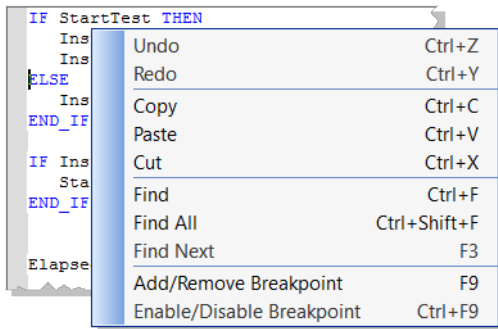


Figure 4-59: Setting Breakpoints

4. A Breakpoint circle is added to or removed from the left margin. The Breakpoint will be set as either active (●) or inactive (○), based on the IDE's connectivity (see [About Breakpoints](#)).

Even when you are **not** connected to the Controller, breakpoints can be placed in programs, sub-programs or UDFBs.

#### NOTE

When you start your application, if the current position is not on a valid line for stepping, the breakpoint is automatically moved to the nearest valid position.

#### IMPORTANT

When you close the connection with the target, all the breakpoints are removed in the KAS Runtime.

#### 6.6.3.1.1 About SFC

There are several things to note about breakpoints in SFC programs:

- In **SFC** programs, breakpoints can only be set on transitions (i.e. in First Level diagram), and not in steps or conditions. With a breakpoint set on a transition, you can debug cycle-by-cycle. Please remember that P1, N and P0 placeholders are designed to contain very simple code.

#### TIP

The recommended way to proceed for SFC sub-level programs is to rely on subprograms, where debugging is allowed.

- Breakpoints can be set and removed in SFC programs, they cannot be enabled and disabled.

#### 6.6.3.2 How to Enable/Disable a Breakpoint

<sup>1</sup>See limitation explained in paragraph below: [About SFC](#)

To enable a breakpoint, right click on an inactive breakpoint and select **Enable / Disable Breakpoint** (or Ctrl + F9). This is only available when the IDE and runtime are connected.

To disable a breakpoint, right click on an active breakpoint and select **Enable / Disable Breakpoint** (or Ctrl + F9). Selecting this option will remove the breakpoint from the runtime; the breakpoint will be remain in the editor and be changed to an inactive state (🔴).

### 6.6.4 Printf Function

You can use the Printf function to display string in debug mode.

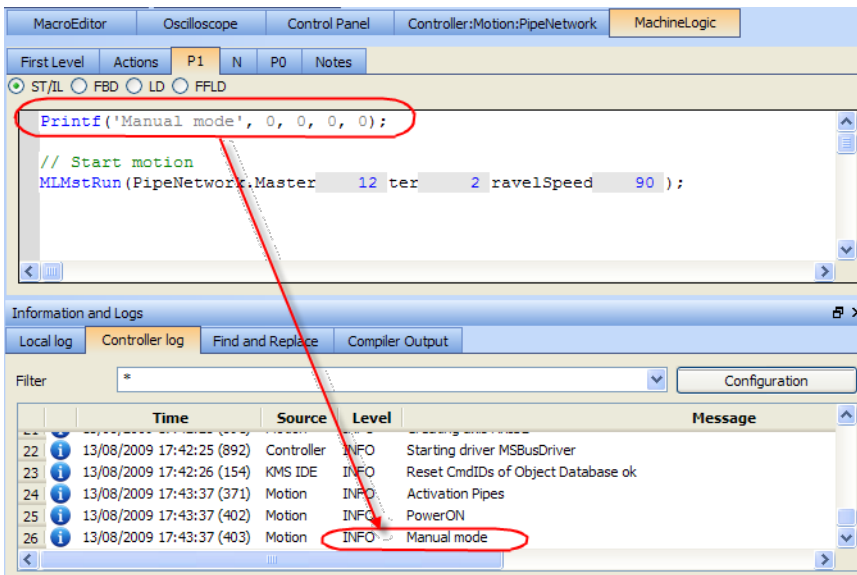


Figure 4-60: Printf Function

This is a good way to trace your SFC programs. Note that you can also use the PrintMessage (Function).

How to customize output in the log window?

#### Raise warnings or errors icons

First column in the log window displays an information icon which can be replaced with a warning or error icon as follows:

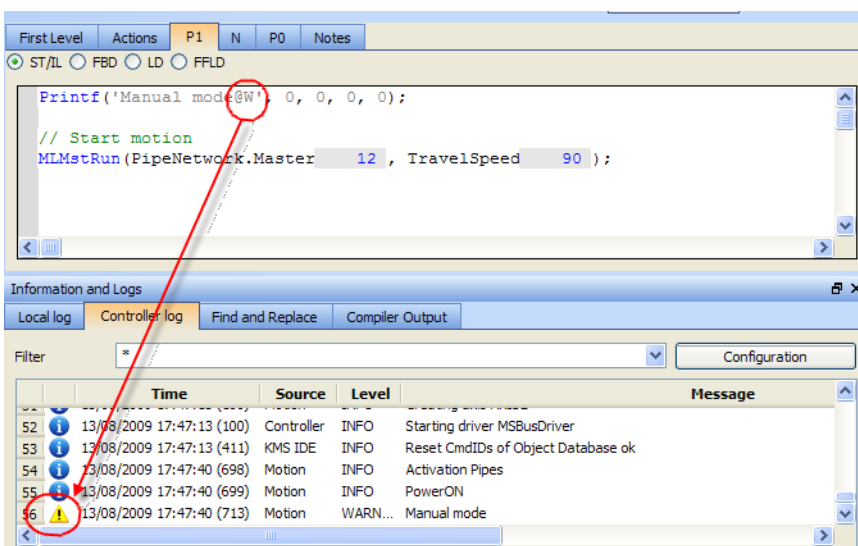
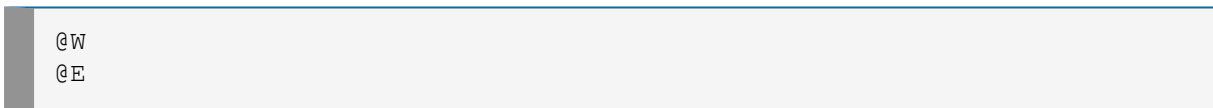


Figure 4-61: Customizing Output for Printf Function



### 6.6.5 Soft Oscilloscope Debugging

It can be interesting to access the values of the variables of the application. It is particularly important during development and debugging.

There is a way of visualizing and changing variables via the Graphics HMI panel (see [IEC 61131-3 Editor Debugging](#)). You can also access and change variables via the Variable Dictionary (see [Variable Monitoring](#)).

However, these two methods can only access and change variables from the PLC part and not from the Motion part of the application. Furthermore, the temporal evolution of the Motion variables would not be very intuitive. The ideal tool to trace the Motion variables is a softscope.

Other typical areas for using the softscope are:

- Recording when an input is sensed in a cycle
- Recording how much correction is being made in each cycle
- Checking the settling time of an axis

To open the Softscope, click the **Oscilloscope** command in the Tools Menu.

For more details on Softscope description and usage, refer to [Soft Oscilloscope](#)

#### 6.6.5.1 How to Plug Motion Variables

##### NOTE

The Softscope retrieves the variable values from the Motion Simulator. You can only plug objects which exist in the Motion Simulator. While the PLC variables exist all the time, the Motion objects are only created after the start of the application.

When your application is running, do the following:

1. Open the **PipeNetwork** of your Controller in the Workspace
2. Right-click on Gear1 to open its menu
3. Choose the command **Plug on channel...**

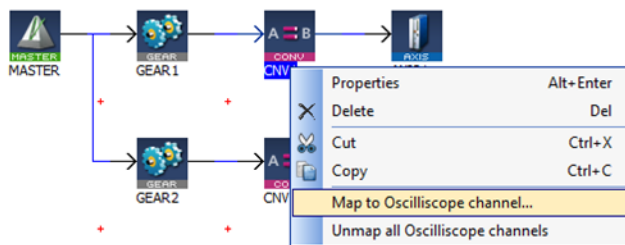


Figure 4-62: Plugging a Motion Variable

##### NOTE

Your application **must be connected and running** to plug a variable to the channel.

4. Set Channel to 1 and choose the relevant Data

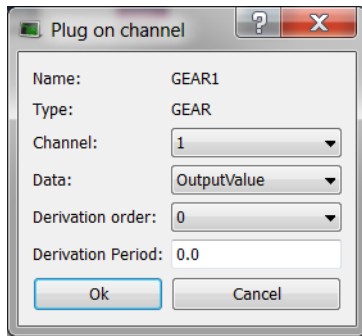


Figure 4-63: Plugging a Motion Variable - Parameters

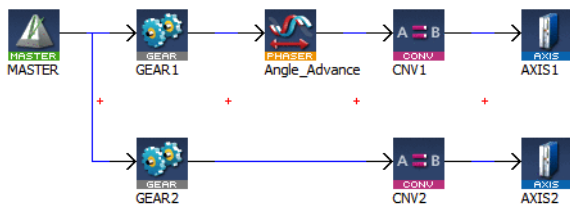
**NOTE**

The complete list of data are only visible when your application is running

For more details on the parameters, refer to [Plugging Probes, Mapping Variables](#)

6.6.5.1.1 Usage example with the Pipe Network

The Softscope allows the recording and display of motion at points any where in a Pipe Network. The following example shows the difference between the input and output of the Phaser Pipe Block (called AngleAdvance).



The red line is the input, the green line is the output and the blue line shows when the phase advance change was active.

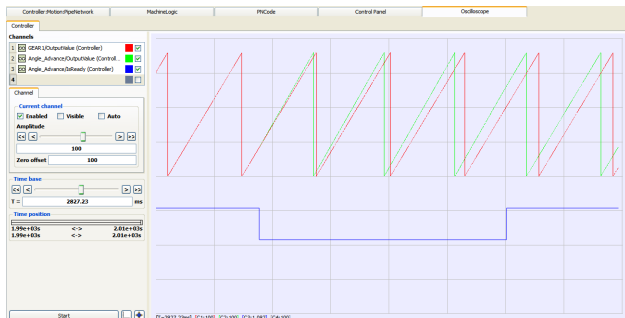
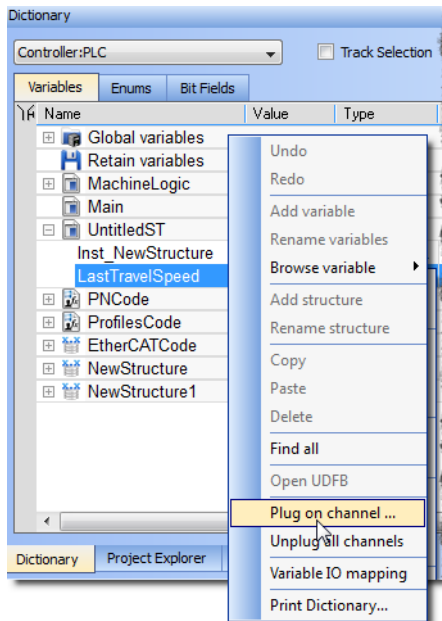


Figure 4-64: Example of Plugging a Pipe Block

[How to Plug PLC Variables](#)

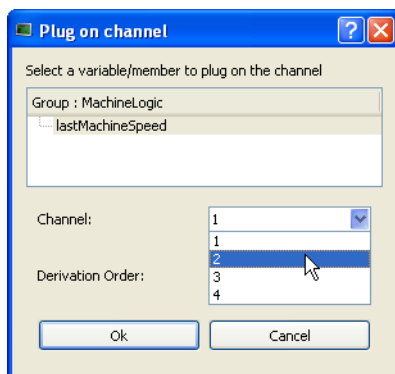
### 6.6.5.2 How to Plug PLC Variables

1. In the Variable Dictionary, right-click on the variable **lastMachineSpeed** to open its menu
2. Choose the command **Plug on channel**



**Figure 4-65:** Plugging a PLC Variable

3. Set Channel to **2**(because channel 1 is already plugged)



**Figure 4-66:** Plugging a PLC Variable - Parameters

You can start the Softscope now to see traces, as shown in the following figure:



Figure 4-67: Traces Displayed with Soft Oscilloscope

**TIP**

Easy probe plugging is assured since you do not need to unplug a probe from a channel before plugging a new probe into the same channel.

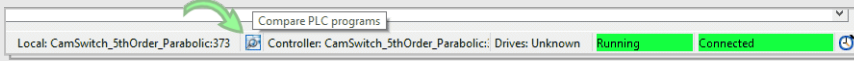
[How to Plug Motion Variables](#)

### 6.6.6 Compare PLC Programs

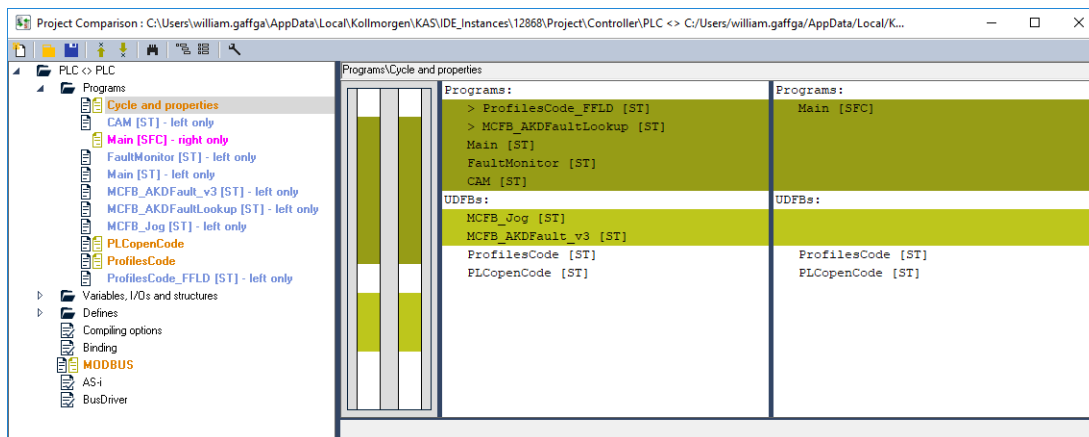
KAS provides a tool to show the differences between the "Local Project" and the project currently on the Controller or another project file.

**TIP**

- To compare project files, use the **Compare Projects** function from the File menu.
- The function for comparing the local project file to that on the controller is accessed from a button on the **Status Bar**, between the Local and Controller versions. It is active when KAS is connected to a controller.

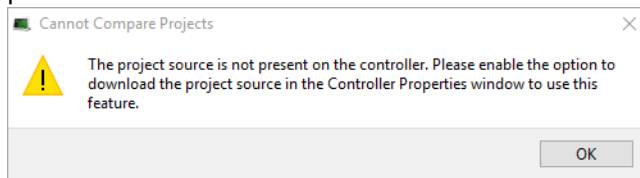


Once files have been selected to compare, a window opens listing the components of each project, and any differences between them. How the projects differ is color coded.. **Red** item indicates where there is a mismatch. Double-click to open an item. The << button brings you back to the list.



**Figure 4-68:** Listing the Differences

The following message will be shown if you click on the compare button but the project is not present on the controller.

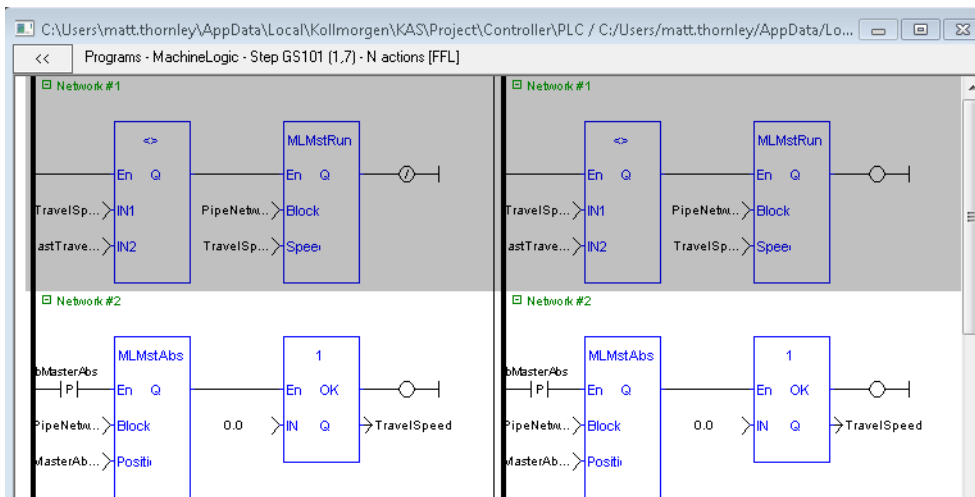


*Limitation with FFLD Diagrams*

If one or more components within a network differ, then the entire network is displayed in the diff view with a gray background. Differences between individual components within the network are not specified in any way.

For example, in the following image we have a comparison of two FFLD programs.

- Network #1 differs, but the actual difference is not called out or highlighted. The difference is the last element: a coil on the left diagram and an inverted coil on the right diagram.
- Network #2 is the same on the left and right, so it has a white background.



## 6.6.7 Variable Animation

When your application is running, all variables in the [IEC 61131-3 Editors](#), in the [Dictionary](#) and in the [Watch Window](#) are animated. This means that the value of each variable is displayed dynamically.

### NOTE

When the value of a variable is displayed, only the value computed at the end of the cycle is displayed.

So if the same variable is set in different programs, the animation in all those programs displays the same value for the variable, which corresponds to the latest program executed within the cycle.

### 6.6.7.0.1 About Online Change

When [Online Change](#) is enabled, the animated values only take place when you are in [Debug](#) mode (and not edit).

### 6.6.7.0.2 Limitations

- The versions on the KAS IDE and the KAS Runtime must be the same
- Animation does not apply to actions in an SFC step

### 6.6.7.1 Variable Monitoring

The Variable Dictionary contains all the IEC 61131-3 variables needed by the application. The variables are listed by categories corresponding to the declared programs, functions and function blocks.

When your application is running:

- all variables in the Dictionary are animated <sup>1</sup> with real-time values displayed in the **Value** column (see call out <sup>1</sup>)
- a specific column is used to indicate the initial values of all variables <sup>2</sup>

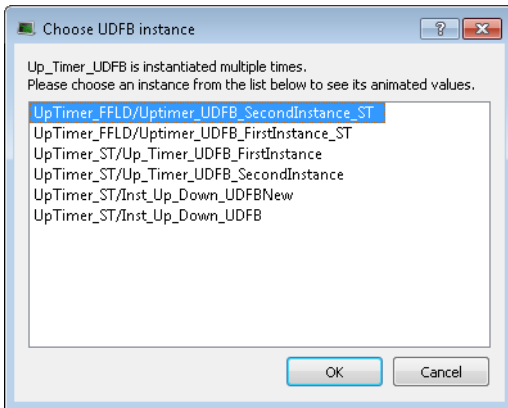
Name	Value	Type	Dim.	Attrib.	Init value	User...	KVE
<b>Global variables</b>							
TravelSpeed	0.0	LREAL			LREAL...		
MasterAbsPos	0.0	LREAL			LREAL...		
MasterDeltaPos	90.0	LREAL			LREAL...		
MachineSpeed	0.0	LREAL			LREAL...		
Axis1Status	185	DINT					
Axis2Status	185	DINT					
MachineState	0	DINT			0		
bMasterAbs	FALSE	BOOL			FALSE		
bMasterRel	FALSE	BOOL			FALSE		
bEStop	FALSE	BOOL			FALSE		
bLedStatus		BOOL	[0..3]				
Profiles		ProfilesCode					
EtherCAT		EtherCATCode		Read Only			
Sample_Inst		NewStructure					
Inst_NewStructure1		NewStructure1					
PipeNetwork		PNCode					

Figure 4-69: Variable Dictionary

#### 6.6.7.1.1 Monitoring UDFBs

<sup>1</sup>To better track variables and expressions of the PLC programs in Test mode, the KAS IDE dynamically computes their value along with the program execution and display the result in gray boxes beside their usage in the instruction lines of the IEC 61131-3 editor.

The real time values for UDFBs can be shown in the workspace while the program is running. Double clicking on a UDFB in the Project View's Subprograms list will open the UDFB in the workspace. You will first be presented with a list of the UDFB's instances if it is instantiated more than once or declared inside of a structure.

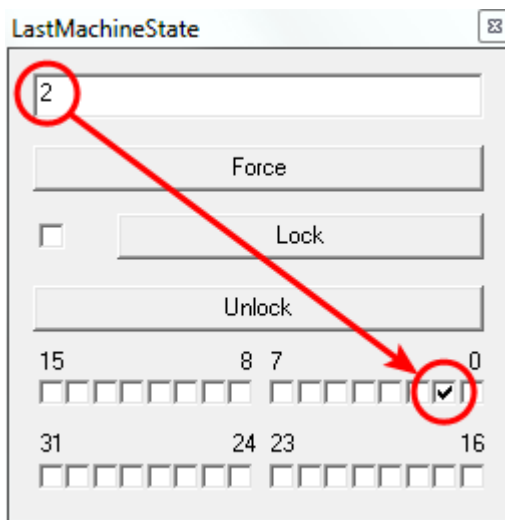


If the UDFB is open in the Workspace before running the program it will not automatically animate because there may be more than once instance of the UDFB. To start the animation double click the entry in the Project View. If there is only one instance, it will open; if there are multiple instances the list will be presented.

### 6.6.7.1.2 Forcing a variable

At run-time, double-click on the value of the variable in the list or press the **ENTER** key when it is selected. A popup window appears and allows you to:

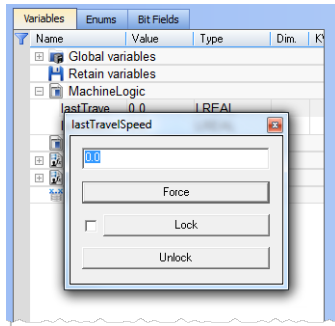
- **Force:** change the value of the selected variable. Depending on the variable type, you have the possibility to define its value either in the text field or with the check boxes.



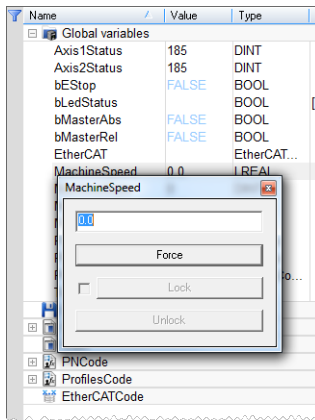
**Figure 4-70:** Forcing a Variable

The variable locking feature can be enabled or disabled via the PLC Options device toolbar button. If enabled, the Lock and Unlock buttons are accessible:





If disabled (default), the Lock and Unlock buttons are not accessible:



### ⚠ IMPORTANT

If PLC variable locking is enabled, the controller Runtime requires an additional 3% to 10% CPU processing power to manage the PLC variable locking. For the best controller performance, disable PLC variable locking.

- **Lock:** When a variable is locked, its value is no longer changed by the runtime. You can then force its value from the debugger, independently from the runtime operations. Note that all variables can be locked and forced at run-time.

### 👉 TIP

The value of a locked variable is displayed with square brackets.

```
ActualMachineState [[ -1 ]] := -1;
```

- **Unlock:** Remove the lock on a variable so it can be changed again by the runtime.

### 6.6.7.2 IEC 61131-3 Editor Debugging

In Test mode (Online or Simulation), all editors are animated <sup>1</sup> with real-time values of the edited objects:

- Values of variables, contacts and coils are displayed in FBD diagrams. Double-click on a variable name to force or lock the variable
- Values of variables, contacts and coils are displayed in FLD diagrams. Double-click on a variable name to force or lock the variable
- Step activities (tokens) are displayed in the SFC editor
- In the text (ST or IL) editor, place the mouse cursor on a variable name to display its real-time value in a tooltip.  
Double-click on the variable name with the **Shift** key pressed to force or lock the variable

```
Repeat
MyCounter (TRUE, FALSE, 16#FFFF);
// CV := MyCounter.CV;
if MyCounter.CV.7 FALSE then
bToggleVal TRUE := TRUE;
Ledlight2 TRUE := bToggleVal TRUE ;
End_if;
bToggleVal TRUE := not bToggleVal TRUE ;
Ledlight2 TRUE := bToggleVal TRUE ;
Until MyCounter.Q FALSE = FALSE
end_repeat;
```

**Figure 4-71:** Animation in Editors

See also [Forcing a variable](#)

### 6.6.7.3 Pipe Network Editor Debugging

See [How to Plug Motion Variables](#).

---

<sup>1</sup>To better track variables and expressions of the PLC programs in Test mode, the KAS IDE dynamically compute their value along with the program execution and display the result in gray boxes beside their usage in the instruction lines of the IEC 61131-3 editor.

## 6.7 Managing a Project

The **New** command in the File menu uses a wizard to help you to define the project.

The **Open...** command opens a window to let you navigate your system and retrieve previous projects.

The **Save** command saves your entire project.

The **Save As...** command allows you to save your project with a custom name and location.

### NOTE

Choose a safe folder for your project. Never select the Installation repository.

The **Close** command prompts you to save first if some modifications have not been saved.

### NOTE

When a project is already open, and you try to create or open another one, the KAS IDE proposes you to save your project before it is closed.

The **Print...** command allows you to create documentation containing editors' programs or diagrams.

For more details on the File menu, also refer to [Menus and Toolbar Overview](#).

### TIP

With the **Recent Projects** command in the File menu, the last four projects can be opened easily.

When editing your project, the KAS IDE has the following restrictions:

- You cannot work with several projects in parallel. You may use multiple instances of the KAS IDE as a work-around.
- Modifications that impact the project structure cannot be reversed with the **Undo** command (you have to make a backup first using the **Save As** command)
- No guarantee is provided by the KAS IDE with respect to the project file's integrity (this means that if you modify your data from outside the KAS IDE, you can spoiled your project)

### Use a Version Control System

To ensure integrity of your project files, you have to rely on tools to control versions.

Generally, such tools also have facilities for:

- Backup management
- Multi-users or multi-site development

## 6.7.1 Print

### 6.7.1.1 Printable Elements

The elements that you can print are:

- All PLC programs (see [PLC](#) node in the Project Explorer)
- Individual programs
- Level 2 SFC
- Level 2 SFC of single transition/state
- The Pipe Network editor
- The Dictionary

You can either print [one specific](#) program or [all the project](#) (PLC, Motion, Dictionary variables)

### 6.7.1.2 Page Setup

This dialog enables you to define the following settings:

#### Page Setup tab

- Orientation:  
Allows you to choose between portrait or landscape.  
Because the orientation can be set in both the page setup and the printer driver, it is recommended to have both settings synchronized.
- Scaling:  
You can select the **Fit to** option to fit on the specified number of pages. You have to enter one of the two values (either Wide or Tall) and the other are filled in automatically to keep a 1:1 aspect ratio of the print.

#### NOTE

These settings are not applicable when printing a [project](#).

#### Margin and Header/Footer tab

If you specify new margins or header/footer for a program, it affects the entire project when printed.

#### About field items used in Header/Footer

Special items can be inserted into the header/footer string as **{@item}**. They are converted to the correct format on printing or for print preview.

About the **Filename** field:

If an SFC level 2 program is being printed, the filename contains the SFC program name, Step or Transition number and the action tab name (e.g. Main, GS3, P1).

#### NOTE

All the settings defined in the Page Setup are saved within your project and are applied to each printed program.

This dialog box also contains two buttons:

- **Print...** displays the Printer dialog box as described below
- **Print Preview** displays a printout on the screen so you can see how it looks like before printing it.

### 6.7.1.3 Print

This dialog enables you to:

- Set the output (a printer, a PDF)
- Set the output preferences to set-up the printer options
- Look for a printer on the network
- Set the number of copies
- Set the page area to be printed
- Start the print

To print an SFC level 2 program, open it in the SFC editor and click the Print icon  (Ctrl + P)

#### 6.7.1.4 Print Preview

This dialog box enables you to display a printout on the screen so you can see how it looks before printing.

##### NOTE

Print preview limits the number of pages to display to the first 30 pages.

#### 6.7.1.5 Print Project

A Print Project dialog displays all the items that are printable. Then you can select those you want to include in your output and click **OK**.

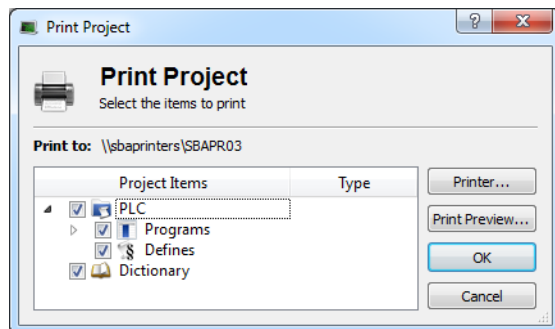


Figure 4-72: Print Project

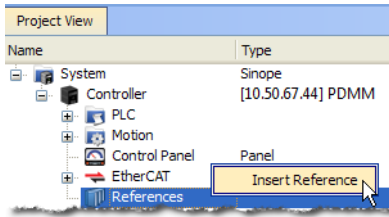
##### NOTE

Selecting an SFC program prints the SFC chart as well as SFC level 2 programs. Automatic scaling is applied for best readability.

## 6.7.2 Use the Reference Folder

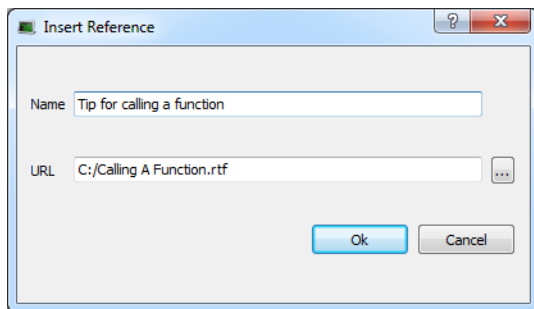
Using the Reference item, you can link as many files as you want to your project.

1. Right-click on the Reference item and select the **Insert Reference** command



**Figure 4-73:** Inserting a Reference

2. Define the **Name** and choose a valid **URL**



**Figure 4-74:** Defining the Reference

3. You can double-click the new reference to open it in the workspace

### NOTE

You can link files that are on your local machine (or to a server shared with a mapped drive) and of the following types: pdf, doc, xls, drawings, etc.. You must ensure the link is not broken if you want the KAS IDE to open it correctly.

## 7 KAS IDE Reference

This section provides in-depth reference to using elements of the KAS IDE interface.

### 7.1 Choose a Workspace Layout

#### 7.1.1 Move Child Windows

In the integrated workspace, all child windows are integrated into a single larger [application window](#).

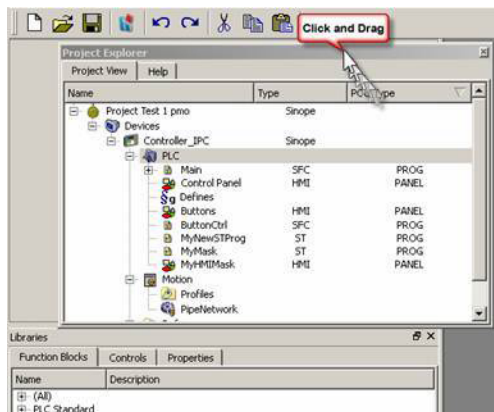
With the MDI/Tabbed workspace command in the Window menu, you can choose to display the child windows either as Tabbed Document Interface (TDI) or as Multiple Document Interface (MDI).

When in MDI mode, you can move and resize the displayed windows.

The Cascade command automatically rearranges all the windows to provide you with easier access to each of them.

#### 7.1.2 Move Toolbox

All toolboxes can be moved within the workspace to a more appropriate location. To customize your workspace, click in the Toolbox header and move the window using drag-and-drop. The other toolboxes are adapted accordingly.



##### 7.1.2.1 Dock Window

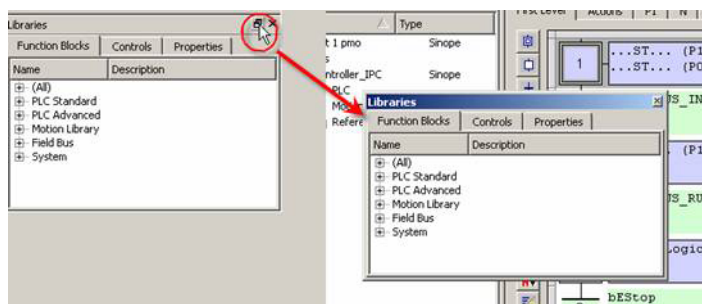
You can separate out a toolbox and change it to a docking window to be placed in the workspace independently of the other toolboxes.

##### How to change a toolbox to a Docking window?

To do so, click the  icon (you can also double-click in the toolbox header).

##### TIP

Double-click to place the window back into its original position.

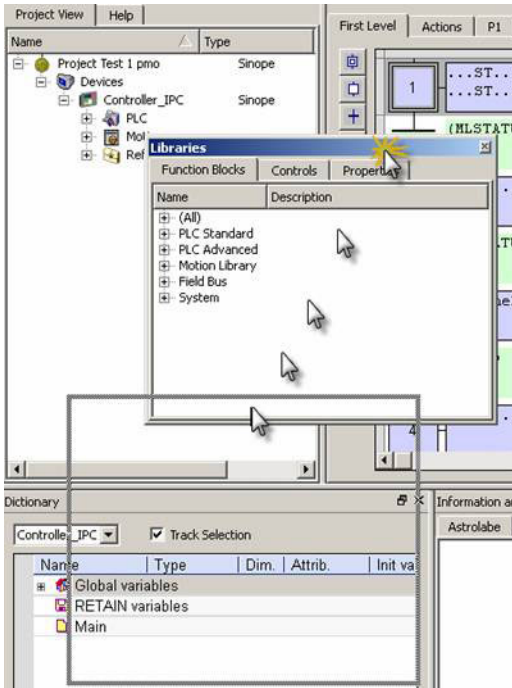


**NOTE**

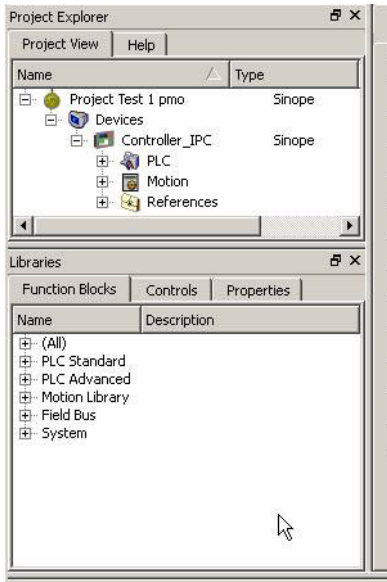
Moving a toolbox to a docking window can lead to problems which can be difficult to recover.

**How to undock a window?**

If problems arise, drag-and-drop the window to a toolbox border as shown below:



Dropped in the bottom border of the Project Explorer toolbox, then the **Libraries** toolbox is moved nearby.



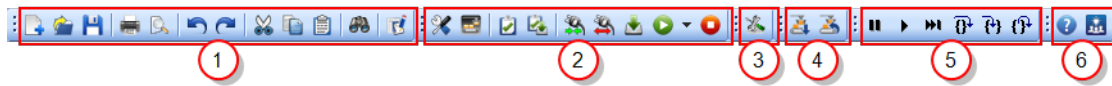


## 7.2 Menus and Toolbar Overview

The KAS IDE contains the five following menus:

- [File](#)
- [Edit](#)
- [Tools](#)
- [Window](#)
- [Help](#)

...and the following toolbars:



1. [Tools](#)
2. [Device](#)
3. [EtherCAT](#)
4. [Online Change](#)
5. [Debug](#)
6. Links to the **Help** and [Kollmorgen Developer Network \(KDN\)](#)

A specific toolbar is also available for the [AKD drive](#).

For details about icons available in the graphical PLC editors, see these sections:

- [FBD toolbar](#)
- [FFLD toolbar](#)
- [SFC toolbar](#)

### 7.2.1 File Menu

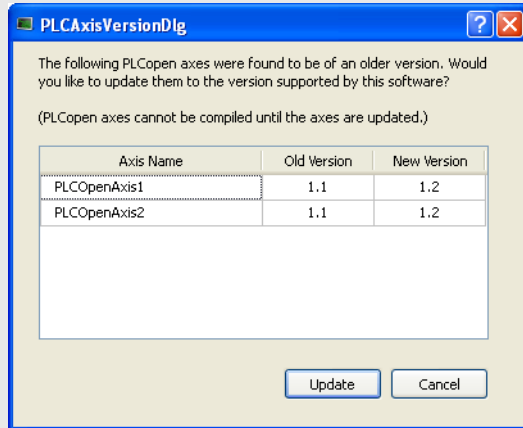
Command	Description
<b>New</b> (Ctrl + N)	Close the current project if any, and then launch the project <a href="#">wizard</a> to create a new one

Command	Description
---------	-------------

**Open...**  
(Ctrl + O)

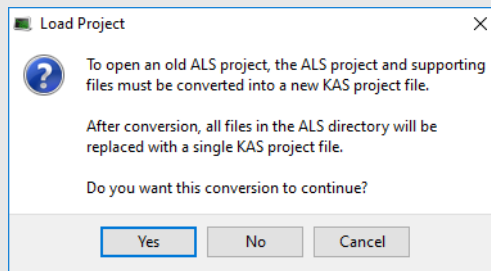
Open an existing project  
*About projects from an older version*

When you try to open a previous project, KAS IDE proposes you to do the conversion to keep the compatibility.



*About ALS project*

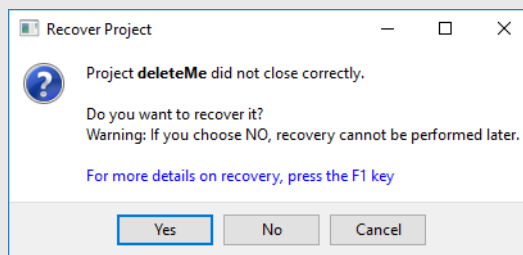
When you try to open a previous project with ALS format, the KAS IDE proposes you to convert it to the current KAS format.



After conversion, all the files located in the folder structure are replaced with a single KAS file.

*About Auto Recovery*

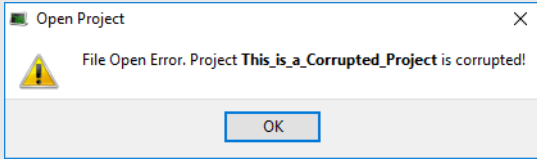
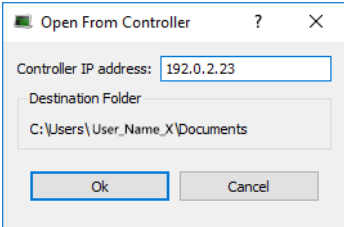
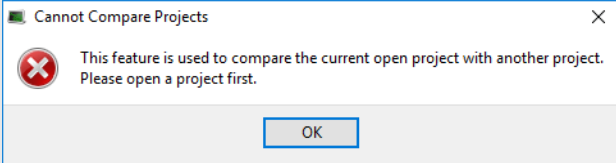
If the KAS IDE crashes when you are working with a project, you can start it again to recover the project from the last successful **Save** (or auto save) operation.



This file can be automatically saved periodically, so you recover recent work instead of the last manually saved file. See [User Options](#) from the [Tools Menu](#).

*About Corruption*

If your project is corrupted, KAS IDE opens a pop-up window

Command	Description
	 <p>Or you can also open an existing <a href="#">library</a></p>
<b>Open From Controller (Alt+Shift+O)</b>	<p>Project files saved on a controller can be retrieved using this command. To retrieve the project, enter the controller's IP Address.</p>  <p>Projects which are successfully retrieved will be saved in <i>My Documents</i> before being opened.</p> <p>See <a href="#">Download the Application</a> for information on storing projects on the controller.</p>
<b>Save</b>	<p>Save the current project</p> <p><i>About Auto Save</i></p> <p>At compile time, an auto-save operation is performed which allows you to recover the project if a crash occurs.</p> <p>Saving when in <a href="#">Debug</a> mode automatically switches the program to Edit mode.</p>
<b>Save As...</b>	<p>Save the current project in a location and with a name that you can define. (the project name is also reflected in the KAS IDE window's title)</p>
<b>Compare Projects (Alt+Shift+C)</b>	<p>This option allows for comparison between an open project and another local project. Once you have browsed for and selected the file for comparison, a window will open which displays the differences between the files. See <a href="#">Compare PLC Programs</a> for more information on the comparison tool.</p> <p>A project must be loaded in the IDE to use this function. If a project is not present you will be presented with the following error.</p> 
<b>Close Project</b>	<p>Close the current project.</p> <p>(if changes have not been saved, a prompt is displayed first)</p>

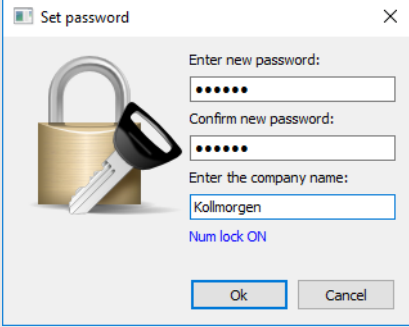


Command	Description
<b>Password Protection</b>	<p>This option provides the ability to set, change and remove password protection on a project file to prevent unauthorized access. The menu options are available when a project is open.</p> <p>Passwords must be 6-20 characters in length and may consist of any alpha-numeric characters (a-z, 0-9, \$, &amp;, *, }, etc.); spaces, tabs, and apostrophes are not permitted. Additionally, a company name must be provided, this should be the name of the company which owns the application source code.</p>  <p>The status bar has a project security status icon in the lower left corner: protected  and unprotected .</p> <p>To protect a project which is stored on the controller, the Project must be compiled and downloaded.</p>
<b>Page Setup...</b>	Define page setup, margins and header/footer
<b>Print... (Ctrl + P)</b>	Print the project element currently open in the workspace
<b>Print Preview...</b>	Display a printout on the screen so you can preview it before printing
<b>Print Project...</b>	Select among the complete project's elements those you want to print
<b>Recent Projects</b>	List the most recently used projects
<b>Exit</b>	Quit KAS IDE

Table 4-3: File Menu Commands

## 7.2.2 Edit Menu

Command	Description
Cut	Cut selected data and copy it to the clipboard
Copy	Copy selected data to the clipboard
Paste	Paste the data currently stored in the clipboard
Undo	Undo last command
	<p><b>NOTE</b></p> <p>This action is not possible for all operations.</p>
Redo	Redo last command
Find...	Show the Find and Replace tab in the <a href="#">Information and Logs</a> toolbox

Table 4-4: Edit Menu Commands

### TIP

See [Can I copy and paste or drag and drop between instances of the KAS IDE](#) for more information on the copy/paste functions, including drag-and-drop.

### **NOTE**

Closing the KAS IDE automatically clears the Clipboard. Anything that was copied is cleared from memory. This means you can not copy something, close the KAS IDE, and then paste.

### 7.2.3 Tools Menu

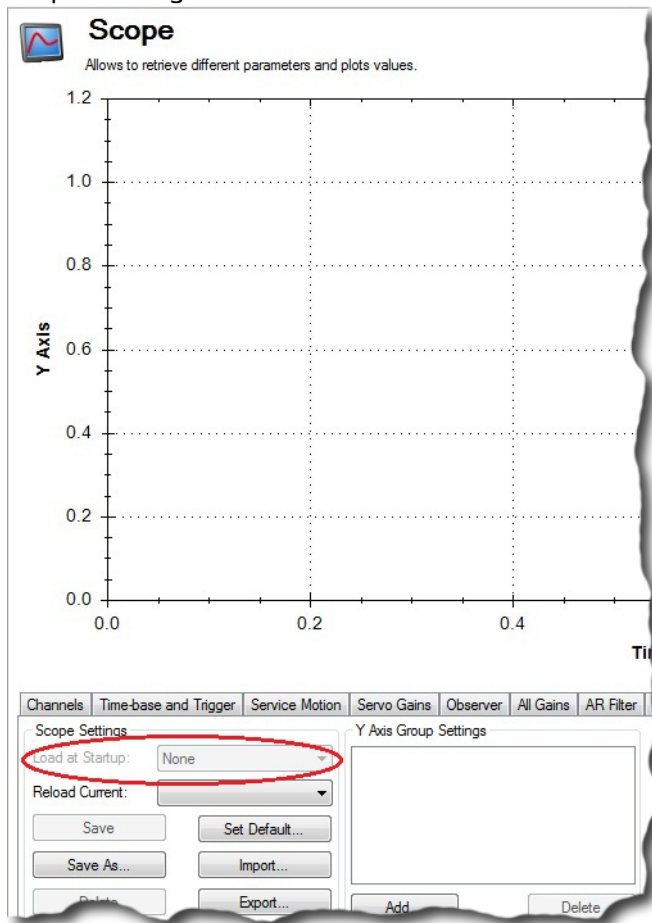
Command	Description
Oscilloscope	Show the <a href="#">soft oscilloscope</a>
Compile (Ctrl + B)	Compile the whole project
Compile & Download (Ctrl + F7)	Compile the whole project and download it to the controller.
User Options	Open the <a href="#">User Options</a> dialog box.

Table 4-5: Tools Menu Commands



#### 7.2.3.1 User Options

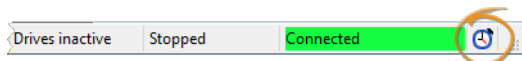
##### 7.2.3.1.1 AKD GUI Scope Settings

The check box found in this dialog enables/disables the “Load at startup” option in the AKD GUI Scope “settings” tab.



### 7.2.3.1.2 AutoSave Recovery File Settings

This option Enables / Disables automatically saving project recovery files at a specified interval. Once set, this applies to all further project recovery files for the user. The stopwatch icon in the bottom right corner of the KAS IDE indicates whether AutoSave Recovery Files is active  or disabled . See [How does file recovery work?](#) for information about recovery files.



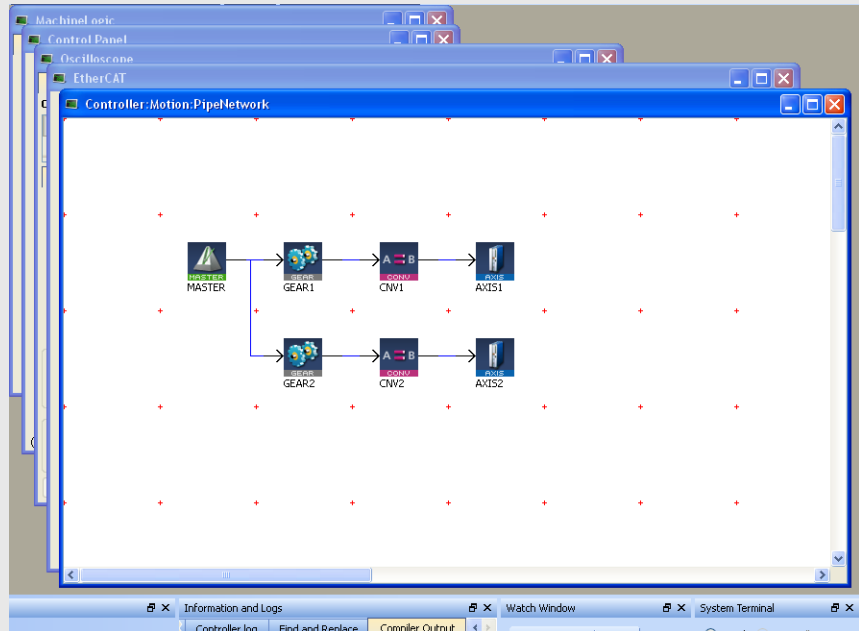
### 7.2.3.1.3 Backup File Settings

This option configures the automatic backup of KAS project or library files when the project or library is saved. The maximum number of backups is specified by the “Backup File Count” spinbox. Backup files are stored in the same directory as the KAS project or library that is being saved.

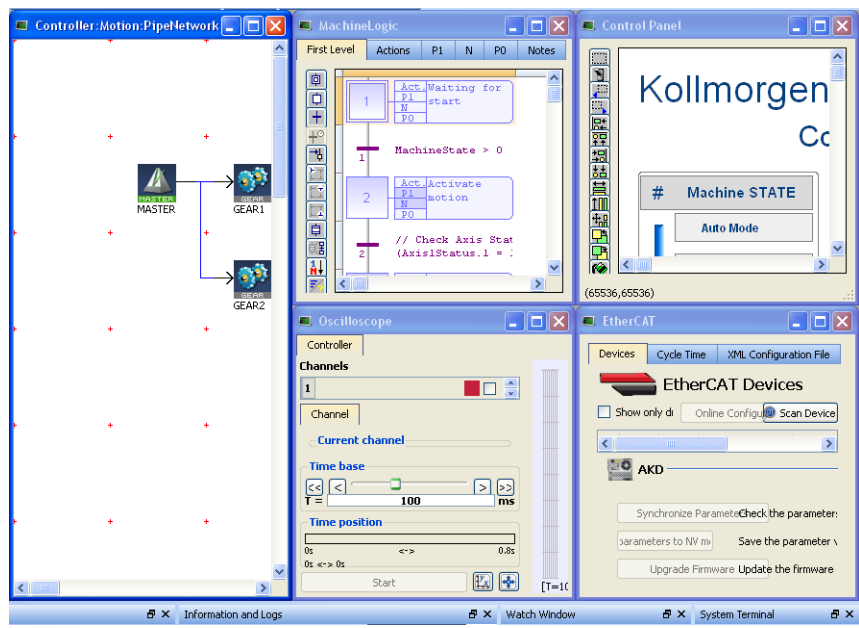
The backup files have the extension “.backup $nn$ .kas” or “.backup $nn$ .kal” where “ $nn$ ” represents two digits. The most recent backup of a KAS project will be “.backup01.kas”; the next most recent backup is “.backup02.kas”, and so on.

### 7.2.4 Window Menu

Command	Shortcut	Description
MDI/Tabbed Workspace	ALT+W	Toggle the workspace between the MDI and the tabbed mode
Cascade	ALT+C	Re-arrange all workspace children windows in cascade mode



Tile	ALT+T	Re-arrange all workspace children windows in tile mode
------	-------	--





Command	Shortcut	Description
<b>Reset IDE Layout</b>	Alt+R	<p>This option will reset widgets to their default location and visibility. If an item is hidden or floating, it will be made visible and returned to its default position. The following interface items are affected by this option.</p> <ul style="list-style-type: none"> <li>• <a href="#">Project Explorer</a></li> <li>• <a href="#">Libraries</a></li> <li>• <a href="#">Dictionary</a></li> <li>• <a href="#">Information and Logs</a></li> <li>• <a href="#">Watch Window</a></li> <li>• <a href="#">Toolbars</a></li> </ul>

Table 4-6: Windows Menu Commands

### 7.2.5 Help Menu

Command	Description
<b>Documentation</b>	Opens the help system.
<b>Help Preference</b>	As stated in <a href="#">Is the Help System Online or Offline?</a> , there is more than one type of help system with different advantages. By default the Online version is used. If you prefer, you may specify which Help version to use, the Online (default) or Offline (built-in) help system.
<b>Kollmorgen Developer Network (KDN)</b>	Opens a web browser to the Kollmorgen Developer Network ( <a href="http://www.kollmorgen.com/developer-network/">www.kollmorgen.com/developer-network/</a> ), a community support site.
<b>About</b>	Show version numbers and other information about the KAS IDE <a href="#">Find KAS IDE Version</a>

Table 4-7: Help Menu Commands

## 7.2.6 Main Toolbar

The main toolbar of the KAS IDE (Tools) contains the following icons:






















Icon	Description
	Create a new project
	Open an existing project
	Save the project
	Print the project item currently open in the workspace For more details, refer to <a href="#">Print</a>
	Print preview
	Undo
	Redo
	Cut
	Copy
	Paste
	Find
	Toggle Edit/Debug mode (enabled when the application is running) For more details, refer to <a href="#">PLC Online Change</a>

Table 4-8: Main Toolbar Icons

## 7.2.7 Device Toolbar

Each icon provided in this toolbox has a brief explanation provided below in order to explain the functionality.

Icon	Description
	<p>Show the PLC options dialog which has parameters for <b>Compilation</b> that can be modified for the target</p> <p>For more details about the compilation options, see <a href="#">"Set the Compilation Options" on page 209</a></p> <p>In the <b>Online Change</b> frame provides for allowing or forbidding Online Changes. For more details, see <a href="#">"PLC Online Change" on page 607</a></p> <p>The <b>PLC Variable Locking</b> frame provides for allowing or forbidding PLC variables to be forced. For more details, see <a href="#">"Forcing a variable" on page 232</a>. Please see the note below.</p> <p>The PLC options selections are stored in a KAS project file and are loaded when the project is opened.</p>
	<p>Change the Controller IP address to connect with KAS Simulator</p> <p>For more details, refer to <a href="#">Using the KAS Simulator</a></p>
	Compile project
	Compile and download project
	Establish a connection with the target Controller (for possible statuses, <a href="#">The following table summarizes all cases for the labels of the status bar.</a> )
	Close connection with the target Controller
	Download the application to the targeted Controller (Note that the application must <b>not</b> be running). For more details, refer to <a href="#">Download the Application</a>
	Start the application. It can be either a <b>Warm</b> or <b>Cold</b> start.
	Stop the application

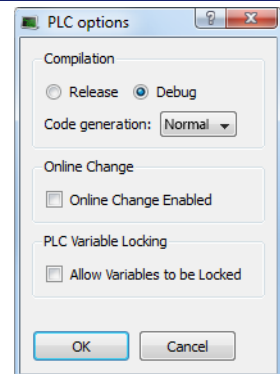


Table 4-9: Device Toolbar Icons

### NOTE

The **Allow Variables to be Locked** option cannot be changed while the **Online Change Enabled** option is selected. If Online Change is enabled, deselect the Online Change Enable option first, enable or disable PLC Variable Locking, then re-enable Online Change, and re-compile the project.

### 7.2.8 EtherCAT Toolbar

Each icon provided in this toolbox has a brief explanation provided below in order to explain the functionality.


Icon	Name	Description
	<b>Online Configuration Mode</b>	<p>This action is only available when the KAS Runtime is connected with the target Controller. Click this toggle button to change the mode (ON / OFF).</p> <p>After the scan has been performed, and your project is compliant with the physical devices on the EtherCAT network, you can activate the <b>Online Configuration Mode</b>. Online Configuration Mode allows setting up AKD and AKD2G drives in an EtherCAT installation.</p> <p>In this mode, KAS IDE communicates with the drives through the integrated views of the AKD Setup Screens or with the AKD Setup Wizard. Additionally, KAS IDE displays a quick status overview of all the drives.</p> <p>The AKD Setup Screens allow functions such as enabling/disabling the drive, service motion, tuning, and a scope where you can plot up to six different parameters from the drive.</p> <p>For more details on the AKD Setup Wizard, <a href="#">AKD Setup Wizard...</a></p> <p><a href="#">See also "FAQ"</a> for a potential issue when resetting the factory parameters.</p>

Table 4-10: EtherCAT Toolbar Icons

### 7.2.9 Online Change Toolbar

Each icon provided in this toolbox has a brief explanation provided below to explain the functionality.



Icon	Description
	<p>When Online Change has been activated in the PLC options; the new code is loaded even if the application is running. See also the Warning in <a href="#">How to Activate Online Change</a></p>
	<p>Revert your changes done after an Online Change, and go back to the previous application</p>

Table 4-11: Debug Toolbar Icons

### 7.2.10 Debug Toolbar

Each icon provided in this toolbox has a brief explanation provided below in order to explain the functionality.







Icon	Description
	Pause application in Cycle to Cycle mode
	Restart application in normal execution mode
	Execute a cycle step
	Step Over the next instruction: If the next instruction is a call of a function block or a sub-program, the execution passes over to the following instruction.
	Step Into the next instruction: The next step will be at the beginning of the called block (if the next instruction is <b>not</b> a call of a function block or a sub-program, then the Step Into behaves like the Step Over)
	Step Out the current block: If the current stepping position is in a called function block or a sub-program, the execution continues up to the end of the current block. Otherwise, the Step out behaves like the Step Over.

Table 4-12: Debug Toolbar Icons

## 7.3 Windows Standard Conventions

### 7.3.1 Windows Manipulation

The following standards apply to the KAS IDE windows:

- Move
- Resize
- Minimize
- Maximize
- Close (Alt+F4)

Press **Esc** to exit a pop-up window.

### 7.3.2 Mouse Manipulation

Double-click an item to open it (e.g. double-click a program in the Project Explorer to open it in the appropriate editor)

Right-click to open the menu and give access to the relevant commands (e.g. to add a variable to the Dictionary)

### 7.3.3 Table Manipulation

#### 7.3.3.1 Sorting Items

If the sort feature is implemented, you can click in the column header to sort all the items according to one of the available parameters.

Click again to alternately sort in ascending or descending order.

#### 7.3.3.2 Selecting a Cell

Click a cell in the table to select it. Once selected, press **F2** to edit the value.

Name	Type	Dim.	Attrib.
NewVar	BOOL		
MachineLogic		2,10,4	

#### **TIP**


A double-click directly opens the pop-up window for editing.

#### 7.3.3.3 Selecting a Row

When available, press the **Spacebar** to toggle the selection mode from cell to row. Then click a cell in the table to select the entire row. Once selected, press **F2** to edit the values of the row.

#### 7.3.3.4 Resizing a Column

If you want to enlarge a column width to make more content visible, put the mouse in the table

header between two columns so the cursor change to the following  and move right or left to resize your column.

After this operation, you need to scroll horizontally to see the other columns.

## 7.4 Keyboard Shortcuts

List of accelerator keys sorted by context:

- [Common Keyboard Shortcuts](#)
- [FBD Editor Keyboard Shortcuts](#)
- [FFLD Editor Keyboard Shortcuts](#)
- [SFC Editor Keyboard Shortcuts](#)
- [ST Editor Keyboard Shortcuts](#)
- [Graphic Editor Keyboard Shortcuts](#)
- [Table Keyboard Shortcuts](#)
- [CAM Editor](#)

### NOTE

A shortcut can be unavailable depending on the context.

### 7.4.1 Common Keyboard Shortcuts

Shortcut	Command
F1	Display the help
-	Collapse
+	Expand
Del	Delete
Ctrl + B	Compile the project
Ctrl + F7	Compile and Download the project
Ctrl + C	Copy
Ctrl + F	<a href="#">Find</a>
Ctrl + Shift + F	Find All
Ctrl + N	New
Ctrl + O	Open
Ctrl + P	<a href="#">Print</a>
Ctrl + S	Save
Ctrl + V	Paste
Ctrl + X	Cut
Ctrl + Y	Redo
Ctrl + Z	Undo
Alt + Shift + O	Open from controller
Alt + Shift + C	Compare projects
Page Up/Down	Scroll Page up/down
RETURN	Equivalent to double-click

Table 4-13: List of Common Keyboard Shortcuts

### TIP

See [Can I copy and paste or drag and drop between instances of the KAS IDE](#) for more information on the copy/paste functions, including drag-and-drop.

### NOTE

Closing the KAS IDE automatically clears the Clipboard. Anything that was copied is cleared from memory. This means you can not copy something, close the KAS IDE, and then paste.



## 7.4.2 FBD Editor Keyboard Shortcuts

- [FBD Editor \(common\)](#)
- [FBD Editor \(when editing\)](#)
- [FBD Editor \(during debug\)](#)

### 7.4.2.1 FBD Editor (common)

Shortcut	Command
Arrows	Scroll window
Ctrl + d	Display FBD execution order
Ctrl + page UP/DOWN	Go to previous/next section
Ctrl + mouse-wheel or + / - on the keypad	Zoom in / out
Escape	Cancel linking/resizing/dragging if selection: deselect if no selection: select mode active
Page UP/DOWN	Scroll page up/down
Return	Equivalent to double-click
Ctrl + Shift + End	Select all items from the cursor position to the end of the document
Ctrl + Shift + Home	Select all items from the begin to the cursor position
Tab	Select next position item
Tab + shift	Select previous position item
Ctrl+F2	Toggle Bookmark (Note that you first have to select the Network header)
Shift+F2	Go to Next Bookmark
Ctrl+Shift+F2	Go to Previous Bookmark

Table 4-14: List of FBD Shortcuts

### 7.4.2.2 FBD Editor (when editing)

Shortcut	Command
char	Start editing a symbol (variable, constant, instance) On jump/comment/break: open dialog box to enter text
Ctrl + arrows	Align selected items
Del	Delete selection
Shift + arrows	Move selection
Shift + page UP/DOWN	Move selection (4 cells)
Spacebar	Swap item style
Ctrl + Shift + down	Insert blank lines at the position of the mouse

### 7.4.2.3 FBD Editor (during debug)

Shortcut	Command
Spacebar	Swap TRUE/FALSE Boolean value
*	Lock var
/	Unlock var

### 7.4.3 FFLD Editor Keyboard Shortcuts

- [FFLD Editor \(when editing\)](#)
- [FFLD Editor \(during debug\)](#)

#### 7.4.3.1 FFLD Editor (when editing)

List of accelerator keys (sorted by action types)

##### 7.4.3.1.1 Insert

Shortcut	Command
Ctrl+Shift+D	Insert Coil De-Energize
Ctrl+Shift+E	Insert Coil Energize
Ctrl+Shift+R	Insert Coil Reset (Unlatch)
Ctrl+Shift+S	Insert Coil Set (Latch)
Ctrl+Shift+K	Insert a positive coil to the destination cell
Ctrl+Shift+L	Insert a negative coil to the destination cell
Ctrl+Shift+C	Insert Contact NC
Ctrl+Shift+A	Insert Contact NC, Negative Transition
Ctrl+Shift+I	Insert Contact NC, Positive Transition
Ctrl+Shift+O	Insert Contact NO
Ctrl+Shift+N	Insert Contact NO, Negative Transition
Ctrl+Shift+P	Insert Contact NO, Positive Transition
Ctrl+Shift+M	Insert Data In
Ctrl+Shift+W	Insert Data In Inverted
Ctrl+Shift+Q	Insert Data Out
Ctrl+Shift+B	Insert Wire (both)
Ctrl+Shift+H	Insert Horizontal Wire
Ctrl+Shift+V	Insert Vertical Wire
Shift+C	Insert Comment
Shift+Insert	Insert Network
Ctrl+Shift+J	Insert Jump
Ctrl+Shift+T	Insert Return
Insert Key	Insert Row
F8	Insert FB

Table 4-15: List of FFLD Shortcuts

##### 7.4.3.1.2 Trace

Shortcut	Command
Ctrl+J	Trace Horizontal Wire Left
Ctrl+K	Trace Horizontal Wire Right
Ctrl+M	Trace Vertical Wire Down
Ctrl+I	Trace Vertical Wire Up

**7.4.3.1.3 Move**

Shortcut	Command
Ctrl+End	Go to End of Network
Ctrl+End followed by Ctrl+End	Go to End of Ladder
Ctrl+Home or Home followed by Home	Go to Top of Network
Ctrl+Home followed by Ctrl+Home	Go to Top of Ladder
Ctrl+Page Up	Go to Previous Network
Ctrl+Page Down	Go to Next Network
Ctrl+Left Arrow or Home	Move focus to begin of row.
Ctrl+Right Arrow or End	Move focus to end of row.
Tab	Move focus cell right
Shift+Tab	Move focus cell left
Arrows	Move focus cell or scroll through ladder
Page up	Scroll 1 page up
Page Down	Scroll 1 page down

**7.4.3.1.4 Select**

Shortcut	Command
Shift+Arrow	Multiselect cells
Shift+left Arrow	Select current cell and one cell to left
Shift+right Arrow	Select current cell and one cell to right
Ctrl+Shift+ right Arrow or Shift+End	Select from current cell to end of line
Ctrl+Shift+ End	Select from current cell to end of network (Bottom element of network and the furthest to the right)
Ctrl+Shift+ left Arrow or Shift+Home	Select from current cell to beginning of line
Ctrl+Shift+ Home	Select from current cell to beginning of network
Shift+up Arrow	Select Cell above or below when focus is on cell.
Shift+down Arrow	Select Row above or below when focus is on left rail
Ctrl+A	Select the contents of a network/rung
Ctrl+A followed by Ctrl+A	Select the entire ladder
Shift+Page Up	Selection Page-Up
Shift+Page Down	Selection Page-Down

**7.4.3.1.5 Edit**

Shortcut	Command
Ctrl+C	Copy Item
Ctrl+X	Cut Item
Ctrl+V	Paste Item
Return	Equivalent to double click
Space	Change contact or coil
Ctrl+Y	Redo
Ctrl+Z	Undo
Ctrl + mouse-wheel or + / - on the keypad	Zoom in / out
Ctrl+S	Save
Esc or Shift-ESC	Close the rename widget. Exit Dialog

**7.4.3.1.6 Find**

Shortcut	Command
Ctrl+F	Find
Ctrl + Shift + F	Find All
F3	Find Next

**7.4.3.1.7 Delete**

Shortcut	Command
Delete Key	Delete cell, selection, or row
Shift+Delete	Delete Network

**7.4.3.1.8 Bookmark**

Shortcut	Command
Ctrl+F2	Toggle Bookmark (you must first select the Network header)
Shift+F2	Go to Next Bookmark
Ctrl+Shift+F2	Go to Previous Bookmark

**7.4.3.2 FFLD Editor (during debug)**

Shortcut	Command
Spacebar	Swap TRUE/FALSE Boolean value
*	Lock var
/	Unlock var

### 7.4.4 SFC Editor Keyboard Shortcuts

Shortcut	Command
?	Show/Hide notes
arrows	Move caret
Page UP/DOWN	Scroll page up/down
Return	Equivalent to double-click
Shift + arrows	Select multiple cells
Shift + Home	Select from left to caret
Shift + Page Up/Down	Selection Page Up/down
b or B	Insert macro body
c or C	Insert convergence
Ctrl + return	Edit reference
d or D	Insert divergence
Del	Delete selection
i or I	Insert step initial
j or J	Insert jump
m or M	Insert macro
s or S	Insert step
Spacebar	Swap item style
t or T	Insert transition
x or X	Insert the left side corner of a divergence/convergence
Ctrl + mouse-wheel or + / - on the keypad	Zoom in / out

Table 4-16: List of SFC Shortcuts

## 7.4.5 ST Editor Keyboard Shortcuts

- [ST Editor \(common\)](#)
- [ST Editor \(when editing\)](#)
- [ST Editor \(during debug\)](#)

### 7.4.5.1 ST Editor (common)

Shortcut	Command
Arrows	Move caret
Shift + arrows	Selection
Ctrl + left/right arrow	Go to previous/next word
Shift + Ctrl + left/right arrow	Select previous/next word
Ctrl+F2	Toggle Bookmark (Note that you first have to select the Network header)
Shift+F2	Go to Next Bookmark
Ctrl+Shift+F2	Go to Previous Bookmark
Ctrl + mouse-wheel or + / - on the keypad	Zoom in / out

Table 4-17: List of ST Shortcuts

### 7.4.5.2 ST Editor (when editing)

Shortcut	Command
.	Select member of a structure or instance
Ctrl + Spacebar	Auto completion or Open the variable selector dialog
Ctrl + Shift + Spacebar	Opens a list of all the standard functions
Ctrl + J	Auto completion or Open the variable selector dialog (an alternative method)

### 7.4.5.3 ST Editor (during debug)

Shortcut	Command
*	Lock variable
/	Unlock variable
Shift + double-click	Force a variable
Spacebar	Toggle Boolean value or bring the dialog to force, lock, unlock the variable (equivalent of Shift + double click)

### 7.4.6 Graphic Editor Keyboard Shortcuts

Shortcut	Command
Ctrl + mouse-wheel down or Shift+MINUS Sign (-) on the numerical keypad	Zoom out
Ctrl + mouse-wheel up or Shift+PLUS Sign (+) on the numerical keypad	Zoom in
Arrow	Scroll
Ctrl + F2	Toggle bookmark
Ctrl + arrow	Align on main selected item
Del	Delete selection
Escape	Cancel resizing/dragging if selection: unselect if no selection: select mode active
Ctrl + Shift + End	Select all items from the cursor position to the end of the document
Ctrl + Shift + Home	Select all items from the begin to the cursor position
Shift + F2	Go to next bookmark
Shift + Page UP/DOWN	Offset selection
Shift + Arrow	Move selection
Tab	Select next position item
Tab + shift	Select previous position item

Table 4-18: List of Graphics Editor Shortcuts

### 7.4.7 Table Keyboard Shortcuts

Shortcut	Command
Arrows	Move selection
Shift + Tab	Move selection to the left
Spacebar	Line selection/cell selection
Tab	Move selection to the right

Table 4-19: List of Table Shortcuts

## 7.5 Bookmarks

Bookmarks are used for navigating in a document. You can insert bookmarks anywhere in a document. Then you can jump from one bookmark to another with a single command for browsing the document. Bookmarks are supported in all program editors and the Variable editor.

Below are the available commands for using bookmarks:

<b>Ctrl + F2</b>	Toggle the bookmark at the current position
<b>Shift + F2</b>	Go to the next bookmark

According to the type of document, the possible locations for a bookmark are:

- In the text editor, a bookmark is placed on a line of text.
- In the SFC editor, a bookmark is placed on an SFC symbol (step, transition, jump...).
- In the FBD editor, a bookmark is placed on any FBD object (not on a line).
- In the FFLD editor, a bookmark is placed on a rung header.
- In the Variable editor, a bookmark is placed on any line of the grid (variable or group).

### NOTE

Bookmarks are valid only while the editing window is open; they are not stored in the document when the window is closed.



## 7.6 Create Programs

This chapter provides details on the syntax, structure and use of the declarations and statements supported by the KAS IDE application language.

### 7.6.1 Project Structure

Structuring the application with care is important in creating your project (see [Project Structure Guidelines](#) in [Advanced Topics](#)).

### 7.6.2 IEC 61131-3 Editors

The KAS IDE programming environment provides language dedicated editors for:

- [Sequential Function Chart \(SFC\)](#)
- [Function Block Diagram \(FBD\)](#)
- [Free Form Ladder Diagram \(FFLD\)](#)
- [Structure Text \(ST\) and Instruction List \(IL\)](#)

*When SFC must be used?*

- SFC must be used when you need to manage sequences of stable process states.
- Using SFC avoids complex switches and the declaration of multiple flags in programs.

*When SFC must **not** be used?*

- SFC must never be used as a decision diagram or flow chart for describing an algorithm (i.e. when you think "If / Then / Else..."). This leads to complex SFC charts and bad performances at run-time.
- Never use a step to represent an intermediate point within a calculation. Use ST in this case.

See also [Program Limitations](#) and the [PLC Online Change](#) feature.

### 7.6.3 Some Tips...

#### 7.6.3.1 About Drag-and-Drop

The editor provides you with an ideal programming environment, including drag-and-drop features:

- Drag a variable from [Dictionary](#) and drop it into the program to insert it
- Drag a definition from [Libraries](#) and drop it into the program to insert its name
- Drag a block and drop it into the program to insert it (you can even select the block from an external text file).
- Drag a function block to the variable list to declare an instance

#### 7.6.3.2 About Autocompletion

When you type the name of a function block instance (use either as an instance or a data structure), pressing the point "." after the name of the instance opens a pop-up list with the names of possible elements. Click the relevant element and validate it with the check mark.

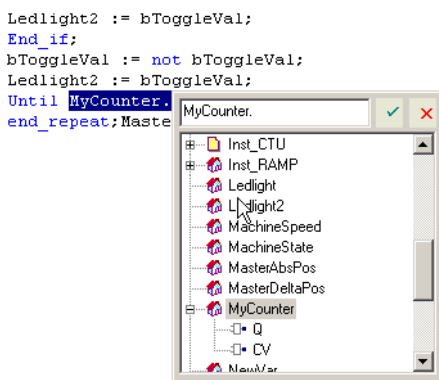


Figure 4-75: Autocompletion

#### Auto-completion of words

#### 7.6.3.3 About tooltip on variable

When you leave the mouse cursor on a variable in Editors, a tooltip is displayed to give you more details on the item.

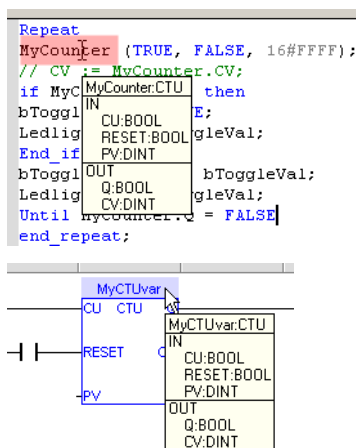


Figure 4-76: Tooltip on Variable

The header of the tooltip displays the name of the variable and its [type](#).

### 7.6.3.4 About Bookmarks

See [Bookmarks](#)

### 7.6.4 Select Function Blocks

All available Operators, functions and function blocks are listed in the [Libraries](#) toolbox. The list of available blocks is sorted into categories. The "(All)" category enables you to see the complete list of available blocks.

To insert a block in a program, select it and drag-and-drop it to the desired position in the Editor.

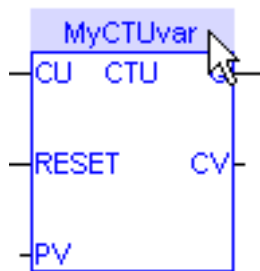
#### TIP

First drag a function block from the Libraries and drop it in the variable list (Dictionary) to declare a new instance. Then drag this instance from the Dictionary and drop it in the program.

### 7.6.5 Select Variables and Instances

Symbols of variables and instances are selected using the variable list in the [Dictionary](#). Selecting variables is available from all editors:

- In FBD diagrams, double-click on a variable box, an FB instance name, a contact or a coil to select the associated variable.
- In FFLD diagrams, double-click on a contact, a coil or a block input or output to select the variable. Double-click on the top of an FB rectangle to select an instance.



- When the variable editor is visible in the editor window, you can drag a variable from the list and drop it in the program to insert it.

*How to access a single bit of an Integer variable?*

<variable>.<Bit number> (e.g. MachineState.7)

### 7.6.6 Drag and Drop Programming

The editor provides you with an ideal programming environment, including drag-and-drop features:

- Drag a variable from [Dictionary](#) and drop it into the program to insert it
- Drag a definition from [Libraries](#) and drop it into the program to insert its name
- Drag a block and drop it into the program to insert it (you can even select the block from an external text file).
- Drag a function block to the variable list to declare an instance

### 7.6.7 Autocompletion When Programming

When you type the name of a function block instance (use either as an instance or a data structure), pressing the point "." after the name of the instance opens a pop-up list with the names of possible elements. Click the relevant element and validate it with the check mark.

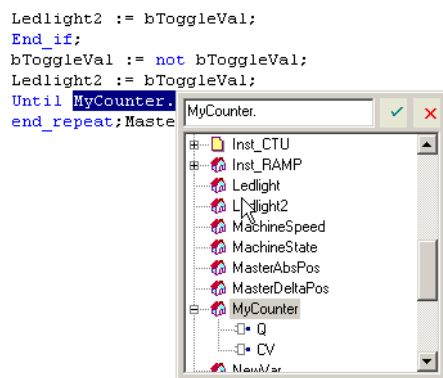
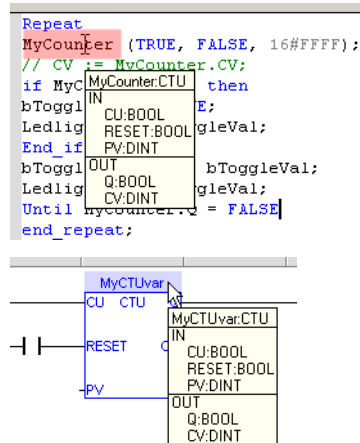


Figure 4-77: Autocompletion

[Auto-completion of words](#)

### 7.6.8 Tooltips on Variables

When you leave the mouse cursor on a variable in Editors, a tooltip is displayed to give you more details on the item.



**Figure 4-78:** Tooltip on Variable

The header of the tooltip displays the name of the variable and its [type](#).

### 7.6.9 Selecting Function Blocks

All available Operators, functions and function blocks are listed in the [Libraries](#) toolbox. The list of available blocks is sorted into categories. The "(All)" category enables you to see the complete list of available blocks.

To insert a block in a program, select it and drag-and-drop it to the desired position in the Editor.

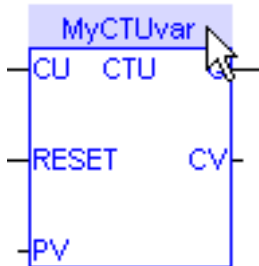
#### **TIP**

First drag a function block from the Libraries and drop it in the variable list (Dictionary) to declare a new instance. Then drag this instance from the Dictionary and drop it in the program.

### 7.6.10 Selecting Variables and Instances

Symbols of variables and instances are selected using the variable list in the **Dictionary**. Selecting variables is available from all editors:

- In FBD diagrams, double-click on a variable box, an FB instance name, a contact or a coil to select the associated variable.
- In FFLD diagrams, double-click on a contact, a coil or a block input or output to select the variable. Double-click on the top of an FB rectangle to select an instance.



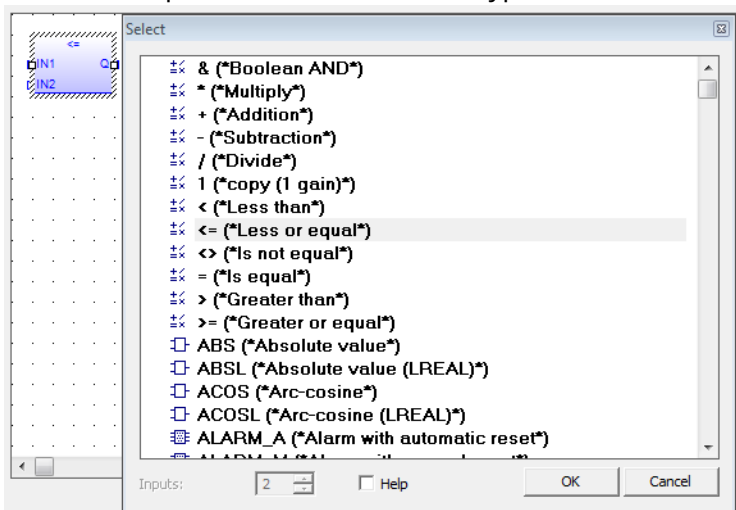
- When the variable editor is visible in the editor window, you can drag a variable from the list and drop it in the program to insert it.

*How to access a single bit of an Integer variable?*

<variable>.<Bit number> (e.g. MachineState.7)

### 7.6.11 Change Operators and Functions in FFLD and FBD Editors

Functions and operators can easily be edited and changed in the FFLD and FBD Editors. Double-clicking on a item in the editor opens a **Select** window which lets you change the entry to another function or operator, even of another type.



- When appropriate you can select the number of inputs. This is most common for mathematical operations.
- Clicking the **Help** option enables opening the help topic for any selected operator, function or function block.

#### **TIP**

This list is inclusive of function blocks. A function or operator can be changed to a function block but variable assignment will need to be managed.

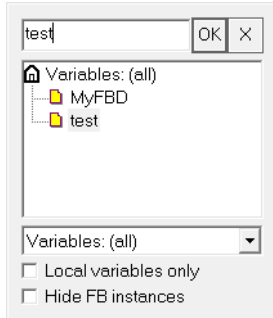
**NOTE**

Please note that this list includes numerous functions that are not valid for KAS. They are either not supported or have been deprecated.

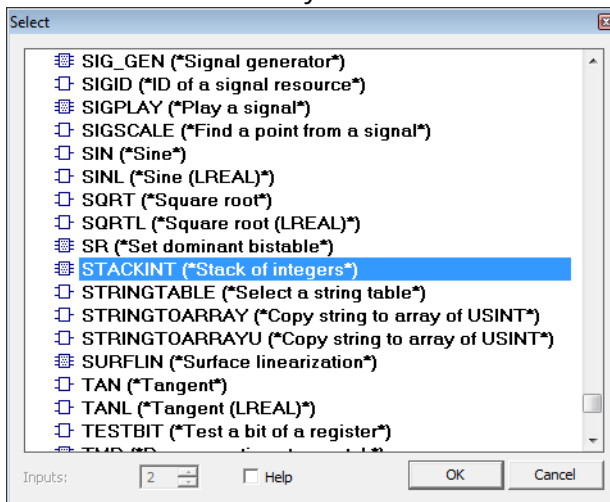
See also [Change Function Blocks in FFLD and FBD Editors](#).

### 7.6.12 Change Function Blocks in FFLD and FBD Editors

- Double-clicking on the name of a function block in the FFLD or FBD Editors allows you to select a different function block from a list of other FB instances.



- Double-clicking on the main area of a function block in the FFLD or FBD editors allows you to change to a different operator, function, or function block. Please be aware of any assigned variables which will likely need modification.

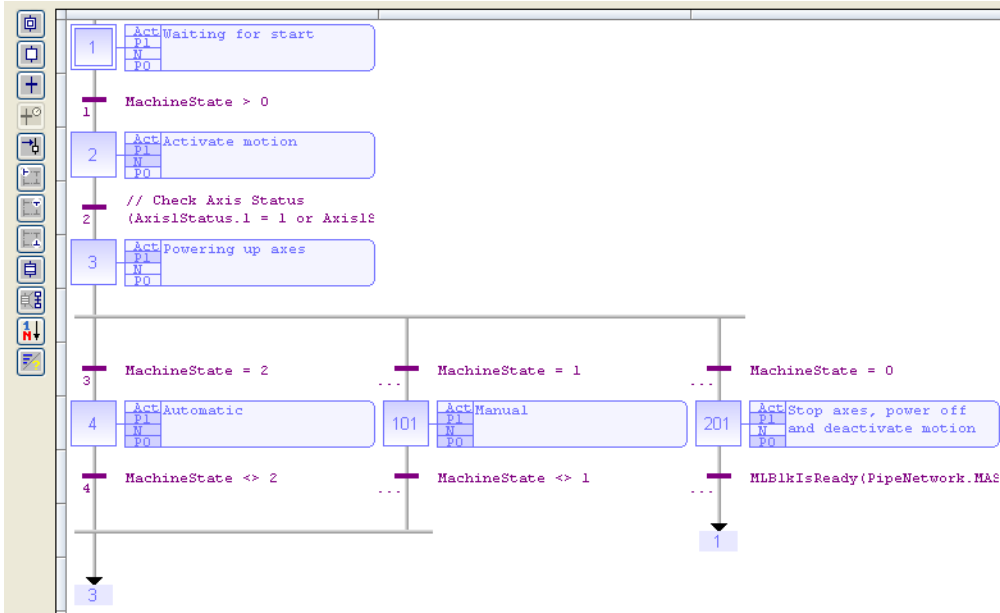
**NOTE**

Please note that this list includes numerous functions that are not valid for KAS. They are either not supported or have been deprecated.

See also [Change Operators and Functions in FFLD and FBD Editors](#)

### 7.6.13 Sequential Function Chart (SFC) Editor

The SFC Editor is a powerful graphical tool that enables you to enter and manage Sequential Function Chart according to the IEC 61131-3 standard. The editor supports advanced graphic features such as drag-and-drop, so that you can freely and rapidly arrange the elements of your diagram. It also supports automatic chart formatting when inserting or deleting items, and thus enables quick input using the keyboard.



**NOTE**

For each step, the cells referring to P1, N and P0 actions are **colored** when they are defined.

SFC diagram components	Related Sections
Steps	<a href="#">Using the SFC toolbar</a>
Transitions	<a href="#">Drawing divergences</a>
Divergences	<a href="#">Viewing the chart</a>
Parallel branches	<a href="#">Printing the chart</a>
Jump to a step	<a href="#">Moving or copying parts of the chart</a>
Macro steps	<a href="#">Entering macro-steps</a>
Actions	<a href="#">Renumbering steps and transitions</a>
Conditions	<a href="#">Entering actions of a step</a>
Timeout check	<a href="#">Entering condition of a transition</a>
	<a href="#">Notes for steps and transitions</a>
	<a href="#">Bookmarks</a>
	<a href="#">Program Limitations</a>

**TIP**

- To change the number of a step, transition or jump, select it and press the **Ctrl+ENTER** keys.
- Hit **Spacebar** on the main corner (on the left) of a divergence or convergence, to set either double or single horizontal line style.



### 7.6.13.1 Using the SFC toolbar

The vertical toolbar on the left side of the editor contains buttons for inserting items in the chart. Items are always inserted before the selected item, and the chart is automatically re-arranged when a new item is inserted.











Icon	Description
	Insert an initial step
	Insert a step
	Insert a transition
	Insert a jump to a step
	Insert the main (left side) corner of a divergence or convergence
	Insert a divergence corner
	Insert a convergence corner
	Insert a macro-step
	Insert the body of a macro-step

Table 4-20: SFC Toolbar - List of Icons

Use the following keyboard commands when an item is selected:

- **ENTER**: edit the level 2 of a step or transition
- **Ctrl+ENTER**: change the number of a step, transition or jump




The last button of the toolbar enables you to switch between possible displays:

-  Swap between possible overviews of level 2 in the level 1 chart:
  - display code of actions and conditions
  - display notes attached to steps and transitions

### 7.6.13.2 Draw SFC divergences

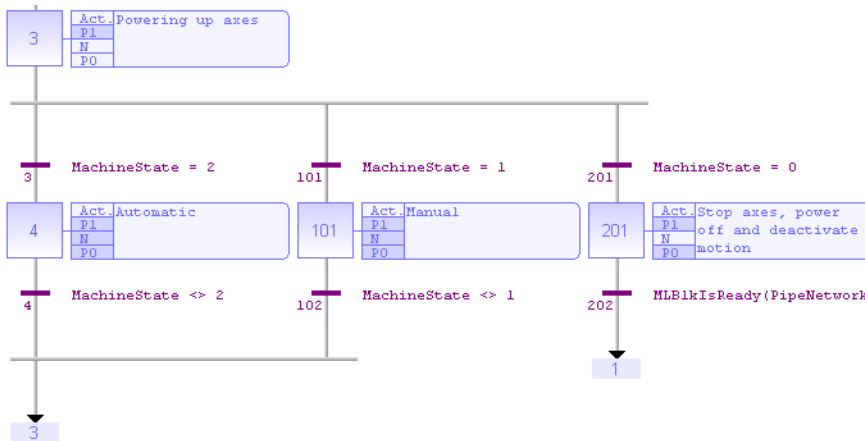
When using the SFC editor, you just need to place items in the grid. The editor calculates and draws lines automatically to link the steps, transitions, and adjusts your place in the chart.

The same method is used for drawing divergences: you just need to place the "corners" that identify divergences, convergences and branches. The editor takes care of drawing vertical and horizontal lines. Use the following buttons in the SFC toolbar:

Icon	Description
	Insert the main (left side) corner of a divergence or convergence
	Insert a divergence corner
	Insert a convergence corner

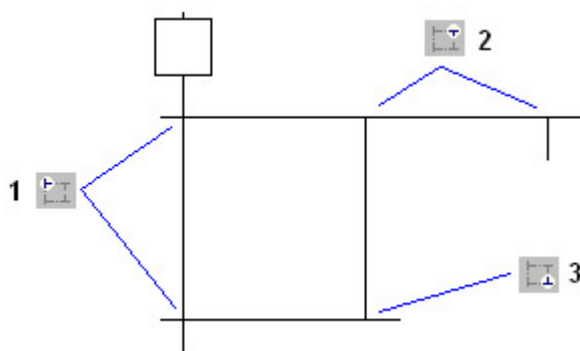
#### ⚠ IMPORTANT

Divergences are always drawn from left to right. The first branch, on the left, contains the "corners" that identify the divergence. It must be aligned with the preceding step or transition:



#### How to proceed?

- 1- Insert the main corner (on the left-hand side branch) of the divergence and the convergence
- 2- Insert corners at the top of each branch (divergence)
- 3- Insert corners at the bottom of the branches where a divergence is required



#### Simple or double divergence lines:

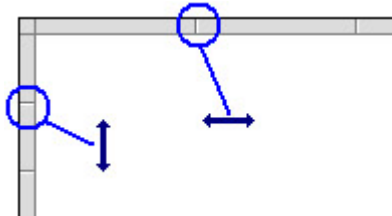
You can change the drawing of a divergence or convergence horizontal line, for drawing simple or double lines according to the SFC definition. To do this, move the selection on the main corner (on the left) and press the **Spacebar**.

### 7.6.13.3 View SFC charts


The chart is entered in a logical grid, and all objects are snapped to the grid. The (x,y) coordinates of the mouse cursor are displayed in the status bar. This helps you to locate errors detected by the compiler, or to align objects in the chart.

At any moment you can zoom in or out of the edited diagram using a Ctrl + mouse-wheel operation. You can also press the [+] and [-] keys of the numerical keypad to zoom the diagram in or out.

You can also drag the separation lines in vertical and horizontal rulers to resize the cells of the grid:



The SFC Editor adjusts the size of the font according to the zoom ratio. When a cell is wide enough, a text is displayed with the contents of the step or transition (level 2). The last button of the toolbar enables you to switch between displays:

	<p>Swap between possible overviews of level 2 in the level 1 chart:</p> <ul style="list-style-type: none"> <li>- display code of actions and conditions</li> <li>- display notes attached to steps and transitions</li> </ul>
---	---

#### 7.6.13.4 Move or copy SFC charts

The SFC Editor fully supports drag-and-drop for moving or copying items. To move an item, select and drag it to the desired position.

To copy an item, do the same, and just press the **Ctrl** key while dragging. It is also possible to drag pieces of a chart from one program to another if both are open and visible on the screen.

At any moment, while dragging items, you can press ESCAPE to cancel the operation.

Alternatively, you can use the Copy / Cut / Paste commands from the Edit menu. The Paste action is performed at the current position.

#### 7.6.13.5 Enter SFC macro-steps

A macro step is a special symbol that represents, within an SFC chart, a part of the chart that begins with a step and ends with a step. The body of the macro-step must be declared in the same program. The body of a macro-step begins with a special "begin" step with no link before, and ends with a special "end" step with no link after. The symbol of the macro step in the main chart has double horizontal lines.

Use the following buttons of the SFC toolbar to enter macro-steps:

	Insert a macro-step
	Insert the body of a macro-step

#### **!** IMPORTANT

The symbol of the macro-step and the first step of its body must have the same number. Press Ctrl+ENTER when a macro-step symbol or a first step is selected to change its number.

#### 7.6.13.6 Renumber steps and transitions

Each step or transition is identified by a number. A jump to a step is also identified by the number of the destination step. The SFC Editor allocates a new number to each step or transition inserted in the chart.

To change the number of a step, transition or jump, select it and press **Ctrl+ENTER**.

It is not possible to change the number of a step or a transition if its level 2 is currently open for editing. The number is used for identifying the step or transition in the level 2 editing window.

In compiler reports, a step is identified by its number prefixed by "GS". A transition is identified by its number prefixed by "GT".

### 7.6.13.7 Enter actions of a step

Actions and notes attached to a step (level 2) are entered in a separate window. To open the level 2 editing window of a step or transition, double-click on its symbol in the chart, or select it and press ENTER.

The level 2 editing window proposes five views for entering different types of level 2 information:

- simple actions entered as text
- P1 actions than can be programmed in ST/IL text, FFLD or FBD
- N actions than can be programmed in ST/IL text, FFLD or FBD
- P0 actions than can be programmed in ST/IL text, FFLD or FBD
- text notes

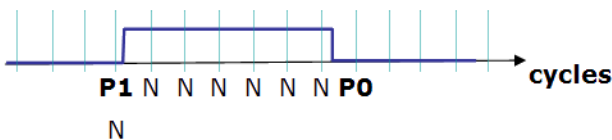
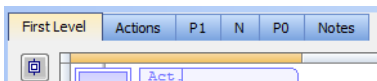


Figure 4-79: SFC Step Action Blocks

Use the tab buttons in the level 2 editing window to select a view:



When editing P1, N or P0 actions, use the radio buttons to select the programming language.

The first view ("Action") contains all simple actions for controlling a Boolean variable or a child SFC chart. However, it is possible to directly enter action blocks programmed in ST together with other actions in this view. Use the following syntax for entering ST action blocks in the first pane:

```
ACTION ( qualifier ) :
statements...
END_ACTION;
```

Where *qualifier* is "P1", "N" or "P0".

### 7.6.13.8 Enter the condition of a transition

The conditions and notes attached to a transition (level 2) are entered in a separate window. To open the level 2 editing window of a step or transition, double-click on its symbol in the chart, or select it and press ENTER.

The level 2 editing window proposes two views for entering different types of level 2 information:  
 - condition programmed in ST/IL text or FFLD  
 - text notes

Use the tab buttons in the level 2 editing window for selecting a view:



When editing the condition, use the "Edit / Set Language" menu command to select the programming language. This command is not available if the condition is not empty. FBD cannot be used to program a condition.

### 7.6.13.9 Enter notes for steps and transitions

The SFC editor supports the definition of text notes for each step and transition. The notes are entered in the level 2 editing window of steps and transitions. Refer to the following topics for further information about the level 2 editing window:

- [entering Level 2 for steps](#)
- [entering Level 2 for transitions](#)

Notes can be displayed in the chart. The last button of the toolbar enables you to switch between possible displays:



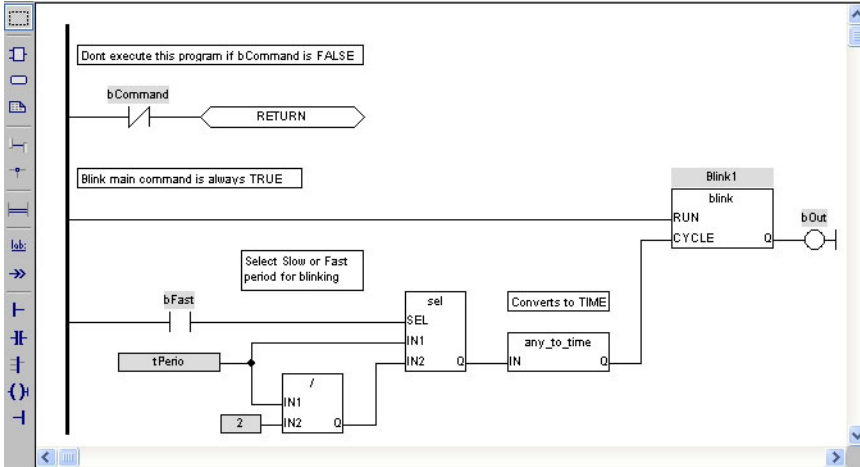
Swap between possible overviews of level 2 in the level 1 chart:

- display code of actions and conditions
- display notes attached to steps and transitions

Notes have no meaning for the execution of the chart. Entering notes for steps and transitions enables you to enhance the auto-documentation of your programs. It also provides an easy way to write and exchange specifications of an SFC program before actions and conditions are programmed.

### 7.6.14 Function Block Diagram (FBD) Editor

The FBD Editor is a powerful graphical tool that enables you to enter and manages Function Block Diagrams according to the IEC 61131-3 standard. The editor supports advanced graphic features such as drag-and-drop, object resizing and connection line routing features, so that you can rapidly and freely arrange the elements of your diagram. It also enables you to insert in a FBD diagram graphic elements of the FFLD (Ladder Diagram) language such as contacts and coils.



FBD diagram components	Related sections
Function blocks	<a href="#">Using the FBD toolbar</a>
<a href="#">Variable tags</a>	<a href="#">Selecting function blocks</a>
<a href="#">Comment texts</a>	<a href="#">Drawing connection lines</a>
<a href="#">Corners</a>	<a href="#">Selecting and entering variables and FB instances</a>
<a href="#">Network breaks</a>	<a href="#">Viewing the diagram</a>
<a href="#">Labels</a>	<a href="#">Moving or copying parts of the diagram</a>
<a href="#">Jumps</a>	<a href="#">Inserting an object on a line</a>
<a href="#">Use of ST instructions</a>	<a href="#">Resizing objects</a>
	<a href="#">Bookmarks</a>
FFLD components:	
<a href="#">Contacts</a>	
<a href="#">Coils</a>	
<a href="#">"OR" vertical rail</a>	
<a href="#">Power rails</a>	

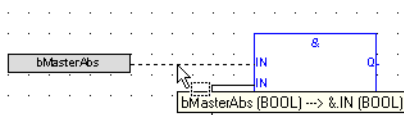
**NOTE**

When a contact or a coil is selected, you can press the **Spacebar** to change its type (e.g. normal, negated, pulse)

Boolean connections can be negative at the entry of a block.

*How to toggle the connection to make it negative?*

1. Select the Boolean connection



Connections in FBD Programs

- Press the **Spacebar** (a small circle is displayed)

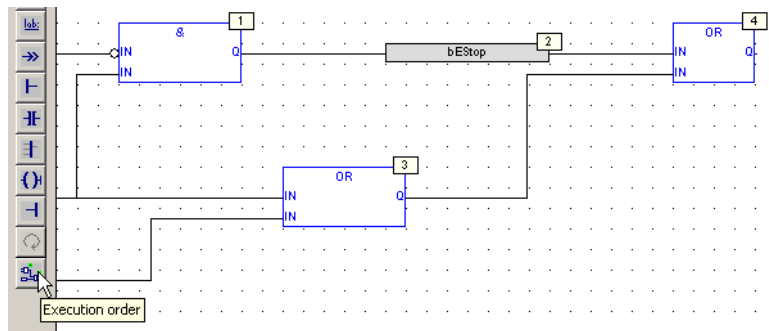


### Toggle Connection in FBD Programs

Execution order can be displayed.

*How to display the execution order?*

Data flow is executed from top left to bottom right (**CTRL+d** shows the execution order)



**Figure 4-80:** Execution Order on FBD

#### 7.6.14.1 Using the FBD toolbar

The vertical toolbar on the left-hand side of the editor contains buttons for all available editing features. Push the desired button before using the mouse in the graphic area.

Icon	Description
	<b>Selection:</b> In this mode, you cannot insert any elements in the diagram. The mouse is used to select object and lines, select tag name areas, or move or copy objects in the diagram. At any moment you can press the ESCAPE key to go back to the Selection mode.
	<b>Insert Block:</b> In this mode, the mouse is used for inserting blocks in the diagram. Click in the diagram and drag the new block to the desired position. The type of block inserted is the one currently selected in the list of the main toolbar.
	<b>Insert variable:</b> In this mode, the mouse is used for inserting variable tags. Variable tags can then be wired to the input and output pins of the blocks. Click in the diagram and drag the new variable to the desired position.
	<b>Insert comment text:</b> In this mode, the mouse is used for inserting comment text areas in the diagram. Comment texts can be entered anywhere. Click in the diagram and drag the text block to the desired position. The text area can then be selected and resized.
	<b>Insert connection line:</b> In this mode, the mouse is used to wire the input and output pins of the diagram objects. The line must always be drawn in the direction of the data flow: from an output pin to an input pin. The FBD editor automatically selects the best routing for the new line. You can change the default routing by inserting corners on lines. (see below). You also can drag a line from an output pin to an empty space. In this case, the editor automatically finishes the line with a user-defined corner so that you can continue drawing the connection to the desired pin and force the routing while you are drawing the line.













Icon	Description
	<b>Insert corner:</b> In this mode, the mouse is used for inserting a user-defined corner on a line. Corners are used to force the routing of connection lines, as the FBD editor imposes a default routing only between two pins or user-defined corners. Corners can then be selected and moved to change the routing of existing lines.
	<b>Insert network break:</b> In this mode, the mouse is used for inserting a horizontal line that acts as a break in the diagram. Breaks have no meaning for the execution of the program; they just help in understanding big diagrams, by splitting them into a list of networks.
	<b>Insert label:</b> In this mode, the mouse is used for inserting a label in the diagram. A label is used as a destination for jump symbols (see below).
	<b>Insert jump:</b> In this mode, the mouse is used to insert jump symbols in the diagram. A jump indicates that the execution must be directed to the corresponding label (having the same name as the jump symbol). Jumps are conditional instructions. They must be linked on their left-hand side to a Boolean data flow.
	<b>Insert left power rail:</b> In this mode, the mouse is used to insert a left power rail in the diagram. A left power rail is an element of the FFLD language, and represents a "TRUE" state that can be used to initiate a data flow. Power rails can then be selected and resized vertically according to the desired network height.
	<b>Insert contact:</b> In this mode, the mouse is used to insert a contact in the diagram, as in Ladder Diagrams.
	<b>Insert "OR" rail:</b> In this mode, the mouse is used to insert a rail that collects several Boolean data flows for an "OR" operation, in order to insert parallel contacts, as in Ladder Diagrams.
	<b>Insert coil:</b> In this mode, the mouse is used to insert a coil in the diagram, as in Ladder Diagrams. It is not mandatory that a coil be connected on its right-hand side.
	<b>Insert right power rail:</b> In this mode, the mouse is used to insert a right power rail in the diagram. A right power rail is an element of the FFLD language, and is commonly used for terminating Boolean data flows. However, it is not mandatory to connect coils to power rails. Right power rails have no meaning for the execution of the diagram.
	<b>Swap item style:</b> change the text justification
	<b>Execution order:</b> Display the execution order of the elements in the diagram.

Table 4-21: FBD Toolbar - List of Icons

#### 7.6.14.1.1 FBD variables

All variable symbols and constant expressions are entered in FBD diagrams using small boxes.

1. Press the following button in the FBD toolbar to insert a variable tag:

 **Insert variable:** In this mode, the mouse is used for inserting variable tags. Click in the diagram and drag the new variable to the desired position.

2. Double-click on a variable tag to open the [variable selection box](#)
3. Either select the symbol of the desired variable or enter a constant expression.

Variables tags must then be linked to other objects such as block inputs and outputs using connection lines.

4. You can resize a variable box vertically in order to display, together with the variable name, its tag (short comment text), its description text, plus its I/O location if the variable is mapped to an I/O channel.

The variable name is always displayed at the bottom of the rectangle:



```

tag
description
%location
name

```

For more details on Variable Tag and Description, see ["Variables tab" on page 67](#)

#### 7.6.14.1.2 FBD comments

Comment text area can be entered anywhere in an FBD diagram.

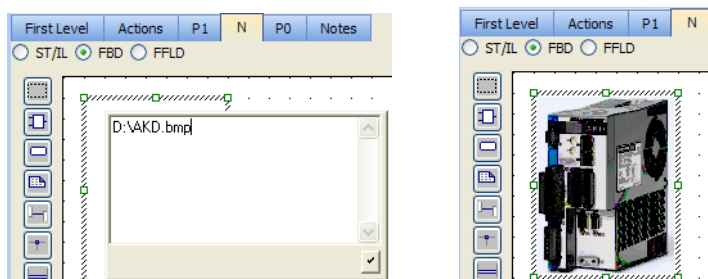
Press the following button  in the FBD toolbar to insert a new comment area.

In this mode, the mouse is used to insert comment text areas anywhere in the diagram.

Double-click on the comments area to enter or change the attached text. When selected, comment texts can be [resized](#).

#### NOTE

You can insert hyperlink on external files as shown below. Only TXT and BMP extensions are allowed. When the link is valid, the hyperlink is replaced with the file's content.




**Figure 4-81: FBD Comments - Inserting Graphic**

#### 7.6.14.1.3 FBD corners

Corners are used to force the routing of connection lines, as the FBD editor imposes a default routing only between two pins or user-defined corners. All variable symbols and constant expressions are entered in FBD diagrams using small boxes.

Press the following button in the FBD toolbar to insert a corner on a line:


 **Insert corner:** In this mode, the mouse is used to insert a user-defined corner on a line.

You can drag a new line from an output pin to an empty space. In this case, the editor automatically finishes the line with a user-defined corner, so that you can continue drawing the connection to the desired pin and force the routing while you are drawing the line.

Corners can then be selected and moved to change the routing of existing lines.

#### 7.6.14.1.4 FBD network breaks

Network breaks can be entered anywhere in an FBD diagram. Breaks have no meaning for the execution of the program; they just help in understanding big diagrams, by splitting them into a list of networks. Press the following button in the FBD toolbar to insert a new break:


 **Insert network break:** In this mode, the mouse is used for inserting a horizontal line that acts as a break in the diagram.

The break line is drawn on the whole diagram width. No other object can overlap a network break. Break lines can then be selected and moved vertically to another location.

Network breaks can also be used to browse the diagram. Press the **Ctrl+Page Up** or **Ctrl+Page Down** keys to move the selection to the next or previous network break.


### 7.6.14.1.5 FBD "OR" vertical rail

The FBD Editor enables the drawing of FFLD rungs. The "OR" rail can be inserted on a rung in order to connect parallel contacts together. Press the following button in the FBD toolbar to insert a new "OR" rail:

 **Insert "OR" rail:** In this mode, the mouse is used for inserting a rail that collects several Boolean data flows for an "OR" operation, in order to insert parallel contacts, as in Ladder Diagrams.

The "OR" rail has exactly the same meaning as an "OR" block regarding the execution of the diagram.

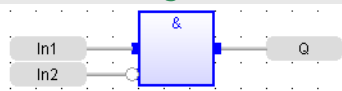
### 7.6.14.2 Draw FBD connection lines

 *Press this button before inserting a new line.*

#### NOTE

As shown below, the editor enables you to terminate a connection line with a Boolean negation represented by a small circle:

(\* use of a negated link: Q is IN1 AND NOT IN2 \*)











To set or remove the Boolean negation, select the line and press the **Spacebar**.

Connection lines must always be drawn in the direction of the data flow: from an output pin to an input pin. The FBD editor automatically selects the best routing for the new line. Connection lines indicate a data flow between the following possible objects:

#### NOTE

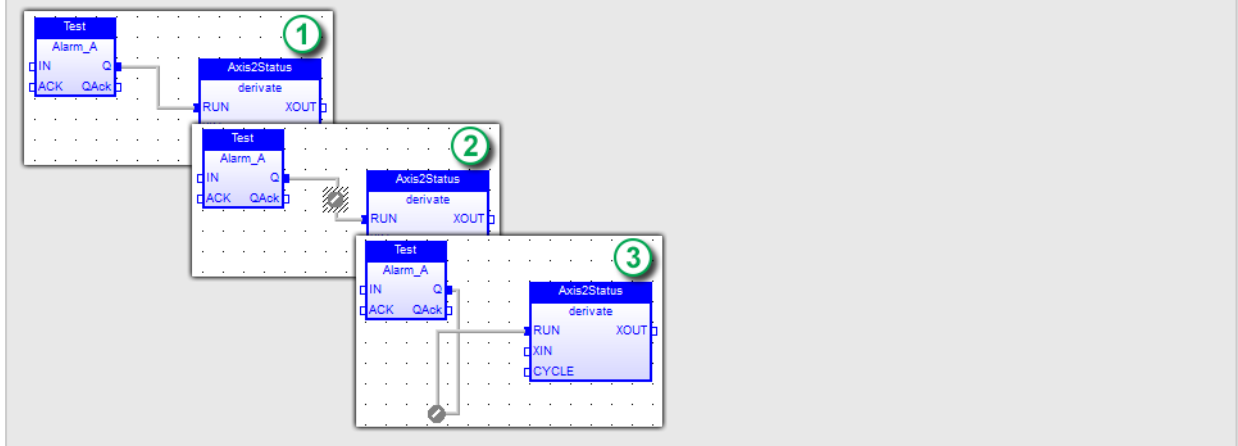
Line is colored in **red** when the two linked items are not the same type.

-  **Block:** Refer to the help on the block for the description of its input and output pins, and the expected data types for the coherence of the diagram.
-  **Variable:** A variable can be connected on its right-hand side (to initiate a flow) or on their left-hand side to force the variable, if it is not "read only". The flow must fit the data type of the variable.
-  **Jump:** a jump must be connected on its left-hand side to a Boolean data flow.
-  **Left power rail:** Left power rails represent a TRUE state and can be connected to a non limited number of objects on their right-hand side.
-  **Contact:** A contact must be connected on its left-hand side and on its right-hand side to Boolean data flows.
-  **"OR" rail:** Such a rail collects several Boolean data flows for an "OR" operation, in order to insert parallel contacts, as in Ladder Diagrams. It may have several connections on its left-hand side and on its right-hand side. All connected data flows must be Boolean.
-  **Coil:** A coil must be connected on its left-hand side to a Boolean data flow. It is not mandatory that a coil be connected on its right-hand side.
-  **Right power rail:** A right power rail is an element of the FFLD language, and is commonly used for terminating Boolean data flows. It has an unlimited number of connections on its left-hand side. It is not mandatory to connect coils to power rails.


#### TIP

Connection lines automatically move and follow FBD elements (1) and you can also manually specify the corners. Simply double-click on the line to show a "handle" (2) which can be







repositioned (3).



### 7.6.14.3 Select FBD variables and instances

 Press this button or press ESCAPE before any selection.

To select the name of the declared variable to be attached to a graphic symbol, you must be in "Selection" mode. Simply double-click on the tag-name gray area. The following types of object must be linked to valid symbols:

	<b>Block:</b> If it is a function block, you must specify the name of a valid declared instance of the corresponding type.
	<b>Variable:</b> Must be attached to a declared variable. Alternatively, a variable box may contain the text of a valid constant expression.
	<b>Label:</b> Must have a name. The name must be unique within the diagram.
	<b>Jump:</b> Must have the same name as its destination label.
	<b>Contact:</b> Must be attached to a declared Boolean variable.
	<b>Coil:</b> Must be attached to a declared Boolean variable.

Symbols of variables and instances are selected using a variable list, that can be used as the variable editor. Simply enter a symbol or constant expression in the edit box and press OK. You can also select a name in the list of declared objects, or declare a new variable by pressing the "Create" button.

For more details, [Select Variables and Instances](#)

### 7.6.14.4 View FBD diagrams

The diagram is entered in a logical grid. All objects are snapped to the grid. The (x,y) coordinates of the mouse cursor are displayed in the status bar. This helps you to locate errors detected by the compiler, or to align objects in the diagram.

At any moment you can zoom in or out of the edited diagram by means of a Ctrl + mouse-wheel operation. You can also press the [+] and [-] keys of the numerical keypad to zoom the diagram in or out.

### 7.6.14.5 Move or copy FBD objects

 Press this button or press ESCAPE before selecting objects

The FBD editor fully supports drag-and-drop for moving or copying objects. To move objects, select and drag them to the desired position.

To copy objects, you can do the same, and just press the CONTROL key while dragging. It is also possible to drag pieces of diagrams from one program to another if both are open and visible on the screen.

At any time while dragging objects, you can press ESCAPE to cancel the operation.

Alternatively, you can use the Copy / Cut / Paste commands from the Edit menu. When you run the Paste command, the editor changes into "Paste" mode, with a special mouse cursor. Click in the diagram and move the mouse cursor to the desired position for inserting pasted objects.

### Using the keyboard

When graphic objects are selected, you can move them in the diagram by pressing the following keys:

Shift + Up	Move to the top
Shift + Down	Move to the bottom
Shift + Left	Move to left
Shift + Right	Move to right

When an object is selected, you can extend the selection by pressing the following keys:

Shift + Control + Home	Extend to the top: select all objects before the selected one
Shift + Control + End	Extend to the bottom: select all objects after the selected one

To insert or delete space in the diagram, you can simply select an object, press Shift+Ctrl+End to extend the selection, and then move selected objects up or down.

### Auto alignment

When objects are selected, the following keystrokes automatically align them:

Control + Up	To the top
Control + Down	To the bottom
Control + Left	To left
Control + Right	To right

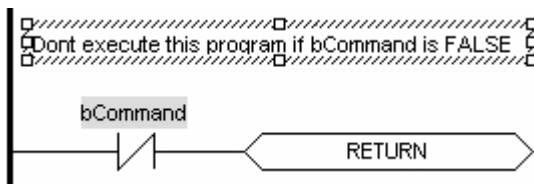
#### 7.6.14.6 Insert FBD objects on a line

The FBD editor enables you to insert an object on an existing line and automatically connect it to the line. This feature is available for all objects having one input pin and one output pin, such as variable boxes, contacts and coils. This feature is mainly useful when entering elements of Ladder Diagrams. Just draw a horizontal line between left and right power rails: it is the rung. Then you can simply insert contacts and coils on the line to build the FFLD rung.

#### 7.6.14.7 Resize FBD objects

 Press this button or press ESCAPE before selecting objects.

When an object is selected, small square boxes indicate how to resize it with the mouse. Click on the small square boxes to resize the object in the desired direction.



Not all objects can be resized. The following table indicates possible operations:

<b>Variable</b>	Horizontally and vertically (*)
<b>Block</b>	Horizontally
<b>Labels and jumps</b>	Horizontally
<b>Power rails</b>	Vertically
<b>OR rail</b>	Vertically
<b>Comment area</b>	In all directions

(\*) Resizing a variable box vertically enables you to display together with the variable name its tag (short comment text), its description text, plus its I/O location if the variable is mapped to an I/O channel. The variable name is always displayed at the bottom of the rectangle:

```
% location  
description  
tag  
name
```

### 7.6.15 Structured Text (ST) / Instruction List (IL) Editor

The ST / IL editor is a powerful language-sensitive text editor dedicated to IEC 61131-3 languages. The editor supports advanced graphic features such as drag-and-drop, syntax coloring and active tooltips for efficient input and test of programs in ST and IL.

```

Blinker (TRUE, t#2s);
Trigger (Blinker.Q);

bSig := Trigger.Q;

Counter (
  bSig,          (* blinking input *)
  not bCommand, (* reset the counter if command *)
  255
);
iValue := Counter.CV;

```

*Related sections:*

- [Language selection](#)
- [Syntax coloring](#)
- [Autocompletion of words](#)
- [Drag-and-drop](#)
- [Active tooltips](#)
- [Selecting function blocks](#)
- [Inserting variable and FB instances symbols](#)
- [Reading output of a FB instance](#)
- [Bookmarks](#)

#### **TIP**

Ctrl + Spacebar opens the [Variable Editor](#) dialog box

#### 7.6.15.1 ST / IL Language selection

The KAS IDE allows you to mix ST and IL languages in textual program. ST is the default language. When you enter IL instructions, the program must be entered between "BEGIN\_IL" and "END\_IL" keywords, such as in the following example

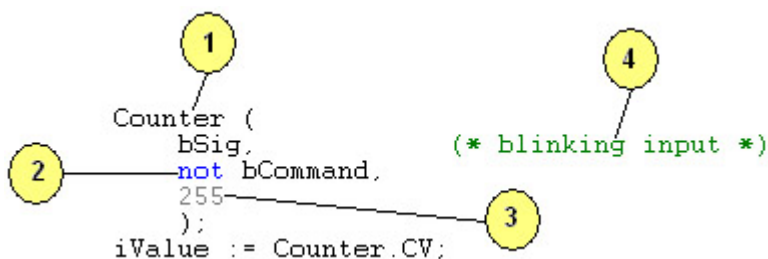
```

BEGIN_IL
  FFLD   var1
  ST     var2
END_IL

```

#### 7.6.15.2 ST / IL Syntax coloring

The ST / IL editor supports syntax coloring according to the selected programming language (ST or IL). The editor uses different colors for the following kinds of words:



1. Default (identifiers, separators)
2. Reserved keywords of the language
3. Constant expressions
4. Comments

**TIP**

Reserved words such as function names are blue. Subprograms including Kollmorgen UDFBs are not reserved and will show as black.

**7.6.15.3 Intellisense**

The following features are available with Intellisense in ST and FBD programs:

**NOTE**

They do not apply to actions in an SFC step.

Conditional compiling coloring

Parts of conditional compiling code (declared with `#ifdef` pragmas) that are not validated are grayed

```
#define CONDITION

#ifdef CONDITION

if tryGetSpike = true then
    MachineState := 2;
    MachineSpeed := 2000;
end_if;

#else

Printf('Manual mode', 0, 0, 0, 0);

// Start motion
MLMstRun(PipeNetwork.Master, TravelSpeed);

#endif
```

Commenting the `CONDITION` changes the active part of the program

```
// #define CONDITION

#ifdef CONDITION

if tryGetSpike = true then
    MachineState := 2;
    MachineSpeed := 2000;
end_if;

#else

Printf('Manual mode', 0, 0, 0, 0);

// Start motion
MLMstRun(PipeNetwork.Master, TravelSpeed);

#endif
```

**NOTE**

Save your project to have the code with the correct colors.

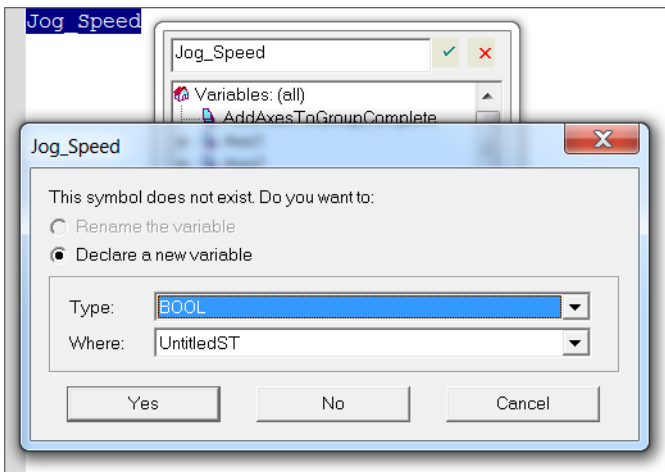
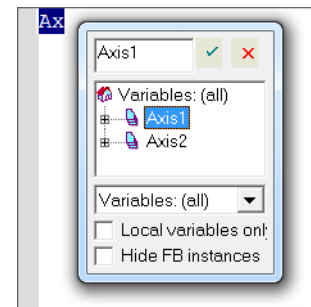
Auto-indentation

Lines are automatically indented on the left when you enter structured ST statements

Autocompletion**7.6.15.4 Auto-completion of words**

The ST / IL editor includes powerful commands for automatic completion of typed words, according to declared variables and data types. Pressing Ctrl-space (or alternatively, Ctrl-J) activates the auto-completion.

If you use auto-complete and there is no match for a variable name in the Dictionary, pressing ENTER will open the variable creation box where you can define a new variable.



The following features are available:

Auto-completion of:	Action
<b>Variable Name</b>	If you enter the first letters of a variable name, you can press <b>CTRL+J</b> to automatically complete the name. A pop-up list is displayed with possible choices if several declared variable names match the typed characters.
<b>Missing Symbols</b>	When you press <b>ENTER</b> at the end of a line containing an unknown variable symbol, you are prompted to declare it immediately.
<b>FB Member</b>	When you type the name of a function block instance (used either as an instance or a data structure), pressing the point "." after the name of the instance opens a pop-up list with the names of possible members.
<b>FB Call</b>	Type the name of a function block followed by an opening parenthesis <div style="border: 1px solid black; padding: 5px; margin: 5px 0;"> <pre>MLMstRun (</pre> </div> Press the <b>ENTER</b> key to complete the instructions with the appropriate argument list, including comments and possibly default values so that you are guided through the list of values to be passed to the called function. <div style="border: 1px solid black; padding: 5px; margin: 5px 0;"> <pre>MLMstRun (     (* BlockID : DINT *) ,     (* Speed : LREAL *) );</pre> </div>



Auto-completion of:	Action
<b>ST Block Statement</b>	<p>On an empty line, enter the main keyword of a ST statement such as "for", "if"...</p> <pre>FOR</pre> <p>Press the <b>ENTER</b> key to complete the whole statement, including comments that will guide you through the syntax.</p> <pre>FOR (* DINT var *) := (* minimum : DINT *) TO (* maximum : DINT *) BY 1 DO  END_FOR;</pre>

#### 7.6.15.4.1 Other syntax related commands

When lines are selected, you can automatically indent them. Press **TAB** or **Shift+TAB** to shift the lines to the left or right, by adding or removing blank characters on the left.

#### 7.6.15.5 ST / IL Drag-and-drop features

The ST / IL Editor supports powerful drag-and-drop features that help you to develop and test your programs. You can:

- Drag text (words or lines) from the ST / IL editor to another application (such as a text editor)
- Do the opposite
- Drag a variable symbol from the variable editor to the ST / IL editor
- Drag a variable symbol from the ST / IL editor to the watch list (\*)

(\*) When dragging the symbol of an array to the watch list, all items in the array are added to the watch list.

#### 7.6.15.6 How to Read Output of a MC Function Block in ST

In the following example:

```
A6_Inst_MC_MoveRelative( 1, Axis6, -90, 5, 300000, 300000, 0, 0 );
```

A6\_Inst\_MC\_MoveRelative is an instance of MC\_MoveRelative PLCopen Motion function block. The values given in parenthesis correspond to the 8 inputs of this FB.

The syntax to read one of the outputs in ST for this instance is:

```
<FUNCTION BLOCK NAME>.<OUTPUT>
```

This FB has the following 5 outputs: Busy, Active, CommandAborted, Error, and Error ID. So for instance, the **Active** output has the following ST expression: A6\_Inst\_MC\_MoveRelative.**Active**

Example 1:

```
UserVariable1 := A6_Inst_MC_MoveRelative.Error ;
```

Example 2:

```
    IF A6_Inst_MC_MoveRelative.Active THEN  
UserVariable2 := 1 ;  
ELSE  
UserVariable2 := 0 ;  
END_IF;
```

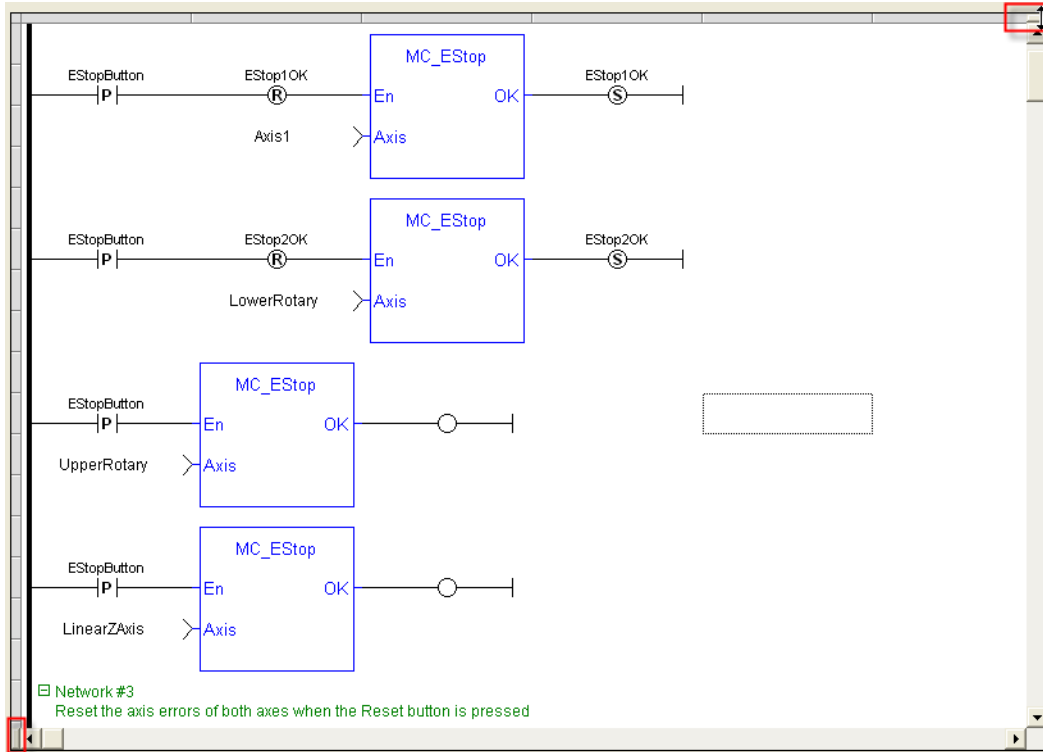
### 7.6.16 Free Form Ladder Diagram (FFLD) Editor

The FFLD Editor is a powerful graphical tool that enables you to enter and manage Ladder Diagrams according to the IEC 61131-3 standard. This Editor enables free drawing and arrangement of FFLD items, and supports advanced graphic features such as:

#### Drag-and-drop operation

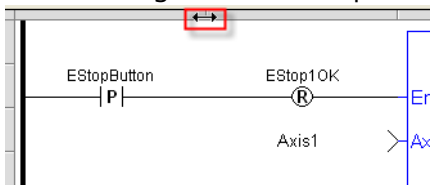
##### Split Window capability

Allows multiple views of the same ladder program to be displayed simultaneously. You can drag the two splitters located in the vertical and horizontal scroll bars to organize your windows.

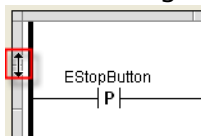


##### Change the size of the ladder diagram

You can drag the column separator to increase or decrease the size of the columns.

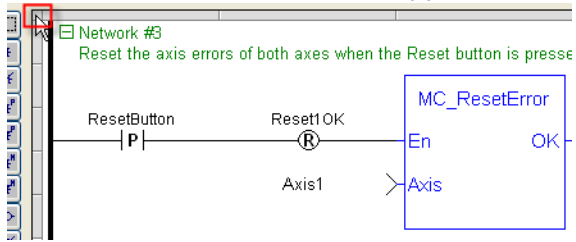


You can drag the row separator to increase or decrease the size of the rows as well as the texts.



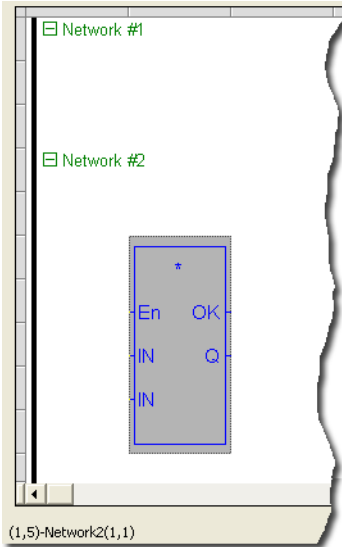
##### Select all the Ladder Diagram

You can click the border in the upper left corner to select the entire ladder.



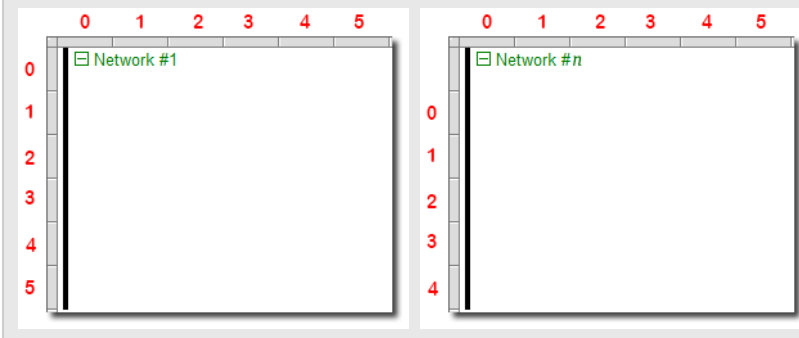
*Coordinate-based position*

The coordinates for a selected function are displayed at the bottom-left corner of the editor. The coordinates are shown as (absolute coordinates)-(network number).(relative coordinates) in (column, row) format.



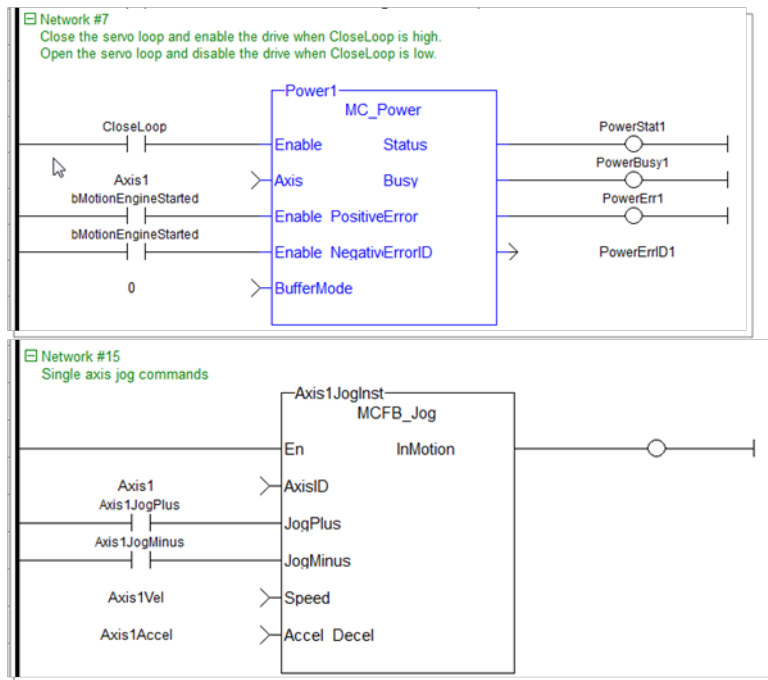
**NOTE**

Relative coordinates are measured from below the Network label.



*Color Coded Function*

Functions and Function Blocks are color-coded as blue, while subprograms and Kollmorgen UDFBs are black.



FFLD diagram components	Related sections
<a href="#">Networks</a> <a href="#">Power rail and lines</a> <a href="#">Contacts and coils</a> <a href="#">Function blocks</a> <a href="#">Data In/Out</a> <a href="#">Jumps and RETURN</a>	<a href="#">Using the FFLD toolbar</a> <a href="#">Selection grid</a> <a href="#">Moving and copying items</a> <a href="#">Run-time</a>

### TIP

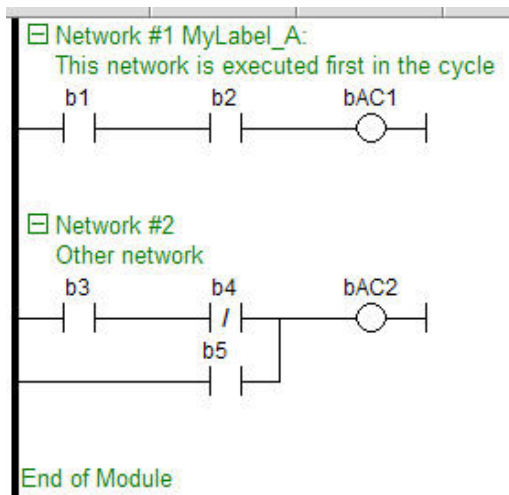
When a contact or a coil is selected, you can press the **Spacebar** to change its type (normal, negated, pulse)

### Networks

A program is entered as a list of independent networks. Networks are executed sequentially from the top to the bottom. The head of a network is drawn on a full row in the editor, grouping the following pieces of information:

- The number of the network (from 1 to N)
- (Optional) A label name used as a target for [jump](#) operations
- (Optional) A directive for [conditional compiling](#)
- (Optional) A multiple line description (comment)

No item can be put on a network header row. No line can go through it. The end of a program is marked with a special "End of module" row. Nothing can be inserted after this row.



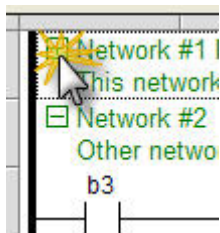
Double-click on the header of a network to enter its label, directive (sometimes called pragma) and description. Network headers are green, but they became blue when a directive is defined (see also [Conditional Compiling](#) for more details).

New networks can be inserted on empty rows.

When a network is selected, pressing "DEL" merges its content with the previous network. When the first network is selected, pressing "DEL" removes the network and its whole contents.

There cannot be two networks having the same label in a program. If such a situation occurs in the case of a copy operation, you will be prompted to either specify another label name for the new network, or remove the label on the new network.

You can also collapse/expand a network with the minus/plus sign located next to the Network number in the header.



### **Run-time**

When your application is running, you can force and lock a variable or a contact directly in the editor with a double-click operation. For more information, see ["Forcing a variable" on page 232](#).

#### **NOTE**

In FFLD, when a function, function block or UDFB is not connected on the left, then it is ignored (removed at compiling time).

### **7.6.16.1 Using the FFLD Editor**

This section describes the [Toolbar](#) icons and [Contextual Menu](#) of the FFLD Editor.

For FFLD accelerator keys, refer to [FFLD Editor Keyboard Shortcuts](#)

#### **7.6.16.1.1 Toolbar**

The vertical toolbar on the left-hand side of the Free Form Ladder editor contains buttons for inserting items in the diagrams. Items are inserted at the current position in the diagram.

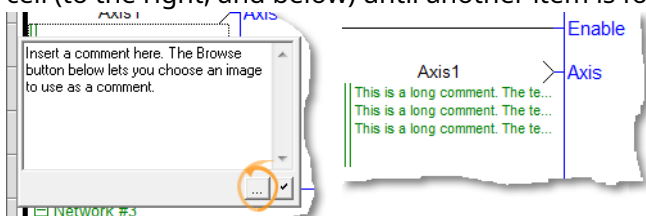
Icon	Shortcut	Description
		Mode selection
	Ctrl+Shift+O	Insert a contact to the destination cell
	Ctrl+Shift+C	Insert an inverted contact to the destination cell
	Ctrl+Shift+P	Insert a Pulse contact to the destination cell
	Ctrl+Shift+I	Insert an inverted Pulse contact to the destination cell
	Ctrl+Shift+N	Insert a N contact to the destination cell
	Ctrl+Shift+A	Insert an inverted N contact to the destination cell
	Ctrl+Shift+E	Insert a coil to the destination cell
	Ctrl+Shift+D	Insert an inverted coil to the destination cell
	Ctrl+Shift+S	Insert a set coil to the destination cell
	Ctrl+Shift+R	Insert a reset coil to the destination cell
	Ctrl+Shift+K	Insert a positive coil to the destination cell
	Ctrl+Shift+L	Insert a negative coil to the destination cell
	Ctrl+Shift+H	Trace a horizontal line to the destination cell
	Ctrl+Shift+V	Trace a vertical line to the destination cell
	Ctrl+Shift+B	Trace a vertical and horizontal line to the destination cell
		Toggle trace mode: click and move the mouse to draw a line spanning on several adjacent cells
	Shift+Insert	Insert a network
	Ctrl+Shift+J	Insert a jump
	Ctrl+Shift+T	Insert a return
	Ctrl+Shift+F	Insert a data in
	F8	Insert a function block
	Ctrl+Shift+Q	Insert a data out
	Spacebar	Swap item style of the current cell for a contact or coil
		Define a network label, pragma, and/or comment
	Shift+C	Define a comment in a cell. Click the button, then select the cell you wish to add a comment to. For more information, see <a href="#">Contextual Menu</a> for a description of cell comments.

Table 4-22: FFLD Toolbar - List of Icons

### 7.6.16.1.2 Contextual Menu

A right-click in the FFLD workspace gives you access to the following commands:

- Insert Comment  
Comments may be added to cells. If the text is long it can display beyond the bounds of the cell (to the right, and below) until another item is found.



- Insert Network
- Insert Row
- Delete Cell

- Delete Network
- Delete Row

### 7.6.16.2 Power rail and lines

Vertical power rails are used in FFLD language to represent the limits of a rung.

The power rail on the left represents the TRUE value and initiates the rung state. Any object connected to this rail is always powered.

Horizontal lines always represent a data flow from the left to the right.

If a vertical line has several items connected on the left, then it represents an OR operation.

You can insert a segment of horizontal line at any location in order to freely draw flow lines. The "vertical line" button enables you to set or remove (toggle) a segment of vertical line on the right of the selected cell.

### 7.6.16.3 Contacts and coils

The table below contains a list of the contact and coil types available:

Contacts	Coils
Normally Open -   -	Energize -( )-
Normally Closed - / -	De-energize -( / )-
Positive Transition - P -	Set (Latch) -(S)-
Negative Transition - N -	Reset (Unlatch) -(R)-
Normally closed positive transition - /P -	Positive transition sensing coil -(P)-
Normally closed negative transition - /N -	Negative transition sensing coil -(N)-

### 7.6.16.4 Function blocks

Functions and function blocks can be used in FFLD diagrams. Blocks are always connected to the flow line (powered) by their first input and first output. If the first input of a block is not Boolean, a special input called "EN" is added, and means that the block is not executed if the input flow is FALSE. If the first output is not Boolean, a special output called "OK" is added. The special "OK" output always has the same state as the first input (the flow).

In the case of a function block, the instance of the block must be specified and is shown on the top of the block. Double-click on the top of the block to select the instance. You can also double-click elsewhere in the block to change its type.

Boolean inputs and outputs of blocks can be directly linked to contacts and coils. Block inputs and outputs can also be specified using specific data in/out items (see below).

#### NOTE

Function and function blocks cannot be put in column 1 of the grid. This would not make sense because data inputs require a column.

You cannot change a function block after it has been inserted.

When a Function is not connected on the right, then it is ignored (removed at compiling time). It is the case for Functions only - **not** for function blocks.

A Function is just part of an expression (same as a contact) and is just intended to provide a result. In case of FFLD, KAS accepts that the output is not connected because it accepts pending "dead" expressions to be removed at compiling time (same as contacts with no coil or FBs after).

#### TIP

If you want another function block, you first have to select it in the [Libraries](#) toolbox before inserting it.



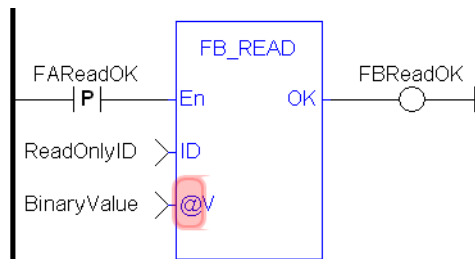
### 7.6.16.5 Data In/Out

The "data in" and "data out" items are used to initiate a flow (line) with the value of a variable, or to force a variable on output with the value of a flow:

VarIn>- ..... ->VarOut

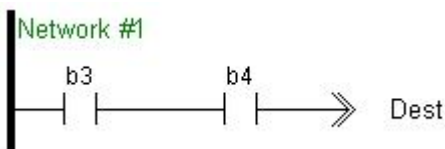
When used with a block, the "data in" and "data out" items can be put close to the block, without any line in-between to connect a variable to an input or output of the block.

In the following example, the @ symbol in front of the V variable indicates it is used for input and output.



### 7.6.16.6 Jumps and RETURN

A jump to a label branches the execution of the program after the specified label. In FFLD language, the ">>" symbol (followed by the target label name) is used as a coil at the end of a rung.



The jump is performed only if the rung state on input is TRUE. The destination label must be specified on a network of the same program.

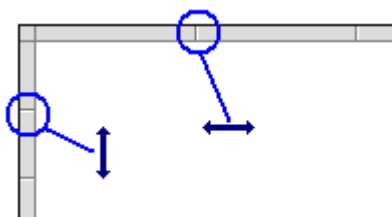
To specify the destination, double-click the cell to display a drop-down menu that lists all the available labels.

**NOTE**  
The special "<RETURN>" destination specifies a jump to the end of the program.

### 7.6.16.7 Selection grid

The diagram is entered in a logical grid, and all objects are snapped to the grid.

At any moment you can zoom in or out of the edited diagram (for shortcuts about zooming, see "Using the FFLD Editor" on page 294). You can also drag the separation lines in vertical and horizontal rulers to freely resize the cells of the grid:



**NOTE**

If a split window is in use, the zoom applies only to the currently selected split window (each split window can be zoomed to different levels).

The current position in the grid is always highlighted by a dotted cell and its coordinates (row, column) are displayed at the bottom left-hand corner of the editor.

If you click on the current position, then the cell is drawn as gray, meaning that it can be dragged somewhere else in the diagram (see below). You can also select multiple cells with the mouse, or use the arrows of the keyboard with the SHIFT key pressed.

Click on the power rail (gray ruler at the left border) to select a full row.

Other selection commands are available from the keyboard:

Keystroke	Description
<b>Home</b>	moves the caret to the left of the line if pressed again, moves the caret to the head of the network
<b>End</b>	moves the caret to the end of the line if pressed again, moves the caret to the end of the network
<b>Ctrl + Page Up / Down</b>	moves the caret to previous or next network header
<b>Ctrl + Home/End twice</b>	moves the caret to the beginning or the end of the program
<b>Ctrl + A</b>	selects the whole network if pressed again, selects the whole program
<b>Page Up / Down</b>	scroll 1 page
<b>Shift-Page Up / Down</b>	selection page up or down
<b>Return</b>	equivalent to a double-click
<b>Space</b>	change contact or coil
<b>Tab</b>	move focus cell right
<b>Shift-Tab</b>	move focus cell left
<b>Arrows</b>	move focus cell or scroll through ladder
<b>Shift-Arrow</b>	multi-select cells
<b>Ctrl + F</b>	performs a Search and Replace (+ add hyperlink on the topic) within the whole program
<b>Ctrl-Shift-F2</b>	go to previous bookmark
<b>Esc / Shift-Esc</b>	close the rename widget

### 7.6.16.8 Move and copy items

When you click on the current position, then the cell is drawn as gray, meaning that it can be copied or moved. Click again on the selection to drag it with the mouse.



Dragging the selected items moves them to the specified location. If you press the **CTRL** key while dragging, then items are copied (for shortcuts, [Using the FFLD Editor](#)).

To move a function block, you must select it entirely.

If you move or copy items on a non-empty area, you will be prompted to confirm the overwriting of items in the area.

When you move or copy items only on a network header, the network is automatically moved in order to make the required extra space for moved items.

The "Copy / Cut / Paste" commands can also be used as an alternative to drag-and-drop.

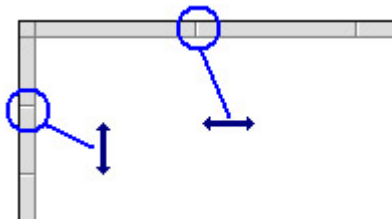
A rectangular selection within the diagram cannot cross a network header, i.e. all selected items must be within the same network. To select a complete network or more, you must select complete rows. To do this, move the caret to the left border or click on the left-hand side ruler (gray).

#### 7.6.16.9 View FFLD diagrams

The diagram is entered in a logical grid. All objects are snapped to the grid. The (x,y) coordinates of the mouse cursor are displayed in the status bar. This helps you to locate errors detected by the compiler, or to align objects in the diagram.

At any moment you can zoom in or out of the edited diagram by means of a Ctrl + mouse-wheel operation. You also can press the [+] and [-] keys of the numerical keypad to zoom the diagram in or out.

You can also drag the separation lines in vertical and horizontal rulers to freely resize the cells of the grid:



The FFLD editor adjusts the size of the font according to the zoom ratio so that the name of variables associated with contacts and coils are always visible. If cells have sufficient height, variable names are completed with other pieces of information about the variable:

- its tag (short description)
- its description text
- its I/O name (%...) if the variable has a user-defined name.

## 7.7 Create Variables

### **TIP**

As a naming convention for variables, it is recommended to use the initial to reflect the variable type (e.g. Boolean with **b**; long integer with **L**)

### 7.7.1 Use the Dictionary

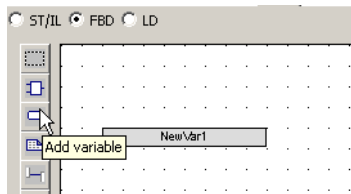
For explanations on dictionary usage, including how to create and [rename](#) variables, [Dictionary Contextual Menu](#)

### 7.7.2 Create Variables from the Editors

You can create variables directly from the IEC 61131-3 editors, as follows:

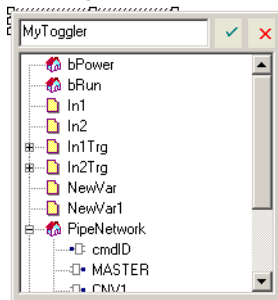
#### FBD editor

1. Click the dedicated button



**Figure 4-82:** Add Variable in FBD Editor

2. Click a location in the editor (or double-click the variable if it is already created)
3. Edit the name in the [Variable Editor](#) (or select an existing variable within the list which is already filtered according to their relevant data type)



**Figure 4-83:** Define Variable Name in FBD Editor

4. The KAS IDE automatically checks if the variable already exists. If it is new, you have to:
  - Select its type in the drop-down menu: for FBD and FFLD, it is set by default according to the In or Out data type of the function block
  - Specify where it is defined: the default is the current PLC program, but you can choose to make the variable Global or declared as a retain variable



Figure 4-84: Define Variable Type in FBD Editor

## FBD variables

### FFLD editor

1. Double-click the in or out pins of the function block

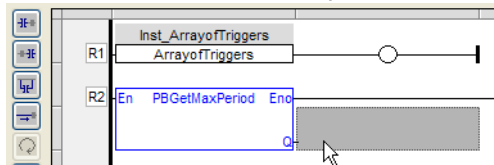


Figure 4-85: Add a Variable in the FFLD Editor

2. Edit the name (or select an existing variable within the list which is already filtered according to their relevant data type)

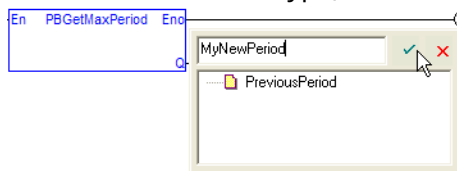


Figure 4-86: Define a Variable Name in the FFLD Editor

3. The KAS IDE automatically checks if the variable already exists. If it is new, you have to:
  - Select its type in the drop-down menu (by default, it is set according to the In or Out data type of the function block)
  - Specify where it is defined

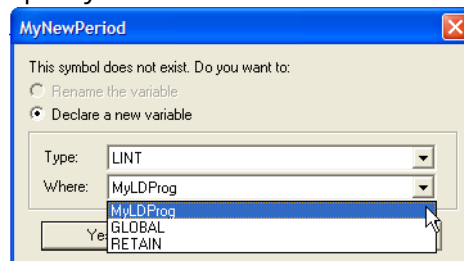


Figure 4-87: Define a Variable Type in the FFLD Editor

## 7.7.3 Data Types

You can create a variable of available Data Types.

*How to declare an array?*

1. Double-click in the corresponding cell of the variable editor (i.e. the **Dim.** column)
2. Enter its dimension (**Note:** for a multi-dimension array, enter dimensions separated by commas (ex: 2,10,4))

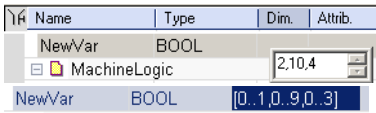


Figure 4-88: Declare an Array for an Internal Variable

Arrays

7.7.4 Complex Structures

Complex variables are arrays, structures, and instances of function blocks. The following features are allowed for programming:

- Use arrays of structures
- Use arrays of FB instances
- Pass any complex data (array, structure, instance) to a UDFB or sub-program

There is almost no limitation in the amount of complex data declared (theoretically up to 4GB, but practically limited by the memory available in the runtime)

For more explanations on the **Structure** concept, refer to [Structures](#)

Declare the structure

1. Right-click in the Dictionary to open the menu
2. Select the **Add structure** command

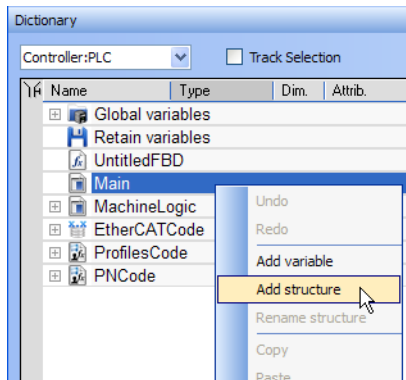


Figure 4-89: Add a Complex Structure

3. Right-click on the newly created structure and select the **Rename structure** command

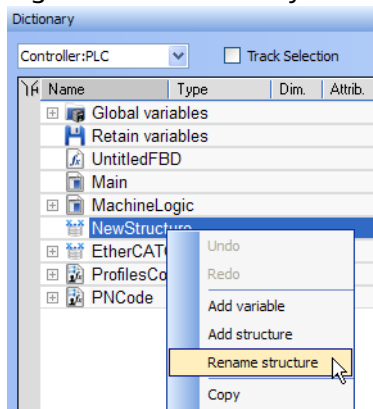
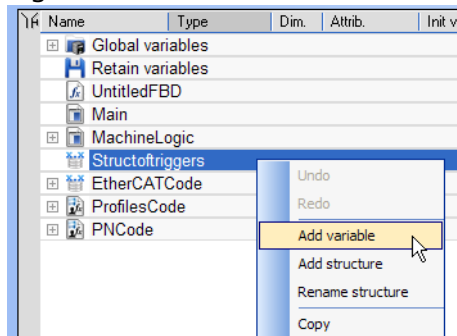


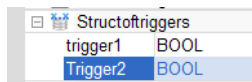
Figure 4-90: Rename Complex Structure

- Right-click on the new structure and select the **Add variable** command



**Figure 4-91:** Add Variable to a Complex Structure

- Expand the new structure
- Double-click on the new nested variable and define its name and type

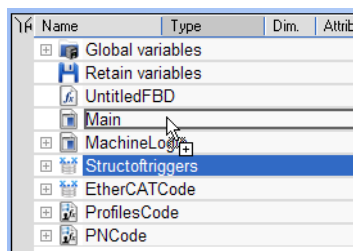


- Repeat steps 5 and 7 to add all the requested variables

### Create an instance of the structure

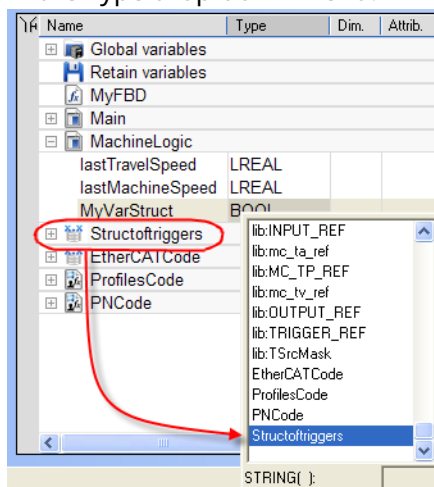
When finalized, you can drag-and-drop the structure from the library in the **(Project)** node to a program just like any other function block. A new instance is automatically created.

- Select the new structure and move it with a drag-and-drop operation to the program declaration within the Dictionary



**Figure 4-92:** Create an Instance of the Structure

- You can also add a variable in the Dictionary with the **Add variable** command. Then double-click on the new variable to define its type by selecting the structure type which is displayed in the Type drop-down menu.



- Then you can drag this new instance and drop it in your program like any other variable

### 7.7.5 Variable Editor

You can edit variables directly from each IEC 61131-3 editor.

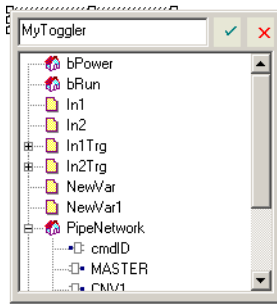


Figure 4-93: Edit the Name in the Variable Editor

#### TIP

Ctrl + Spacebar opens this dialog box

KAS IDE automatically checks if the variable already exists. When the variable is new, you have to:

- Select its type in the drop-down menu: for FBD and FFLD, it is set by default according to the In or Out data type of the function block
- Specify where it is defined: the default is the current PLC program, but you can choose to make the variable Global or declared as a retain variable

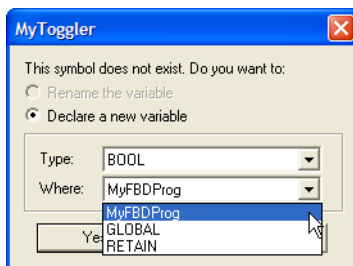
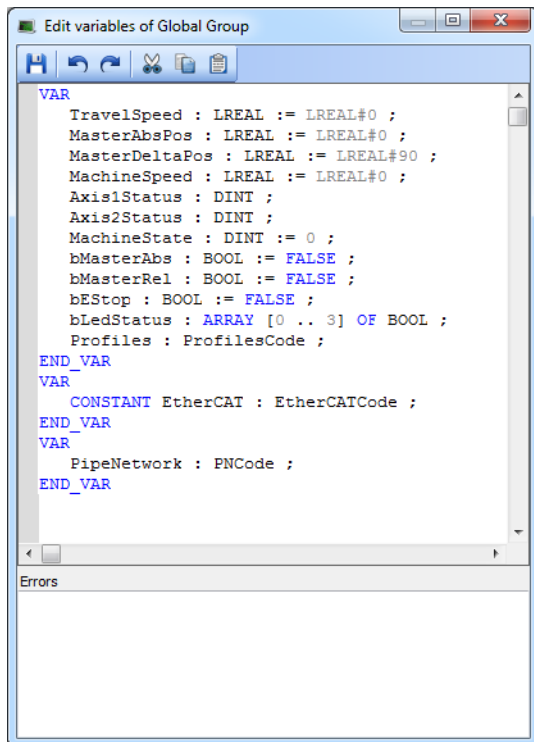


Figure 4-94: Define Type and Scope of the Variable

### 7.7.6 Editing Variables as Text

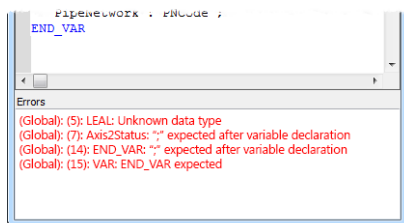
As an alternative for editing variables, they may be edited as text. Text editing applies to all the variables of a group. Selecting "Edit variables as text" from the right mouse menu opens a dialog box which contains all of the variables in the group. From here, variables may be added, deleted, or edited using the IEC61131-3 format.





**Figure 4-95:** Editing variables as text

The editing dialog contains the basic text-editing functions, Save, Undo, Redo, Cut, Copy, and Paste. Upon saving changes, the variables are validated. If there are errors in the declaration, the changes are not saved and errors are listed in the "Errors" section of the dialog box.



**Figure 4-96:** Errors caused by editing variables.

Each error includes the group name, the line number where the error occurs, and what the error is. Double-clicking on an entry will put the cursor at the point of the error.

### ⓘ IMPORTANT

If variables that are used in PDO mapping are renamed or deleted, the mapping will be lost upon saving changes. If this occurs, you will be informed with an alert.

For more information on editing variables as text, [see "Editing variables as text using IEC 61131-3 syntax" on page 80.](#)

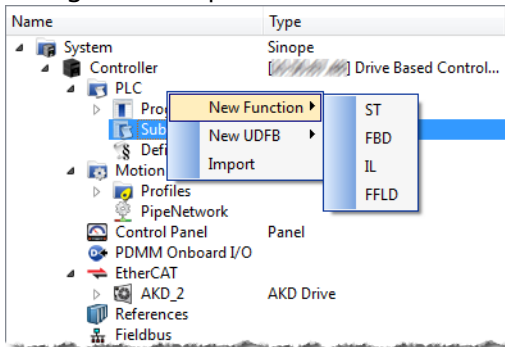
## 7.8 Create Functions and Function Blocks

For explanation about the difference between functions and function blocks, refer to [Program Organization Units](#).

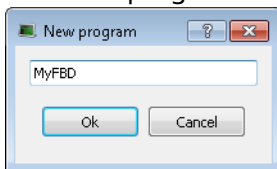
### 7.8.1 Declare Functions or Function Blocks

This section explains how to create a new function or UDFB.

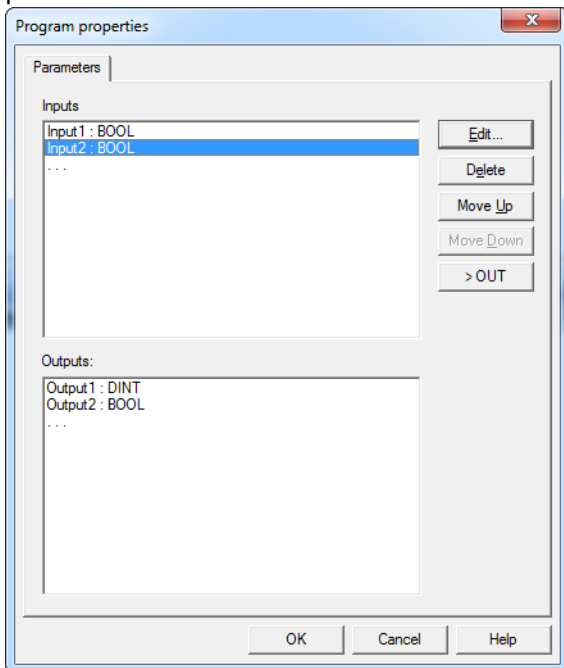
1. Expand the PLC node in the Project Explorer, and right-click on the [Subprograms](#) item and select the action you wish to perform (i.e., *New Function* or *New UDFB*). The New Program dialog box will open.



2. Enter the program name (MyFBD, for example), and click OK.

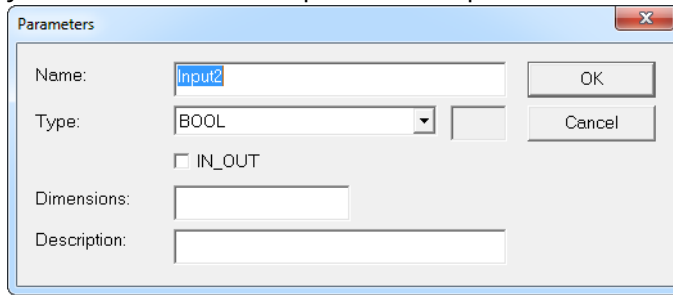


3. The *Program Properties* dialog box opens. This is where you will enter the input and output parameters for the function or UDFB.



4. Clicking on the editable space ("...") in Inputs or Outputs enables the **Edit** button. Clicking the Edit button or double-clicking on the editable space opens the *Parameters* dialog which lets

you add variables for Inputs and Outputs.



Item	Description
<b>Edit</b>	Press this button to change the definition of the selected parameter. Pressing this button when the last line (...) of the list is selected lets you add a new parameter.
<b>Delete</b>	Press this button to remove the selected parameter.
<b>Move Up/Down</b>	Press these buttons to move the selected parameter in the list to arrange the order of parameters. The order is very important as it defines the calling prototype of the UDFB or sub-program.
<b>Swap</b>	Pressing this item moves the selected parameter between the Inputs and Outputs lists. When an item is selected the button's text changes to show where the parameter will be moved.

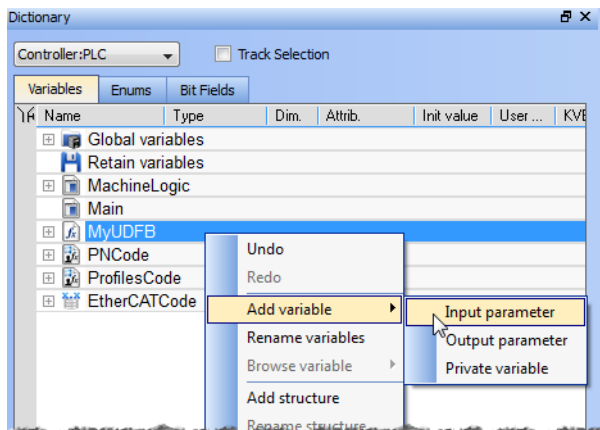
#### NOTE

New UDFBs are added to the **(Project)** node in the [Library toolbox](#)

## 7.8.2 Define Parameters and Private Variables

For a Function or UDFB, input and output parameters (as well as private variables) are declared in the Dictionary toolbox as local variables of the item. The **Add variable** command let you add the following:

- **Input1** Parameter
- **Output2** Parameter
- **Private3** Variable



**Figure 4-97:** Parameters and Private Variables

<sup>1</sup>Externally supplied, not modifiable within the organization unit

<sup>2</sup>Supplied by the organization unit to external entities

<sup>3</sup>Supplied by external entities - can be modified within organization unit

Input and Output parameters always appear at the beginning of a UDFB group.

Pressing the **INSERT** key when the item is selected adds a private variable.

#### NOTE

There are some things to be aware of with UDFB parameters:

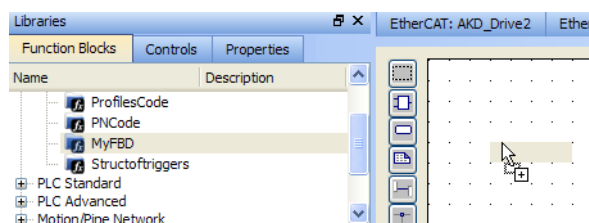
- UDFB cannot contain parameters being both for Input and Output
- Simple parameters (scalar type) can be either IN, OUT, or IN\_OUT
- Complex parameters (an array or structure) should be declared as IN but is systemically considered to be IN\_OUT.
- UDFB cannot have more than 32 input parameters or 32 output parameters
- Output parameters can only be simple [data type](#)

### 7.8.3 Finalize Functions or Function Blocks

Double-click the item in the Project Explorer to open and complete it in its corresponding editor.

### 7.8.4 Call Functions or Function Blocks

When finalized, you can drag-and-drop UDFBs from the library in the **(Project)** node to a program just like any other function block. A new instance is automatically created.



**Figure 4-98:** Create an Instance of UDFB in a Program

#### NOTE

- A single data type parameter defined as IN is passed by the calling program to the UDFB and the body UDFB cannot change its value
- A single data type parameter defined as OUT is set in the body UDFB and always actuated in the calling program after the call
- A parameter which is an array or a structure is always declared as IN (visible on the left of the block). Both the calling program and the body of the UDFB can read and write such a parameter

## 7.9 Use the Defines List

The Defines list consists of defined constants, (an expression with a fixed value). Defines are both pre-defined (internal) and user-created (global and local). Defines are used to determine which parts of a program's code will be compiled using an `ifdef` statement (see [Conditional Compiling](#)). This creates more efficient code for a given machine type. For example, you can write a program that covers many machine types but compile for a specific machine with more efficient code.

Defined constants have three levels of scope:

Level	Scope
<a href="#">Internal Defines</a>	All the projects present on your machine
<a href="#">Global Defines</a>	All the programs within your project. These are user-defined.
<a href="#">Local definitions</a>	Only the current program currently open

### NOTE

**Important!** To guarantee precision when evaluating the expression, you need to pay special attention to the [data types](#) of variables used in the expression. For example when mixing **LREAL** and **REAL**, the precision will be that of **REAL**.

KAS IDE supports the definition of aliases. An alias is a unique identifier that can be used in programs to replace other text. See [Alias Definitions](#) for more information.

### 7.9.1 Internal Defines

These are pre-defined, common constant definitions which are declared for all projects.

#### ⓘ IMPORTANT

To ensure consistency, you should not modify these declarations.

To see the set of declarations currently installed on your machine, you can view the file (named: *lib.eqv*) located under: [C:\Program Files\Kollmorgen\Kollmorgen Automation Suite\Astrolabe\Bin\HwDef](#) (the folder location differs if you chose another location when installing KAS).

Below is a an example of predefined constants that you may find in your system.

```
#define MLPN_CREATE_OBJECTS 1 (* Creation of blocks and pipes *)
#define MLPN_ACTIVATE 2 (*Activation of pipes*)
#define MLPN_CONNECT 3 (*Connections from convertors to axes*)
#define MLPN_POWER_ON 4 (*Power ON of axes*)
#define MLPN_POWER_OFF 5 (*Power OFF of axes*)
#define MLPN_DEACTIVATE 6 (*Deactivation of pipes*)

#define MLSTATUS_NOT_INITIALISED 0 (*Motion not initialised*)
#define MLSTATUS_RUNNING 1 (*Motion is running*)
#define MLSTATUS_STOPPED 2 (*Motion is stopped*)
#define MLSTATUS_ERROR 3 (*Motion is in error*)
#define MLSTATUS_INITIALISED 2 (*--DEPRECATED-- Motion is initialised*)

#define MLPR_CREATE_PROFILES 1 (* Creation and initialization of profiles
*)

#define MLFI_FIRST 0 (* ID of the first FastInput of an axis *)
#define MLFI_SECOND 1 (* ID of the second FastInput of an axis *)
```

```
#define MLFI_DISABLE 0 (* configures a FastInput as disabled *)
#define MLFI_RISING_EDGE 1 (* FastInput is sensible to rising edges *)
#define MLFI_FALLING_EDGE 2 (* FastInput is sensible to falling edges *)

#define PI 3.1415926535897932

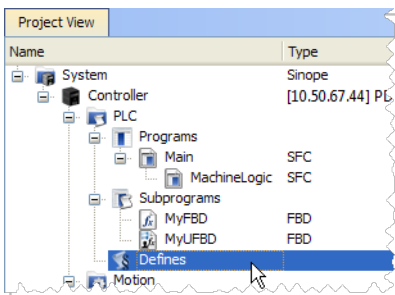
#define EC_POSITION_DEMAND_VALUE 10000
#define EC_VELOCITY_DEMAND_VALUE 10001
#define EC_TORQUE_DEMAND_VALUE 10002
#define EC_ADDITIVE_TORQUE_VALUE 10003
#define EC_MAX_TORQUE 10004
#define EC_OPERATION_MODE 10005
#define EC_CONTROL_WORD 10006
#define EC_LATCH_CONTROL_WORD 10007
#define EC_ANALOG_OUTPUT 10009
```

**NOTE**

The exact contents of the list depend on the version of the KAS IDE.

**7.9.2 Global Defines**

Global Defines are user-generated constants to be used in a program. Global Defines let you write code and add an `ifdef` statement to call the Define only if it is used for a particular machine. They are created and edited from the Project Explorer toolbox under **PLC**.



**Figure 4-99:** Global Defines

Double-click the **Defines** item to open your global definitions file (named: *appli.eqv*) in a text editor as follows:

```
#define DefReqTime T#10ms // 10 ms
#define BitMask 2#00100111 // binary
#define BitMaskHex 16#12AE // hexadecimal

#define OFF FALSE (* redefinition of FALSE constant *)
#define PI 3.14 (* numerical constant *)
#define ALARM (bLevel > 100) (* complex expression *)
```

**Figure 4-100:** Edit the Global Definitions

Each definition must be entered on one line of text according to the following syntax:

```
#define Identifier Equivalence (* comments *)
```

You may use a definition within the contents of another definition. The definition used in the second must be declared first. See example below:

```
#define PI      3.14
#define TWOPI  (PI * 2.0)
```

### 7.9.3 Local definitions

Local definitions are user-created defines that are being used within the corresponding program through an `ifdef` statement.

#### **TIP**

Using definitions disturbs the program monitoring and makes error reports more complex. It is recommended to restrict the use of definitions to simple expressions to avoid misunderstandings when reading or debugging a program.

## 7.10 Map Input and Output to Variables

This procedure describes how to map EtherCAT motion bus I/O or AKD PDMM/PCMM Onboard I/O to PLC variables.

### NOTE

This operation is disabled when the controller is running.

For more information see:

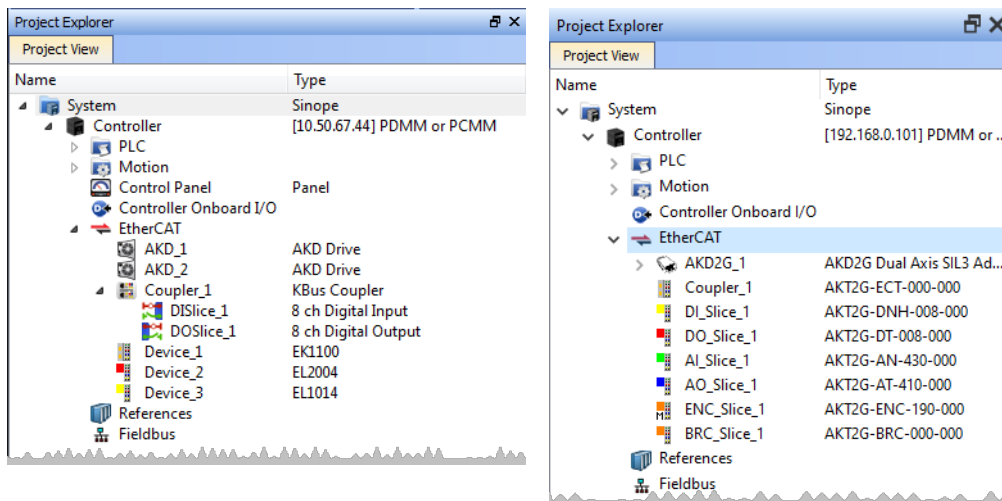
- [Map I/O from the Project Explorer](#)
- [PLC Variable Creation Wizard](#)
- [PLC Variable Selector](#)
- [Analog I/O Parameters](#)

### 7.10.1 Map I/O from the Project Explorer

#### TIP

For information on mapping I/O with third-party devices, please see [PDO Selection/Mapping Tab](#).

1. In the Project Explorer, expand the Controller node and the EtherCAT node(s) to access your devices.



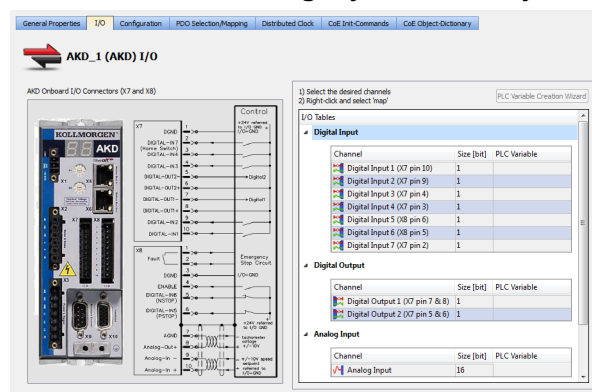
AKT (K-Bus) devices

AKT2G (E-Bus) devices

2. Depending upon the device, the properties are accessed in slightly different ways.

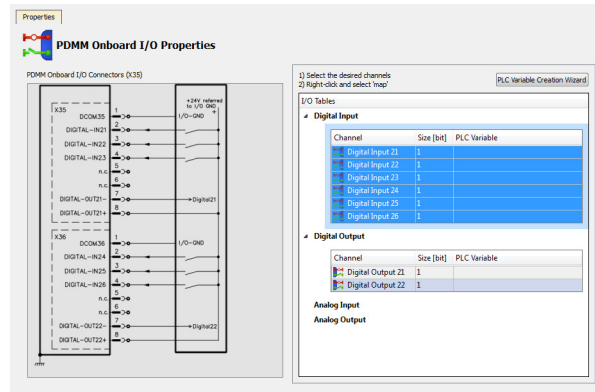
**AKD or AKD2G:**

Double-click on the drive and select the I/O tab



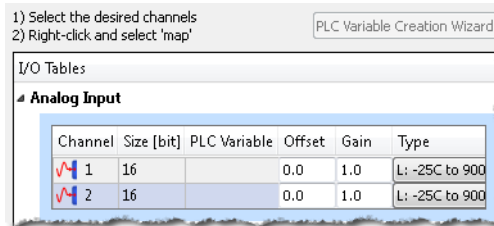
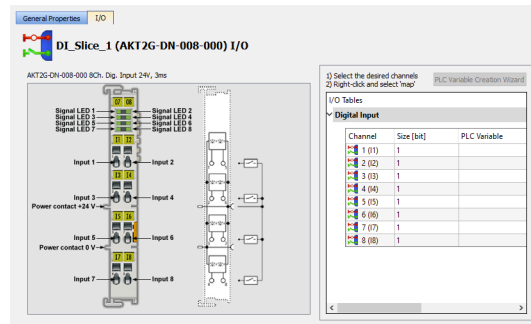


**AKD PDMM or PCMM**  
 Double-click Controller Onboard I/O



**I/O Slice:**  
 Double-click the I/O Slice under EtherCAT (E-Bus) or in the Coupler entry (K-Bus).

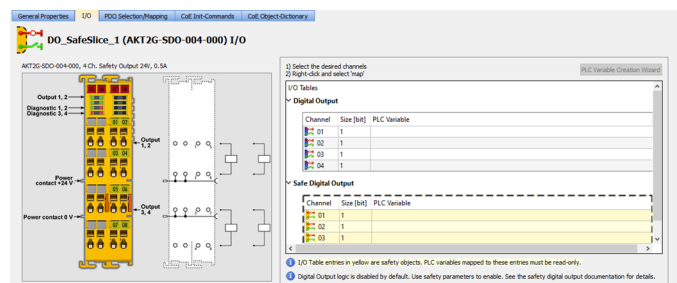
For Slice Analog I/O and thermocouples, you also have to define offset and gain parameters.



**Safety I/O Slice:**  
 Double-click the Safety I/O Slice under EtherCAT (E-Bus).

Safe Digital Inputs and Output channels are clearly identifiable by a yellow table in the I/O tab.

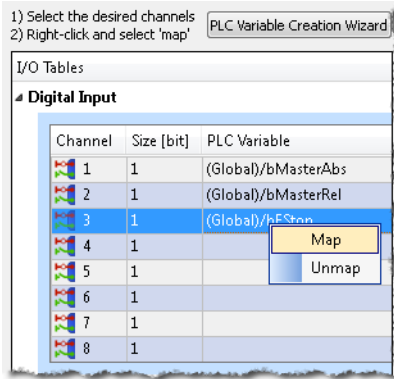
Digital output logic on the safety slices / devices is typically disabled by default from the standard PLC. See the safety digital output documentation and the [AKT2G-SDO-004-000Safety Parameters](#) table for details.



- There are several ways to map variables:
  - Drag and drop a variable from the Dictionary onto a table entry.
  - Use the [PLC Variable Creation Wizard](#) for Kollmorgen devices.
  - Directly map/unmap the Inputs/Outputs to PLC variables using the [PLC Variable Selector](#).

Select the channel(s) you want to map. Selection may be done by click-dragging or shift-clicking a range of entries. The entire table may be selected by clicking on its border. After the selection is made, right-click and select **Map** or press the **Enter** key.

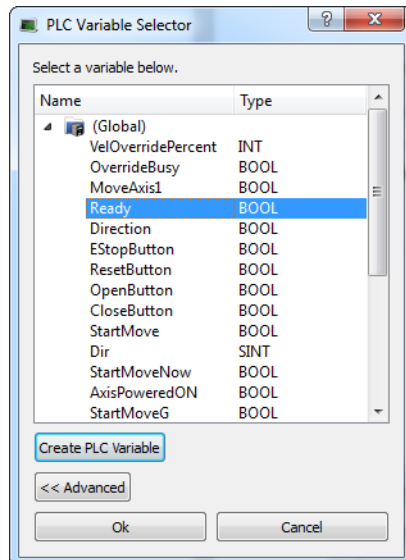
This opens the **PLC Variable Selector**.



### TIP

The **Unmap** command in the contextual menu allows you to remove the link between the variable and the associated channel(s). In addition, deleting a variable from the dictionary which is mapped to the channel(s) also removes the link(s).

Choose the variable to be linked to the channel(s) or PDO object.



### NOTE

The list of variables is filtered to display only those with relevant types.

- PLC Variable Selector displays Read Only variables when mapping the Input channels from the I/O tab.
- Variables assigned to the Safe Digital Input and Safe Digital Output channels must be Read Only.

### IMPORTANT

Please be aware of the following limitations of PLC variable mapping.

- Each PLC variable can be mapped to an EtherCAT I/O and exclusively to either:
  - a Controller Onboard I/O
  - an external driver
- Because a variable can only be mapped to one channel or PDO object, when you link a variable to a new channel or PDO object, the previous mapping is removed (even if linked to another slice or device).
- Individual bits within a variable can be mapped to multiple I/O channels on different devices (AKD, AKD PDMM, PCMM, or Slice).

For example, the same PLC variable cannot be mapped to both Profinet and a Controller Onboard I/O but it is possible with a regular EtherCAT I/O.

*What happens to variables that are mapped to a device I/O which was removed using the ECATDeviceAction function block?*

When an device is removed using [ECATDeviceAction](#), what happens to variables depends on what they are mapped to.

- **Variable mapped to an Analog or Digital Input Type:** The value of the variable becomes '0' when the mapped device is removed. The device I/O value will be automatically reconnected to the variable when the device is reconnected to the network.
- **Variable mapped to an Analog or Digital Output:** The value will not be sent to the device when the mapped device is removed. When the device is reconnected, the value will be sent to the I/O automatically.

#### See Also:

- For details on the **Create PLC Variable** and **Advanced** buttons, see [PLC Variable Selector](#).
- For more details on parameters, see [Analog I/O Parameters](#).
- For more information on the AKD Onboard EtherCAT I/Os, see [Configure Onboard I/O](#).
- For more information on the AKD PDMM or PCMM local digital I/Os, see [Configure Controller Onboard I/O](#).

### 7.10.2 PLC Variable Creation Wizard

This wizard allows you to automatically create a list of variables used for the mapping. The variable type is **Boolean** for digital I/Os and **UINT** for analog I/Os.

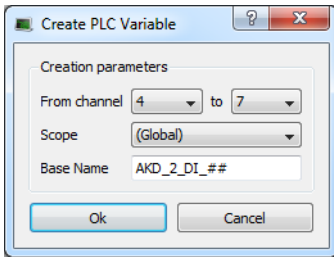


Figure 4-101: Wizard to Create PLC Variable - Parameters

Field	Description
<b>From channel</b>	Defines the range of channels you want to map automatically
<b>Scope</b>	Defines where the variables are created (if you select the <b>Global</b> scope, then the variables are created under the Global node in the Dictionary)
<b>Base Name</b>	Pattern used for variable naming with the name of the object in the Project Tree, followed by a 2-letter acronym (2LA) representing the I/O type. The channel number will replace <b>##</b> .

Channel	Variable Name
1	
2	
3	
4	DInput_4
5	DInput_5
6	DInput_6
7	DInput_7
8	

Figure 4-102: Wizard to Create PLC Variable - Mapped Channels

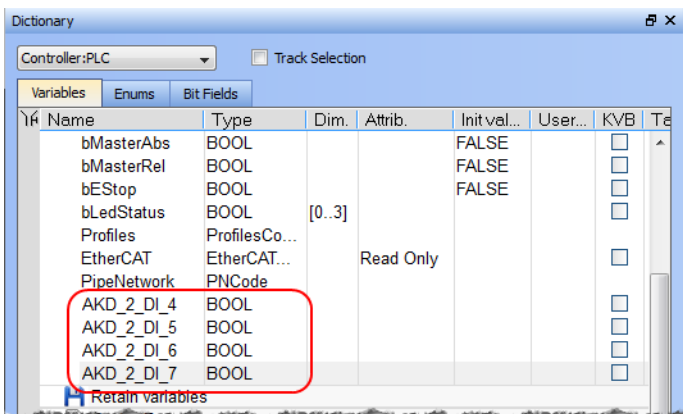
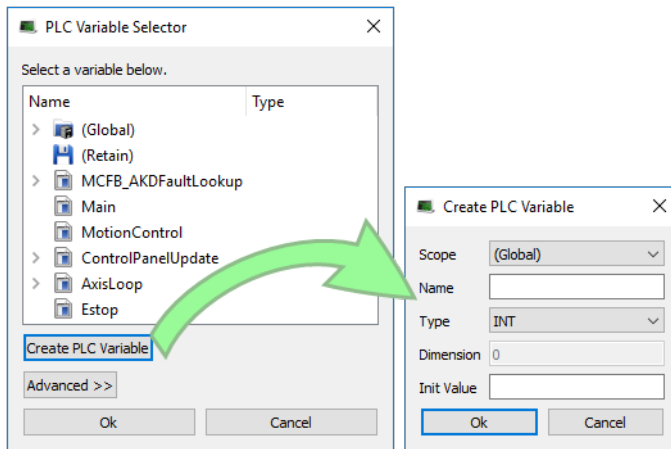


Figure 4-103: Wizard to Create PLC Variable - Variables in the Dictionary

## 7.10.3 PLC Variable Selector

### 7.10.3.1 Create PLC Variable Button

This button allows the creation of a new variable to be mapped to the selected item(s).



Field	Description
<b>Scope</b>	Defines where the variable is created
<b>Name</b>	See <a href="#">Name a variable</a>
<b>Type</b>	Defines the <a href="#">Type</a> of the variable, and its <a href="#">Dimension</a> if the variable is an <a href="#">array</a>
<b>Dimension</b>	The dimension of the variable (read-only). The dimension is set automatically based on the necessary size and the selected variable type.
<b>Init Value</b>	See <a href="#">Initial Value of a Variable</a>

Based on the variable **Type** we select, the KAS IDE will automatically set the **Dimension** (the size of the array) and assign the bits across the objects.

See the [online help for a video](#) that demonstrates mapping a variable to several PDO objects.

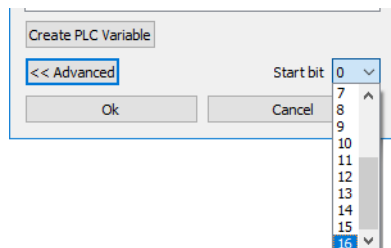
#### TIP

- Safety PDOs are highlighted in yellow with a border.
- Any variable assigned to a FSoE PDO must be Read Only. See [About Safety, PDOs, and Variables](#) for more information.
- See [How To Set Up a Variable to Monitor Safety Data](#) for step-by-step instructions.

See also [Create Variables](#)

### 7.10.3.2 Advanced Button

For integer variables with types stored on several bits, the Advanced button gives access to the **Start bit** definition. This allows you to link a set of channels to a specific range of bits within an integer variable.



For example, when you select three channels ranging from 1 to 3 and map them to a DINT variable (stored on 32 bits ranging from 0 to 31), the first channel can be linked to position ranging from 0 to 29. The three channels are mapped to the last three bits ranging from 29 to 31.

1) Select the desired channels  
2) Right-click and select 'map'

Channel	Variable Name
1	MachineState.29
2	MachineState.30
3	MachineState.31
4	DInput_4

**NOTE**  
When the selected variable is an array, the "Start bit" is disabled. An offset is not allowed when mapping to an array.

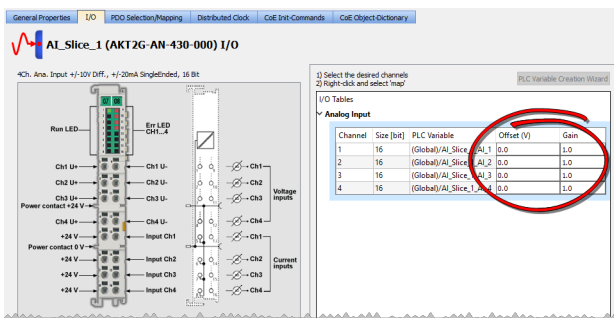
### 7.10.4 Analog I/O Parameters

#### 7.10.4.1 AKT2G Input Terminals

The offset and gain is applied to calculate the value converted from the analog input:

$$\text{PLC Variable Value} = \text{Analog to Digital Converter Value} \times \text{Gain} + \text{Offset}$$

Where the Gain and Offset are configured from the terminal's I/O tab.



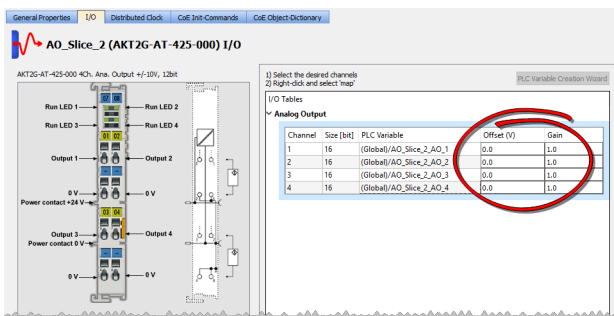
See also [E-Bus Analog I/O Data Representation](#)

#### 7.10.4.2 AKT2G Output Terminals

The offset and gain is applied to calculate the value to be converted to the analog output:

$$\text{Digital to Analog Converter Value} = \text{PLC Variable Value} \times \text{Gain} + \text{Offset}$$

Where the Gain and Offset are configured from the terminal's I/O tab.



See also [E-Bus Analog I/O Data Representation](#)

#### 7.10.4.3 AKT Input Terminals

The process data that are transferred to the Bus Coupler are calculated using the following equation:

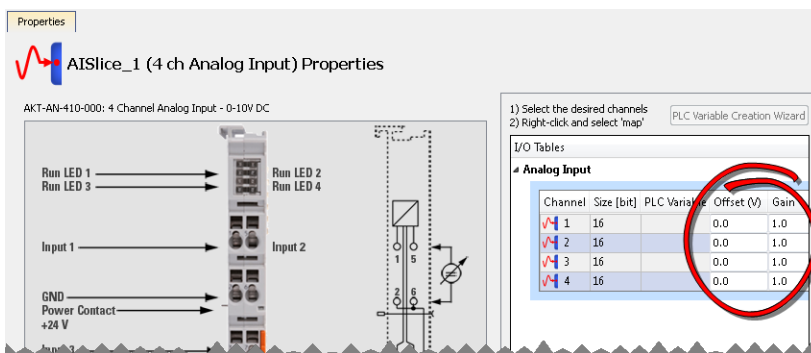
$$Y_a = ( B_a + X_{ADC} ) \times A_a$$

$$Y_{out} = B_w + ( ( A_w + A_h ) \times Y_a )$$

With the following parameters:

<b>X<sub>ADC</sub></b>	Output values of the Analog Input Modules A/D converter
<b>Y<sub>out</sub></b>	Process data to the controller
<b>B<sub>a</sub> , A<sub>a</sub></b>	Manufacturer offset and gain compensation ‡
<b>A<sub>h</sub></b>	Manufacturer scaling: default gain ‡
<b>B<sub>w</sub> , A<sub>w</sub></b>	User scaling: Offset and Gain as set in the <i>Analog Input Properties</i> (see image below).

‡ For the thermocouple input terminals, AKT-AN-200-000 and AKT-AN-400-000, the manufacturer default gain is 160. For all other supported terminals, the manufacturer default gain is 1. The manufacturer default offset is zero for all supported terminals.



#### 7.10.4.4 AKT Output Terminals

The process data that are transferred to the Bus Coupler from the controller are calculated using the following equations:

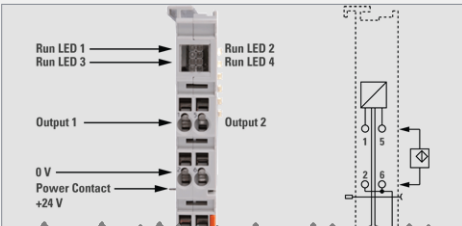
$$Y_2 = B_w + ( ( A_w \times A_h ) \times X )$$

$$Y_{dac} = Y_2 \times A_a + B_a$$

<b>X</b>	Controller Process data
<b>Y<sub>dac</sub></b>	Controller data to analog output module D/A converter
<b>B<sub>a</sub> , A<sub>a</sub></b>	Manufacturer offset and gain compensation ‡
<b>A<sub>h</sub></b>	Manufacturer scaling: default gain ‡
<b>B<sub>w</sub> , A<sub>w</sub></b>	User scaling: offset and gain as set in the <i>Analog Output Properties</i> (see image below).

‡ The manufacturer default offset is zero for all supported terminals. The manufacturer default gain is 1 for all supported terminals.

Properties

 AOSlice\_1 (4 ch Analog Output) Properties

AKT-AT-410-000: 4 Channel Analog Output - 0-10V DC

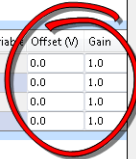
1) Select the desired channels  
2) Right-click and select 'map'

PLC Variable Creation Wizard

I/O Tables

4 Analog Output

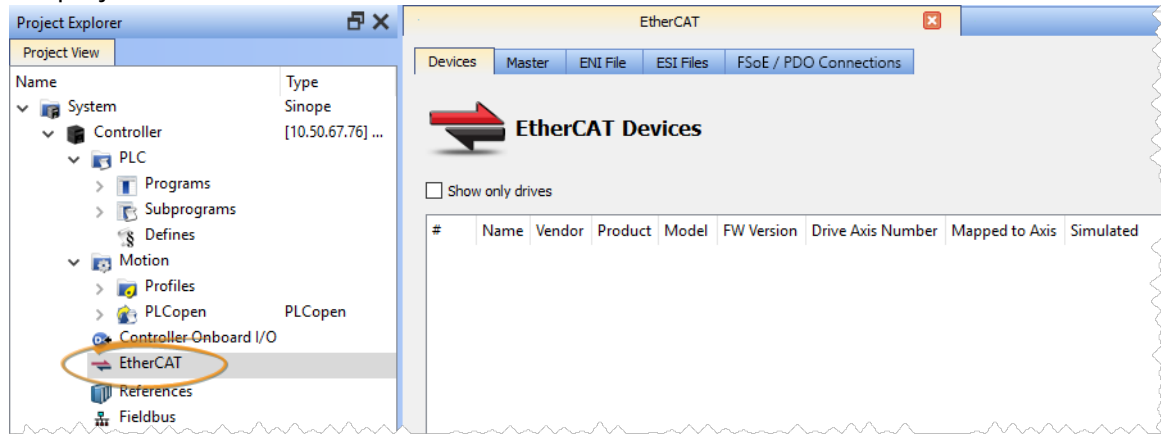
Channel	Size [bit]	PLC Variable	Offset (V)	Gain
1	16		0.0	1.0
2	16		0.0	1.0
3	16		0.0	1.0
4	16		0.0	1.0





## 7.11 Configuring EtherCAT

This section provides reference information for the EtherCAT entry in the Project Tree. Specifically this includes the [EtherCAT Devices tab](#), [EtherCAT Master Settings Tab](#), [ENI File Tab](#), [ESI Files Tab](#), and [FSoE / PDO Connections Tab](#) tabs. These are accessed by double-clicking on the EtherCAT item in the Project Tree or right-clicking and selecting **Properties**. Reference information for devices in the project can be found in [EtherCAT Devices](#).



### 7.11.1 EtherCAT Devices tab

The EtherCAT Devices tab lists all the EtherCAT devices and provides for the ability to discover and map their use. The interface is slightly different for AKD and AKD2G drives. The AKD2G prompts you to select the mapping by the drive axis.

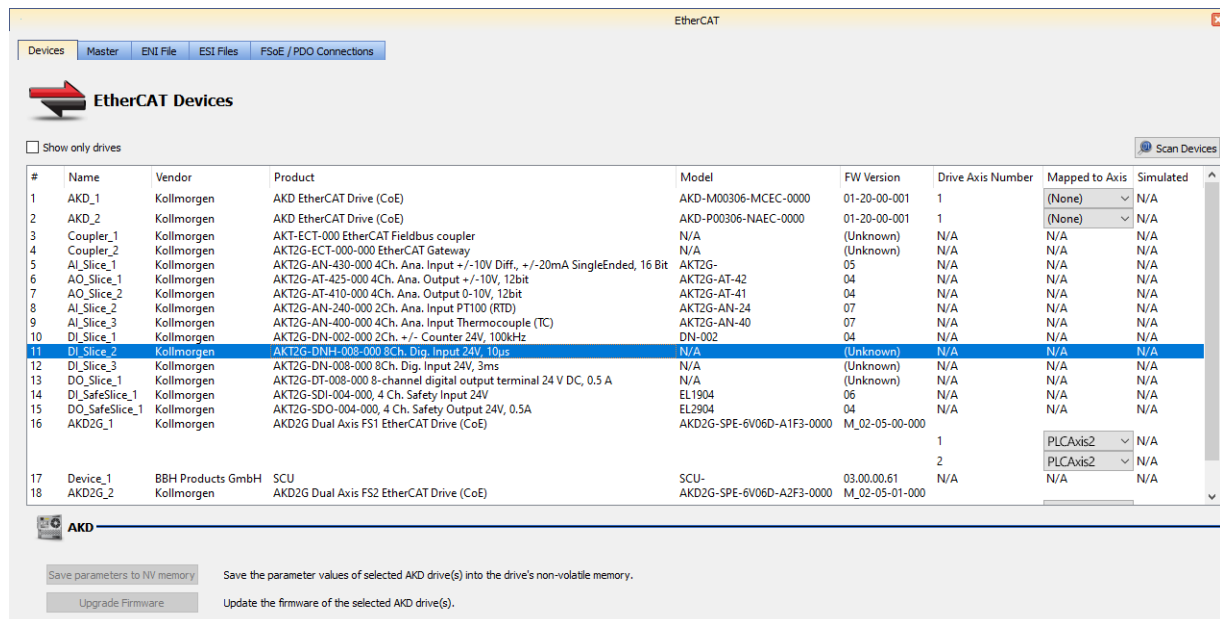


Figure 4-104: EtherCAT Devices Summary Form

First generation AKD drives support one axis per drive, while AKD2G drives support two axes per drive. When configuring drives in the [EtherCAT Devices tab](#), you will need to designate which motion axis applies to each "drive axis", which is the actual axis the drive controls.

Item	Description
Show only drives	This option hides from the list all EtherCAT slaves that are not drives

Item	Description						
<b>Scan Devices</b>	The KAS Runtime sends EtherCAT messages to discover the physical devices and modules present in the network (see explanation <a href="#">below</a> )						
<b>Name, Vendor, Product, and Model</b>	The project device name, vendor name, product name (or code number), and model number for each device is displayed and ordered by the position in the tree. The model (when available) includes the extension and connectivity options, <sup>NBEC</sup> for example.						
<b>FW Version</b>	<p>For Kollmorgen drives, the firmware version is displayed on the conditions that:</p> <ul style="list-style-type: none"> <li>• The "Scan Devices" routine was run successfully on the EtherCAT network (for drives with version 01-08-000-00 firmware or later)</li> <li>• Firmware was downloaded to the drive(s) while the KAS project was open in the KAS IDE.</li> </ul> <p>For other devices, the firmware version is displayed if the Scan Devices routine was successfully run and the device supports reading this information over EtherCAT (i.e. it uses CoE object 0x100A:0).</p> <p>Otherwise <b>(Unknown)</b> is displayed.</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p style="text-align: center;"><b>NOTE</b></p> <p>If a Kollmorgen drive is showing the firmware version as "Unknown", the drive has valid resident firmware (from which to boot), but does not have valid operational firmware. Please download and install the latest operational firmware and reboot the AKD/AKD2G.</p> </div>						
<b>Drive Axis Number</b>	This number designates which axis, of those available on a drive, is being defined. For most drives, this number will just be a "1", but for AKD2Gs, there are two entries, Drive Axis Number 1 and Drive Axis Number 2,						
<b>Mapped to Axis</b>	<p>For each drive, it is displayed if it is:</p> <ul style="list-style-type: none"> <li>• Unassigned: from the drop-down menu, you can choose an axis that has not been assigned (it is applicable either for PLCopen or PipeNetwork motion engines).</li> <li>• Already mapped to a physical device: the mapping operation is done using the <b>Scan Devices</b> command. See details in <a href="#">Scan Devices</a>.</li> <li>• If a drive node is removed from the EtherCAT network using <a href="#">ECATDeviceAction</a> then the mapped axis (if any) will automatically be forced to simulation.</li> </ul>						
<b>Simulated</b>	<p>Select this option when you want to simulate the device, which means that the device is not used and no communication to this device is performed through the fieldbus.</p> <table border="1" style="margin-top: 10px; width: 100%;"> <thead> <tr> <th>Mapped to an Axis?</th> <th>Simulated State</th> </tr> </thead> <tbody> <tr> <td>No</td> <td>Simulation is not applicable</td> </tr> <tr> <td>Yes</td> <td> <ul style="list-style-type: none"> <li>• If Drive is mapped to a physical drive, then the simulation is Enabled, so you can set state to Yes/No --&gt; Display checkbox</li> <li>• If Drive is <b>not</b> mapped to a physical drive, Simulation is forced to <b>Yes</b></li> </ul> </td> </tr> </tbody> </table>	Mapped to an Axis?	Simulated State	No	Simulation is not applicable	Yes	<ul style="list-style-type: none"> <li>• If Drive is mapped to a physical drive, then the simulation is Enabled, so you can set state to Yes/No --&gt; Display checkbox</li> <li>• If Drive is <b>not</b> mapped to a physical drive, Simulation is forced to <b>Yes</b></li> </ul>
Mapped to an Axis?	Simulated State						
No	Simulation is not applicable						
Yes	<ul style="list-style-type: none"> <li>• If Drive is mapped to a physical drive, then the simulation is Enabled, so you can set state to Yes/No --&gt; Display checkbox</li> <li>• If Drive is <b>not</b> mapped to a physical drive, Simulation is forced to <b>Yes</b></li> </ul>						

Item	Description
<b>Save parameters to NV memory</b>	<p>Allows you to save the drives' parameters to the <b>NVRAM</b> of each drive currently selected in the list. This action is enabled only when the Online Mode is activated</p> <p>You will be automatically be prompted with the option to save modified drive parameters if this action has not been performed prior to the following circumstances.</p> <ul style="list-style-type: none"> <li>• Exiting <b>Online Configuration Mode</b></li> <li>• Disconnecting from the controller</li> <li>• Closing the project</li> <li>• Exiting the IDE</li> </ul>
<b>Upgrade Firmware</b>	<p>This command triggers a firmware upgrade for the selected drives (you can use <b>Ctrl+A</b> shortcut to select all drives).</p> <p>For more details, refer to <b>FAQ</b> section.</p>

Table 4-23: EtherCAT Devices

### 7.11.1.1 Scan Devices

The scan process allows the following tasks:

- Discover the devices and modules physically present in the fieldbus network (see [EtherCAT Network - Physical View](#))
  - Map them to items under the EtherCAT node of the Project Explorer (see [EtherCAT Network - Logical View. K-Bus on left, E-Bus on right](#))
- Note that the order of the devices in the tree is the same as in the real fieldbus network.

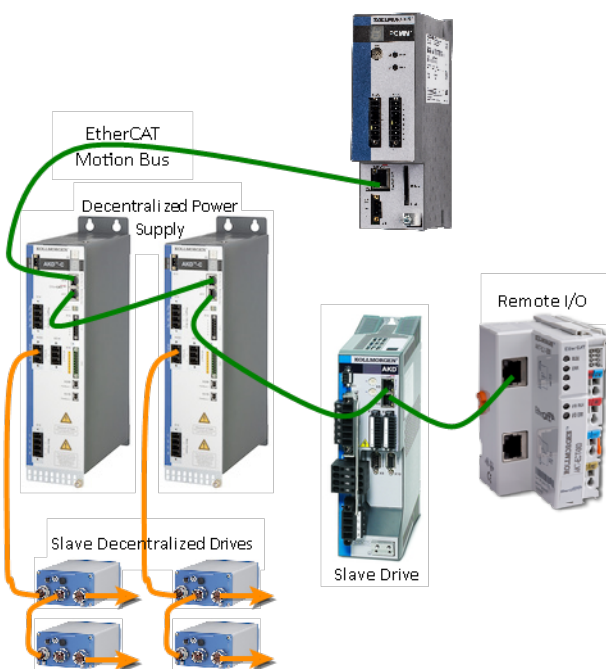


Figure 4-105: EtherCAT Network - Physical View



**Figure 4-106:** EtherCAT Network - Logical View. K-Bus on left, E-Bus on right

During the scan operation, all physical devices connected to the EtherCAT network will be discovered and identified. The KAS IDE will list the devices in the order they are connected. Kollmorgen devices, which include AKD and S300/S700 drives, Standard I/O Couplers, and remote I/O terminals (for a list, see ["Remote Input/Output Terminals" on page 749](#)) will include detailed information for each device.

The status of devices can be determined by their icon in the logical view.

Icon	Meaning	Description
(device icon)	Normal	The device has been added by scanning the system. The associated ESI file has been found. The icon varies by device, but is the icon set by the manufacturer in the ESI file.
	Excluded	The device has been added manually or has been disassociated from a discovered device.
	ESI missing	The ESI file is missing.
	Error	For AKD, shown when there is an error.

Table 4-24: EtherCAT device icon descriptions.

### 7.11.1.1.1 Scan Limitations

- I/O slices for Standard I/O Coupler do not reveal their Device IDs.
- The discovery feature does not differentiate between AKT-DN-004-000 and AKT-DNH-004-000 I/O terminals. Nor between AKT-DN-008-000 and AKT-DNH-008-000.
- Devices other than those made by Kollmorgen will be identified by the Vendor Name (or ID number) and Product Name (or ID number). If the device is missing an ESI file, then you will need to import the ESI file supplied by the device Vendor. The ESI file is required by the IDE to decode and display the Product Name, Device Description, and other details.
- ESI files for any MDP devices connected to the network should be added to KAS IDE's ESI library before starting a scan. If the ESI file for an MDP device is not available, then the scan cannot identify the device and cannot discover any physical or logical modules under the device.

### 7.11.2 EtherCAT Master Settings Tab

This tab includes configurations for the EtherCAT bus master.

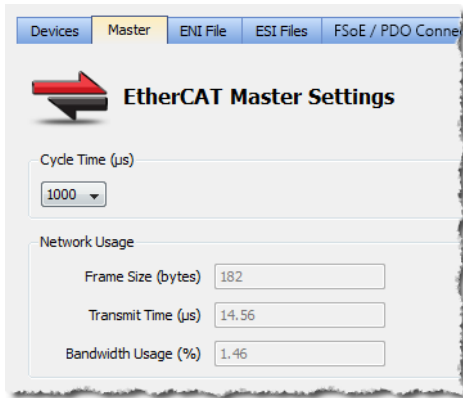


Figure 4-107: EtherCAT Master Settings

Item	Description
Cycle Time (µs)	Duration of one cycle in microseconds to define the time base period for scheduling the motion and the PLC programs (for more details on scheduling, see <a href="#">"Tasking Model / Scheduling" on page 463</a> ). Available times include: <ul style="list-style-type: none"> <li>• 250</li> <li>• 500</li> <li>• 1000</li> <li>• 2000</li> <li>• 4000</li> </ul>
Frame Size	It is the total size (in bytes) of the EtherCAT frame which is sent cyclically. The size is proportional to the number of EtherCAT slaves (and consequently the <b>PDO</b> data) on the network. The EtherCAT frame size may be between 84 to 1500 bytes.
Transmit Time	It is the time (in microseconds) that it takes to send a frame
Bandwidth Usage	It is an estimation of the percentage of the cycle time used to transmit a frame of data. Bandwidth value goes up when cycle time decreases (see calculation below)
Working Counter Error Limit	A configurable threshold for generating an E30 error and shutdown of EtherCAT communication. The default setting is 3 Errors in 1000 Frames. Other options are: <b>Trigger on 1st Error, 2 Errors in 1000 Frames, 10 Errors in 1000 Frames, 100 Errors in 1000 Frames, or Disabled</b> . If the <b>Disabled</b> option is selected the Master will keep the EtherCAT network running in op mode. It is up to the application to decide if it is appropriate to shut down the Master's EtherCAT communication ( <a href="#">MLMotionStop</a> ). <ul style="list-style-type: none"> <li>• The slave device state with regards to EtherCAT communication errors is determined by the manufacturer.</li> <li>• AKD drives transition to Init state when 7 consecutive packet failures are encountered, and the drive executes a controlled stop.</li> </ul> See <a href="#">EtherCAT Error Messages</a> for additional working counter information.

Table 4-25: EtherCAT Cycle Settings - Form Description

The three read-only fields display (**unknown**) when the [Use imported file](#) option is selected (see [ENI File Tab](#) ). Otherwise, they are recalculated and refreshed each time that:

- A device is added or removed
- A device [simulation state](#) changes
- The [Use imported file](#) check box is cleared

### Bandwidth calculation algorithm

The Bandwidth (BW) usage calculation takes into account the calculated frame size and the Ethernet speed (100 Megabits per second).

$$BW\% = \text{Transmission time} / \text{Cycle Time}$$

With Transmission time ( $\mu\text{sec}$ ) = (Frame Size in bytes \* 8) bits / 100 \* 106 bps

For example:

If Frame Size = 100 bytes  
**then** Transmission Time =  $100 \cdot 8 / (100 \cdot 106) = 8 \mu\text{sec}$

If cycle time = 1000  $\mu\text{sec}$   
**then** BW% =  $8 / 1000 = 0.8 \%$

### 7.11.3 ENI File Tab

During the [compilation](#), the KAS IDE generates the ENI (EtherCAT Network Information) file based on the EtherCAT devices defined in your project.

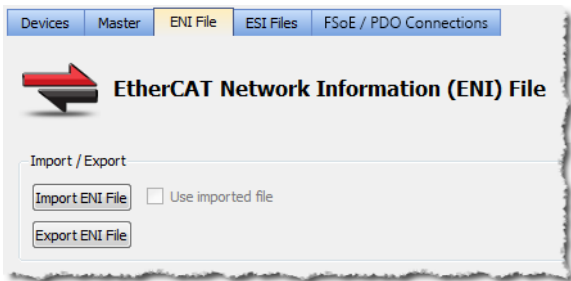


Figure 4-108: ENI File tab

Item	Description
<b>Import ENI File</b>	Enables you to browse and select an ENI file to be imported. If the file is successfully imported into the project, the <b>Use imported file</b> option is automatically selected. Once imported, the ENI file is added to your project.
<b>Use imported ENI file</b>	This enables you to include EtherCAT devices in your project that are not natively supported by KAS. For more details, refer to <a href="#">Add Unsupported EtherCAT Device</a> . Allows you to specify whether or not to use the imported ENI file. See also the paragraph below.
<b>Export ENI File</b>	Enables you to export the ENI file generated by the KAS IDE. You can specify the name and directory for the file. Only the logical devices in the project tree that are mapped to a physical device (and <b>not</b> simulated) are taken into account when generating the ENI file. This export can be useful if you want to use the file in another context or with another program.

Table 4-26: ENI File - Form Description

**⚠ IMPORTANT**

Importing an external ENI file overrides all EtherCAT project device information and configuration settings in the IDE. The following views and configurations are *not* applicable when using an imported ENI file:

- Project View: All devices located under the EtherCAT node
- EtherCAT Device View tabs:
  - General Properties
  - PDO Selection/Mapping
  - Distributed Clock
  - CoE Init-Commands
- Slice I/O Properties
- Mapping PLC Variables to Slice I/O or PDO objects

Information displayed in the views may not match the imported ENI file.

### Using an Imported ENI file

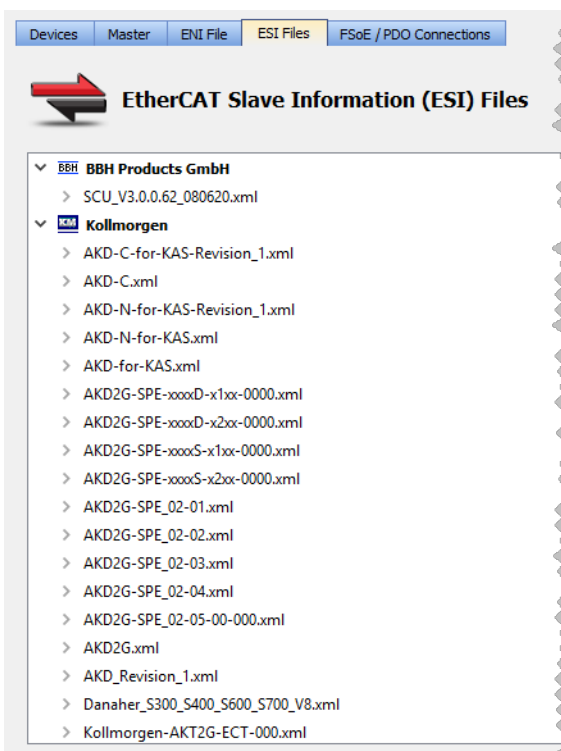
- The KAS IDE works in a degraded mode when using an imported ENI file, and the [Mapped to Axis](#) settings are disabled. This is because the information about the devices in the project tree and the EtherCAT widget table is no longer relevant.
- When using an imported configuration file the following parameters must be manually set for each axis:
  - the type of motion bus
  - its address on the fieldbus ring

This is done by right-clicking on the Axis Pipe Block and selecting the **Properties** command.

- Scan Devices must be run from [EtherCAT Devices](#) before downloading the application to the controller.

### 7.11.4 ESI Files Tab

This tab lists the available ESI (EtherCAT Slave Information) files and provides the ability to add and remove files.



**Figure 4-109:** The ESI Files tab

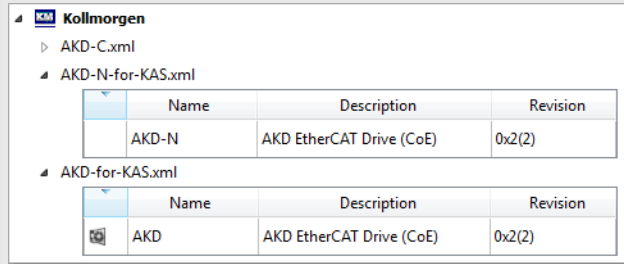
ESI files are sorted by vendor and then the files provided by the vendor. Each file lists the device it supports including

- Device name
- Device description
- Device revision number

ESI files are easily added to the list by clicking the *Add* button and navigating to where the file is saved. The *Delete* button will remove a file from the list but not your hard drive.

### TIP

KAS IDE manages Kollmorgen devices (AKD2G, AKD, AKD-N, etc.) and will automatically install and/or upgrade their ESI files to the latest version. The KAS IDE allows Kollmorgen ESI files to be upgraded, but downgrading is not possible. For example, the Kollmorgen ESI files: "AKD-for-KAS.xml" or "AKD-N-for-KAS.xml" should not be manually imported.



**Figure 4-110:** Do not overwrite these files.

Under special circumstances, Kollmorgen ESI files may be manually upgraded. We recommend working with a Kollmorgen representative when doing so.

For more information on working with ESI files, see the [ESI File Management](#) topic.

### IMPORTANT

**WARNING:** Replacing an ESI file changes the configuration of any associated device to the new file. You will be prompted to confirm or cancel replacing an ESI file and the associated devices' configurations.

ESI files will be replaced when:

- Opening a project file that contains an ESI file that is different from the file in the KAS Internal Library of ESI Files stored on the PC.
  - You are prompted to select "**term\_A**" (use the version in the project) or "**term\_B**" (use the ESI Internal Library version). Selecting "**term\_A**" will overwrite the ESI Internal Library version.
- Pressing the **Add** button in the ESI Files tab view and:
  - importing an ESI file name that already exists in the KAS Internal Library of ESI Files stored on the PC.
  - importing an ESI file that contains duplicate device information that already exists in the KAS Internal Library of ESI Files stored on the PC.

### IMPORTANT

**It is strongly recommended that you do not add or remove ESI files while multiple instances of the KAS IDE are open.** Doing so modifies a global cache of ESI files. Changes to the global ESI file cache from one instance of the KAS IDE could create unexpected consequences in another instance.

## 7.11.5 FSoE / PDO Connections Tab

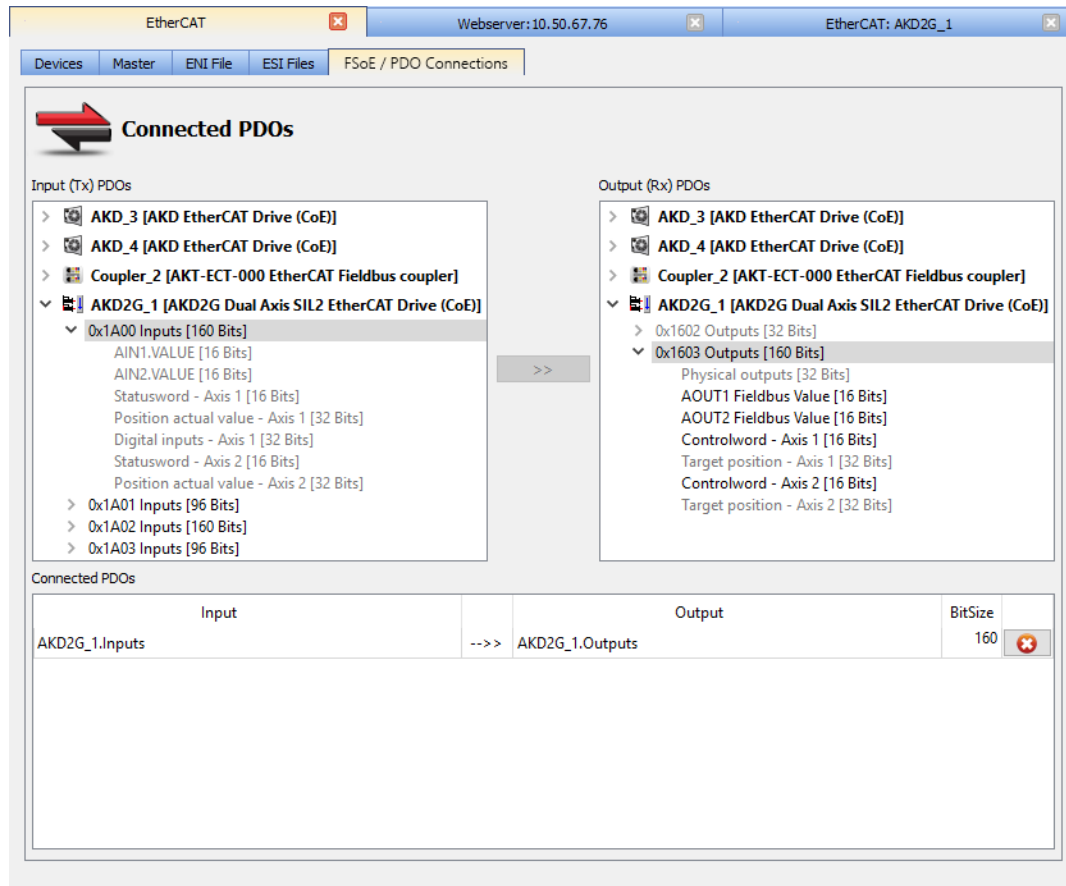
The FSoE (FailSafe over EtherCAT) tab displays devices discovered on the EtherCAT network, listing their Input (Tx) and Output (Rx) PDOs. This provides you with a way to connect an Input PDO to an Output PDO.



The KAS Runtime will copy the PDO data between the connected PDOs when EtherCAT is in operational mode. This feature makes it possible to transfer safety-critical process data via the EtherCAT frames. The FSoE protocol, communication, and state machine are handled by the EtherCAT Safety devices.

### TIP

Connected PDOs are intended for FSoE operation. This feature is not limited to FSoE, it could be useful for any application that need to copy PDO data between EtherCAT devices.



#### How to Map FSoE PDOs

These instructions assume you have already scanned the EtherCAT network.

1. Double-click (or right-click) on EtherCAT in the Project Explorer, and select the FSoE tab.
2. Select the desired Input PDO.

Output PDOs which may be mapped (or paired) to the input are shown in black. Outputs of a different size are grayed out.

3. Select the desired Output PDO.
4. Click on the Connect PDOs button.

Connected PDOs are listed at the bottom of the tab.

### TIP

An example of mapping PDOs may be found in the topic [Set Up FSoE Master and an AKD2G with SafeMotion Monitor](#).

#### How to Unmap FSoE PDOs

1. Double-click (or right-click) on EtherCAT in the Project Explorer, and select the FSoE tab.
2. Select the desired pair of Connected PDOs from the list at the bottom of the tab.
3. Click the Delete Connection button.

*What happens to the connected PDO(s) when a device is removed using the `ECATDeviceAction` function block?*

The PDO's connection is removed internally when the device is removed using the `ECATDeviceAction` function block and has its PDO connected to another device. The PDOs will automatically be reconnected when the device is connected back to the EtherCAT network.

#### NOTE

Inputs and Outputs must have identical Bit sizes in order to be paired.

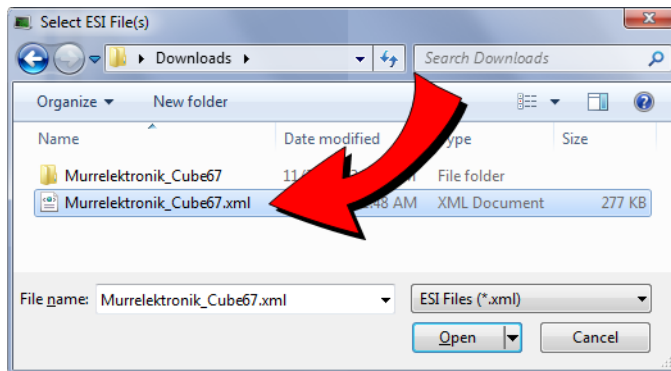
**See Also:** [Safety over EtherCAT](#)

### 7.11.6 ESI Files with References (MDP)

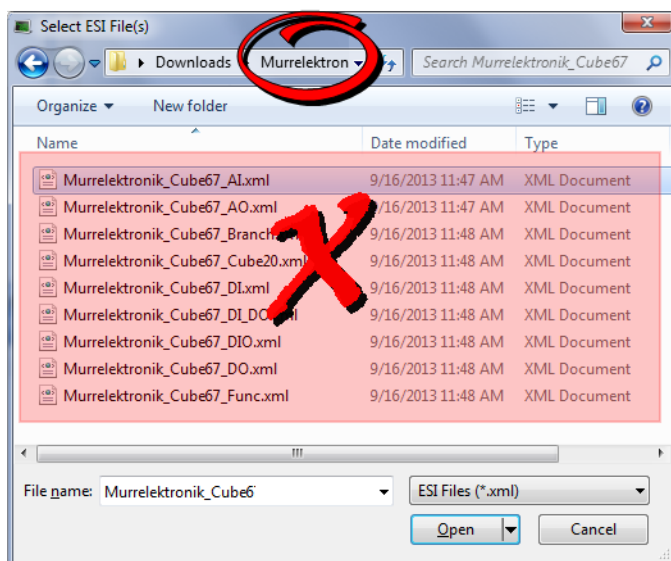
ESI files may contain references to other ESI files or to EDS files. The referenced files may be located in the same directory or sub-directories. ESI files with references are common with MDP (Modular Device Profile) devices. This is because the ESI files for the modules are shared between several device Products and/or Revision Numbers.

When adding an ESI file with references to the IDE's ESI library, *only* add the top-level file that contains the references to the other files and sub-directories. The IDE will check the selected ESI file for reference files and automatically add the reference files.

For example, when adding the Murrelektronik Cube67 MDP (modular) device ESI file, select the top-level ESI file.



Do not select the lower-level module ESI files found in a sub-folder.



#### ! IMPORTANT

**WARNING:** Always add the top-level ESI file to the IDE's ESI library. Do not add, delete, or upgrade lower-level module files to the IDE's ESI library. Removing or replacing lower-level files may cause unexpected or unknown behavior. If any reference files are missing when the top-level file is added to the IDE's ESI library, an error message will notify you which sub-files are missing.

#### 7.11.6.1 EtherCAT Device Configuration - MDP Devices

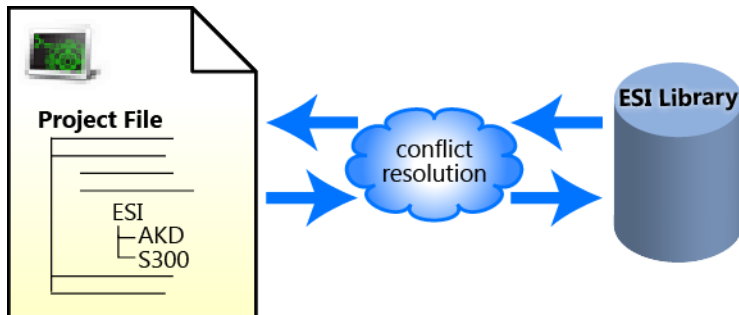
The KAS IDE incorporates the ability to configure and manage EtherCAT MDP (modular device profile) devices. Devices which meet the ETG.5001 standard may be added to the EtherCAT network

in the Project Explorer and modules may be mapped to them. This includes devices such as I/O device modules, safety device modules, or fieldbus gateway modules.

### 7.11.7 ESI File Management

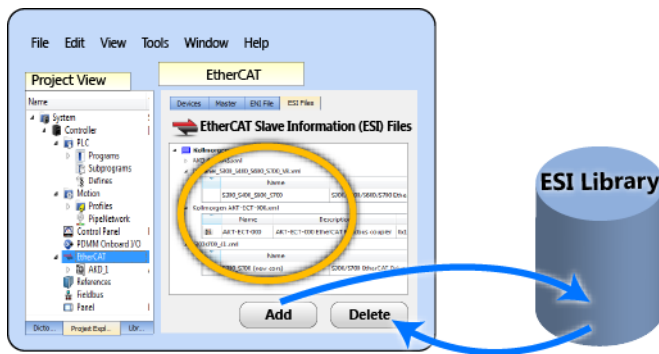
ESI files which are referenced from a project's EtherCAT devices are stored in the project.

- Upon opening a project the ESI files are copied to the local folder.



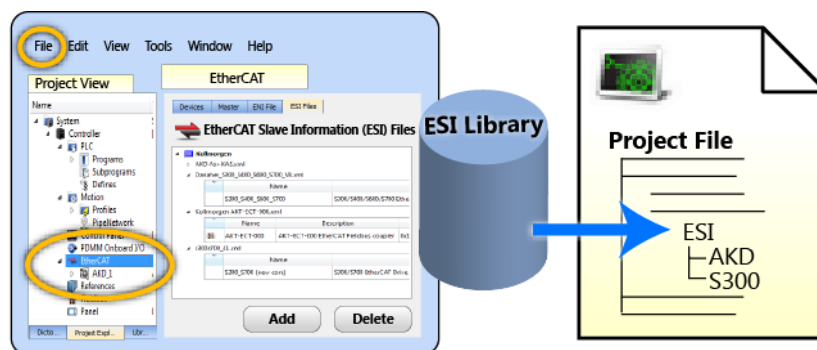
**Figure 4-111: Opening** — Upon opening a KAS project, the project's ESI files are compared to the internal library. If there are conflicts, you are prompted to resolve them.

- Adding or deleting ESI files from KAS IDE affects the internal library



**Figure 4-112: Adding/Deleting** — Adding or deleting an ESI file from the KAS IDE affects KAS's internal library of ESI files.

- Upon saving a project the ESI files are copied to (project folder)\Controller\ESI.



**Figure 4-113: Saving** — When a KAS project is saved, a copy of the ESI file(s) is included in the project file.

**TIP**

The ESI files are stored along with the project so the project can easily be moved to another location and it will still compile. It still compiles because the ESI files are copied back to the local folder.

**NOTE**

There is the possibility that the names and/or contents of files can be conflicting. The system checks the files before adding them. You will be prompted to chose the correct file should any conflicts be found.

**! IMPORTANT**

**WARNING:** Replacing an ESI file changes the configuration of any associated device to the new file. You will be prompted to confirm or cancel replacing an ESI file and the associated devices' configurations.

ESI files will be replaced when:

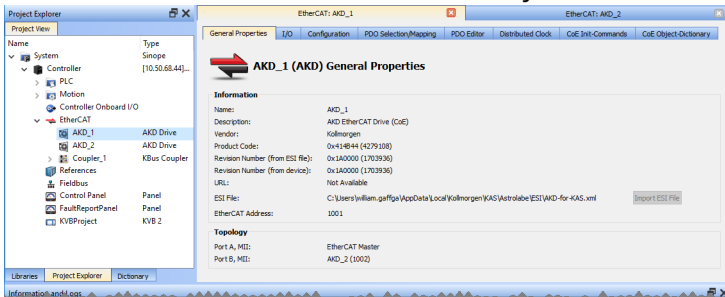
- Opening a project file that contains an ESI file that is different from the file in the KAS Internal Library of ESI Files stored on the PC.
  - You are prompted to select "**term\_A**" (use the version in the project) or "**term\_B**" (use the ESI Internal Library version). Selecting "**term\_A**" will overwrite the ESI Internal Library version.
- Pressing the **Add** button in the ESI Files tab view and:
  - importing an ESI file name that already exists in the KAS Internal Library of ESI Files stored on the PC.
  - importing an ESI file that contains duplicate device information that already exists in the KAS Internal Library of ESI Files stored on the PC.

**! IMPORTANT**

**It is strongly recommended that you do not add or remove ESI files while multiple instances of the KAS IDE are open.** Doing so modifies a global cache of ESI files. Changes to the global ESI file cache from one instance of the KAS IDE could create unexpected consequences in another instance.

## 7.12 EtherCAT Devices

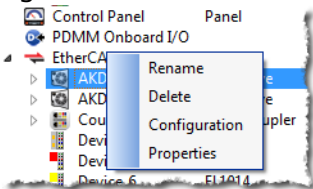
This section provides reference for defining EtherCAT devices. Devices are added by right-clicking the **EtherCAT** entry in the Project Explorer. Devices are seen as branches of the EtherCAT item. Double-clicking a device opens a tab labeled as "**EtherCAT: [device name]**". See [Configuring EtherCAT](#) for information on the tree entry.



### 7.12.1 EtherCAT Device General Properties Tab

This tab provides information about the selected EtherCAT device and the associated ESI file. It also provides the ability to import an ESI file if one is not present. There are two ways to access this tab:

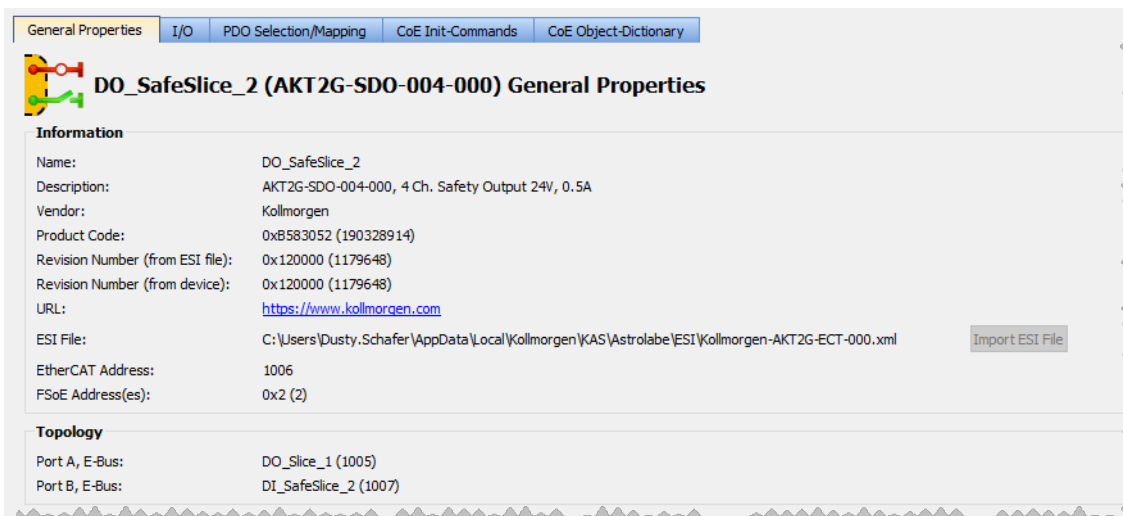
- double-click on an EtherCAT device
- right-click on an EtherCAT device and select Properties.



There are two main sections in the dialog: Information, and Topology.

#### 7.12.1.1 Information

This section details the device's basic information and is not editable. In addition to the device and vendor information is the path to the ESI file associated with the device. The **Import ESI File** button is available if the ESI file is missing. This lets you select a file to associate with the device. Lastly, this section lists the **EtherCAT Address** for devices and **FSOE Address** for safety devices which have been discovered.




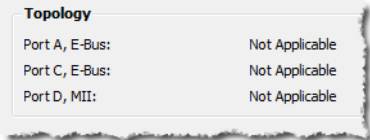
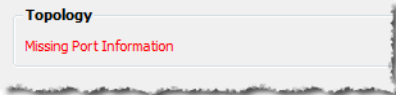
*Why are the Revision Numbers different?*

It is possible that there is a mismatch in the revision of firmware on the device and the revision referenced by the ESI file when the device was scanned. If this occurs, your device may not have access to the full capabilities of the latest firmware.

Revision Number (from ESI file):	0x3010000 (50397184)
Revision Number (from device):	0x3010001 (50397185)

### 7.12.1.2 Topology

This section lists the device's ports and assignments. There are three possible states for this information.

Condition	Appearance
Physical Devices	
Virtual Device	
ESI file is missing	

### 7.12.2 Configuration Tab

Parameters for AKD2Gs, AKDs, and AKT2G stepper modules can be set by double-clicking on the device in the project tree, or by right-clicking and selecting **Configuration**. This opens a Configuration tab which allows you to set the parameters for the device.

See the following for more information.

- [Configure the AKD2G/AKD Drive](#)
- [AKT2G-SM-Lxx Stepper Drive Configuration Tab](#)

### 7.12.3 PDO Selection/Mapping Tab

This tab includes the PDO configurations for an EtherCAT device. Assigned PDOs and their objects are viewable for the Inputs (Tx) and Outputs (Rx). The PDOs become active when the EtherCAT network is initialized to operation mode.

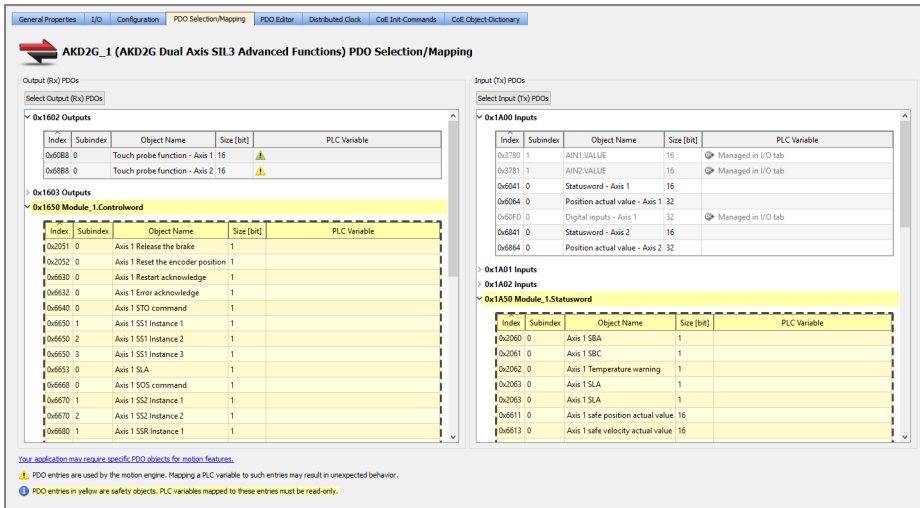


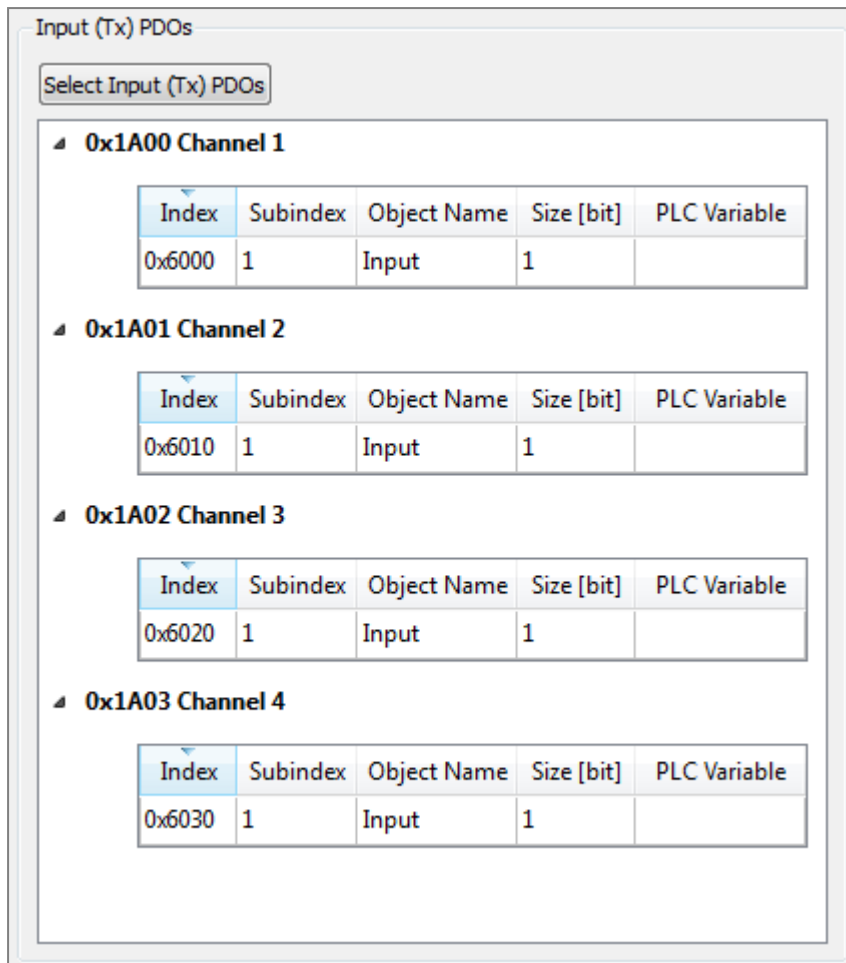
Figure 4-114: The PDO Selection/Mapping tab including FSoE objects in yellow.

**TIP**

- Safety PDOs are highlighted in yellow with a border.
- Any variable assigned to a FSoE PDO must be Read Only. See [About Safety, PDOs, and Variables](#) for more information.
- See [How To Set Up a Variable to Monitor Safety Data](#) for step-by-step instructions.

Each assigned PDO is listed by its Index (hex) and Name. The objects associated with each PDO are listed below the name. The objects are identified by their object dictionary **Index** and **Subindex**. The **Object Name** provides a simple description. The **Size** determines the data length. The **PLC Variable** mapped to the PDO objects is also listed.

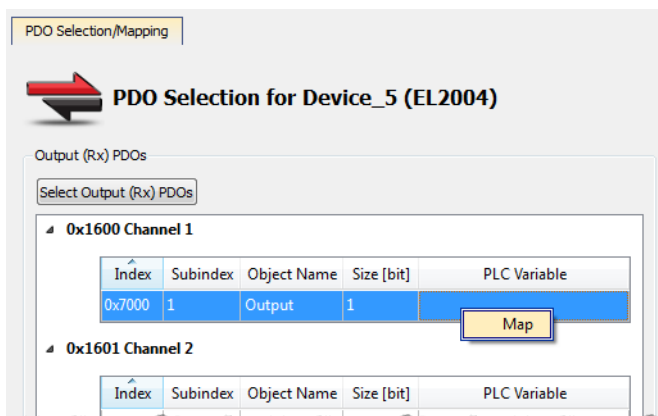




### 7.12.3.1 Map PLC Variable to PDO Object

PLC variables can be mapped to PDO objects by:

- Double-clicking in a PLC variable cell to open the PLC Variable Selector.
- Right-clicking in a PLC variable cell and select **Map** or **Unmap**.



- Drag-and-drop a variable from the Dictionary to a PLC variable cell.
- Multiple objects may be selected by:
  - Shift-clicking a contiguous range of objects.
  - Ctrl-clicking non-contiguous objects.

- Do not map PLC variables to third-party drive PDO objects. They may conflict with the KAS Runtime's internal motion engine read/write operations.

Special information regarding mapping PLC variables to safety objects

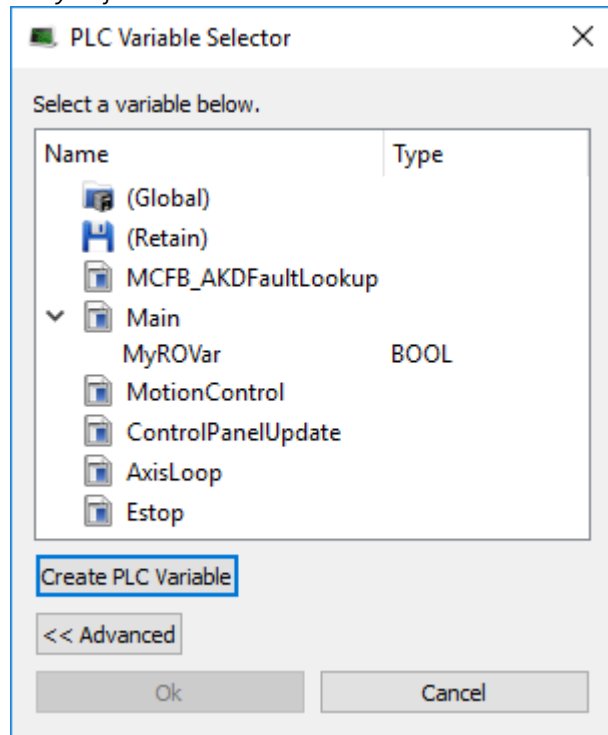
PLC variables may be mapped to safety objects in exactly the same way as any other object, but the variable must be set as Read Only.

- The PLC Variable Selector will display only Read Only variables.
- Variables created by clicking the Create PLC Variable button, when accessed from a safety PDO, will automatically be set to Read Only.
- There will be no warning if an assigned variable is changed to Read/Write, but an error will be reported when the project is compiled.

What happens to variables that are mapped to a device PDO which was removed using the ECATDeviceAction function block?

When an device is removed using ECATDeviceAction, what happens to variables depends on what they are mapped to.

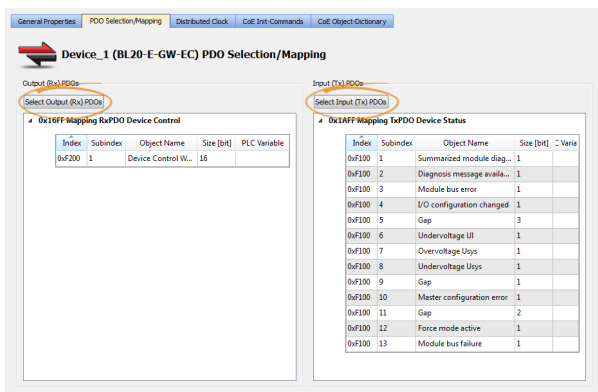
- **Variable mapped to an Input (Tx) PDO object:** The value of the variable becomes '0' when the mapped device is removed. The device PDO object will be automatically reconnected to the variable when the device is reconnected to the network.
- **Variable mapped to an Output (Rx) object:** The link to the device PDO is removed when the mapped device is removed. When the device is reconnected, the PDO object will be automatically reconnected to the variable.



For more details, please see [Map Input and Output to Variables](#).

### 7.12.3.2 Select Input and Output PDOs

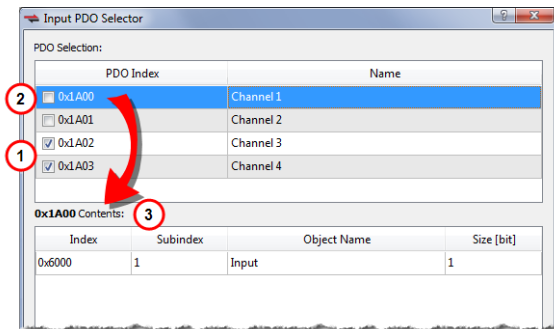
Press the **Select Input (Tx) PDOs** or **Select Output (Rx) PDOs** button to choose the Input or Output PDOs. The appropriate dialog box will open.



- The upper portion contains checkboxes to select specific PDOs for the EtherCAT network cyclic data.
- The lower portion contains the list of object(s) included within a specific PDO.

### 7.12.3.3 Viewing the contents of a PDO

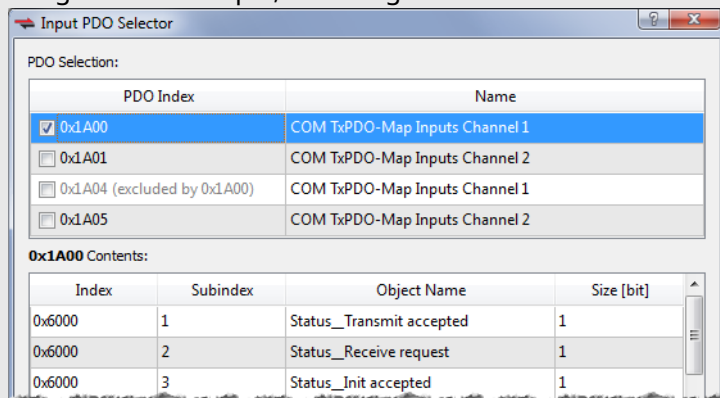
Click on a row in the upper portion of the PDO Selector dialog box to view the contents of that PDO. The lower portion will update to list the associated objects. Using the following example, the PDOs 0x1A02 and 0x1A03 are selected for the input objects but the contents for PDO 0x1A00 are listed because that row is selected..



1. Selected PDOs
2. Selected row
3. Contents of selected row's PDO.

## NOTE

- Some EtherCAT devices may not have selectable input and/or output PDOs.
- Some PDOs allow you to select more than one PDO at a time while others are exclusive.
- Exclusive PDOs prevent simultaneously selecting certain other PDOs. Using the following image as an example, choosing PDO 0x1A00 excludes selecting PDO 0x1A04.



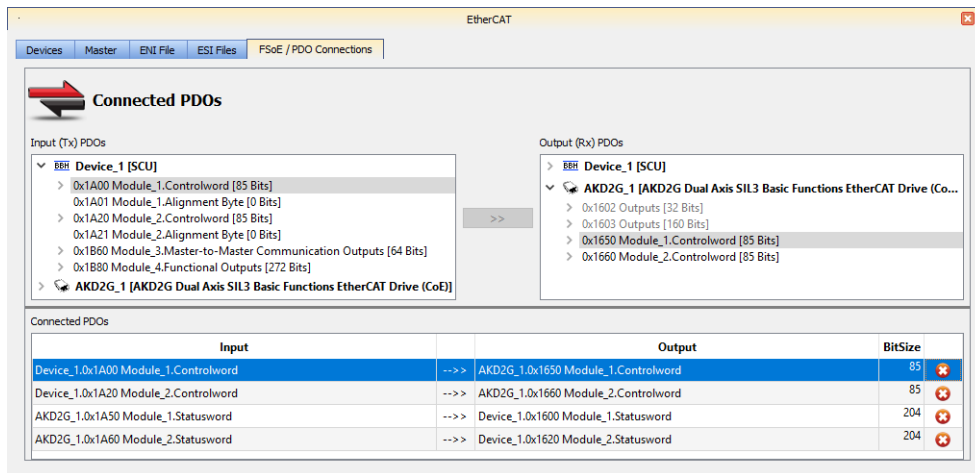
- Device vendors determine the PDO(s), content and possible selection exclusivity. This information is defined inside the device vendor's ESI file. Please contact the device vendor for details about a specific device.

### 7.12.3.4 How To Set Up a Variable to Monitor Safety Data

This example walks you through setting up a variable which can be used to monitor data from a safety device. The project for this example has a BBH SCU and an AKD2G.

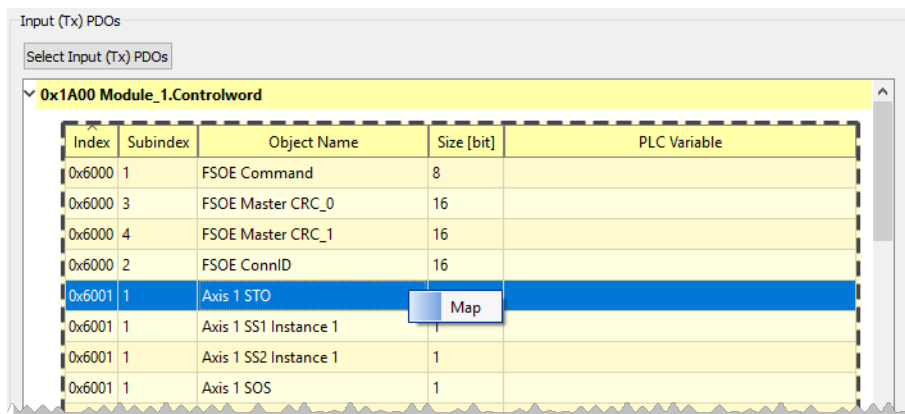
1. Connect the SCU and drive PDOs. (See [FSOE / PDO Connections Tab](#))

While this step is optional, it is quite useful and will save you time and effort.

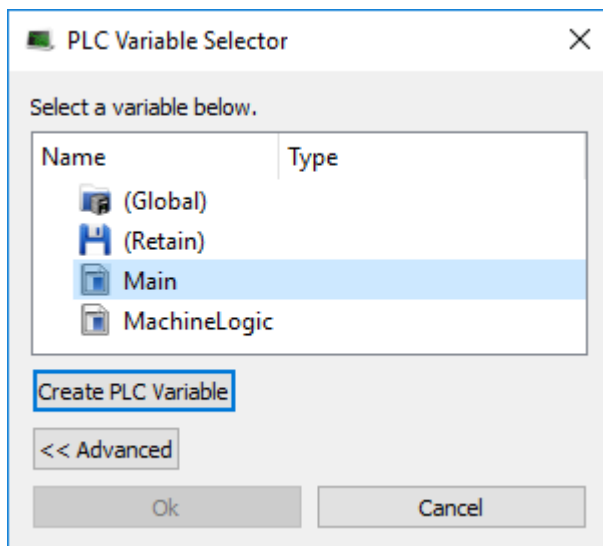


2. Right-click (or double-click) on an **Input (Tx) PDOs** object that will be monitored and select **Map**.

It is recommended that you use Tx PDOs as the source for a variable as they update every cycle, where Rx PDOs may not be current.



3. Read-Only variables are the only ones that will be shown in the **PLC Variable Selector**. Click **Create PLC Variable**. In this case we are creating the variable inside of **Main**.



4. Define a variable by giving it a name, type, and optionally an initial value, and click **OK**.

Create PLC Variable

Scope: Main

Name: SafeWatch\_1

Type: BOOL

Dimension: 0

Init Value:

Ok Cancel

5. The **PLC Variable Selector** now displays variable. Click **OK**.

PLC Variable Selector

Select a variable below.

Name	Type
(Global)	
(Retain)	
Main	
SafeWatch_1	BOOL
MachineLogic	

Create PLC Variable

<< Advanced

Ok Cancel

The variable is now assigned and available to monitor any object in the same Index:Subindex.

Input (Tx) PDOs

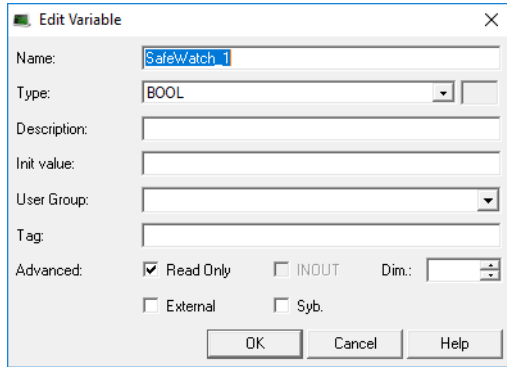
Select Input (Tx) PDOs

0x1A00 Module\_1.Controlword

Index	Subindex	Object Name	Size [bit]	PLC Variable
0x6000	1	F5OE Command	8	
0x6000	3	F5OE Master CRC_0	16	
0x6000	4	F5OE Master CRC_1	16	
0x6000	2	F5OE ConnID	16	
0x6001	1	Axis 1 STO	1	Main/SafeWatch_1
0x6001	1	Axis 1 SS1 Instance 1	1	Main/SafeWatch_1
0x6001	1	Axis 1 SS2 Instance 1	1	Main/SafeWatch_1
0x6001	1	Axis 1 SOS	1	Main/SafeWatch_1
0x6001	1	Axis 1 SSR Instance 1	1	Main/SafeWatch_1
0x6001	1	Axis 1 SDI_p	1	Main/SafeWatch_1
0x6001	1	Axis 1 SDI_n	1	Main/SafeWatch_1
0x6001	0	Axis 1 Acknowledge error	1	
0x6001	2	Axis 1 SS1 Instance 2	1	

6. **OPTIONAL:** The variable may be further defined, such as giving it a description. Looking at it, we can see that it was created as **Read Only**. It is recommended that the variable is not changed to Read/Write. If it is changed an error will be generated when the project is

compiled.



### 7.12.4 PDO Editor Tab

This tab provides the ability to modify editable PDOs. There are two sections, [Editable PDOs](#) and [PDO Content](#). See [PDO Restrictions and Compiler Errors](#) for information about compiler errors and certain constraints.

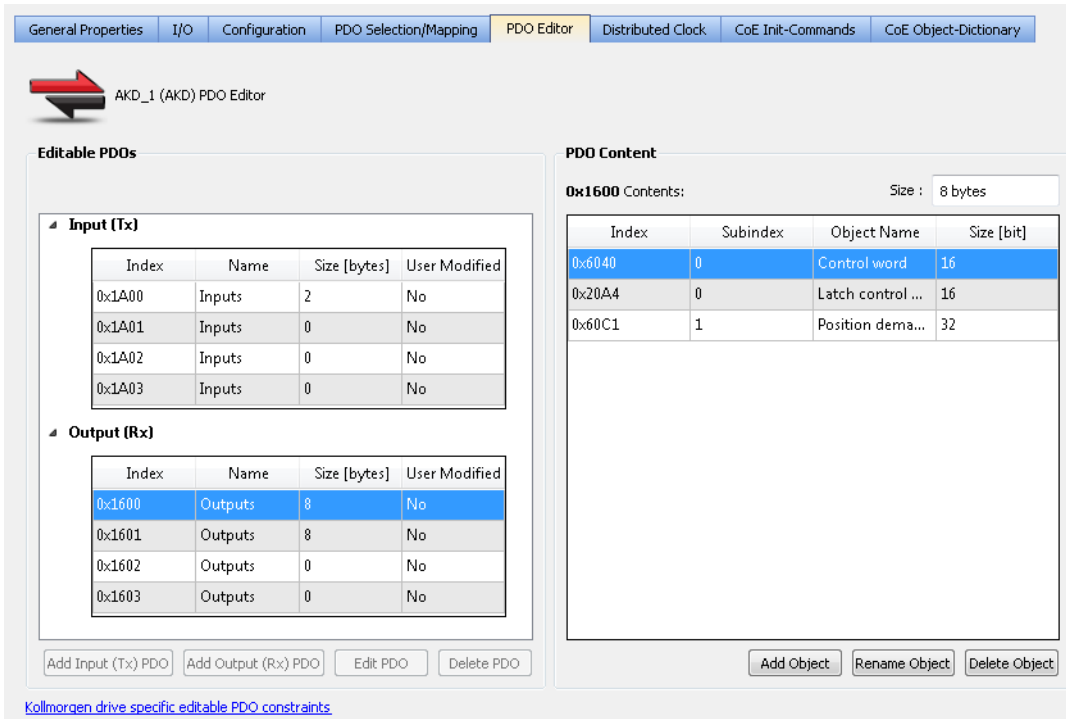
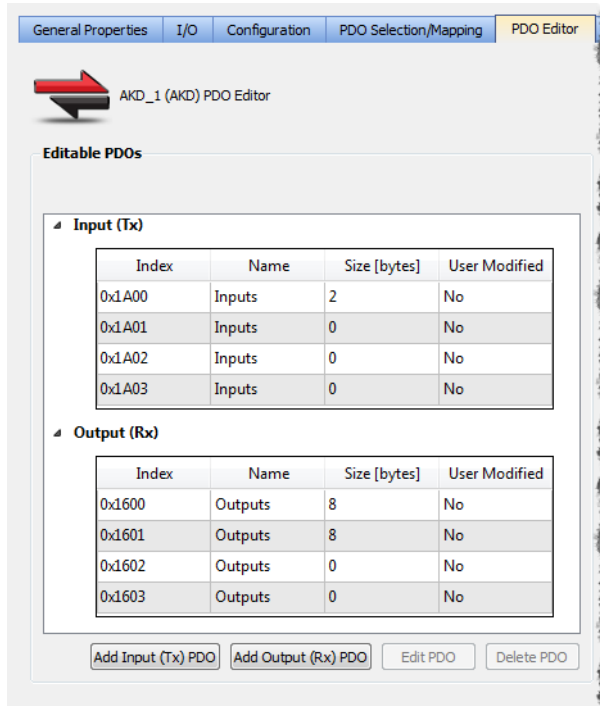


Figure 4-115: The PDO Editor tab

#### 7.12.4.1 Editable PDOs

This section lists all of the PDOs which may be edited. PDOs may be created (Add PDO), modified (Edit PDO), or removed from the project (Delete PDO). Please see the EtherCAT device-specific

documentation for more details.



The Input (Tx) and Output (Rx) PDOs are grouped separately. Each PDO table contains the following information:

<b>Index</b>	Identification value for the PDO. The range is defined by the EtherCAT specification. RxPDOs: 0x1600 to 0x17FF TxPDOs: 0x1A00 to 0x1BFF
<b>Name</b>	Text label to identify the PDO. The name is provided either by the device vendor or user, and can be changed by the user.
<b>Size [bytes]</b>	Total length for the PDO object(s) data.
<b>User Modified</b>	Yes: indicates the user has changed the PDO or its content No: indicates the PDO is the default provided by the device vendor.

The KAS IDE compiler enforces restrictions on the sizes of individual PDOs as well as on the sum of the sizes of PDOs. See [PDO Restrictions and Compiler Errors](#) for more information.

#### 7.12.4.1.1 Add a Flexible PDO

1. Click the **Add Input (Tx) PDO** or **Add Output (Rx) PDO** button to open the **Add Input** or **Add Output** window.
2. Select an **Index** value and enter a **Name** for the PDO. The dialog performs a live check for valid entries.
3. Click **OK**.

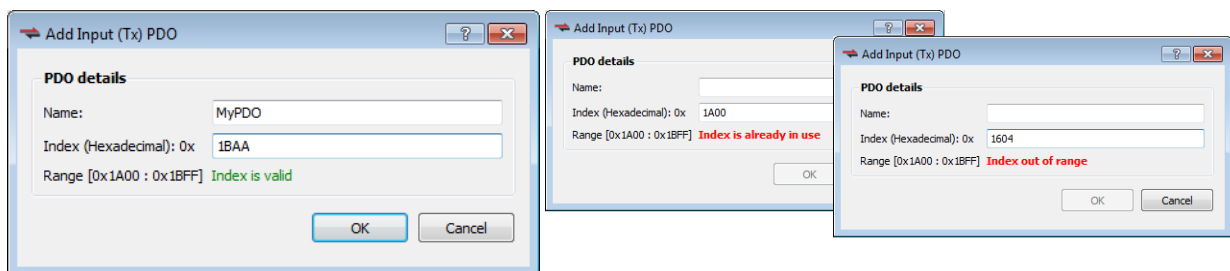


Figure 4-116: Examples of live checking for valid entries.

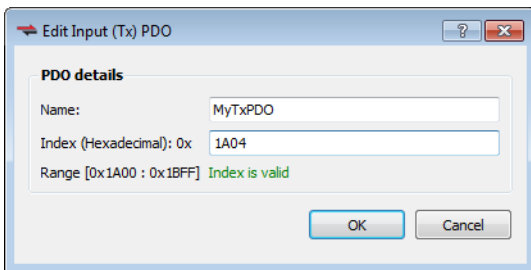
Objects may be added once the PDO has been created. See [Add a CoE Object](#).

EtherCAT device support for flexible PDOs is optional. Please consult the manufacturer's device-specific manual for details about support for flexible PDOs and which PDO index values are supported.

#### 7.12.4.1.2 Edit a Flexible PDO

A flexible PDO's Name and Index may be edited after it has been added.

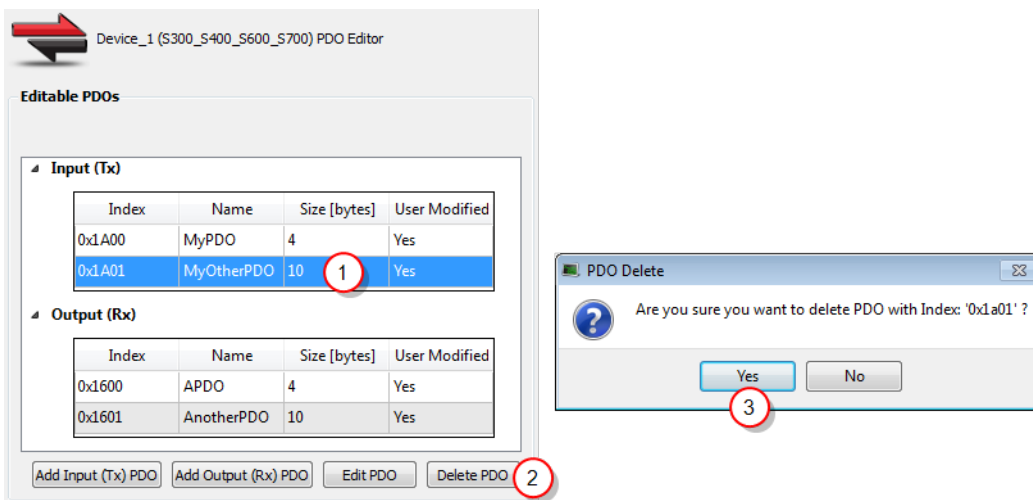
1. Select a flexible PDO from the list of Editable PDOs.
2. Click the **Edit PDO** button.
3. Make your changes. The dialog performs a live check, enabling the OK button when an entry is valid.
4. Click **OK**.



#### 7.12.4.1.3 Delete a Flexible PDO

A flexible PDO, which has previously been added, may be deleted.

1. Select a flexible PDO from the list of Editable PDOs.
2. Click the **Delete PDO** button.
3. Confirm your deletion by clicking **OK**.

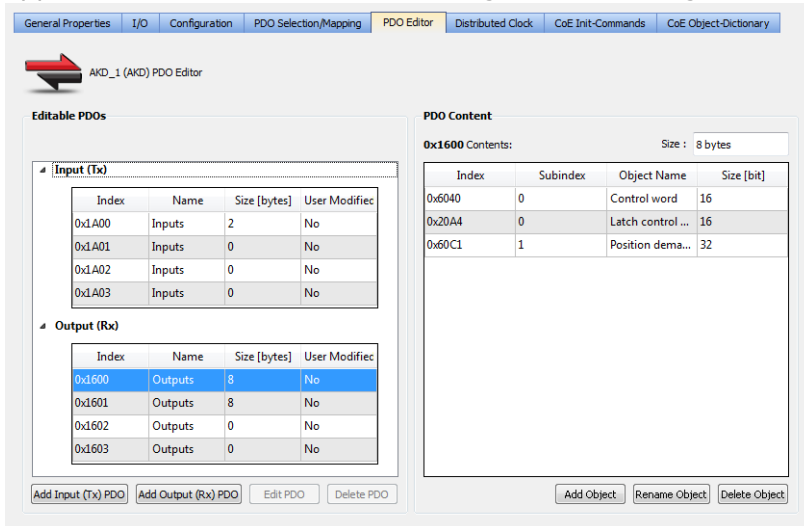


#### 7.12.4.2 PDO Content

This section lists the objects within the currently selected PDO. Objects may be created (**Add Object**), modified (**Edit Object**), or removed from the project (**Delete Object**). Select a PDO in the **Editable PDOs** section (left side of dialog box) to view the objects in a PDO. The list of objects will



appear in the **PDO Content** section (right side of dialog box).



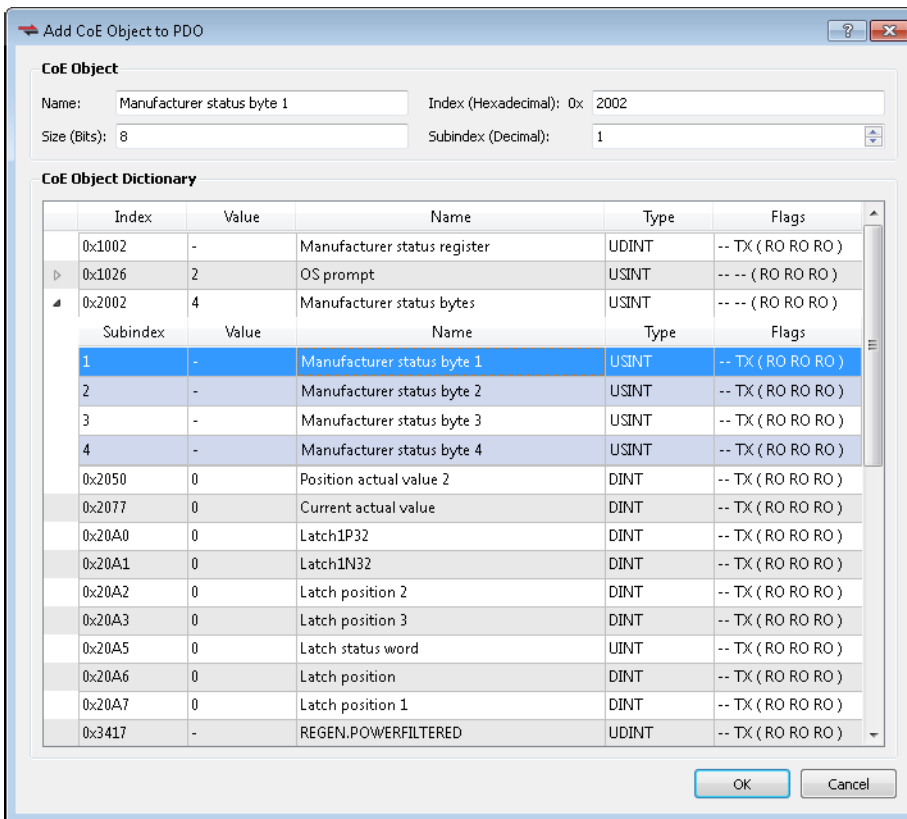
<b>Index</b>	Identification value for the PDO. The range is defined by the EtherCAT specification. RxPDOs: 0x1600 to 0x17FF TxPDOs: 0x1A00 to 0x1BFF
<b>Name</b>	Text label to identify the PDO. The name is provided either by the device vendor or user, and can be changed by the user.
<b>Size [bytes]</b>	Total length for the PDO object(s) data.
<b>User Modified</b>	Yes: indicates the user has changed the PDO or its content No: indicates the PDO is the default provided by the device vendor.

#### NOTE

**Constraints for the Kollmorgen AKD device family:** The PDO content **Size** field will be highlighted in red if the Editable PDO content size is greater than 8 bytes.

#### 7.12.4.2.1 Add a CoE Object

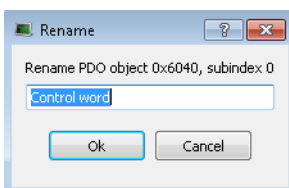
1. Select a PDO in the **Editable PDOs** section.
2. Click the **Add Object** button to show the **Add CoE Object to PDO** window.
3. Select an object from the **CoE Object Dictionary** or manually enter the Name, Index, Subindex and Size data, and click **Ok**.

**NOTE**

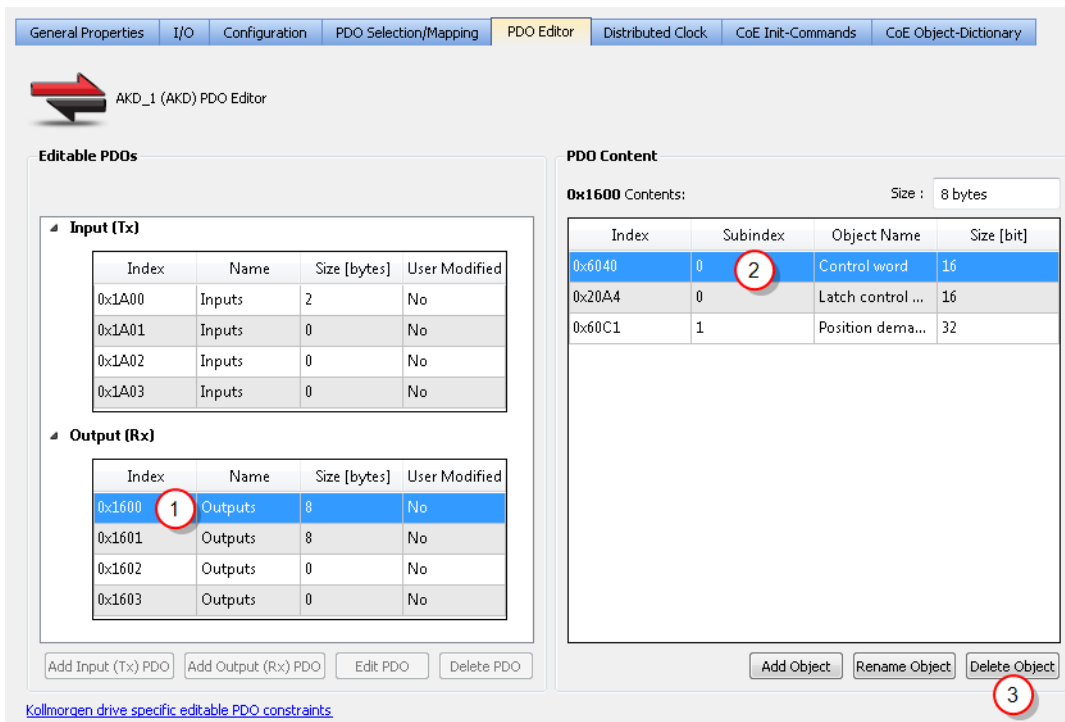
The Flags indicate whether an object can be selected for an Input (Tx) or an Output (Rx) PDO. If the Flags are "-- --" then the object is not eligible for the PDOs. Selecting the wrong object type will generate an error.

**7.12.4.2.2 Rename a CoE Object**

1. Select a PDO from the **Editable PDOs** list.
2. Select the object to be renamed from the PDO Content.
3. Click the **Rename Object** button.
4. In the **Rename** pop up, enter the new name.
5. Click **Ok**.

**7.12.4.2.3 Delete a CoE Object**

1. Select a PDO from the **Editable PDOs** list.
2. Select the object to be deleted from the PDO Content.
3. Click the **Delete Object** button.
4. Confirm the deletion.



### 7.12.4.3 PDO Restrictions and Compiler Errors

#### 7.12.4.3.1 Restrictions for AKD Devices

The KAS IDE compiler enforces the following restrictions to the editable PDOs for AKD devices:

- The maximum byte count for the content of an individual PDO (both selected and unselected) is 8 bytes. The PDO content **Size** field will be highlighted in red if the Editable PDO content size is greater than 8 bytes.
- The total maximum byte count for the content of all Tx PDOs (both selected and unselected) is 32 bytes.
- The total maximum byte count for the content of all Rx PDOs (both selected and unselected) depends upon the firmware.

Firmware Version	Byte Limit	Notes
v1-17 or later	32 bytes	This is the byte limit when the EtherCAT parameter <a href="#">ECAT.LEGACYREV</a> is set to 0.
v1-16 or later	20 bytes	This is the byte limit when the EtherCAT parameter <a href="#">ECAT.LEGACYREV</a> is set to 1.
v1-15 or earlier	20 bytes	

- The total byte count for the content of all selected Tx PDOs must be **even**. This does not apply to AKD2G devices.
- The total byte count for the content of all selected Rx PDOs must be **even**. This does not apply to AKD2G devices.

#### TIP

The object 0x00 can be used to pad the PDO to get the total byte count to an even value.

Compiler errors will be generated if these restrictions are not met. Example compiler errors:

```
In AKD_1, the size of Rx PDO 0x1600 is greater than the limit: 9 > 8
```

In AKD\_1, the total byte count for all editable Rx PDOs is greater than the limit: 39 > 32

In AKD\_1, the total byte count for all editable Rx PDOs that are assigned is not even: 29

#### 7.12.4.3.2 Restrictions for AKD2G Devices

The KAS IDE compiler enforces the following restrictions to the editable PDOs for AKD2G devices:

- PDOs have a size constraint of 128 bytes per PDO.
- The total maximum byte count for the content of all Rx PDOs (both selected and unselected) is 512 bytes [(128 per PDO) x 4 = 512 bytes].
- The total maximum byte count for the content of all Tx PDOs (both selected and unselected) is 512 bytes [(128 per PDO) x 4 = 512 bytes].
- There is a limitation to the number of PDO objects, based on the the Sample Rate / Cycle Time.

Sample Rate	Cycle Time	Max. PDOs
4kHz	250µs	14 Rx, 14 Tx
2kHz	>500µs	30 Rx, 30 Tx

#### 7.12.4.3.3 Restrictions for All Other Devices

The KAS IDE compiler enforces the following restriction to the editable PDOs for all devices:

- The maximum byte count for the content of an individual PDO (both selected and unselected) is 64 bytes.

Compiler errors will be generated if these restrictions are not met. Example compiler errors:

In Device\_1, the size of Tx PDO 0x1A16 is greater than the limit: 66 > 64

#### 7.12.4.4 Redundant PDO Entries

It is possible within the KAS IDE to have the same PDO entry occur in multiple PDOs (or even in a single PDO). A PDO entry is uniquely defined by an index and a subindex. The name of the entry is irrelevant in regards to identification.

In the example below, the entry defined by index 0x6040, subindex 0, appears in the Rx PDOs 0x1600 and 0x1604. Similarly, the entry 0x3470, subindex 4 appears in two different PDOs.

AKD\_1 (AKD) PDO Selection/Mapping

Output (Rx) PDOs

Select Output (Rx) PDOs

0x1600 Outputs

Index	Subindex	Object Name	Size [bit]	PLC Variable
0x20A4	0	Latch control word	16	⚠
0x6040	0	Control word	16	⚠
0x60C1	1	Position demand v...	32	⚠

0x1601 Outputs

0x1604 my\_Rx\_PDO

Index	Subindex	Object Name	Size [bit]	PLC Variable
0x6040	0	Control word	16	⚠

Input (Tx) PDOs

Select Input (Tx) PDOs

Index	Subindex	Object Name
0x3470	4	AOUT_INPUT_VAL...

0x1B23 Inputs

Index	Subindex	Object Name
0x2050	0	Position actual valu...
0x20A5	0	Latch status word
0x20A6	0	Latch position
0x3470	4	AIN.VALUE
0x6041	0	Status word
0x6063	0	Position actual value

- A redundant Rx PDO entry will generate a compiler error. For example, the redundant Rx PDO illustrated above would generate the following compiler error:

```
EtherCAT: ERROR: In AKD_1, PDO object index 0x6040, subIndex 0 is redundant. It is in the following PDO(s): 0x1600, 0x1604
```

- Attempting to export the ENI file will fail with the error:

```
Failed to export ENI file, check the logs for more details.
```

- A redundant Tx PDO entry will generate a compiler warning, such as the example below:

```
EtherCAT: Warning : In AKD_1, PDO object index 0x3470, subIndex 4 is redundant. It is in the following PDO(s): 0x1A04, 0x1B23
```

- A redundant Tx PDO entry will not prevent a successful compilation nor will it generate an error when exporting the ENI file.

#### 7.12.4.5 About Safety, PDOs, and Variables

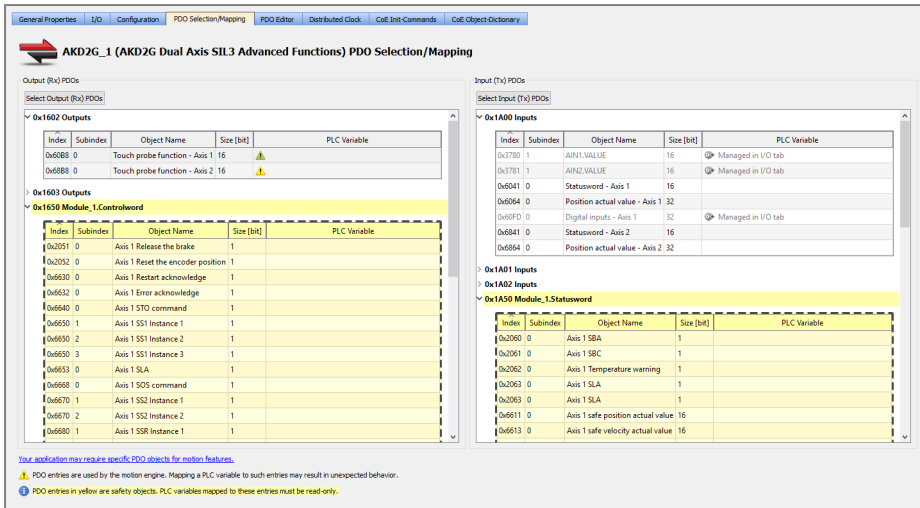
This topic discusses PDOs, safety objects, and variable mapping. Within the PDO Selection / Mapping tab any safety-related objects are clearly identifiable by being in a yellow table. Mapping a variable to an object is easily done by any of the following methods.

- Double-clicking on the object and then selecting or creating a variable
- Right-clicking on the object and selecting Map and then selecting or creating a variable
- Dragging an existing variable from the Dictionary to the object

Variables assigned to a safety object must be Read Only. Variables that are Read/Write will not show up in dialogs, nor will drag-and-drop work.

#### NOTE

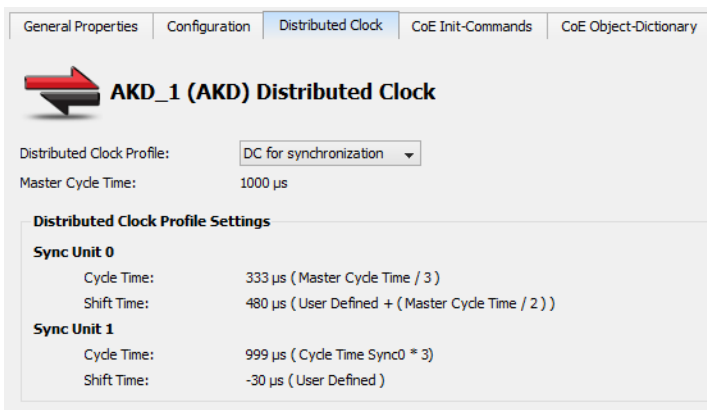
Once a variable is assigned, there is no further validation until the project is compiled. That means a variable could be changed to Read/Write and still be assigned. When you compile the project an error will be generated.



See [How To Set Up a Variable to Monitor Safety Data](#) for step-by-step instructions.

### 7.12.5 Distributed Clock tab

The contents of this tab allow you to change settings related to the Distributed Clock for all Kollmorgen and third-party devices (both discovered and manually added).



Element	Description
<b>Distributed Clock Profile</b>	Select the Distributed Clock (DC) operation mode. These modes cannot be edited.  <b>FreeRun, DC unused, SM-Synchronous:</b> Device cyclic operation is triggered by the EtherCAT master's SyncManager via the cyclic frame. Devices are not synchronized by the reference clock.  <b>DC-Synchron, DC for synchronization, DC-Synchronous:</b> Device cyclic operation is triggered by the local distributed clock. Devices are synchronized by the reference clock.  <b>DC-Synchron (input based):</b> as DC-Synchron mode, with the cyclic start time chosen such that it coincides with other input slaves in the EtherCAT system.
<b>Master Cycle Time</b>	Base interval in microseconds, which will be used by the master. This is changed and automatically updated by changing the <b>Cycle Time</b> value on the <a href="#">EtherCAT Master Settings Tab</a> tab.

Element	Description
<b>Sync Unit 0</b>	Cycle Time: <ul style="list-style-type: none"> <li>• Sync Unit Cycle: Unit is synchronized relative to the Master Cycle Time</li> <li>• User defined: Unit has its own interval</li> </ul> Shift Time: <ul style="list-style-type: none"> <li>• Unit is adjusted by the shift time</li> </ul>
<b>Sync Unit 1</b>	Cycle Time: <ul style="list-style-type: none"> <li>• Sync Unit Cycle: Unit is synchronized relative to the Master Cycle Time</li> <li>• Sync 1 Cycle: Unit is synchronized relative to the First Sync Unit</li> <li>• User defined: Unit has its own interval</li> </ul> Shift Time: <ul style="list-style-type: none"> <li>• Unit is adjusted by the shift time</li> </ul>

**NOTE**

Some or none of the content will be available under the following scenarios:

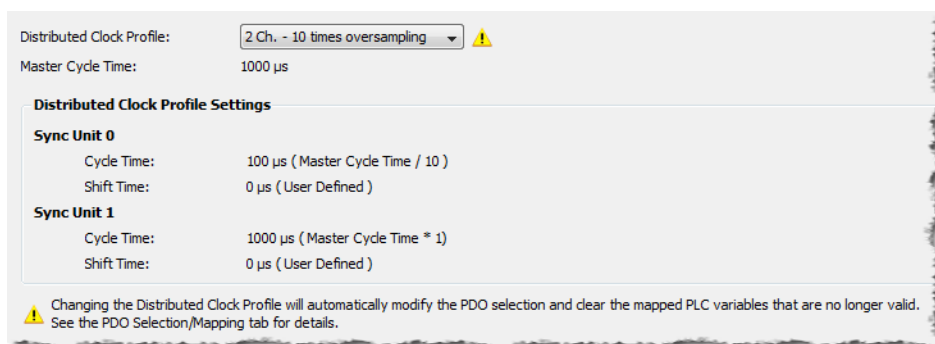
- The Sync 0 or Sync 1 parameter is not present
- Distributed Clock is not supported by the device
- The ESI file is missing.

**7.12.5.1 Oversampling devices**

Some EtherCAT devices have oversampling features. An oversampling device is typically able to record (input) or provide (output) signals at a higher rate than the EtherCAT cycle time. This rate is called the *oversampling factor*. For example, with an oversampling factor of 10 and an EtherCAT cycle time of 1ms (1Khz), an input device can record values every 100µs (1000 divided by 10).

Oversampling devices have as many PDO objects in their cyclic frames as the oversampling factor in order to achieve the higher rate. Each of these PDO objects corresponds to one sample. For example, an output device with an oversampling factor of 4 will typically have 4 PDO objects: Output 1, Output 2, Output 3 and Output 4.

The oversampling factor is tied to the Distributed Clock Profile. A warning is displayed next to the DC profile selection box in the Distributed Clocks tab when a device has oversampling features, as seen below.







**Figure 4-117:** Example of a device with oversampling.

Changing the Distributed Clock Profile will automatically change the PDO selection. All PDOs corresponding to the selected Distributed Clock Profile and its according oversampling factor will be selected. These maps will be discarded if one or several PLC variables were mapped to a PDO that is no longer selected.

### 7.12.6 CoE Init Commands tab

This tab displays the EtherCAT device's CoE Init commands. The Init commands are grouped based on the EtherCAT transition state. The sequence within each group defines the order in which the commands are executed on the device.

The order of CoE Init Commands can be changed to define the correct command sequence required for device operation. Selecting a command enables buttons to move a command Up , Down , to the Top of the table , and the Bottom of the table .

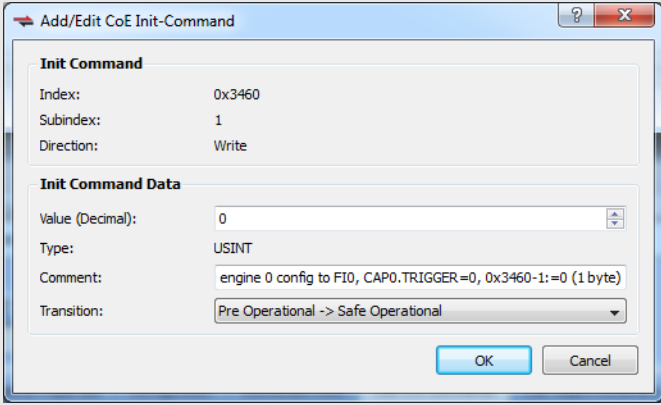
**NOTE**

- This tab is grayed out if the device does not support the CoE Protocol or the ESI file is unavailable.
- Fixed Init-commands have been deprecated and will not be displayed.

Field	Description
<b>Index</b>	The hex value of the CoE-Index
<b>Subindex</b>	The CoE-Subindex
<b>Value</b>	Value of the init command
<b>Comment</b>	Description of the init command
<b>Direction</b>	Specifies if the command is <b>Read</b> or <b>Write</b> .
<b>Source</b>	The source of the init command. There are two possible values. <ul style="list-style-type: none"> <li>• ESI File denotes that the command comes from the ESI file</li> <li>• User denotes that the command is created by the user.</li> </ul>

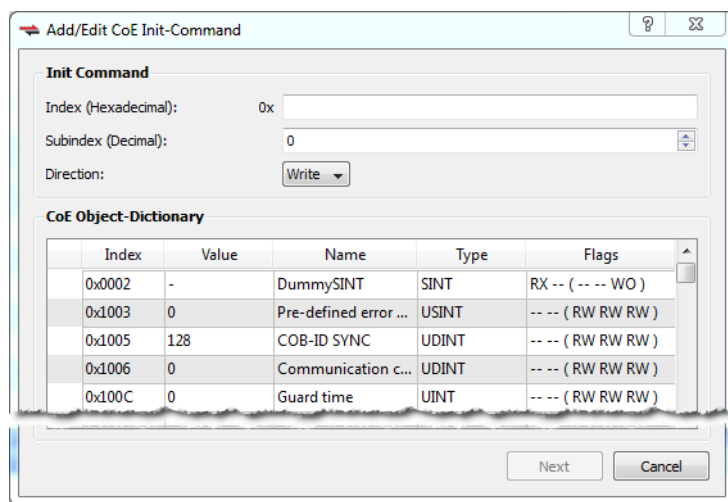
Function	Description
<b>Add</b>	Clicking Add opens a dialog which allows you to create a new Init Command. See <a href="#">Adding CoE Commands</a> for more information.



Function	Description
<b>Edit</b>	This function allows you to edit an existing Init Command's attributes. Select an entry and click this button or double-click an entry. The dialog allows you to edit the command data including the <b>Value</b> , <b>Comment</b> , and <b>Transition</b> .
	
<b>Delete</b>	Delete the selected Init Command row.

### 7.12.6.1 Adding CoE Commands

This command allows you to construct a new Init Command by supplying Index, Subindex, Value, Comment, Transition, and Direction attributes. The dialog lists the device's object dictionary and the objects are filtered based on the **Direction** currently selected, e.g. if "Write" is selected then the Read-only objects are not shown.



#### NOTE

- The **Index** field accepts four characters at most.
- The **Subindex** has a range of 0 - 255.
- When setting the Direction you may select **Read** or **Write**.
- Selecting an item in the CoE Object-Dictionary auto-populates the **Index** and **Subindex** entries.

The **Next** button is available once the **Index** and **Subindex** fields have values.

*If the Init Command is already present in the dictionary:*

If the Init Command is already in the dictionary (i.e. the Index and Subindex you specified are already used) then you are presented with a dialog which allows you to specify the **Value**, a **Comment**, and the **Transition**.

- The Transition may be **Pre Operational -> Safe Operational** or **Safe Operational -> Operational**.
- If the **Type** is "STRING (20)" or "STRING (50)", you may only enter up to 20 or 50 characters.

Clicking **OK** adds the command to the appropriate transition group in the [CoE Init Commands tab](#) and sets the Source attribute to **User**.

The screenshot shows a dialog box titled "Add/Edit CoE Init-Command". It has two main sections: "Init Command" and "Init Command Data".

- Init Command:**
  - Index: 0x0002
  - Subindex: 0
  - Direction: Write
- Init Command Data:**
  - Value (Decimal): 0
  - Type: SINT
  - Comment: DummySINT
  - Transition: Pre Operational -> Safe Operational

At the bottom, there are "OK" and "Cancel" buttons.

If the Init Command data type is an array of bytes, then all the bytes can be entered in HexBinary format in the **Value** text box. The size of the array determines the maximum data that can be entered.

The screenshot shows a dialog box titled "Add/Edit CoE Init-Command". It has two main sections: "Init Command" and "Init Command Data".

- Init Command:**
  - Index: 0x2000
  - Subindex: 3
  - Direction: Read
- Init Command Data:**
  - Value (HexBinary): 00 01
  - Type: ARRAY [0..2] OF BYTE
  - Comment: Reply
  - Transition: Pre Operational -> Safe Operational

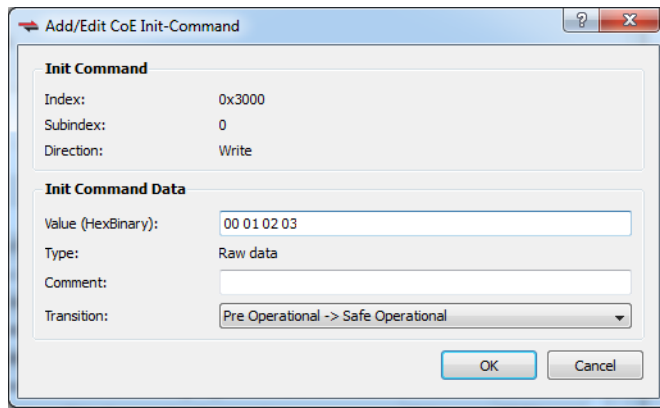
At the bottom, there are "OK" and "Cancel" buttons.

*If the Init Command is not present in the dictionary:*

Adding a new command (i.e. the Index and Subindex are not in the dictionary) presents you with a dialog which allows you to specify a **Value**, **Comment**, and **Transition**.

- The Value must be entered in HexBinary format as shown below.
- The Transition may be **Pre Operational -> Safe Operational** or **Safe Operational -> Operational**.

Clicking **OK** adds the command to the appropriate transition group in the [CoE Init Commands tab](#) and sets the Source attribute to **User**.



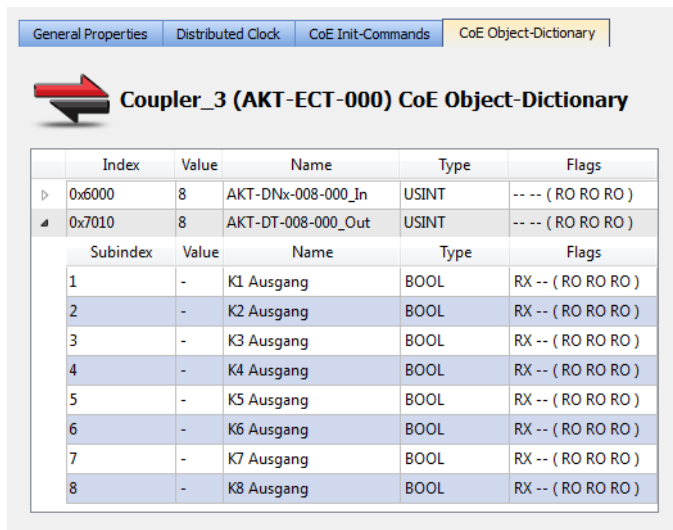
### 7.12.7 CoE Object Dictionary Tab

This tab displays the all the CoE (CAN over EtherCAT) objects associated with the EtherCAT device. It is used as reference to add new CoE Init Commands for the EtherCAT device. The fields in the table are described below. The CoE objects can be used for three different actions, depending upon the access flags.

- PDO mapping for cyclic communication during application execution
- CoE Init commands during application start-up
- SDO communication during application execution

#### NOTE

This tab is grayed out if the ESI file does not have CoE object information, or if the file is unavailable.



CoE objects can be composed of a simple data type or a complex type. In case of a complex data type, each simple data type composing it has a SubIndex . As shown in the example below, the CoE object can be expanded to show all the subindex's details that are the same as the details for a simple type CoE object.

Index	Value	Name	Type	Flags
0x10F3	5	Diagnostics: History buffer	USINT	--- (RO RO RO)
Subindex				
1	32	Max. number of messages	USINT	--- (RO RO RO)
2	0	Latest message	USINT	--- (RO RO RO)
3	0	Latest acknowledged message	USINT	--- (RW RW RW)
4	0	New active message	USINT	-- TX (RO RO RO)
5	0	Control bits	UINT	--- (RW RW RW)

Field	Description
<b>Index / SubIndex</b>	CoE Index or SubIndex number
<b>Name</b>	Name of the CoE object
<b>Value</b>	Default value of the CoE object
<b>Type</b>	Data type of the CoE object
<b>Flags</b>	<p>This column contains 5 values in the format <b>XX YY (AA BB CC)</b>, each representing an access option.</p> <p>The values of XX and YY provide PDO mapping options:</p> <p>XX      Specifies if the CoE object can be mapped as RxPDO. It is represented as <b>RX</b> if it can be mapped.</p> <p>YY      Specifies if the CoE object can be mapped as TxPDO. It is represented as <b>TX</b> if it can be mapped.</p> <p>The values for AA, BB, and CC provide the CoE Access type. The values can be read only (<b>RO</b>), read-write (<b>RW</b>), or write only (<b>WO</b>)</p> <p>AA      Access type of the object in PREOP state</p> <p>BB      Access type of the object in SAFEOP state</p> <p>CC      Access type of the object in OP state</p>
<b>Module</b>	Name of the module that is associated with the CoE Object

## 8 Understanding KAS

This chapter gives explanation about the most important **concepts** that you need to understand to use KAS.

### ⚠ IMPORTANT

To take full advantage of KAS functions, a basic understanding of automation (programming languages and motion control) is required.

### 8.1 IEC 61131-3

#### 8.1.1 Introduction

To create programs for the implementation of the PLC part of your application, the KAS IDE complies with IEC 61131-3. This standard currently defines five programming languages for programmable control systems.

The KAS IDE implements this standard to provide you with well-defined and well-known programming languages.

#### 8.1.2 Data Types

Data types are defined within the common elements of IEC 61131-3.

##### Why Data typing?

Data typing is implemented to define the type of any parameter used, which helps to prevent errors early on in the programming phase. This avoids for example dividing a Date by an Integer.

When you have defined whether the data is a string, a date, an integer or a 16-bit Boolean input, there is no longer any confusion, nor any conflict between different people using the textual representation (i.e. the name of the variable).

##### Different kinds of Data types

Common data types are Boolean, Integer, Real, Byte, Word, Date, Time\_of\_Day, and String. Based on these, you can define your own personal data types, known as derived data types. In this way you can define an analog input channel as a data type, and re-use it.

##### List of Data types

Below are the available basic data types:

Types	Description	Values	Prefixes
<b>BOOL</b>	Boolean (bit)	FALSE or TRUE - stored in 1 byte	
<b>SINT</b>	Small signed integer in 8 bits	-128 to +127	SINT#
<b>USINT</b>	Small unsigned integer in 8 bits	0 to +255	USINT#
<b>BYTE</b>	<i>Same as USINT</i>		
<b>INT</b>	Signed integer in 16 bits	-32768 to +32767	INT#
<b>UINT</b>	Unsigned integer in 16 bits	0 to +65535	UINT#
<b>WORD</b>	<i>Same as UINT</i>		
<b>DINT</b>	Signed double precision integer in 32 bits	-2147483648 to +2147483647	
<b>UDINT</b>	Unsigned integer in 32 bits	0 to +4294967295	UDINT#
<b>DWORD</b>	<i>Same as UDINT</i>		

Types	Description	Values	Prefixes
<b>LINT</b>	Long signed integer in 64 bits		LINT#
<b>ULINT</b>	Long unsigned integer in 64 bits		ULINT#
<b>LWORD</b>	<i>Same as ULINT</i>		
<b>REAL ‡</b>	Single precision floating point stored in 32 bits	-3.4E38 to 3.4E38 and -3.4E-38 to 3.4E-38 (6 to 7 significant digits of accuracy)	LREAL# ‡‡
<b>LREAL ‡</b>	Double precision floating point stored in 64 bits	-1.7E308 to 1.7E308 and -1.7E-308 to 1.7E-308 (14 to 15 significant digits of accuracy)	LREAL#
<b>TIME</b>	Time data type is used to specify a time variable - accuracy is 1ms. See <a href="#">TIME</a> for more information.	0ms to 24h	T# or TIME#
<b>STRING</b>	Variable length string with declared maximum length Each character is store on 1 byte (i.e. on 8 bits)	Maximum length cannot exceed 255 characters	

**NOTE**

‡ REAL variables are limited to 6 digits of accuracy. To achieve greater accuracy, a longer mantissa may be specified by prefixing LREAL with #.

**Example:** To achieve an accuracy of 20 digits for the value of pi, rather than what REAL provides (3.14159), set the type to LREAL#3.141592653589793238.

**IMPORTANT**

‡‡ REAL is restrictive, but because it is the default, it is recommended to explicitly declare your real constants with the **LREAL#** prefix.

**TIP**

You can use **2#**, **8#** or **16#** prefixes to specify an integer in binary, octal or hexadecimal basis respectively.

**8.1.2.1 Structures**

A structure is a complex data type defined as a set of members. Members of a structure can have various data types. A member of a structure can have dimensions or can be an instance of another structure.

When a structure is defined, it can be used like other data types to declare variables.

Members of a structure can have an initial value. In that case, corresponding members of all declared variables having this structure type will be initialized with the initial value of the member.

To specify a member of a structured variable in PLC languages, use the following notation:

```
VariableName.MemberName
```

**8.1.2.1.1 Limitation**

If a member of a structure is an instance of another structure, the nested structure must be declared **BEFORE** in the list.

**8.1.2.2 Arrays**

You can declare arrays for internal variables by specifying dimension(s).

To declare an array, enter the number of elements in the **Dim.** column of the **Dictionary** (see procedure [here](#)).

Name	Type	Dim.	Attrib.
NewVar	BOOL		
MachineLogic		2,10,4	
NewVar	BOOL	[0..1,0..9,0..3]	

For a multi-dimensional array (note that arrays have at most three dimensions), enter the number of elements for each dimension separated by commas (for example **2,10,4** is a 3 dimensional array, the first dimension has 2 elements, the second dimension has 10 elements, and the third dimension has 4 elements).

#### ⓘ IMPORTANT

All indexes are 0 based. For example, in the case of a single dimension array, the first element is always identified by *ArrayName[0]*.

### 8.1.3 Variables

The scopes of the variables are normally limited to the organizational unit in which they are declared, e.g. local. This means that their names can be re-used in other parts without any conflict, eliminating another source of errors, e.g. the scratchpad. If the variables have global scope, they must be declared as such. Parameters can be assigned an initial value to have the right setting at start up and cold restart.

#### 8.1.3.1 About Retain Variables

A retain variable is a PLC variable which:

- is non-volatile: stored in non-volatile controller memory (AKD PDMM or PCMM). When using KAS Simulator the retain variables are stored in a normal disk file.
- is known by all programs (when its content is changed, the change is propagated to all equations in which this variable is used)
- normally does not contain real-time critical data.

#### ↖ TIP

- Retain variables are saved when you stop the PLC. The next time the PLC is started the retained values are used. Additionally, retains are auto-saved every two seconds. This applies to both the AKD PDMM and PCMM models.
- Variables stored in non-volatile memory that have changed will be updated every two seconds. Unchanged variables are not updated.

Such a variable is used to store application specific data, like for instance to count a cutting-edge cycle in order to stop for its blade replacement after a specific number of iterations.

- On an application "Start", KAS initializes the retain variables with the value stored in the NVRAM.
- On an application "Cold Start", KAS initializes the retain variables with their default value.
- If the declaration of the retain variables is different between the non-volatile memory and the project, the retain variables are also reset to their default value.

Two parameters are checked to identify if the declaration changed:

- The number of variables of each type
- The length of a STRING type variables

#### ⓘ IMPORTANT

The non-volatile memory size is hardware dependent. If the size of the retained variables is larger

than the non-volatile storage space, an error will be logged and the data will not be stored in non-volatile memory. See [NVRAM](#) for more information.

For the KAS Runtime Simulator, the retained variables are saved in a file in your project repository.

#### NOTE

The following actions will reset retained value(s) to their Init value(s):

- Changing the type of a retain variable
- Changing the length of a string retain variable
- Changing the size of an array variable
- Changing any element of a structure variable

For more information on retained variables see [Retain Variables](#).

### 8.1.3.2 Working with Variables

All variables used in programs must be first declared in the variable editor. Each variable belongs to a group and must be identified by a unique name within its group.

#### 8.1.3.2.1 Groups

A group is a set of variables. A group either refers to a physical class of variables, or identifies the variables local to a program or user-defined function block. Below are the possible groups:

Groups	Description
<b>GLOBAL</b>	Internal variables known by all programs
<b>RETAIN</b>	Non volatile internal variables known by all programs
<b>%I...</b>	Channels of an input board - variables with same data type linked to a physical input device
<b>%Q...</b>	Channels of an output board - variables with same data type linked to a physical output device
<b>PROGRAMxxx</b>	All internal variables local to a program (the name of the group is the name of the program)
<b>UDFBxxx</b>	All internal variables local to a User-Defined Function Block plus its IN and OUT parameters (the name of the group is the name of the program)

#### 8.1.3.2.2 Data type and dimension

Each variable must have a valid data type. It can be either a basic data type or a function block. In the later case, the variable is an instance of the function block. Physical I/Os must have a basic data type. Instances of function blocks can refer either to a standard block, or to a User Defined Function Block.

If the selected data type is STRING, you must specify a maximum length. This cannot exceed 255 characters.

Refer to the list of [available data types](#) for more information. Refer to the section describing function blocks for further information about how to use a function instance.

Additionally, you can specify dimension(s) for an internal variable, in order to declare an [array](#).

#### 8.1.3.2.3 Naming a variable

A variable must be identified by a unique name within its parent group. The variable name cannot be a reserved keyword of the programming languages and cannot have the same name as a standard function or function block. A variable must not have the same name as a program or a user-defined function block.



The name of a variable must begin by a letter or an underscore ("\_") mark, followed by letters, digits or underscore marks. It is not allowed to put two consecutive underscores within a variable name. Naming is case-insensitive. Two names with different cases are considered as the same.

<b>%IX</b>	1 byte input - BOOL or SINT
<b>%QX</b>	1 byte output - BOOL or SINT
<b>%IW</b>	2 bytes input - INT
<b>%QW</b>	2 bytes output - INT
<b>%ID</b>	4 bytes input - DINT or REAL
<b>%QD</b>	4 bytes output - DINT or REAL
<b>%IL</b>	8 bytes input - LINT or LEAL
<b>%QL</b>	8 bytes output - LINT or LEAL
<b>%IS</b>	STRING input
<b>%QS</b>	STRING output

#### 8.1.3.2.4 Attributes of a variable

Physical I/Os are marked as either "**Input**" or "**Output**". Each internal variable can be configured as Read/Write or **Read Only**. Read Only variables can be mapped to Outputs, but not to Inputs. This is because Inputs can change state and a Read Only variable would not be able to change its value to match the input state.

Parameters of User-Defined Function Blocks are marked as either **IN** or **OUT**.

#### 8.1.3.3 Retain Variables

##### 8.1.3.3.1 What is a retain variable?

A retain variable is a PLC variable which:

- is non-volatile: stored in non-volatile controller memory (AKD PDMM or PCMM). When using KAS Simulator the retain variables are stored in a normal disk file.
- is known by all programs (when its content is changed, the change is propagated to all equations in which this variable is used)
- normally does not contain real-time critical data.

#### TIP

- Retain variables are saved when you stop the PLC. The next time the PLC is started the retained values are used. Additionally, retains are auto-saved every two seconds. This applies to both the AKD PDMM and PCMM models.
- Variables stored in non-volatile memory that have changed will be updated every two seconds. Unchanged variables are not updated.

Such a variable is used to store application specific data, like for instance to count a cutting-edge cycle in order to stop for its blade replacement after a specific number of iterations.

- On an application "Start", KAS initializes the retain variables with the value stored in the NVRAM.
- On an application "**Cold Start**", KAS initializes the retain variables with their default value.
- If the declaration of the retain variables is different between the non-volatile memory and the project, the retain variables are also reset to their default value.

Two parameters are checked to identify if the declaration changed:

- The number of variables of each type
- The length of a STRING type variables

**IMPORTANT**

The non-volatile memory size is hardware dependent. If the size of the retained variables is larger than the non-volatile storage space, an error will be logged and the data will not be stored in non-volatile memory. See [NVRAM](#) for more information.

For the KAS Runtime Simulator, the retained variables are saved in a file in your project repository.

**NOTE**

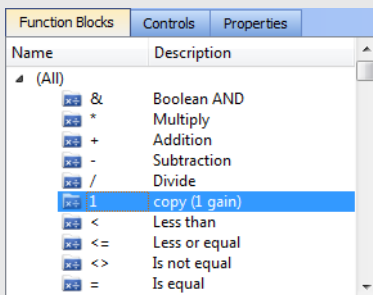
The following actions will reset retained value(s) to their Init value(s):

- Changing the type of a retain variable
- Changing the length of a string retain variable
- Changing the size of an array variable
- Changing any element of a structure variable

**TIP**

*Retained variables should not be used as the output of Function Blocks. Doing so may cause problems with retaining the value if the FB is executed with the Enable input off. The output structure can cause it to be conditionally updated to zero or the old value.*

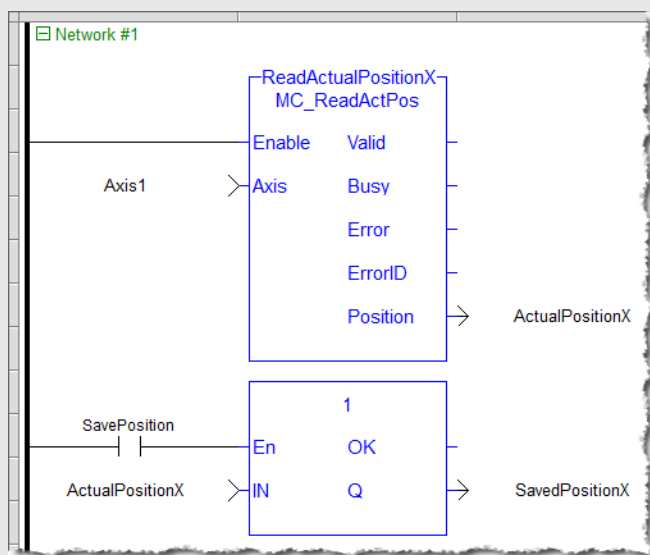
A work-around solution is to use a `copy (1 gain)` function to selectively enable updating the retain variable from an intermediate variable, which is the output of a function block.



*Work-around example*

In the following image `ReadActualPositionX` is continually reading position feedback from an axis, and the copy block is saving a specific value to a retain variable.

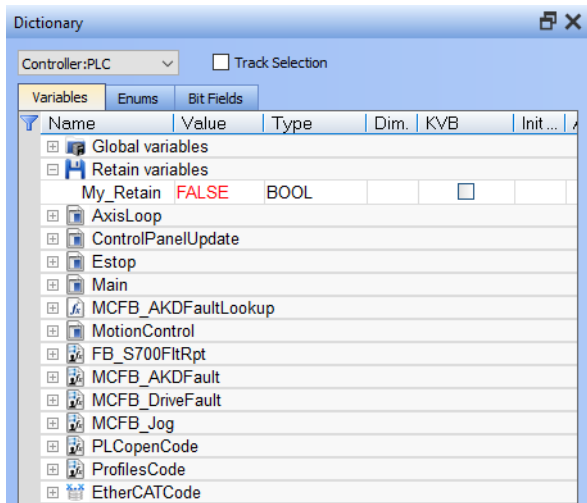
`ActualPositionX` is the intermediate variable and `SavedPositionX` is the retain variable.



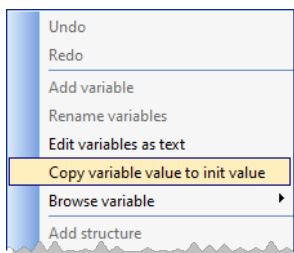
**8.1.3.3.2 Copy a Retained Variable Value to Init Value**

Once determined, retained variable values can be saved to the variable's initial value. This allows for the KAS project to more easily be re-used for other, similar machines. In the following example we will determine a retain variable's value and save the value to the retain variable's initial value.

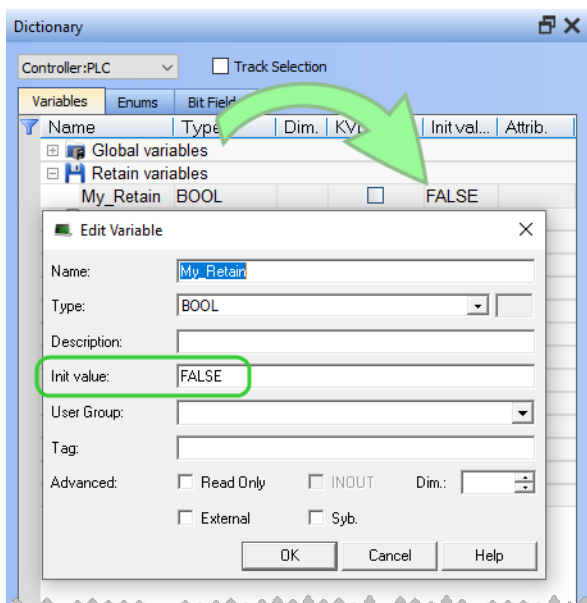
1. Run the program to determine the retain variable's value, which is FALSE.



2. Right-click on the variable and select **Copy variable value to init value**.



3. We see that the Init value field is now "FALSE".



### 8.1.3.3.2.1 Limitations

The copy function has several limitations

- The menu item is only enabled under these conditions:
  - The project must be compiled and running, and the KAS IDE is connected to the controller.
  - One or more retain variables must be selected (an individual variable or a group).
- The program must be recompiled and downloaded for this to take effect.
- Arrays and structures are not supported.

### 8.1.4 Constant Expressions

Constant expressions can be used in all languages for assigning a variable with a value. All constant expressions have a well-defined [data type](#) according to their semantics. If you program an operation between variables and constant expressions having inconsistent data types, it will lead to syntactic errors when the program is compiled. Below are the syntactic rules for constant expressions according to possible data types:

Type	Prefix	Description
<b>BOOL</b>		<b>Boolean.</b> There are only two possible Boolean constant expressions. They are reserved keywords <code>TRUE</code> and <code>FALSE</code> .
<b>SINT</b>	<code>SINT#</code>	<b>Small (8 bit) Integer.</b> Small integer constant expressions are valid integer values (between -128 and 127) and must be prefixed with <code>SINT#</code> . All integer expressions having no prefix are considered as <a href="#">DINT</a> integers.
<b>USINT/BYTE</b>	<code>USINT#</code>	<b>Unsigned 8 bit Integer.</b> Unsigned small integer constant expressions are valid integer values (between 0 and 255) and must be prefixed with <code>USINT#</code> . All integer expressions having no prefix are considered as <a href="#">DINT</a> integers.
<b>INT</b>	<code>INT#</code>	<b>16 bit Integer.</b> 16 bit integer constant expressions are valid integer values (between -32768 and 32767) and must be prefixed with <code>INT#</code> . All integer expressions having no prefix are considered as <a href="#">DINT</a> integers.
<b>UINT/WORD</b>	<code>UINT#</code>	<b>Unsigned 16 bit Integer.</b> Unsigned 16 bit integer constant expressions are valid integer values (between 0 and +65535) and must be prefixed with <code>UINT#</code> . All integer expressions having no prefix are considered as <a href="#">DINT</a> integers.
<b>DINT</b>		<b>32 bit (default) Integer.</b> 32 bit integer constant expressions must be valid numbers between -2147483648 to +2147483647. <a href="#">DINT</a> is the default size for integers: such constant expressions do not need any prefix.  You can use <code>2#</code> , <code>8#</code> or <code>16#</code> prefixes to specify an integer in binary, octal or hexadecimal basis respectively.
<b>UDINT/DWORD</b>	<code>UDINT#</code>	<b>Unsigned 32 bit Integer.</b> Unsigned 32 bit integer constant expressions are valid integer values (between 0 and 4294967295) and must be prefixed with <code>UDINT#</code> . All integer expressions having no prefix are considered as <a href="#">DINT</a> integers.
<b>LINT</b>	<code>LINT#</code>	<b>Long (64 bit) Integer.</b> Long integer constant expressions are valid integer values and must be prefixed with <code>LINT#</code> . All integer expressions having no prefix are considered as <a href="#">DINT</a> integers.
<b>ULINT/LWORD</b>	<code>ULINT#</code>	Unsigned 64-bit integer constant expressions are valid integer values. All integer expressions having no prefix are considered as <a href="#">DINT</a> integers.

Type	Prefix	Description																		
REAL		<p><b>Single precision floating point value.</b> Real constant expressions must be valid numbers, and must include a dot ("."). If you need to enter a real expression having an integer value, add ".0" at the end of the number. You can use "F" or "E" separators for specifying the exponent in case of a scientific representation. REAL is the default precision for floating points: such expressions do not require a prefix.</p> <p><i>REAL is restrictive, but because it is the default, it is recommended to explicitly declare your real constants with the LREAL# prefix.</i></p> <p>REAL constants are limited to 6-7 digits of accuracy. Any digits after these significant digits will be lost, leading to a loss of precision.</p>																		
LREAL	LREAL#	<p><b>Double precision floating point value.</b> Real constant expressions must be valid numbers, must include a dot ("."). If you need to enter a real expression having an integer value, add ".0" at the end of the number. You can use "F" or "E" separators for specifying the exponent in case of a scientific representation.</p> <p>LREAL constants are limited to 14-15 digits of accuracy. Any digits after these significant digits will be lost, leading to a loss of precision.</p>																		
TIME	T# or TIME#	<p><b>Time of day.</b> Time-constant expressions represent durations that must be less than 24 hours. Expressions must be prefixed by either TIME# or T#. They are expressed as a number of hours followed by "h", a number of minutes followed by "m", a number of seconds followed by "s", and a number of milliseconds followed by "ms".</p> <ul style="list-style-type: none"> <li>• The order of units (hour, minutes, seconds, milliseconds) must be respected.</li> <li>• You cannot insert blank characters in the time expression.</li> <li>• There must be at least one valid unit letter in the expression.</li> </ul>																		
STRING		<p><b>Character string.</b> String expressions must be written between single quote marks. The length of the string cannot exceed 255 characters. You can use the following sequences to represent a special or not-printable character within a string:</p> <table border="1"> <thead> <tr> <th>Sequence</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>\$\$</td> <td>a "\$" character</td> </tr> <tr> <td>\$'</td> <td>a single quote</td> </tr> <tr> <td>\$T</td> <td>a tab stop (ASCII code 9)</td> </tr> <tr> <td>\$R</td> <td>a carriage return character (ASCII code 13)</td> </tr> <tr> <td>\$L</td> <td>a line feed character (ASCII code 10)</td> </tr> <tr> <td>\$N</td> <td>carriage return plus line feed characters (ASCII codes 13 and 10)</td> </tr> <tr> <td>\$P</td> <td>a page break character (ASCII code 12)</td> </tr> <tr> <td>\$xx</td> <td>any character (xx is the ASCII code expressed on two hexadecimal digits)</td> </tr> </tbody> </table>	Sequence	Description	\$\$	a "\$" character	\$'	a single quote	\$T	a tab stop (ASCII code 9)	\$R	a carriage return character (ASCII code 13)	\$L	a line feed character (ASCII code 10)	\$N	carriage return plus line feed characters (ASCII codes 13 and 10)	\$P	a page break character (ASCII code 12)	\$xx	any character (xx is the ASCII code expressed on two hexadecimal digits)
Sequence	Description																			
\$\$	a "\$" character																			
\$'	a single quote																			
\$T	a tab stop (ASCII code 9)																			
\$R	a carriage return character (ASCII code 13)																			
\$L	a line feed character (ASCII code 10)																			
\$N	carriage return plus line feed characters (ASCII codes 13 and 10)																			
\$P	a page break character (ASCII code 12)																			
\$xx	any character (xx is the ASCII code expressed on two hexadecimal digits)																			

Table 5-1: List of Prefixes for Constant expressions

### 8.1.4.1 Examples

Below are some examples of valid constant expressions:

TRUE	TRUE Boolean expression
FALSE	FALSE Boolean expression
SINT#127	small integer
INT#2000	16 bit integer
123456	DINT (32 bit) integer
16#abcd	DINT integer in hexadecimal basis
8#34712	DINT integer in octal basis
2#1000100	DINT integer in binary basis
LINT#1	long (64 bit) integer having the value "1"
0.0	0 expressed as a REAL number
1.002E3	1002 expressed as a REAL number in scientist format
LREAL#1E-200	Double precision real number
T#23h59m59s999ms	maximum TIME value
TIME#0s	null TIME value
T#1h123ms	TIME value with some units missing
'hello'	character string
'name\$Tage'	character string with two words separated by a tab
'I\$m here'	character string with a quote inside (I'm here)
'x\$00y'	character string with two characters separated by a null character (ASCII code 0)

Below are some examples of typical errors in constant expressions

<b>BooVar := 1;</b>	0 and 1 cannot be used for Booleans
<b>1a2b</b>	basis prefix ("16#") omitted
<b>1E-200</b>	"LREAL#" prefix omitted for a double precision float
<b>T#12</b>	Time unit missing
<b>'I'm here'</b>	quote within a string with "\$" mark omitted
<b>hello</b>	quotes omitted around a character string

Additionally, there are pre-defined constants. See [Use the Defines List](#) for information about [Internal](#) and [user-defined](#) Defines.

### 8.1.5 Program Organization Units

Within IEC 61131-3, the [Functions](#), [Function Blocks](#), and [Programs](#) are called Program Organization Units (POU).

In addition to the IEC standard, you can write your own code: sub-program or UDFB.

Types	IEC 61131-3	Written by end-user
Basic functions (has no memory)	<a href="#">Functions</a>	<a href="#">Programs / Sub-programs</a>
Instantiated functions (keep track of the past)	<a href="#">Function Blocks (FB)</a>	<a href="#">User-Defined Function Blocks (UDFB)</a>

#### 8.1.5.1 Difference between Functions and Function Blocks

- Functions are expected to complete in one cycle
- Function Blocks can take several cycles to complete

##### 8.1.5.1.1 Description of FB operation

Rather than halt the application, waiting for operations to complete, the FB typically gives control back to the application but does not set its **Done** output.

##### 8.1.5.1.2 Examples of Operations Overrunning the Cycle Duration

- A motion command to move from one location to another can take several cycles to complete.
- Same for operations like reading/writing to files or reading and writing over TCP/IP can also take several cycles to complete.

##### 8.1.5.1.3 Operation Sequence

1. When a FB is called, it starts an operation and possibly does not complete it
2. The FB is called in the next cycle, and it checks to determine if the operation is done
3. If it is done, it sets the **Done** output. If not, it continues on
4. Now the application knows that the operation is complete and can do whatever other processing it needs based on the FB being done

#### 8.1.5.2 Functions

IEC has defined standard functions and also allows you to create your own functions (called user-defined functions). Typically, functions take several inputs and return a single output as the result of processing.

- Standard functions are for example ADD (addition), ABS (absolute), SIN (sine), COS (cosine), GT (Greater Than),....
- User-defined functions, as in the following example, can be used repeatedly once defined.

```
FUNCTION SIMPLE_FUN : REAL
  VAR_INPUT
    A, B : REAL;
    C : REAL := 1.0;
  END_VAR
  SIMPLE_FUN := A*B/C;
END FUNCTION
```

#### 8.1.5.3 Function Blocks

Function Blocks (FBs) take several inputs and return a group of values as the output as the result of processing.

Function Blocks are the equivalent to Integrated Circuits (IC), representing a specialized control function. They are specified at such a level that you quickly recognize the functionality of the function block and specifically what happens if it is activated or connected to other blocks in a sequence of motion commands.

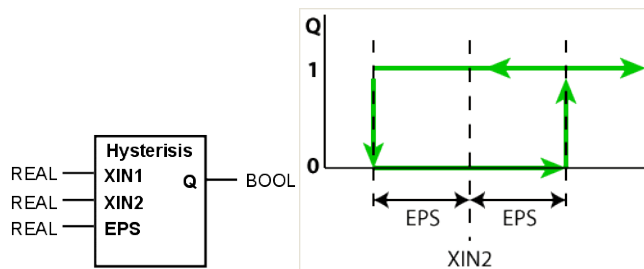
They contain data as well as an algorithm, so they can keep track of the past (which is one of the differences from Functions). They have a well-defined interface and hidden internals, like an IC or a black box. The user only sees the interface, being the inputs and outputs. The code itself is hidden.

Function Blocks can be used in any of the IEC languages. Note that in an SFC program, function blocks can be part of a step or transition created in FFLD, ST, IL and FBD.

Once defined, they can be used repeatedly, in the same program, different programs, or even different projects. This makes them highly re-usable.

There are predefined function blocks (e.g. timers, counters or triggers) and also additional function blocks that can come from libraries produced by you or other suppliers (e.g. a temperature control-loop or PID).

### Example of function blocks



The function block is based on the programming language function block Diagram and has the name Hysterisis. It has three inputs (XIN1, XIN2 and EPS) of datatype REAL on the left, and one output (called Q) of type BOOL on the right-hand side.

#### **TIP**

Input names are not very usable. Please use meaningful names.

Internally, the FB contains the following body code:

```
FUNCTION_BLOCK HYSTERISIS
VAR_INPUT
    XIN1, XIN2 : REAL;
    EPS : REAL; (* Hysterisis ban *)
END_VAR
VAR_OUTPUT
    Q : BOOK := 0
END_VAR
IF Q THEN
    IF XIN1 < ( XIN2 - EPS ) THEN
        Q := 0 (* XIN2 decreasing *)
    END_IF;
    ELSEIF XIN1 > ( XIN2 + EPS ) THEN
        Q := 1; (* XIN2 increasing *)
    END_IF;
END_FUNCTION_BLOCK
```

In this example, the body code is written in the Structured Text language:



- The first part deals with the data structure
- The second with the algorithm
- No additional data is used.

Whatever name was used for this local data inside the body, it does not conflict with matching names in other functions, function blocks, or with global expressions. This example of data encapsulation removes a major source of errors.

#### 8.1.5.4 Programs

With the above-mentioned basic building blocks, a program can be seen as a network of functions and function blocks. Each of them being written in any of the defined programming languages.

##### TIP

Programs and Subprograms can be reordered by drag-and-drop. Dragging an item up the list places it above the program you drop it on, while dragging down places it below.

##### 8.1.5.4.1 Sub-programs

The list of programs is completed with "Sub-programs". Sub-programs are written in FBD, FFLD, ST or IL languages, and can be called by the programs of the application. Input and output parameters plus local variables of a sub-program are declared in the variable editor as local variables of the sub-program.

- A function type sub-program can call another function type subprogram.
- A UDFB type sub-program can call another UDFB type subprogram or a function type subprogram

Unlike UDFB, local variables of a sub-program are not instantiated. This means that the sub-program always works on the same set of local variables. Local variables of a sub-program keep their value among various calls. The code of a sub-program is not duplicated when called several times by parent programs.

A sub-program cannot have more than 32 input parameters or 32 output parameters.

A good programming practice is to break up your programs into smaller modules.

See also [Application Software Structure - Definitions](#).

##### 8.1.5.4.2 Program Guidelines

An application is a list of programs. Programs are executed sequentially within the target cycle, according to the following model:

```

Begin cycle
| exchange I/Os
| execute first program
| ...
| execute last program
| wait for cycle time to be elapsed
End Cycle

```

Programs are executed according to the order defined by the user. All SFC programs must be grouped (it is not possible to insert a program in FBD, FFLD, ST or IL between two SFC programs). Each program is entered using a language chosen when the program is created. Possible languages are Sequential Function Chart (SFC), Function Block Diagram (FBD), Free Form Ladder Diagram (FFLD), Structured Text (ST) or Instruction List (IL). The maximum number of programs in an application is 32767.

##### 8.1.5.4.2.1 Program Name

Programs must have unique names. The name cannot be a reserved keyword of the programming languages and cannot have the same name as a standard or "C" Function or function block. A program must not have the same name as a declared variable. The name of a program must begin by a letter or an underscore ("\_") mark, followed by letters, digits or underscore marks. It is not allowed to put two consecutive underscores within a name. Naming is case-insensitive. Two names with different cases are considered as the same.

#### 8.1.5.4.2.2 Child SFC Programs

You can define a hierarchy of SFC programs, entered as a tree in the list of programs. A child program is controlled within action blocks of the parent SFC program.

#### TIP

Even if you do not want to split your FFLD program, at least separate FFLD from SFC. Simply make a sub-program in FFLD called from the SFC step, and keep only the state machine in the SFC program. This makes everything simpler and more comfortable for editing and debugging.

#### 8.1.5.4.3 Program Limitations

When creating your application you have to consider the following important limitations.

For **SFC** programs:

- Actions in SFC steps cannot be more than 32kB
- Condition in SFC transition cannot exceed 32kB
- Total PLC code size of the program cannot exceed 64kB

For **FFLD** programs:

- Width of any network is limited to 255 columns
- Height of any network is limited to 255 rows

For any program, sub-program or UDFB written in **other** languages:

- Jump limit is 64kB  
For example, in a Free Form Ladder program, if you create a UDFB or program which is over 64kB and then decide to add a jump to label in the first network to the last network, this jump reaches the limit.
- Total PLC code size of the program, sub-program or UDFB cannot exceed 64kB

#### 8.1.5.5 User-Defined Function Blocks

The list of programs is completed with "User-Defined Function Blocks" (UDFBs). UDFBs are described using SFC, FBD, FFLD, ST or IL languages, and can be used as other function blocks in the programs of the application. Input and output parameters plus private variables of a UDFB are declared in the variable editor as local variables of the UDFB.

There is no restriction using any operation in a UDFB. A UDFB can call standard functions and function blocks.

A UDFB can call another UDFB. Note that the called UDFB must be declared before the calling one in the program list.

Each time a UDFB is instantiated, its private variables are duplicated for the declared instance. The code of the UDFB is duplicated on each call in parent programs. This leads to higher performances at run-time, but consumes code space. It is recommended to package small algorithms in UDFBs. Large parts of code must be managed in programs.

### 8.1.6 Programming Languages

Within the IEC 11631 standard, syntax and semantics of the programming languages have been defined, leaving no room for variance. Once you have learned them, you can use a wide variety of systems based on this standard.

The languages consist of two textual and three graphical versions:

#### Textual:

- Instruction List (IL)
- Structured Text (ST)

#### Graphical

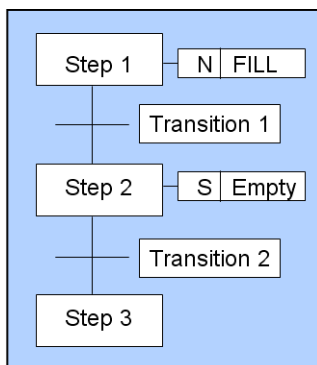
- Sequential Function Chart (SFC)
- Free Form Ladder Diagram (FFLD)
- Function Block Diagram (FBD)

All five languages are interlinked: they provide a common suite. The choice of programming language depends on:

- the programmer's background
- the problem at hand
- the level of describing the problem
- the structure of the control system
- the interface to other people / departments

#### 8.1.6.1 Sequential Function Chart (SFC)

SFC describes graphically the sequential behavior of a control program. It is derived from Petri Nets.



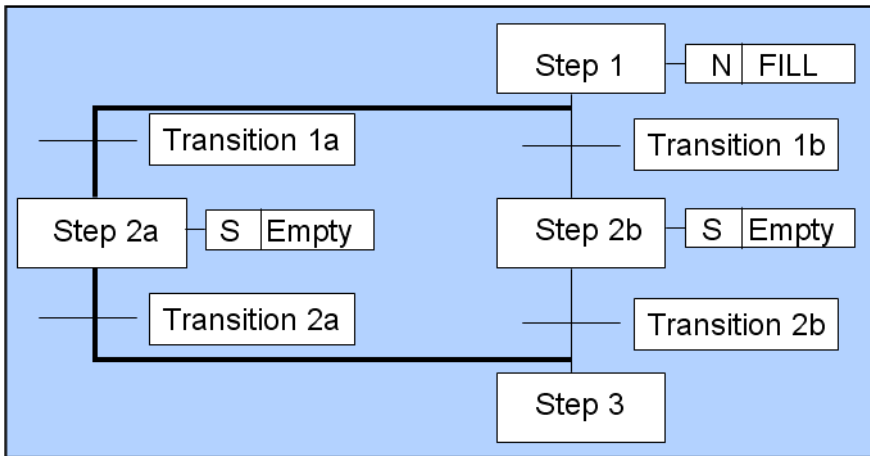
SFC organizes the internal structure of a program, and helps to deconstruct a control problem into manageable parts, while maintaining the overview.

SFC consists of steps, linked with Action Blocks and Transitions. Each step represents a particular state of the systems being controlled. A transition is associated with a condition, which, when true, causes the step before the transition to be deactivated, and the next step to be activated. Steps are linked to action blocks, performing a specific control action. Each element can be programmed in any of the IEC languages, including SFC itself.

#### Alternative and Parallel Sequences

You can use alternative sequences and even parallel sequences, like those commonly required in batch applications. For example, one sequence is used for the primary process, and the second for monitoring the overall operating constraints.

As shown in the following picture, parallel sequences are also possible:



**Figure 5-1:** Example of a Parallel Sequence in SFC

From step 1, it either goes to step 2a or step 2b, depending on which of the transition conditions is met. Both conditions need to exclude each other.

#### 8.1.6.2 Structured Text (ST)

ST is a very powerful high-level language with its roots in ADA, Pascal and "C". It contains all the essential elements of a modern programming language, including selection branches (IF-THEN-ELSE and CASE OF) and iteration loops (FOR, WHILE and REPEAT). These elements can also be nested. It can be used for the definition of complex function blocks, which can be used within any of the other languages.

#### 8.1.6.3 Function Block Diagram (FBD)

FBD is very common to the process industry. It expresses the behavior of functions, function blocks and programs as a set of interconnected graphical blocks, as in electronic circuit diagrams. It looks at a system in terms of the flow of signals between processing elements.

#### 8.1.6.4 Free Form Ladder Diagram (FFLD)

FFLD is based on the graphical presentation of Relay Ladder Logic.

#### 8.1.6.5 Instruction List (IL)

IL is the European counterpart of FFLD. As textual language, it looks like Assembler.

### 8.1.7 Alias Definitions

The compiler supports the definition of aliases (see usage in [Use the Defines List](#)).

An alias is a unique identifier that can be used in programs to replace another text. Definitions are typically used to replace a constant expression and facilitate the maintenance of programs.

There are three levels of definitions:

- Common to all the projects present on your machine
- Global to all programs within your project
- Local to one program

Local definitions are edited together with the corresponding program.

Definitions are entered in a text editor. Each definition must be entered on one line of text according to the following syntax:

```
#define Identifier Equivalence (* comments *)
```

Below are some examples:

```
#define OFF FALSE (* redefinition of FALSE constant *)
#define PI 3.14 (* numerical constant *)
#define ALARM (bLevel > 100) (* complex expression *)
```

You can use a definition within the contents of another definition. The definition used in the other one must be declared first. Below is an example:

```
#define PI 3.14
#define TWOPI (PI * 2.0)
```

Note that a definition can be empty, for example:

```
#define CONDITION
```

The defined word can be used for directing the [conditional compiling](#) directives.

#### TIP

You can enter `#define` lines directly in the source code of programs in IL or ST languages.

The use of definitions can disturb the program monitoring and make error reports more complex. It is recommended to restrict the use of definitions to simple expressions that do not risk creating a misunderstanding when reading or debugging a program.

### 8.1.8 Handling Exceptions

The compiler enables you to write your own exception programs for handling particular system events. The following exceptions can be handled:

- [Startup Exceptions](#) (before the first cycle)
- [Shutdown Exceptions](#) (after the last cycle)
- [Division By Zero Exceptions](#)

#### NOTE

Exception handling is supported for both Normal and Optimized [PLC code generation](#).

In the sub-program that handles the exception you can perform any safety or trace operation. You then have the selection between the following possibilities:

- Return without any special call. In that case the standard handling will be performed: a system error message is generated, the result of the division is replaced by a maximum value and the application continues.
- Call the FatalStop function. The runtime then stops immediately in Fatal Error mode.
- Call the CycleStop function. The runtime finishes the current program and then turns in "cycle setting" mode.

Handlers can also be used in DEBUG mode for tracking the bad operation. Just put a breakpoint in your handler. When stopped, the call stack will show you the location of the division in the source code of the program.

#### 8.1.8.1 Startup Exceptions

You can write your own exception program to be executed before the first application cycle is executed:

1. Create a new main program that will handle the exception. It cannot be an SFC program.
2. In the editor of [global defines](#), insert the following line:

```
#OnStartup ProgramName
```

#### NOTE

**Warning:** The program is executed before all other programs within the first cycle. This implies that the cycle timing can be longer during the first cycle. You cannot put breakpoints in the Startup program.

#### 8.1.8.2 Shutdown Exceptions

You can write your own exception program to be executed after the last application cycle when the runtime system is cleanly stopped:

1. Create a new main program that will handle the exception. It cannot be an SFC program.
2. In the editor of [global defines](#), insert the following line:

```
#OnShutdown ProgramName
```

#### NOTE

**Warning:** You cannot put breakpoints in the Shutdown program.

#### 8.1.8.3 Division By Zero Exceptions

You can write your own exception program for handling the "Division by zero" exception. Below is the procedure you must follow for setting an exception handler:

1. Create a new sub-program without any parameter that will handle the exception
2. In the editor of [global defines](#), insert the following line:

```
#OnDivZero SubProgramName
```

## 8.2 Motion Concepts

### 8.2.1 Introducing Motion

#### 8.2.1.1 Motion Control Main Functions

To ensure accurate positioning and movement, motion control consists of the two following main parts:

- Setpoint generation
- Regulation

#### Setpoint generation

This consists of generating a trajectory defined by **position versus time**. It is purely logical and does not relate to the physical world.

#### Regulation

Even using the very best drives, you cannot maintain accurate positioning without a feedback loop. The regulation consists of following the generated position settings using classical feed-forward or **feedback** control-loops (by means of PID). Regulation is the part which takes care of the physical world of making moving motors.

These two functions can be located on the same hardware (as in a "stand-alone" servo drive) or on two separate hardware devices, linked together by a fieldbus.

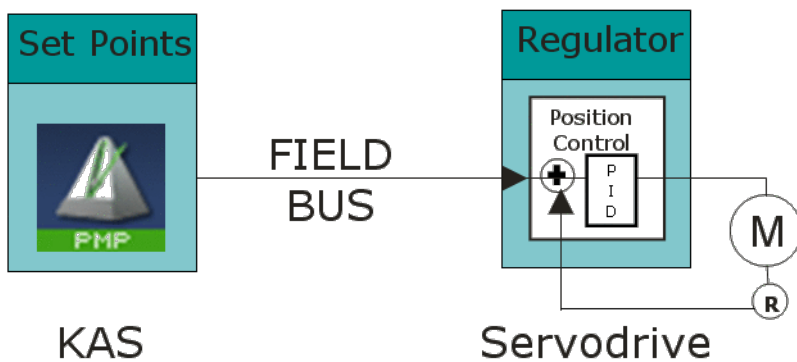


Figure 5-2: Regulation with Remote Drive

#### 8.2.1.2 Single and Multi-Axis Motion

In **Single-Axis**, as shown in the figures above, one setpoint generator is linked to one axis.

**Multi-Axis** motion consists of synchronizing several axes linked to a common motion source. This source can be external, like a physical motor (called master) or an internal profile generator (called virtual master) as shown in **Multi-Axis Driven by a Virtual Master** below.

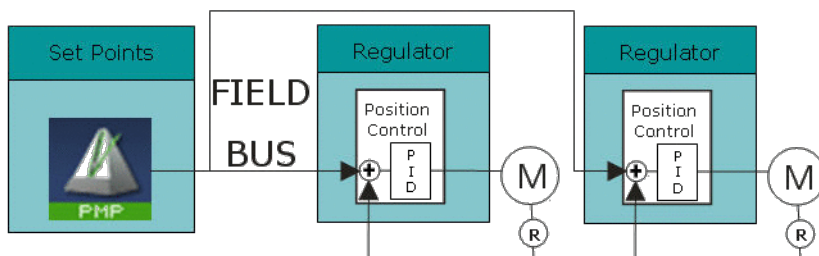


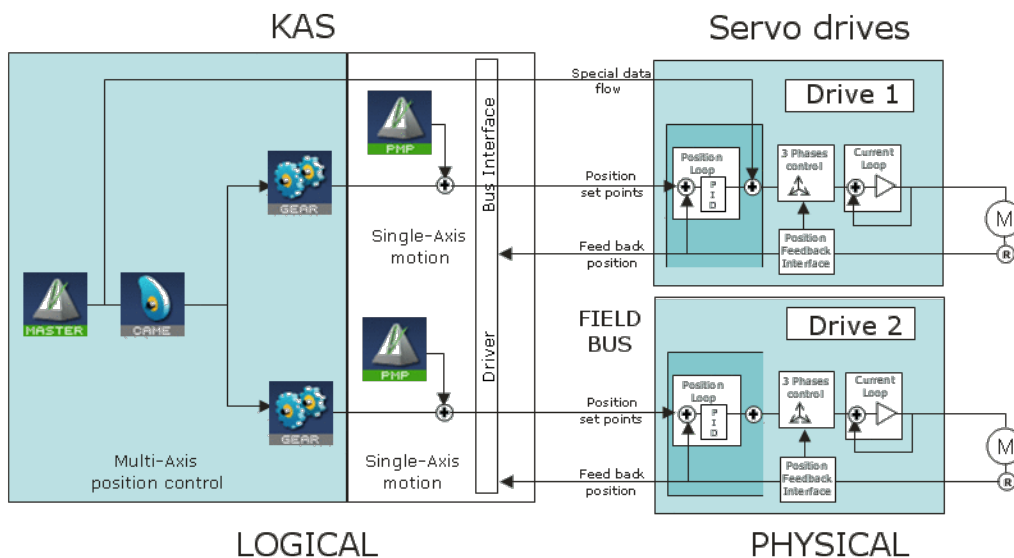
Figure 5-3: Multi-Axis Driven by a Virtual Master

#### 8.2.1.3 Hardware Organization of Motion Functions



A complete motion control "chain" is made of two main parts that can be subdivided into several more basic functionalities. Depending on your hardware system configuration, each of these elementary functions can theoretically be embedded in different hardware modules.

One of the possible configurations is represented in the figure below.



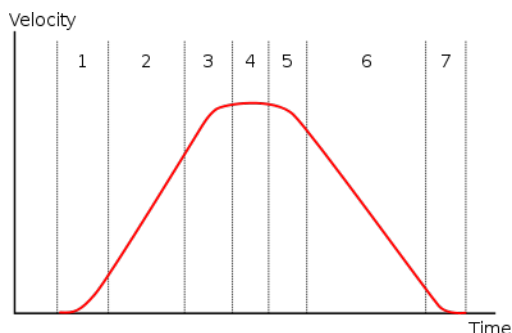
**Figure 5-4:** Hardware Organization of Motion Functions

The scope of Kollmorgen Automation Suite is to manage all the logical parts of the motion control and to ignore the physical aspects (which are handled by the hardware). To make the link between the logical and physical worlds, KAS includes some components that acts as interface.

Therefore, we do no longer consider regulation and the physical world in the following paragraphs. Only setpoint generation are taken into account.

### 8.2.1.4 Motion Profile

In motion control, a common need is to move a system from one steady position to another (point-to-point motion). Following the fastest possible motion within an allowed maximum value for speed, acceleration, and jerk, results in a third-order motion profile as illustrated below:



**Figure 5-5:** Third-order motion profile

The motion profile consists of up to seven phases defined by the following:

- acceleration increase, with maximum positive jerk
- constant maximum acceleration (zero jerk)
- acceleration decrease, approaching the desired maximum velocity, with maximum negative jerk
- constant maximum speed (zero jerk, zero acceleration)
- deceleration increase, approaching the desired deceleration, with maximum negative jerk

- constant maximum deceleration (zero jerk)
- deceleration decrease, approaching the desired position at zero velocity, with maximum positive jerk

If the initial and final positions are sufficiently close together, the maximum acceleration or maximum velocity may never be reached.

### 8.2.1.5 KAS Touch Probe & Fast Input Overview

This page provides links to KAS documentation sections which are helpful in configuring and using the Touch Probe feature, also known as "fast inputs" or "capture engine".

#### 8.2.1.5.1 Configuration

##### 8.2.1.5.1.1 Drive Configuration

- [AKD: AKD Support With MC\\_TouchProbe](#)
- [AKD2G: AKD2G Support With MC\\_TouchProbe](#)
- **EtherCAT Configuration for Drives:** The correct input PDOs must be enabled for touch probes to work correctly. New KAS projects enable the PDOs needed for most touch probe applications. Some applications need special configurations. Those configurations can be changed here:  
[PDO Selection/Mapping Tab](#)

##### 8.2.1.5.1.2 PipeNetwork Configuration

- [MLAxisCfgFastIn](#)
- [MLTrigSetEdge](#)

##### 8.2.1.5.1.3 PLCopen Configuration

- [MC\\_TouchProbe](#)
- [MC\\_Reference](#)
- [MC\\_MachRegist](#)

#### 8.2.1.5.2 List of Function Blocks and Pipe Blocks

##### 8.2.1.5.2.1 PLCopen Function Blocks

I/O

- [MC\\_TouchProbe](#)

Reference

- [MC\\_Reference](#)

Registration

- [MC\\_MachRegist](#)
- [MC\\_MarkRegist](#)
- [MC\\_StopRegist](#)

##### 8.2.1.5.2.2 PipeNetwork Blocks

[Motion Library - Axis](#)

- [MLAxisCfgFastIn](#)
- [MLAxisIsTriggered](#)
- [MLAxisRstFastIn](#)

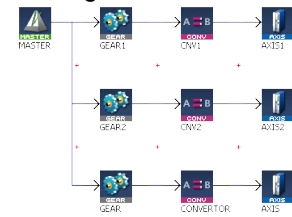
[Motion Library - Trigger](#)

- [MLTrigClearFlag](#)
- [MLTrigInit](#)
- [MLTrigIsTriggered](#)
- [MLTrigReadDelay](#)
- [MLTrigReadPos](#)
- [MLTrigReadTime](#)
- [MLTrigSetEdge](#)
- [MLTrigWriteDelay](#)

### 8.2.2 PipeNetwork or PLCopen

Using KAS there are two ways to generate motion functions and motion profiles: with PipeNetwork or PLCopen.

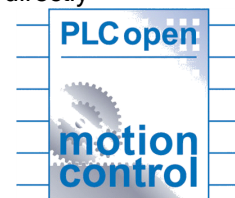
**Pipe Network:** The PipeNetwork enables you to create a high-performance motion algorithm which is tightly integrated to the PLC program with motion library function blocks. For high performance, complex, or synchronized multi-axis applications, the pipe concept in KAS provides a simple conversion of mechanical applications into a graphical representation of application elements and the process flow. This format makes it easy to understand, program, and update the motion profiles and positional relationships.



The KAS application begins with the creation of a PipeNetwork structure linking Master objects (source) to Axes objects (destination) and includes the definition of specific transformer motion profiles. This structure is then controlled from the PLC application using dedicated function blocks in the Motion Library.

To be able to use pipes correctly, it is necessary to first consider some [definitions](#).

**PLCopen :** (see [PLCopen Web site](#)) Standard function blocks can be used and directly incorporated into the PLC application. Programming of motion is done using standard MC function blocks that can be incorporated in [single-axis](#) or [multi-axis](#) applications.



#### 8.2.2.1 Motion Engine Differences of PipeNetwork and PLCopen

The following table outlines some of the main feature differences between the PipeNetwork and the PLCopen motion engines. It also provides their associated function blocks.

Topic	PipeNetwork	PLCopen
<b>Function block format</b>	Begins with <b>ML</b> ex: MLAxisRel	Begins with <b>MC_</b> ex: MC_MoveRelative
<b>Does Function block requires instantiation?</b>	No. Except for MLAxisStop	Most require it
<b>Method to start execution</b>	Most are level triggered	Most are edge triggered
<b>Motion execution status, for function block executing motion</b>	Use MLMotionStatus function block	Each function block includes a standard set of outputs for motion status
<b>Function block standard input format</b>	Requires additional function blocks to define motion parameters (speed, accel, decel, etc.)	Includes standard set of inputs to define motion (speed, accel, decel, etc.)
<b>Axis setup method</b>	Includes in the PipeNetwork Axis block <a href="#">properties</a>	Part of Axis definition screen in the Project tree
<b>How the Axis name is setup?</b>	Automatically done as part of PipeNetwork Axis block <a href="#">properties</a>	Create an instance of a Axis_Ref variable structure in the dictionary, then assign an axis number to it in a PLC program (for procedure, <a href="#">Create PLCopen Axis</a> )

Topic	PipeNetwork	PLCopen
<b>Is there additional motion editor?</b>	Yes (PipeNetwork editor)	No
<b>Motion buffering</b>	Execution of multiple motion commands in a row is handled by the programmer	Function blocks have built in buffering modes
<b>Motion jerk reduction</b>	Primarily available by adding cams to the PipeNetwork	Function blocks have jerk reduction input

Table 5-2: Differences between the PipeNetwork and PLCopen

### 8.2.3 PipeNetwork Concept

To introduce the PipeNetwork concept, we can use a mechanical analogy.

In the figure below, the mechanical system is composed of three-axes and driven by one motor. All axes are connected to the motor through shafts, gears and cams. When the motor is in motion, all axes are moving synchronously. The speed relation between the Master and the Axis is achieved by using a mechanical Gear. A mechanical cam is used to get linear motion from a rotating wheel.

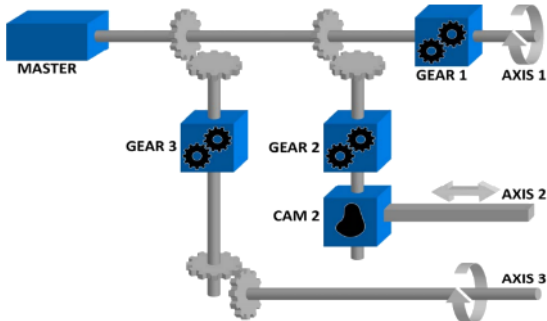


Figure 5-6: Mechanical System

The PipeNetwork in the figure below corresponds to the mechanical system described above. The pipe concept is a one-to-one translation of a mechanical system into the logical world.

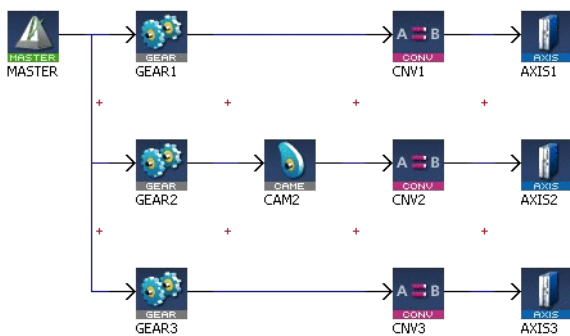


Figure 5-7: Pipe Network Structure

In our PipeNetwork, the analogy is as follows:

- The main motor of the mechanical machine becomes a Virtual Master Pipe Block
- The gear boxes becomes Gear Blocks
- The mechanical cam becomes a Cam Block
- The axes becomes Axis Blocks

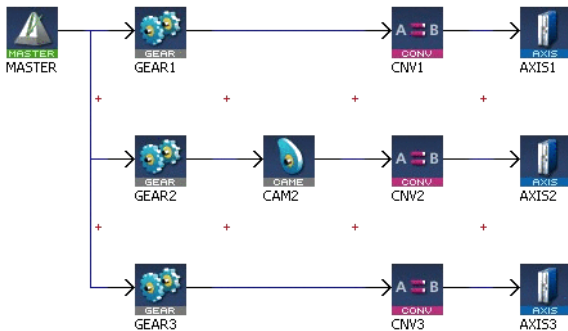
The PipeNetwork concept allows motion engineers to define in a very natural way the physical relationships between the different axes of their machine.

This powerful modular approach provides a solution for almost any multi-axis requirements. It also remains open for new, additional functions that can be required in the future.

### 8.2.3.1 PipeNetwork

To control the machine application with multi axes that are dynamically interconnected, you can design several pipes with the KAS IDE to create the global Pipe Network as shown in [Pipe Network Structure](#).

Relationships between the Axes are developed and connected graphically, allowing you to visualize how the machine functions. Each horizontal flow is considered as a separate pipe. In the application below there are three pipes.



- The Pipe Network can be edited at any time.

#### NOTE

You do not have to finalize the Pipe Network before writing a PLC program, but you must compile your project to have the latest Pipe Network information available in the PLC program editor.

- Program code does not have to be written when setting up the foundation of a program, as the parameters are entered into set-up screens.

#### IMPORTANT

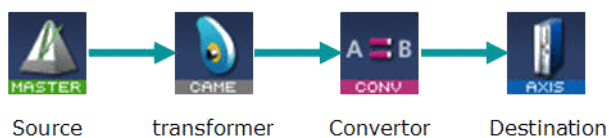
Pipe Network code is generated automatically by the compiler, you should not try to modify it.

- In the programs, you can define activation or deactivation statements to install or remove pipes and Pipe Blocks. This allows the dynamic adjustment of the machine behavior depending on the result.
- The Pipe Network is used for more than just coordinated motion. It contains a full library of single-axis motion commands for sections of an application where an axis operates independently.

### 8.2.3.2 Pipe Structure

A pipe is a set of Pipe Blocks linked together (where position flows from one Pipe Block to the next). The general structure of a pipe is quite simple:

1. Start with an input Pipe Block (source)
2. Optionally followed by transformer Pipe Blocks
3. Followed by an output Pipe Block (converter)
4. Finish with the destination Pipe Block



**Figure 5-8:** Typical Pipe Structure

#### **TIP**

To avoid jerk in the pipe network (which ultimately may cause a jerk in motor motion when a cam block is applied to the upstream pipe network positions) the potential position offset between the cam's first point and the input to the cam block must be taken care of in the application program by setting a cam offset or another method.

#### **NOTE**

More about the different kinds of Pipe Blocks are discussed in [Adding Motion](#).



### 8.2.3.3 Pipe Block

Pipes are built using logical entities called Pipe Blocks. A Pipe Block is an object whose purpose is to modify a flow of values with strict time constraints. Pipe Blocks normally have both input and output flows of values. Based on their functions, there are four kinds of Pipe Blocks:

Function	Description
Input (source)	Works as generator of values: <ul style="list-style-type: none"> <li>sample external source objects or create a discrete flow of values as an input to the pipe</li> </ul>
Transformer	<ul style="list-style-type: none"> <li>apply a specific algorithm to the input value to produce their output (transformations can be linear or complex: e.g. cam)</li> <li>can create events depending on the incoming values</li> </ul>
Output (convertor)	Block that can end a pipe: <ul style="list-style-type: none"> <li>convert the incoming values from user units to correct system units for the destination objects</li> </ul>
Destination	Simply models a physical axis of the machine

The following table provides a short description of each Pipe Block:

Function	Pipe Block	Description
Input	Master	Virtual master generating values (position) at each cycle
Input	Sampler	Samples external value (encoder, resolver, PLC variable etc.)
<b>Transformation</b>		
Mathematical	Derivator	Applies a derivation on the input data flow
Mathematical	Integrator	Integrates the input data flow
Mathematical	Adder	Adds two data flows
Event-driven	Synchronizer	Starts and stops a sub-pipe in a controlled way
Event-driven	Delay	Delay the data flow during some cycles
Event-driven	Comparator	Monitor the input data flow and detects the crossing of a particular value
Event-driven	Trigger	Computes the local pipe value from the timestamp of a Fast Input event
Modification	Cam	Applies a cam table (also called Cam Profile) to the input data flow
Modification	Gear	Applies a gearing ration on the input data flow
Modification	Phaser	Applies a phase offset to the input
Output	Convertor	Converts input data flow to a position and forwards it to an axis
Destination	Axis	Models a physical axis

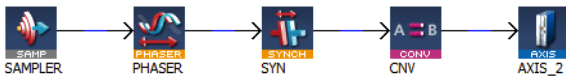
Table 5-3: Pipe Network - List of Pipe Blocks

#### 8.2.3.3.1 Master



Use a Master Pipe Block to create a virtual master to link two or more axes. The Profile generator in the Master block is trapezoidal. If a parabolic type profile is required, use a PMP Pipe Block. If the master is an external encoder or another axis, use the Sampler Pipe Block.

### 8.2.3.3.2 Sampler



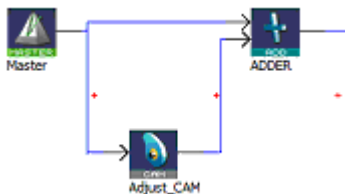
Use a Sampler Pipe Block to read an external encoder as an input signal into the Pipe Network or to directly read the actual position of another axis.

### 8.2.3.3.3 Gear

Use a Gearing Pipe Block to perform electronic gearing. The Gear Pipe Block allows gear ratios and the slope of the gear change to be initially set, then changed from within the application program.

### 8.2.3.3.4 Cam

Use a Cam Pipe Block to optimize the motion profile. Use an Adder block with a Cam block to dynamically change the distance moved during each period (or modulo) of motion.

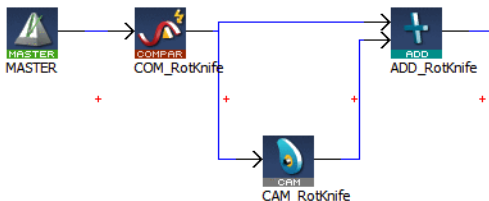


Cam Profiles are created using the [cam creation tool](#).

### 8.2.3.3.5 Comparator

By tracking the position at one point of the Pipe Network, you can use a Comparator Pipe Block to synchronize when code is executed in a PLC application program.

The following example shows the changing of the offset move by changing the amplitude (or offset) of the Cam Pipe Block.



In a PLC application program, the MLCompWriteRef function block is used to arm the comparator block and MLCompCheck function block is used to check the position. By using condition statements in a user program, specific actions (such as changing the move distance of the offset) can then be taken.

Another example shows the use of a Comparator Pipe Block to determine if a high-speed input is within the acceptable position range.

### 8.2.3.3.6 Trigger



Use a Trigger Pipe Block to read the position when a high-speed input is triggered on the machine. The trigger block allows you to “catch” the position at a particular location in the Pipe Network, as required by the application.

### 8.2.3.3.7 Delay

Use a Delay Pipe Block to delay the flow of position through a Pipe Network. One potential use is to place it before a Trigger block in a pipe which is not connected to a drive. There is a delay of five servo update cycles between the dynamic position in the Pipe Network and the triggering of a high-speed digital input.

#### 8.2.3.3.8 Phaser

Use a Phaser Pipe Block to perform a dynamic phase adjustment inside the Pipe Network. This block can be used to phase-advance or phase-retard a position as required to synchronize different motion elements on a machine.

#### 8.2.3.3.9 Synchronizer

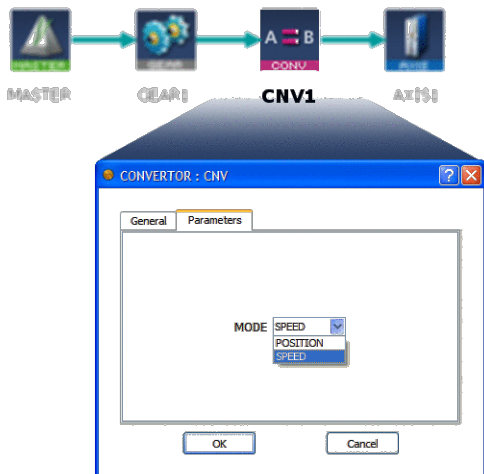
Use a Synchronizer Pipe Block to synchronize two axes. This Pipe Block is useful in applications where it is necessary to start the motion of a second axis and sync to the first.

#### 8.2.3.3.10 Axis

Models the link from the Pipe Network to a physical axis.

#### 8.2.3.3.11 Changing Information Flow from Position to Velocity

You can change the Pipe Network flow of information from position to velocity by using the **Convertor** Pipe Block. This Pipe Block is normally set up to receive position, so it must be changed to receive the expected input signal type as shown below:



Change the mode of Converter block to SPEED (and not POSITION mode).

### 8.2.3.4 Axis Pipe Block

The Axis pipe block is the link between the logical and physical worlds, managing position data for a physical or simulated axis.

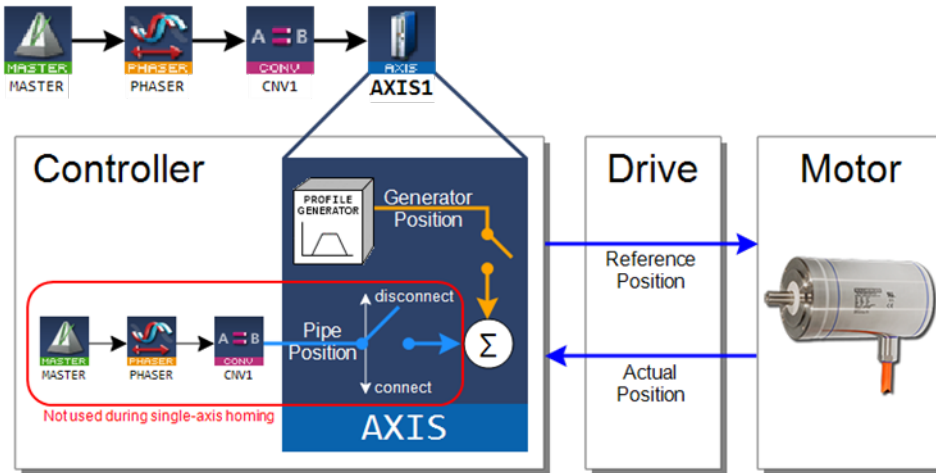


Figure 5-9: Axis Pipe Block Positions

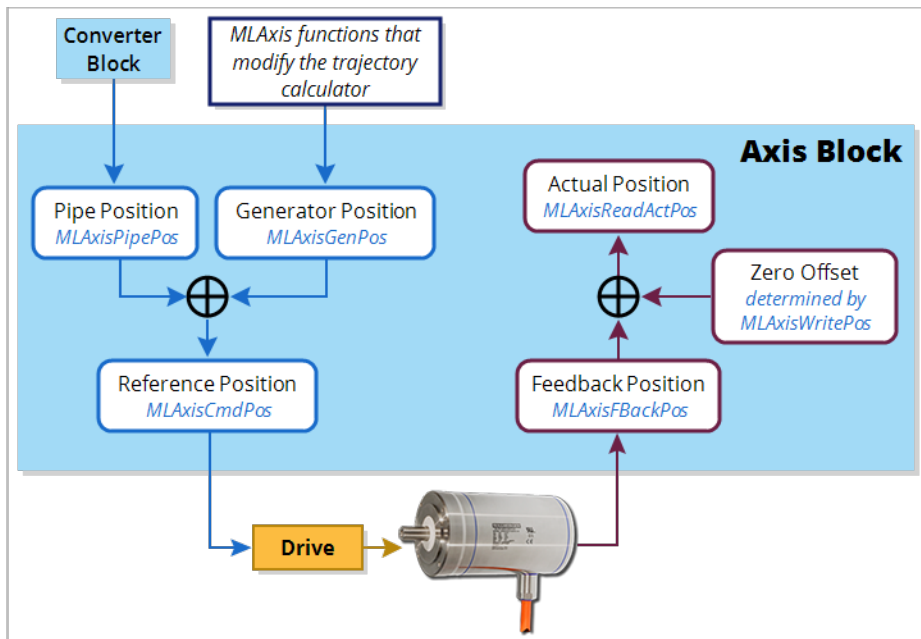
#### 8.2.3.4.1 About Associated Data on Positions

The following data are illustrated in the figure below.

**NOTE**

All positions are in user units with modulo applied if active, unless specified.

Position / Offset	Description
<b>Actual Position</b>	<b>Actual Position</b> refers to the actual position of the underlying axis as reported by the drive. It is the sum of the feedback value (Position actual value) returned from the communication link to the drive and any zero-offset due to an MLWritePos function block ( <a href="#">MLAxisWritePipPos</a> , <a href="#">MLAxisWritePos</a> ).
$\text{ActualPos} := \text{FeedbackPos} + \text{ZeroOffset}$	
<b>Feedback Position</b>	<b>Feedback Position</b> is the current position the drive reports for an axis, scaled to user units. It does not take into account the value of the Zero Offset or axis modulo.
<b>Generator Position</b>	<b>Generator Position</b> is the summation of all previous commands (i.e. calls to functions which perform motion such as <a href="#">MLAxisAbs</a> , <a href="#">MLAxisMoveVel</a> , and <a href="#">MLAxisRel</a> ) to the Axis internal motion generator. It is modified by <a href="#">MLAxisWritePos</a> in order to insure no jumps in the <b>Reference Position</b> command. It also accumulates changes in pipe position due to activate and deactivation of the pipe the Axis block is associated with.
<b>Pipe Position</b>	The output of the convertor block is written into the <b>Pipe Position</b> value whenever the <a href="#">Convertor block</a> is connected to the axis and the pipe is active.
<b>Reference Position</b>	<b>Reference Position</b> is the commanded axis position sent to the drive. It is the summation of <b>Pipe Position</b> and <b>Generator Position</b> .
$\text{ReferencePosition} = \text{Pipe Position} + \text{Generator Position}$	
<b>Zero Offset</b>	The <b>Zero Offset</b> adjusts the coordinate system so that the Actual Position reports correct values after homing or using <a href="#">MLAxisWritePos</a> .



### 8.2.3.4.2 About Units

User Units (abbreviated UU) are the dimensions that will be used in the application code to define axis position or distance for an axis.

The "user units per revolution" field is used to set how many of these custom units are associated with one revolution of the motor. Some examples for this field are:

User Units	User Units per Revolution Value	Notes
Degrees	360.0	
Gradians	400.0	
Radians	3.14159265358979324	
Inches	15.7079632679489662	This equals pi*5. This could represent the circumference of a 5 inch diameter drum.

KAS standard units for velocity and acceleration are UU/sec and UU/sec<sup>2</sup>.

### 8.2.3.4.3 Functions That View Axis Block Positions

<a href="#">MLAxisReadActPos</a>	Returns the Actual Position
<a href="#">MLAxisCmdPos</a>	Returns the Reference Position
<a href="#">MLAxisFBackPos</a>	Returns the Feedback Position
<a href="#">MLAxisGenPos</a>	Returns the Generator Position
<a href="#">MLAxisPipePos</a>	Returns the PipePosition

### 8.2.3.4.4 Functions That Change Axis Positions

<a href="#">MLAxisAdd</a>	This function adds a relative distance to the current target Generator Position.
<a href="#">MLAxisRel</a>	This function adds a relative distance to the current Generator Position.
<a href="#">MLAxisAbs</a>	This function sets a new target Generator Position.
<a href="#">MLAxisMoveVel</a>	This function sets the generator position moving at a programmed speed.

<a href="#">MLAxisStop</a>	This function stops any current Generator Motion. It also causes the axis to start ignoring any changes in Pipe position to be added into the reference position. It decelerates, if moving, at a programmed rate.
<a href="#">MLAxisReAlign</a>	Causes the Axis to move by a programmed amount without changing the Reference Position following an <a href="#">MLAxisStop</a> . Also allows the Pipe Position to be used following an <a href="#">MLAxisStop</a> .
<a href="#">MLAxisWritePos</a>	If convertor is not connected, Zero Pipe Position and Pipe Offset. If convertor is connected (pipe active also), the pipe position and offset are left alone. The actual position is then set equal to the target position, and the Zero Offset is adjusted for no motion. The Reference position and Generator Positions are then realigned so that the new reference position creates no step in motion, with the lag between reference position and actual position being absorbed in the generator position.
<a href="#">MLAxisWritePipPos</a>	Changes the pipe position to be the new value. May cause a jump in motion.
<a href="#">MLCNVConnect</a>	Initializes the pipe position to the Convertor block output value and adjusts the axis Pipe Offset so that no jump in motion is generated.
<a href="#">MLCNVDisconnect</a>	Stops sending the convertor output to the Pipe Position, and disconnects the convertor from the axis.
<a href="#">MLPipeAct</a>	Starts calculating Pipe data, If the convertor block is connected to the axis it will reconnect the convertor and start updating the pipe position.
<a href="#">MLPipeDeact</a>	Stops sending Pipe data to the Axis Block Pipe Position and disconnects the convertor output from the axis.

#### 8.2.3.4.5 Axis Block Initialization

A call to the [MLAxisInit](#) function block is required to implement motion for the axis.

- All positions and offsets are set to zero
- The Axis Block motion generator is initialized with the proper ranges
- The values are "aligned": ReferencePosition = Pipe Position + Generator Position

#### 8.2.3.4.6 Axis Connection to a Pipe

A call to the [MLCNVConnect](#) function block is required to get motion generated in the pipe to the Axis

- Pipe Offset is calculated as follows: Pipe Offset = Pipe Position – Reference Position
- The values are "aligned": Reference Position = Pipe Position + Generator Position

#### 8.2.3.4.7 Realigning Positions

A call to the [MLAxisReAlign](#) function block is used to realign the axis after an error occurs

- Motion must come to a stop first
- The [MLAxisReAlign](#) is executed  
You must set the movement of this block to [MLAxisReadActPos](#) - [MLAxisCmdPos](#)
- The target position must be reached before any additional motion can occur.  
It can be checked by using the [MLAxisReAlignRdy](#) function block

#### 8.2.3.4.8 Set Zero Axis

A call to the [MLAxisWritePos](#) function block is used to set a position offset at the Axis when the Pipe Network is not yet connected

- Pipe Position and Pipe Offset are set to zero
- Generator Position is set to equal to Zero Position

- (Zero Position is defined in [MLAxisWritePos](#) function block)
- Then Reference Position equals Pipe Position + Generator Position

#### 8.2.3.4.9 Homing

Homing is the process of moving the motor to a known physical reference point on the machine.

Homing can be performed with the **MLFB\_Home\*** [Kollmorgen UDFBs](#) or by adding custom application code that uses only the [Motion Library - Axis](#) function blocks. The PipeNetwork is typically not used in homing.

Custom homing is done with [MLAxisRel](#) and [MLAxisAbs](#) to make motion and [MLAxisWritePos](#) to set a position offset.

#### 8.2.3.4.10 Single-Axis Operation

This includes motion done on an individual axis block: jogging, absolute move, or incremental moves. If these are single-axis based, then motion is executed with the [MLAxisMoveVel](#), [MLAxisAbs](#), and [MLAxisRel](#) FBs. These motions are typically done during machine setup or adjustment and are often referred to as manual mode. For these operations, the Pipe Network does not need to be connected to the axis.

#### 8.2.3.4.11 Multi-Axis Operation

For multi-axis applications, automatic operation requires motion synchronization between two or more axes and the Pipe Network is required to achieve the synchronization. To start up the Pipe Network the following two function calls must be executed in an application program:

```
PipeNetwork (MLPN_ACTIVATE) ;
PipeNetwork (MLPN_CONNECT) ;
```

Multi-axis synchronized motion is then accomplished using a motion block associated with one of the three input Pipe Blocks:

- Master: [MLMstRun](#), [MLMstRel](#), and [MLMstAbs](#)
- PMP: [MLPmpAbs](#), [MLPmpRel](#)
- Sampler: [MLSmpConnect](#), [MLSmpConECAT](#), [MLSmpConPNAxis](#), [MLSmpConPLCAxis](#)

#### 8.2.3.4.12 Monitoring an Axis

There are function blocks to monitor the performance and status of an axis. The key function blocks are as follows:

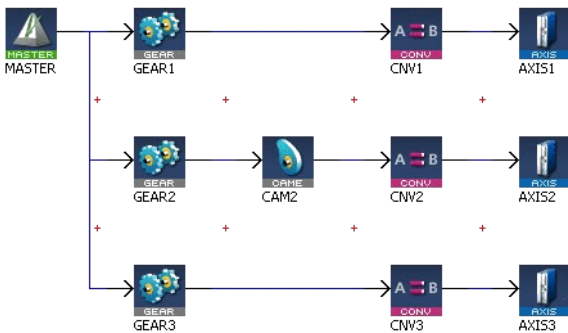
- [MLAxisCmdPos](#) - The commanded position to an axis.
- [MLAxisReadActPos](#) - The actual position of the axis.
- [MLAxisStatus](#) - The status of the axis. This includes information on the enabled/disabled state, bus connection, PipeNetwork connection, the drive executing an axis stop function, as well as other status information.
- [MLAxisReadGenStatus](#) - The status of the Axis generator: acceleration, run, deceleration, change designation point, single step.
- [MLAxisGenIsRdy](#) - Specifies whether the Axis generator is ready.

### 8.2.3.5 Executing Motion

Two types of Pipe Blocks are used to command motion in a Pipe Network: Axis block and Input block.

- Axis block starts motion directly on one axis.
- Input blocks start motion that affect all axes that are connected downstream in a Pipe Network. Input blocks can be one of three types:
  - **Master** - Trapezoidal motion
  - **PMP** - Parabolic Motion
  - **Sampler** - Externally generated motion from another axis or external encoder

In the following example, executing **MLAxisMoveVel**, **MLAxisAbs** and **MLAxisRel** Functions can be used to cause motion on a particular axis. Whereas **MLMstRun**, **MLMstAbs** and **MLMstRel** functions cause motion on Axis1, Axis2 and Axis3.



For information on **error management**, see [Error Management](#).

For explanations on **restarting the motion**, refer to [Restarting Motion with PipeNetwork](#)

### 8.2.3.6 Pipe Block Lifetime

#### Activation

The pipe is activated when the output of the Converter Pipe Block is connected to its related Axis (all characteristics are reset to the declaration values and the history of the block begins).

#### Usage

As long as the pipe remains active, its values are cyclically calculated. Functions can be performed and events can be created.

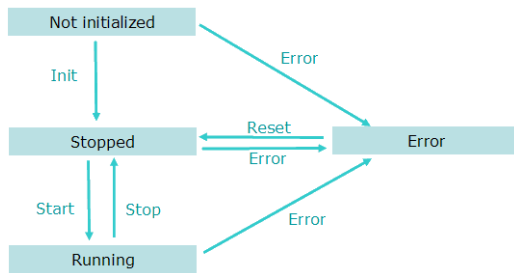
#### Deactivation

The pipe is deactivated when the deactivate function is applied to the pipe (all internal current values are lost and the block no longer exists).



### 8.2.3.7 Motion State Machine

The Motion State Machine is driven by the IEC 61131-3 application with the help of dedicated function blocks.



**Figure 5-10:** Motion State Machine

Each arrow represents a transition from one State to another.

### 8.2.3.8 Phase Execution in the Pipe Network

#### Absolute phase of execution

The absolute phase of execution of a pipe is the elapsed time between any fixed reference and the next computation for the specified pipe.

#### Relative phase of execution

The relative phase of execution between two pipes is the elapsed time between the computation of the first pipe and the second one.

The relative phase of execution between two pipes of the same Pipe Network is zero. The phase of execution between two Pipe Networks cannot be specified by the user and depend on the pipe activation time of the application execution.

### 8.2.3.9 Use Motion Function Block for Pipe Network

Use motion library function blocks in your PLC application program to interface to the Pipe Network (to see the procedure [Design Motion with Pipe Network](#)).

ML function blocks are used to:

1. Create and initialize the Pipe Network
2. Perform motion at a single-axis or multi-axis level
3. Read information from points in the Pipe Network

For example:

- Read a high-speed input position from a Trigger Pipe Block
  - Read Command or Reference position from an Axis Pipe Block
  - Determine when a position has been reached in a Comparator Pipe Block
4. Modify how the blocks work in the Pipe Network

For example:

- Change the phase offset of Phaser Pipe Block
- Change the amplitude or offset of a CAM profile
- Change the speed of a Master Pipe Block

#### 8.2.3.9.1 Buffer Mode

With the Pipe Network engine, when a motion function block is executed while another one is presently executing, there is an immediate change. That means the previous function block is aborted and the new one immediately becomes the active move and begins executing.

#### 8.2.3.9.2 Motion Init

During initialization, the IEC 61131-3 application can create (by means of the `MLMotionInit` function) the different motion objects it needs (pipes, blocks, axes):

- Pipe Create
- Profile Create

#### NOTE

When the state machine leaves the Init state, the creation of new motion objects is no longer allowed, in order to avoid memory allocation problems while running the application.

#### 8.2.3.9.3 Motion Start

The Start method (`MLMotionStart` function) initializes the motion engine, motion bus driver, and initializes EtherCAT network to operational mode. `MLMotionStop` driver deactivates the execution of the motion engine.

The function blocks [MLMotionStart](#), [MLMotionStop](#) and [MLMotionRstErr](#) can be used by the IEC 61131-3 applications to navigate between states: i.e. Not initialized, Running, Stopped and Error.

#### 8.2.3.9.4 Using the Q output of ML Function Blocks for the Pipe Network

There is a Q output on most ML function blocks. The operation of the Q output is different for different ML function blocks. The Q output can be useful in PLC application programs.

Examples:

- `MLAxisRel.Q` is set when function block successfully executes
- `MLAxisStop.Q` is set when motion is stopped (zero speed)
- `MLPrfWriteIOffset.Q` is set if cam offset has been changed to the new value

For more details on Q output, refer to [What is the difference between Q and OK?](#)

### 8.2.3.10 Function - General rules

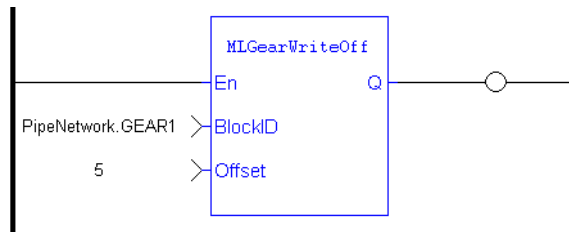
This section outlines rules for using ML function in the Pipe Network context.

#### 8.2.3.10.1 Languages

Function that interact with the Pipe Network start with ML (for example MLAxisRel, MLPrfWriteOScale, or MLMstRel). These function can be used in all four of the 61131-3 PLC languages.

#### 8.2.3.10.2 BlockID Inputs

The BlockID input is a DINT ID. It is the second input to a Pipe Network function when using FFLD:



The BlockID input is the first one if programming in Structured Text:

```
MLGearRatSlp( BlockID (*DINT*));
```

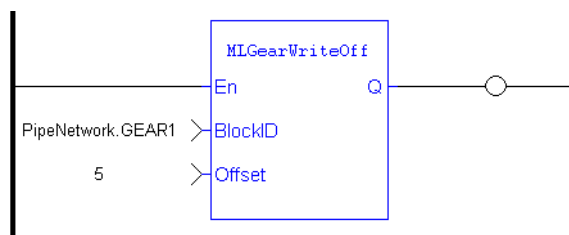
This input identifies the block in the Pipe Network that the function interacts with, and if using the graphical [Pipe Network Editor](#) the used variable starts with **PipeNetwork.xxx** (except if it is acting on a CAM profile, in which case the input is named ProfileID and the variable starts with **Profiles.xxx**).

#### **TIP**

As a general rule, when selecting a Pipe Block as the BlockID for a ML function, choose a Pipe Block with the same type which is in the name of the ML function. For example, MLMstxxx functions expect a Master block to be chosen for the BlockID; MLAxisxxx functions need an Axis block to be chosen for the AxisID input; MLPrfxxx functions need a Profile entered for ProfileID, etc.

#### 8.2.3.10.3 Output status

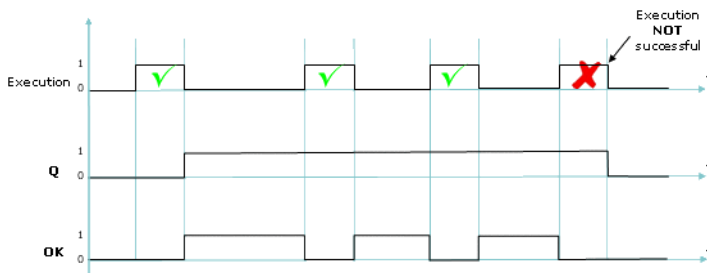
Most Pipe Network function have a default Boolean output labeled either **Q** or **OK**:



#### 8.2.3.10.4 What is the difference between Q and OK?

**OK** returns true when function successfully executes.

**Q** output is initially set to 0 until the first time the block is successfully executed in a running program. After this execution, the Q output is set to 1. It remains to 1 until the function does NOT execute successfully. Alternately, after an unsuccessful execution the Q output is set to 0. It remains to 0 until a successful execution resumes



### 8.2.3.10.5 When Q is set to True?

Some function change the Q output from low to high immediately after it starts executing, but others (including most functions that command motion) wait to change the output until the function has completely finished executing. You need to check the description for each individual function block to be sure how its Q output is behaving.

### 8.2.3.10.6 Input parameters

The En input parameter, which is used to execute the function, is not edge-triggered. If a function is seen in the PLC code and its En input is positive, the function executes. For example, a MLAxisRel command continuously executes relative moves in a program if it is called each program cycle; thus it acts as a Run/Jog command if continually commanded.

### 8.2.3.10.7 Missing input parameters

All inputs to a function must be entered in order for code to compile.

### 8.2.3.10.8 Position versus distance

**Position** is a value defined within a coordinate system.

**DeltaPosition** is a relative measure related to technical units. It is the difference between two positions.

### 8.2.3.10.9 Default Block Parameters

The parameters set when initializing a Pipe Network block are used as defaults when calling functions. These parameters can be modified in a program by using specific functions to set these values. But if a value is never set in a program the parameter entered during initialization is used.

For example:

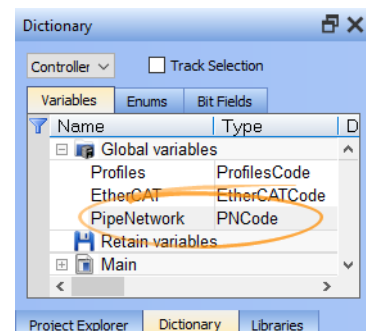
When making a Master relative move (MLMstRel) you input the DeltaPosition, but not the velocity or acceleration. You can set the velocity for the move by using the MLMstWriteSpeed function before calling the relative move. If the speed is not set in the program, the default parameter entered during the initialization (i.e. in the properties dialog box of the Pipe Blocks) is used.

### 8.2.3.11 Using the Generated PipeNetwork Function

When the [PipeNetwork Editor](#) is used, the KAS IDE automatically creates a subprogram named **PNCode** and an instance of that subprogram named **PipeNetwork**, which can be found in the Dictionary under **Global variables**.

The PipeNetwork instance contains the IDs for all the pipe blocks created in the PipeNetwork Editor. It also can execute certain actions when called as a function. These functions are:

- PipeNetwork(MLPN\_CREATE\_OBJECTS) — Creates all PipeNetwork related objects. This needs to be called first, before any other access to the PipeNetwork instance. The



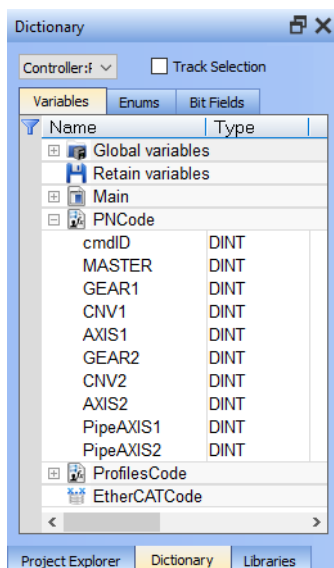
PipeNetwork template applications make this call in the **Main** program.

- PipeNetwork(MLPN\_ACTIVATE) — Activates all pipes defined in the PipeNetwork Editor by calling [MLPipeAct](#) on each pipe.
- PipeNetwork(MLPN\_DEACTIVATE) — Deactivates all pipes defined in the PipeNetwork Editor by calling [MLPipeDeact](#) on each pipe.
- PipeNetwork(MLPN\_CONNECT) — Connects all converter blocks defined in the PipeNetwork Editor to their associated axis blocks by calling [MLCNVConnect](#) on each converter block. This is typically done after all homing is completed.
- PipeNetwork(MLPN\_DISCONNECT) — Disconnects all converter blocks defined in the PipeNetwork Editor from their associated axis blocks by calling [MLCNVDisconnect](#) on each converter block. This is typically done when axes need to be re-homed.
- PipeNetwork(MLPN\_POWER\_ON) — Powers up (Enables) all axes defined in the PipeNetwork Editor, enabling any associated servo or stepper drives mapped to the axes. This is done by calling [MLAxisPower](#) on each axis block.
- PipeNetwork(MLPN\_POWER\_OFF) — Powers down (Disables) all axes defined in the PipeNetwork Editor, disabling any associated servo or stepper drives mapped to the axes. This is done by calling [MLAxisPower](#) on each axis block.

The IDs for the blocks used in the PipeNetwork Editor are needed when writing code that accesses those function blocks. For example, when you need to read the actual position for an axis, you will need to use the pipe block ID in [MLAxisReadActPos](#):

```
// Structured Text Example
// Syntax = Function Name (PN Instance Name.Pipe Block ID)
MLAxisReadActPos (PipeNetwork.AXIS1);
```

The IDs for the pipe blocks can be found in the dictionary under the **PNCode** subprogram as seen below, but are accessed through the PipeNetwork instance.



## 8.2.4 PLCopen®

The Kollmorgen Automation Suite supports the International [PLCopen®](#) motion standard.



The PLCopen international standard was created to obtain uniformity of motion function blocks and motion startup between machine control products. The PLCopen function blocks for Motion Control, is based on IEC 61131-3 [Function Blocks](#) concept with the following factors in consideration:

- Simplicity - ease of use for the application program builder and installation & maintenance
- Efficiency - in the number of function blocks, for efficiency in design (and understanding)
- Consistency - conforming to the IEC 61131-3 standard
- Universality - hardware independent
- Flexibility - future extensions / range of applications
- Completeness

KAS supports PLCopen motion in the following program formats: FFLD, SFC, ST,IL and FBD. PLCopen blocks in KAS start with "MC" (example: MC\_MoveAbsolute). MC blocks are an alternative to using the ML Motion function blocks (example: MLAxisAbs) and associated Pipe Networks in many applications. Using MC Motion function blocks does not require a separate motion editor. Users who are familiar with PLCopen are automatically familiar with PLCopen inside the KAS IDE.

### 8.2.4.1 PLCopen Function Blocks

The following function block (FB) library is designed for the purpose of controlling one or more servo axes using the IEC 61131-3 [PLCopen](#) standard (for more details on FB, refer to [Function Blocks](#)).

To offer flexibility, ease of use and reusability, the library consists of command-oriented function blocks that have a reference to the axis, e.g. the abstract data type [Axis](#)

The PLCopen Library contains function blocks for:

- **Control:** function blocks to define and initialize motion, control power, and reset errors
- **I/O:** function blocks to control interaction with Digital I/O and Touch Probe and trigger [registration](#) functionality
- **Info:** function blocks to provide information on motion, position, status, and the ability to read and write other drive parameters
- **PLCopen Motion:** function blocks to execute different types of motion
- **Profile:** function blocks for master/slave motion
- **Reference:** function blocks to reset position
- **Registration:** function blocks to perform registration

<a href="#">MC_MoveAbsolute</a>	performs a single-axis move to a specified endpoint position.
<a href="#">MC_MoveRelative</a>	performs a single-axis move of a specified distance relative to the actual position at the time of the start of execution.

<b>MC_MoveAdditive</b>	commands a controlled motion of a specified relative distance. Can also be used to interrupt a motion currently being performed. In this case the MotionAdditive FB causes the speed, acceleration, and deceleration of the motion already running to be changed to the parameters specified in the MC_MoveAdditive FB. If the MC_MoveAdditive FB is activated in Continuous Mode, the specified relative distance is added to the actual position (at the time of execution).
<b>MC_MoveSuperimposed</b>	commands a controlled motion of a specified relative distance additional to an existing motion. The existing Motion is not interrupted, but is superimposed by the additional motion.
<b>MC_MoveVelocity</b>	commands a never-ending controlled motion (jog) at a specified velocity.
<b>MC_Stop</b>	commands a controlled motion stop and transfers the axis to the "Stopping" state. It aborts any ongoing function block execution. When the Done output is set, the state transfers to StandStill. While the axis is in Stopping state, no other FB can perform any motion on the same axis.
<b>MC_Power</b>	controls the power stage: enable(on) and disable (off).
<b>MC_ReadStatus</b>	returns Axis status details with respect to the motion currently in progress.
<b>MC_ReadAxisErr</b>	indicates Drive-related errors.
<b>MC_ResetError</b>	makes the transition from the state ErrorStop to StandStill by resetting all internal axis-related errors and clearing pending commands – it does not affect the output of the FB instances.
<b>MC_ReadParameter &amp; MC_ReadBoolParameter</b>	return the value of a Drive parameter. The returned value has to be converted to Real if required. If not possible, the vendor has to provide a supplier-dependent FB for it.
<b>MC_WriteParameter &amp; MC_WriteBoolParameter</b>	modify the value of a Drive parameter.
<b>MC_ReadActualPosition</b>	returns the value of the actual position.
<b>MC_MachRegist</b>	performs Mark-to-Machine registration
<b>MC_MarkRegist</b>	performs Mark-to-Mark registration
<b>MC_StopRegist</b>	turns off registration
<b>MC_CamTblSelect</b>	selects the Cam tables by setting the pointers to the relevant tables.
<b>MC_CamIn</b>	engages the Cam.
<b>MC_CamOut</b>	disengages the slave from the master axis immediately in a cam block.
<b>MC_GearIn</b>	commands a ratio between the VELOCITY of the slave and master axis.
<b>MC_GearOut</b>	disengages the slave from the master axis.
<b>MC_AddSuperAxis</b>	adds a superimposed axis to a specified axis.
<b>MC_RemoveSuperAxis</b>	removes a superimposed axis from an axis.

## 8.2.4.2 PLCopen Function Blocks - Overview

### 8.2.4.2.1 Queuing

A queuing mechanism is provided for all PLCopen motion function blocks including single-axis and master/slave moves. This mechanism allows the application to queue a next move while the active move is executing. The buffer modes, described below, define the transition from the active move to the next move.

### 8.2.4.2.2 Buffer Modes

Some of the FBs have an input called BufferMode. With this input, the FB can either work in a Non-buffered mode (default behavior) or in a Buffered mode. The difference between those modes is when they start their action:

- A command in a non-buffered mode acts immediately, even if this interrupts another motion
- A command in a buffered mode waits until the current FB sets its **Done** output (or **InPosition**, or **InVelocity**,..).

There are six buffer modes that can be specified at the BufferMode input of the function blocks.

Buffer mode	Value	Short name	Description
MC_BUFFER_MODE_ABORT	0	Abort	A move that specifies Abort aborts the active move, removes the next move from the queue, and immediately becomes the active move and begins executing
MC_BUFFER_MODE_BUFFERED	1	Buffer	One of three events can happen with a move that specifies Buffer: <ul style="list-style-type: none"> <li>• Case 1. If there is no active move, this move immediately becomes the active move and begin executing.</li> <li>• Case 2. If there is an active move but no next move queued, this move is queued as the next move, and begins executing when the active move has completed and decelerated to zero velocity.</li> <li>• Case 3. If there is an active move and a queued next move, this move does not execute but returns the error "queue full" at the ErrorID output.</li> </ul>
MC_BUFFER_MODE_BLENDING_PREVIOUS	2	Blend to Previous	A move specifying Blend-to-Active behaves the same as Buffer in cases 1 and 3. In case 2, this move is queued as the next move. The active move stays at its programmed velocity to its endpoint. When the active move reaches its endpoint, this move becomes active and begins to accelerate or decelerate to its programmed velocity
MC_BUFFER_MODE_BLENDING_NEXT	3	Blend to Next	A move specifying Blend-to-Next behaves the same as Buffer in cases 1 and 3. In case 2, this move is queued as the next move. When the expected time is reached, the active move begins to accelerate or decelerate so that it reaches this move's programmed velocity at the time the active move reaches its endpoint
MC_BUFFER_MODE_BLENDING_LOW	4	Blend to Low	A move specifying Blend-to-Low behaves like Blend-to-Active if the active move's velocity is lower than this move's velocity. It behaves like Blend-to-Next if this move's velocity is lower than the active move's velocity



Buffer mode	Value	Short name	Description
MC_BUFFER_MODE_BLENDED_HIGH	5	Blend to High	A move specifying Blend-to-High behaves like Blend-to-Active if the active move's velocity is higher than this move's velocity. It behaves like Blend-to-Next if this move's velocity is higher than the active move's velocity.

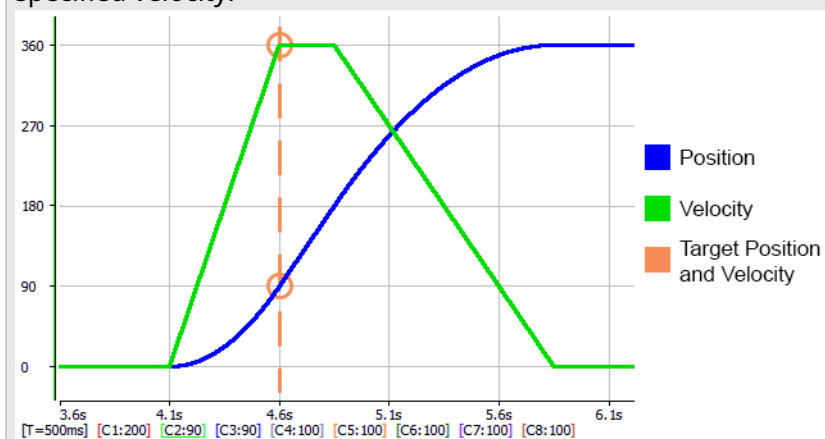
**! IMPORTANT**

The MC\_BUFFER\_MODE\_BLENDED\_NEXT and MC\_BUFFER\_MODE\_BLENDED\_HIGH buffer modes may increase the acceleration to meet the target position at the specified velocity. The acceleration will be automatically re-calculated to reach the target position at the specified velocity if the specified acceleration is too small to reach the target velocity within the specified distance.

For example, suppose the following absolute position moves are commanded:

- Move #1:  
*Position = 90, Velocity = 180, Accel = 360*
- Move #2 (with Blending High or Blending Next)  
*Position = 360, Velocity = 360, Accel = 360*

To reach velocity = 360 within a distance of 90, the acceleration must be 720. The function block will automatically calculate and apply an acceleration of 720 to reach the target position at the specified velocity:



### 8.2.4.2.3 S-curve and Trapezoidal Acceleration/Deceleration

#### 8.2.4.2.3.1 S-curve

If the Jerk input of a motion function block is non-zero, S-curve acceleration/deceleration is used. The Acceleration input specifies the maximum acceleration/deceleration reached during changes in velocity. The Deceleration input is unused. The Jerk input specifies the constant rate of change of acceleration and deceleration used to cause a smooth transition to and from maximum acceleration/deceleration.

The [Small Jerk Acceleration](#) below is a velocity plot of the acceleration of a move when Jerk is a small value. The smaller the Jerk value, the more gradual the rate of change of acceleration/deceleration when transitioning from one velocity to another.

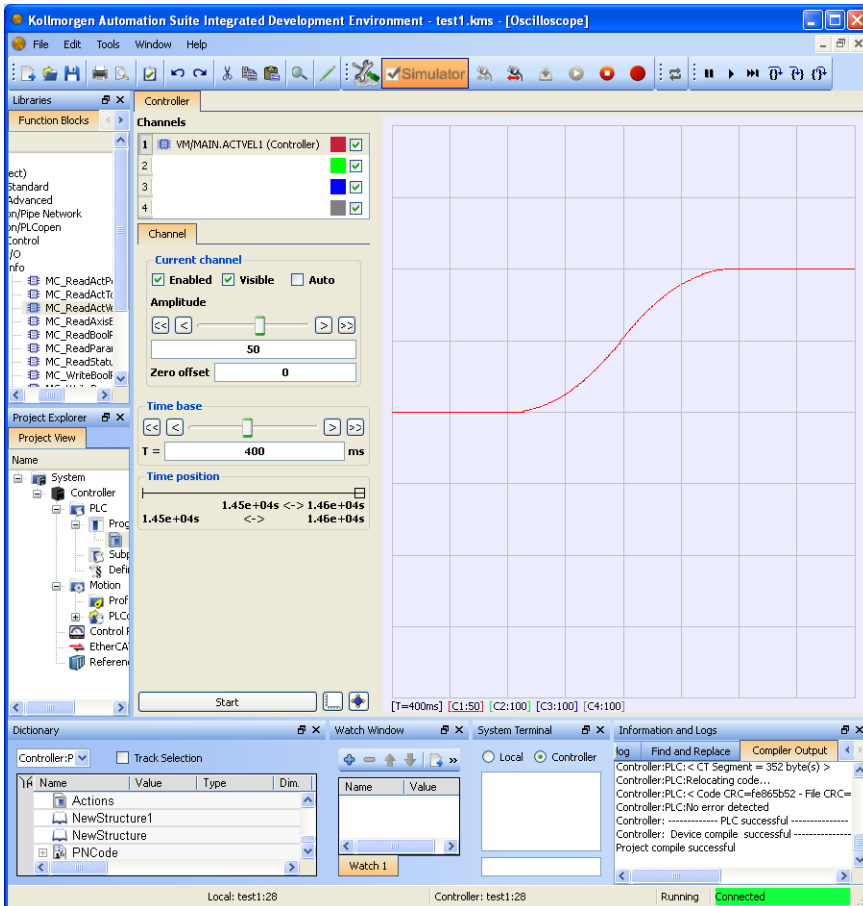


Figure 5-11: Small Jerk Acceleration

The [Large Jerk Acceleration](#) below is a velocity plot of the acceleration of a move when Jerk is a large value. The larger the Jerk value, the more abrupt the rate of change of acceleration/deceleration when transitioning from one velocity to another.

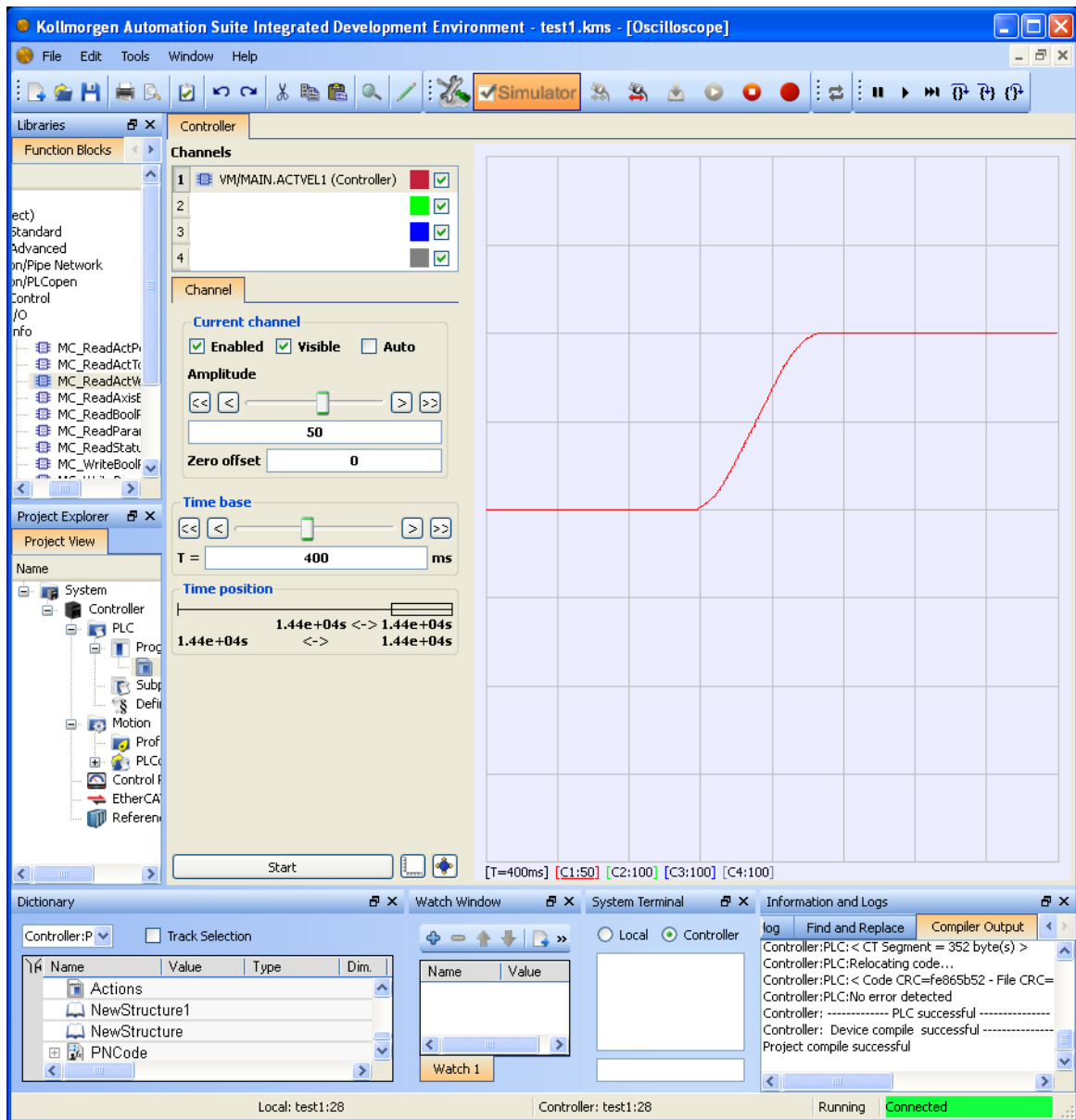


Figure 5-12: Large Jerk Acceleration

### 8.2.4.2.3.2 Trapezoidal

If the Jerk input of a motion function block is zero, trapezoidal acceleration/deceleration is used. The Acceleration input specifies the linear acceleration rate. The Deceleration input specifies the linear deceleration rate.

The **Trapezoidal Acceleration** below is a velocity plot of the acceleration of a move when trapezoidal acceleration is used (Jerk = 0).

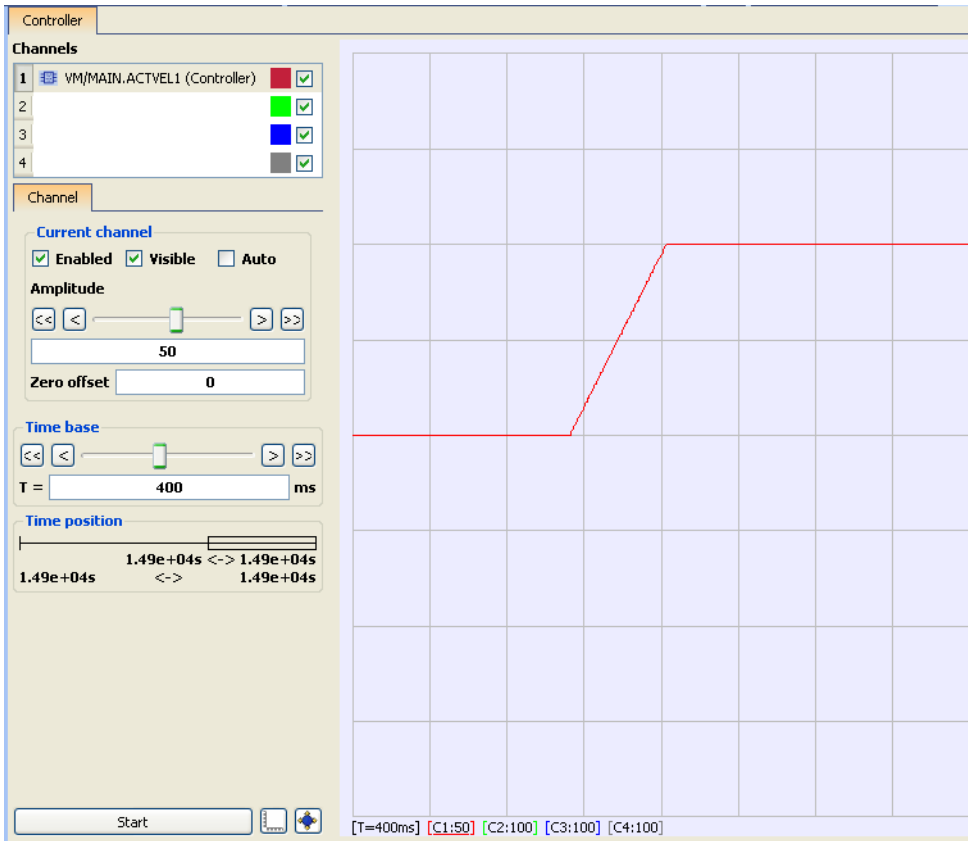


Figure 5-13: Trapezoidal Acceleration

#### 8.2.4.2.4 Selection of Acceleration and Jerk Parameters for Function Blocks

##### Definition

Acceleration is the first derivative of velocity, or the rate of change of velocity. The Acceleration rate therefore specifies how quickly an axis may change its velocity.

Jerk is the second derivative of velocity, or the rate change of acceleration. The Jerk rate therefore specifies how quickly an axis may change its acceleration. Jerk therefore controls how abrupt the axis begins and ends the acceleration and deceleration curves.

See also [Motion Profile](#).

##### Rules

The amount of time an axis takes to change its velocity from one velocity to another is a function of both Acceleration and Jerk. The larger the values of acceleration and jerk, the more quickly an axis will attain its programmed velocity. The following are generalizations that can be made about acceleration, jerk and their relationships to each other.

- The higher the acceleration rate, the faster the axis will obtain programmed velocity
- The higher the jerk rate, the more responsive the axis will be to changes in command
- Excessive jerk typically, more noticeably contributes to harsh acceleration than excessive acceleration
- Too low of a jerk value contributes to slow axis responsiveness to changing commands
- Lower jerks tend to soften the beginning and end of acceleration, while higher jerks sharpen the beginning and end of acceleration
- Typically, Jerk > Acceleration, Acceleration > Velocity

##### Methods

There are several methodologies to determine proper acceleration and jerk values. These methodologies allow you to calculate parameters given different desired profiles. Once parameters are calculated, you can then modify them as desired to obtain the results you want. Acceleration and Jerk values are subject to the limits of ratios as explained below.

1/3,1/3,1/3 time, given velocity and time. This allows you to calculate an appropriate acceleration and jerk, if you would like an axis acceleration/deceleration profile to "jerk" or ramp acceleration up for 1/3 of the time, accelerate 1/3 of the time and ramp acceleration down 1/3 of the time. Time is the desired amount of time to reach desired velocity. Note, this is the time to change velocity, not the time to complete the move.

```
Acceleration = (3 * Velocity) / (2 * time)
Jerk = 3 * Acceleration / time
```

1/3,1/3,1/3 velocity, given velocity and time. This allows you to calculate an appropriate acceleration and jerk, if you would like an axis acceleration/deceleration profile to "jerk" or ramp acceleration up for 1/3 of the velocity change, accelerate 1/3 of the velocity change and ramp acceleration down 1/3 of the velocity change. Where velocity is the desired velocity change, and time is the desired amount of time to reach the desired velocity change. Note, this is the time to change velocity, not the time to complete the move.

```
Acceleration = (5 * Velocity) / (3 * time)
Jerk = (3 * Acceleration ^2) / (2 * velocity)
```

Calculate Jerk, given Velocity, acceleration and time. If you already know the maximum acceleration of the axis, and want to simply calculate a Jerk given the velocity and time, you can use the following equation. Note, this is the time to change velocity, not the time to complete the move.

```
Jerk = (2 * Acceleration) / (time - (velocity / (2 * acceleration)))
```

#### 8.2.4.2.4.1 Limitations on Acceleration and Jerk

The ratios of Acceleration to Jerk and Velocity to Jerk are limited on most function blocks.

- The ratio of Velocity to Acceleration must be less than **20**. A value of 20 suggests a time to accelerate to velocity of approximately 20 seconds, assuming infinite jerk. As jerk is decreased, this acceleration time would be increased.
- The ratio of Acceleration to Jerk must be less than **2**. A value of 2 suggests the time to jerk to the acceleration rate is approximately 2 seconds.

#### 8.2.4.2.5 Profile Generator

Each servo axis has three Profile Generators which has its own queue. The three Profile Generators are: Normal, Superimposed, and Phasing.

- Normal handles all single-axis and master-slave moves
- Superimposed handles MC\_MoveSuperimp moves exclusively
- Phasing handles MC\_Phasing phase shifts exclusively

The three Profile Generators allow these types of moves to execute simultaneously.

#### 8.2.4.2.6 AXIS\_REF Structure

The PLCopen specification indicates a [data structure](#) to be used for identifying the axis at a function block input. AXIS\_REF contains two members:

Member	Type	Description
AXIS_NUM	INT	The axis number
AXIS_ENGINE	UINT	Reserved. Do not modify this variable or rely on its value.

For more details on Axis Number, [About Axis Name and Number](#)

You have to create and initialize this data structure in your application.

#### 8.2.4.2.7 Axis Parameters

##### Boolean Parameters

The table below is a list of Boolean parameters currently supported. These parameters are read and written by the function blocks [MC\\_ReadBoolPar](#) and [MC\\_WriteBoolPar](#).



Parameter	ID	Name	R/W	Update Rate Type	Description
MC_AXIS_PARAM_APPLY_SUPERIMPOSED_DISTANCE	1033	Apply Superimposed Distance	Read/Write	Controller	<p>If false, <code>MC_MoveAbsolute</code> does not take into account any added superimposed distance due to <code>MC_MoveSuperimp</code>. The final reported position of the axis, which does include superimposed distance, may not be equal to the <code>Position</code> input to <code>MC_MoveAbsolute</code>.</p> <p><i>Example of False</i></p> <pre> Param 1033 = False MC_ MoveSuperimp move 1      +50 MC_ MoveSuperimp move 2      -35 MC_ MoveAbsolute            125 (superimposed distance = 50- 35 = 15) End Position = 140 </pre> <p>If true, <code>MC_MoveAbsolute</code> compensates for all added superimposed distance from <code>MC_MoveSuperimp</code> calls at the time of the initial <code>MC_MoveAbsolute</code> call. The final reported position of the axis will be equal to the <code>Position</code> input to <code>MC_MoveAbsolute</code>. If <code>MC_MoveSuperimp</code> is called during <code>MC_MoveAbsolute</code>, then the final position will include the superimposed distance.</p> <p><i>Example of True</i></p> <pre> Param 1033 </pre>

Parameter	ID	Name	R/W	Update Rate Type	Description
					<pre> = True MC_ MoveSuperimp move 1      +50 MC_ MoveSuperimp move 2      -35 MC_ MoveAbsolute            125 End Position            = 125 </pre> <p>The default value for this parameter is False.</p>
MC_AXIS_PARAM_DRIVE_WARNING	1013	Drive Warning	Read Only	EtherCAT Cyclic	Drive Warning Status. <sup>1</sup>
MC_AXIS_PARAM_FI_OCCURRED	1027	Fast Input Occurred	Read Only	EtherCAT Non-Cyclic	This parameter is deprecated. The recommended replacement is to use the "Done" output of the <a href="#">MC_TouchProbe</a> function block to verify if the fast input has occurred. <i>Deprecated behavior:</i> True if a fast input occurred on either Capture Engine 0 or Engine 1. This Boolean will automatically reset after it has been read. <sup>3</sup>
MC_AXIS_PARAM_IN_POSITION	1011	Axis In-Position	Read Only	Controller	True if the axis has no active or next move queued, the command delta is 0, and the actual position is within the in-position bandwidth of the command position. False otherwise, Boolean.
MC_AXIS_PARAM_REGIST_BAD	1026	Bad Registration Mark Occurred	Read Only	EtherCAT Non-Cyclic	True indicates that a bad registration mark was encountered. This Boolean will be automatically reset after it has been read. <sup>2</sup>
MC_AXIS_PARAM_REGIST_GOOD	1025	Good Registration Mark Occurred	Read Only	EtherCAT Non-Cyclic	True indicates that a good registration mark was encountered. This Boolean will be automatically reset after it has been read. <sup>2</sup>

## Non-Boolean Parameters

The table below is a list of non-Boolean parameters currently supported. These parameters are read and written by the function blocks [MC\\_ReadParam](#) and [MC\\_WriteParam](#).

Parameter	ID	Name	R/W	Update Rate Type	Description
<b>MC_AXIS_PARAM_ACT_VEL</b>	10	Actual Velocity	Read Only	<a href="#">EtherCAT Cyclic</a>	Axis actual velocity, User unit/sec
<b>MC_AXIS_PARAM_BUS_ADDRESS</b>	1017	Drive Address	Read Only	<a href="#">Static</a>	Drive address value to be used in EtherCAT fieldbus functions as drive address. Before using in fieldbus functions, this value needs to be converted to integer by using a convert any to DINT function. <sup>1</sup>
<b>MC_AXIS_PARAM_CM_ACT_CMD_POS</b>	1029	Coordinated Motion Applied Command Position	Read Only	<a href="#">Controller</a>	Amount of motion actually applied to the PLCopen axis drive by Coordinated Motion commands.
<b>MC_AXIS_PARAM_CM_CMD_POS</b>	1030	Coordinated Motion Command Position	Read Only	<a href="#">Controller</a>	Amount of motion requested of a PLCopen axis by the Coordinated Motion commands.
<b>MC_AXIS_PARAM_CMD_POS</b>	1	Command Position	Read Only	<a href="#">Controller</a>	Axis command position – includes any command deltas from superimposed axes, user units.  If a set position is in process (due to homing or registration, there may be a 1 cycle delay.
<b>MC_AXIS_PARAM_CMD_VEL</b>	11	Command Velocity	Read Only	<a href="#">Controller</a>	Axis command velocity – includes any command deltas from superimposed axes, User unit/sec
<b>MC_AXIS_PARAM_DRIVE_AXIS_NUMBER</b>	1032	Drive Axis Number	Read Only	<a href="#">Static</a>	One-based number that specifies the axis on the drive
<b>MC_AXIS_PARAM_DRIVE_STATUS</b>	1014	Drive Status	Read Only	<a href="#">EtherCAT Cyclic</a>	Drive Status Word (Similar to MAxisStatus)  Some status bits are set only at program startup.
<b>MC_AXIS_PARAM_FEEDBACK_LAST</b>	1007	Raw Feedback	Read Only	<a href="#">EtherCAT Cyclic</a>	Raw Feedback position in user units. Keeps track of the actual position based on the physical feedback device connected to the drive and will not be offset as a result of calling <a href="#">MC_SetPos</a> .

Parameter	ID	Name	R/W	Update Rate Type	Description
MC_AXIS_PARAM_IN_POSITION_BAND	1012	Axis In-Position Bandwidth	Read/Write	Controller	The bandwidth about the command position to determine the state of the in-position flag. <sup>2</sup>
MC_AXIS_PARAM_INGEAR_BANDWIDTH	1031	"In Gear" bandwidth	Read/Write	Controller	The bandwidth about the target slave velocity in which the slave axis will lock onto the master axis and the "InGear" output will turn on for the <a href="#">MC_GearIn</a> function block; User unit/sec (Default value 0.1 User units/sec). <sup>1</sup>
MC_AXIS_PARAM_INTERP_CMD_POS	1019	Interpolated Command Position	Read Only	Controller	Command position solely from this axis's interpolator (in user units). This value does not include any command deltas from other axes that are currently superimposed upon it.
MC_AXIS_PARAM_INTERP_CMD_VEL	1020	Interpolated Command Velocity	Read Only	Controller	Command velocity solely from this axis's interpolator (in user units). This value does not include any command deltas from other axes that are currently superimposed upon it.
MC_AXIS_PARAM_MASTER_OFFSET	1002	Master Offset	Read / Write	Controller	Write: the amount to increment the master offset for an active master/slave move, user units. Read: the amount of master offset applied, user units.
MC_AXIS_PARAM_MOVE_TYPE_ACTIVE	1004	Active Move Type	Read Only	Controller	The active move type (see table in <a href="#">Move Types</a> )
MC_AXIS_PARAM_MOVE_TYPE_NEXT	1005	Next Move Type	Read Only	Controller	The queued (next) move type (see table in <a href="#">Move Types</a> )
MC_AXIS_PARAM_PHASE_SHIFT	1000	Phase Shift	Read Only	Controller	The amount of phase shift applied by MC_Phasing, in slave axis' user units
MC_AXIS_PARAM_POSITION_ERROR	1006	Position Error	Read Only	EtherCAT Cyclic	Position error in user units

Parameter	ID	Name	R/W	Update Rate Type	Description
MC_AXIS_PARAM_REGIST_BAD_CNT	1024	Number of Consecutive Bad Registration Marks	Read/Write	EtherCAT Non-Cyclic	Number of consecutive bad registration marks. This value is incremented each time a bad registration mark is encountered and automatically zeroed when a good registration mark is encountered. The ability to write this parameter is provided to allow the application to zero this value. <sup>2</sup>
MC_AXIS_PARAM_REGIST_COMP	1021	Registration Compensation	Read Only	Controller	The latest calculated registration compensation value. This value is updated each time a good registration mark is encountered. This value is in User Units.
MC_AXIS_PARAM_REGIST_DIST	1022	Distance Between the Last Two Good Registration Marks	Read Only	EtherCAT Non-Cyclic	Distance between the last two good registration marks. This value is in User Units. <sup>2</sup>
MC_AXIS_PARAM_REGIST_GOOD_CNT	1023	Number of Consecutive Good Registration Marks	Read/Write	EtherCAT Non-Cyclic	Number of consecutive good registration marks. This value is incremented each time a good registration mark is encountered and automatically zeroed when a bad registration mark is encountered. The ability to write this parameter is provided to allow the application to zero this value. <sup>2</sup>
MC_AXIS_PARAM_ROLLOVER_POSITION	1008	Rollover	Read/Write	Controller	The axis rollover position in user units. If the axis is a servo axis, this parameter can only be written when there are no moves in the queue. <sup>1</sup>
MC_AXIS_PARAM_SENSOR_DELAY	1018	Sensor Delay	Read/Write	Controller	Compensation for Physical sensor delay for MC_TouchProbe (FunctionBlock), in microseconds. <sup>1</sup>

Parameter	ID	Name	R/W	Update Rate Type	Description
MC_AXIS_PARAM_SLAVE_OFFSET	1003	Slave Offset	Read/Write	Controller	Write: the amount to increment the slave offset for an active master/slave move, user units. Read: the amount of slave offset applied, user units.
MC_AXIS_PARAM_SUPERIMPOSED_DISTANCE	1001	Superimposed Distance	Read Only	Controller	The cumulative distance traveled via MC_MoveSuperimp moves, user units
MC_AXIS_PARAM_TORQUE_ACTUAL	1016	Actual Torque	Read Only	EtherCAT Cyclic	The actual torque being delivered by the drive, expressed in thousandths of max torque. See <a href="#">Example of reading the current to units of Amps.</a>
MC_AXIS_PARAM_UU_FB_RATIO	1034	User unit to feedback unit ratio	Read Only	Static	The user units to feedback units ratio for the axis. See <a href="#">About the User Units to Feedback Units Ratio</a> below.
MC_AXIS_PARAM_UU_FB_RATIO_DEN	1028	User Unit to feedback unit ratio denominator	Read Only	Static	The denominator of the user-units-to-feedback-units ratio (UU/FBU Ratio). See <a href="#">About the User Units to Feedback Units Ratio</a> below.
MC_AXIS_PARAM_UU_FB_RATIO_NUM	1015	User Unit to feedback unit ratio numerator	Read Only	Static	The numerator of the user-units-to-feedback-units ratio (UU/FBU Ratio). See <a href="#">About the User Units to Feedback Units Ratio</a> below. <sup>1</sup>
MC_AXIS_PARAM_UU_REV_RATIO	1037	User Unit revolution ratio	Read/Write	Controller	The user units per revolution ratio for the axis. See <a href="#">Axis Data Parameters</a> for details.

**⚠ WARNING**

Do NOT change the user units per revolution ratio when the axis is enabled, geared or Cammed to another axis. Undefined behavior may result.

Parameter	ID	Name	R/W	Update Rate Type	Description
MC_AXIS_PARAM_VELCOMP_FACTOR	1009	Velocity Compensation Factor	Read/Write	Controller	The factor used to multiply the velocity compensation value to account for the number of updates of delay in transmission of the feedback value from the drive to the control. See <a href="#">Velocity Compensation</a> for more information. <sup>1 4</sup>
MC_AXIS_PARAM_VELCOMP_FILTER	1010	Velocity Compensation Filter	Read/Write	Controller	The number of updates in which to apply a change in velocity compensation. See <a href="#">Velocity Compensation</a> for more information. <sup>1 4</sup>

1. This is a configuration parameter.
2. There is some delay in acquiring fast input information from drives as well as calculating the registration marks. While the information is evaluated cyclically, there may be a few cycles between when the fast input occurs and the system records the registration marks. Please see [Tuning Controller Performance for EtherCAT Communication Latency](#) for more details.
3. There is some delay in acquiring fast input information from drives as well as calculating the fast input position. While the information is evaluated cyclically, there may be a few cycles between when the fast input occurs and the system records the fast input data. Please see [Tuning Controller Performance for EtherCAT Communication Latency](#) for more details.
4. There is some delay in acquiring the actual position values. While the information is evaluated cyclically, there may be a few cycles between when the actual position is updated in the drive and the system records the fast input data. Please see [Tuning Controller Performance for EtherCAT Communication Latency](#) for more details.

Update Rate Type	Description
EtherCAT Cyclic	Update rate depends on the EtherCAT (link to ECAT update page <a href="#">EtherCAT Master Settings Tab</a> ) and KAS application program (link to <a href="#">Define the PLC Cycle</a> ) update rates
EtherCAT Non-Cyclic	Update rate depends on the update rate of reading the parameter through Ethercat (Link to times to read non-cyclic parameters [? does the time come from ECATReadSDO FB or from the MCRReadParam DriveReadParam FB <a href="#">EtherCAT Library</a> ]) and KAS application program update rate
Controller	Update rate depends on KAS application program update rate (link to <a href="#">Define the PLC Cycle</a> )
Static	Values do not change after axis is created.

#### TIP

##### About the User Units to Feedback Units Ratio

Parameters 1015 and 1028 are set during the [MC\\_CreatePLCAxis](#) function block execution. These two parameters work together to form the User Units to Feedback Units Ratio (UU/FBU Ratio). The drive interface units are fixed by the drive and define the drive units per revolution, which is used to command the drive per the ratio.

For a drive where the drive interface units are set to 1048576 units per revolution, the value of the *UU-per-revolution* ratio can be calculated as follows:

$$\text{UU per revolution} = \text{MC\_AXIS\_PARAM\_UU\_FB\_RATIO\_NUM} * 1048576 / \text{MC\_AXIS\_PARAM\_UU\_FB\_RATIO\_DEN}$$

**Example**

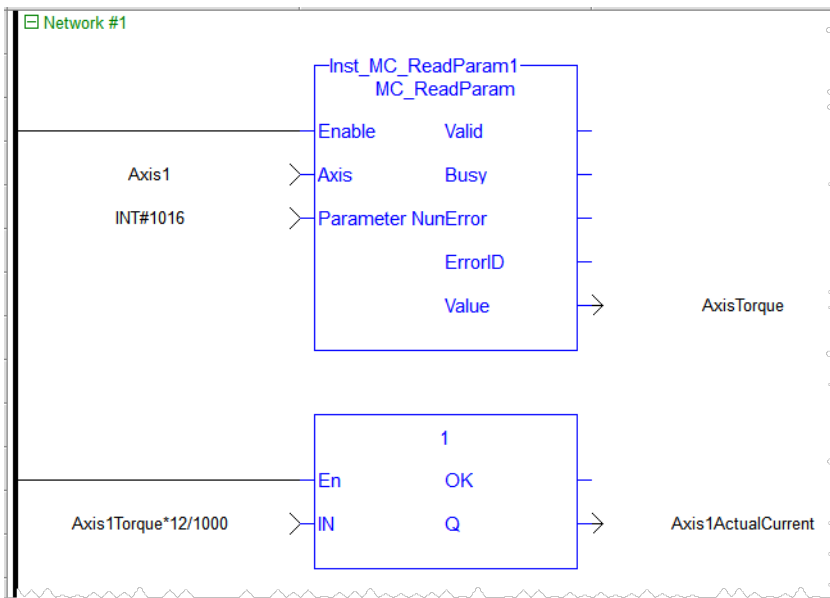
For a drive:

- A ratio of 360 UU / 1048576 FBU will generate 360 UU per revolution of the drive motor and a 0.000343323 UU-to-FBU ratio.
- A ratio of 11379 UU / 1898996404 FBU will generate 6.283185 UU per revolution of the drive motor and a 5.99211e-6 UU-to-FBU ratio.

*Example of reading the current to units of Amps.*

After reading the current via MC\_ReadParam, the following equation converts current to amps for a 12 amp peak drive:

$$\text{MC\_AXIS\_PARAM\_TORQUE\_ACTUAL} * \text{Drive Peak Current Rating} / 1000$$





### 8.2.4.2.8 Axes Group Parameters

The table below is a list of Boolean parameters currently supported. These parameters are read and written by the function blocks [MC\\_GrpReadBoolPar](#) and [MC\\_GrpWriteBoolPar](#).

Parameter	ID	Name	R/W	Description
MC_GRP_PARAM_IGNORE_AXIS_ESTOP	1000	Ignore Axis EStop	Read/Write	Controls whether the axes group will continue performing motion if one of the member axes is experiencing an Estop. A value of FALSE indicates that the axes group should enter an Estop state if a member axis enters the Estop state. A value of TRUE indicates that the axes group should continue performing motion.
MC_GRP_PARAM_AXIS_ESTOP_ACTIVE	1001	Axis EStop Active	Read Only	This Read-only parameter will be asserted TRUE whenever an axis in the group is experiencing an axis Estop Error. When there are no axis Estop Errors present on the axes in a group, this parameter will be set to FALSE.

The table below is a list of non-Boolean parameters currently supported. These parameters are read and written by the function blocks [MC\\_GrpReadParam](#) and [MC\\_GrpWriteParam](#).

Parameter	ID	Name	R/W	Description
MC_GRP_PARAM_CIRCLE_TOLERANCE	2000	Command Velocity	Read/Write	This parameter controls how large an error will be acceptable when constructing circles using circle mode MC_CIRC_MODE_CENTER. A value of zero will default to the 1 part in 100,000 tolerance. For more information see <a href="#">Precision Requirements for Circular Move Input Parameters</a> .

### 8.2.4.2.9 Axis Positions Data

**CommandPosition** is the command position (without modulo and in the axis' User Units) that is sent to the drive interface to command an axis. This position is tied to the Status output of the MC\_Power function block:

- When the Status = 1 the command position is a combination of the Normal, Superimposed and Phasing commands
- When the Status = 0 the command position tracks the Actual Position

**CoordinatedMotionCmdPos** is the command position generated by the Coordinated Motion Interpolator when interpolating coordinated motion moves.

**NormalCmdPos** is the command position generated by the Normal Interpolator when interpolating a single axis move or a slave move.

**PhaseCmd Pos** is the command position generated by the Phasing Interpolator when interpolating a MC\_Phasing master phase shift. It is also incorporated in the generation of the NormalCmdPos when interpolating a slave move.

**SuperimposedCmdPos** is the command position generated by the Superimposed Interpolator when interpolating a MC\_Superimp move

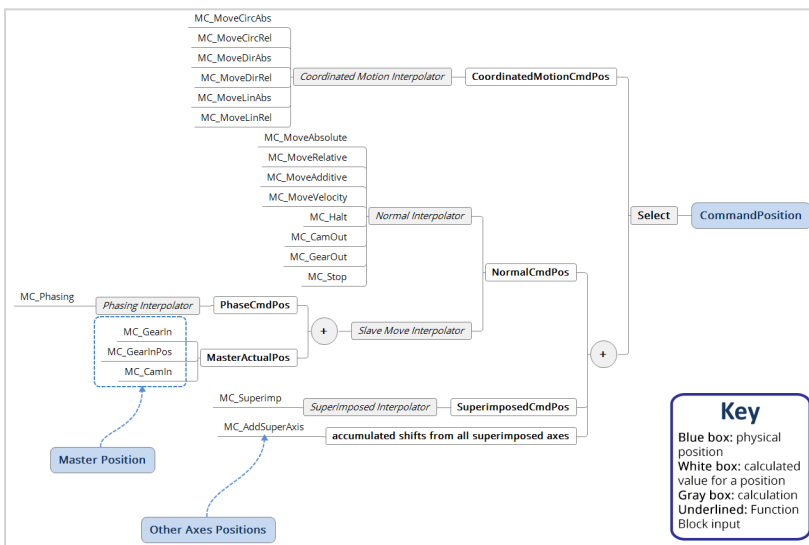


Figure 5-14: Graphic of how PLCOpen axis position data is calculated.

**Actual Position** is the position of the axis read from the drive interface, which is read from the feedback device,

**Group Actual Position** is the set of positions of an axes group read from the axis interface,

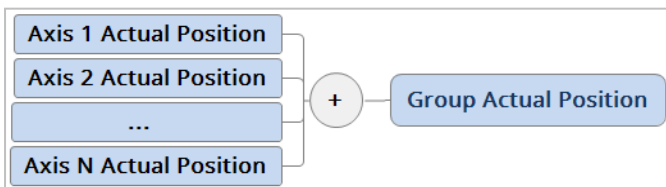
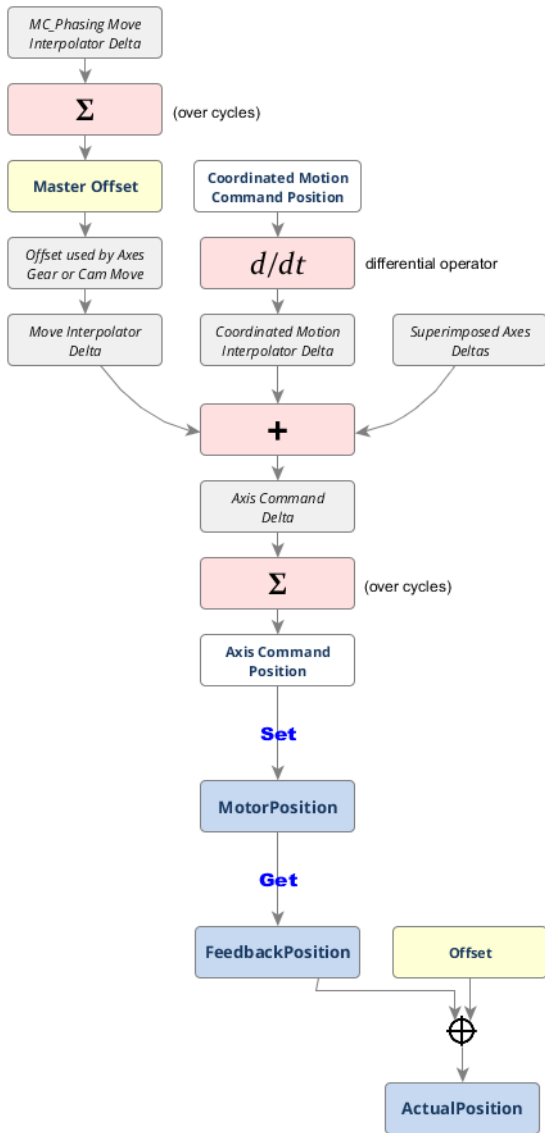


Figure 5-15: How Actual Position is calculated.



#### 8.2.4.2.10 Axis Velocity Data

The following velocity data are related to PLCopen Axis

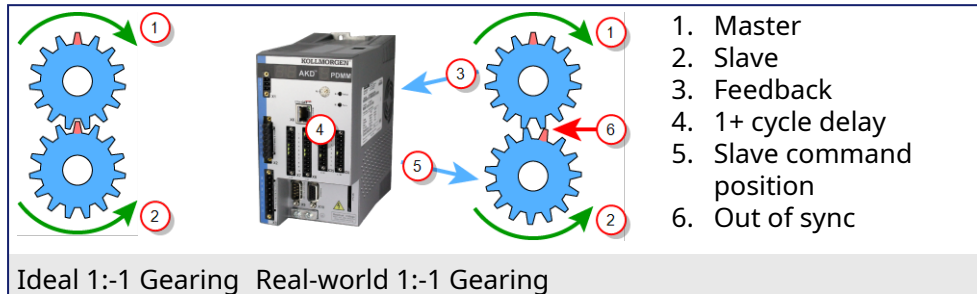
- **ActualVelocity:** is the rate of change of the actual position of the axis read from the drive interface which is read from the feedback device.



- **Command Velocity:** the rate of change in command position. This velocity is tied to the Status output of the **MC\_Power** function block. Since command velocity is the rate change of command position:
  - When the **Status = 1** the command velocity is the rate of change of the command position sent to the drive interface to command the axis.
  - When the **Status = 0** the command velocity tracks the Actual Velocity

### 8.2.4.2.11 Velocity Compensation

Velocity compensation is a value that is added to the master actual position in order to predict where the master will be when the new slave command position is sent to the slave. In other words, the compensation is used to get rid of the delay in reading the master's feedback. This involves prediction about where the master will be, so there is likely some small error in the prediction.



The above diagrams represent a 1:-1 gearing.

- On the left, we have ideal gearing where the master and slave are always perfectly synchronized. When the red tooth of the master gear is at the top, the red tooth of the slave gear is also at the top.
- On the right, we have a real-world gearing example. There is a time delay in reading the master feedback position, communicating that value to the controller, and the controller communicating the new command position to the slave axis. This delay will cause the slave to lag behind in its position. This is demonstrated by the slave's red tooth being slightly out of position.

To compensate for the delay, we want to predict where the master axis will be when the new slave position is applied to the slave. This is the purpose of the velocity compensation.

Velocity compensation is controlled by two axis parameters, **Velocity Compensation Factor** and **Velocity Compensation Filter**, described in more detail below. Both parameters are properties of master axes. When an axis is synchronized to a master via camming or gearing, the PLCopen motion engine uses these two properties of the master axis to predict where the master axis will be in the next motion cycle. The PLCopen motion engine then uses this predicted master position to calculate the new slave position.

The **Velocity Compensation Factor** (Axis Parameter 1009) is the number of motion cycles to compensate for the feedback latency. It should be set equal to the number of motion cycles between when the feedback device acquires the position and when the motion controller processes the position feedback. A value of 0 indicates no compensation will be applied. The larger the value of *VelocityCompensationFactor*, the more prediction takes place and the more prediction error there will be. For physical axes, the value for *VelocityCompensationFactor* will normally be 1 or 2. Some feedback devices or drives may require larger values. For virtual masters, the value will usually be 0. Virtual masters do not need compensation since their position is always known by the controller.

The **Velocity Compensation Filter** (Axis Parameter 1010) is a way to filter noisy feedback systems and/or masters that may have jitter due to servo action. Numerically, the *VelocityCompensationFilter* is the number of motion cycles over which a previously calculated compensation is applied. This helps smooth noisy systems. The default value for *VelocityCompensationFilter* is 1, which means that no filtering takes place; all corrections are applied in a single cycle. Larger values smooth the noise, but can also cause the prediction of the master position to be further off by what can be characterized as additional following error or phase delay. Balancing between the level of noise and the amount of phase delay is application specific and is best determined by experimentation. Good values typically lie in the range of 1 to 5.

See also: [Axis Parameters](#).

**8.2.4.2.12 Move Types**

MoveID	Description	Related FB
0	No move	
1	Distance move	, and
2	Position move	
3	Velocity move	
4	Halt move	
5	Gear-in move	
6	Gear-out move	
7	Reference move	
8	Stop move	
9	Gear-in pos. move	
10	Cam profile move	
11	Cam-out move	

**8.2.4.2.13 Rollover**

The Rollover Position is specified in user units in the PLCopen Axis Data dialog. When this value is non-zero, the axis' position is reset to zero when it reaches the rollover position.

For example, if the rollover position is 360 and the axis is traveling in the positive direction, the axis position counts up until it reaches 360 where it resets to 0 and then continues counting up from there.

If the axis is traveling in the negative direction, the axis position counts down until it reaches 0, where it resets to 360 and then continues counting down from there.

Refer to MC\_MoveAbsolute's description for an explanation of its operation when Rollover Position is nonzero.

When the Rollover Position is zero, rollover is not in effect and the axis position continues to count up when traveling in the positive direction and count down when traveling in the negative direction.

#### 8.2.4.2.14 PLCopen Function Block ErrorID Output

These are the possible errors that could be returned at the ErrorID output of the function blocks.

ErrorID	Description
0	no error
1	queue full
2	abort mode required
3	invalid axis
4	One of four possible scenarios: <ul style="list-style-type: none"> <li>• invalid master axis</li> <li>• master axis and slave axis are the same</li> <li>• master axis is currently slaved to the specified slave axis</li> <li>• master axis and slave axis do not have the same update rate.</li> </ul>
5	invalid parameter number
6	invalid move
7	invalid override
8	buffer mode required
9	invalid parameter data
10	move cannot be executed because an axis error exists, the axis is in the stopping state or the axis is disabled.
11	invalid buffer mode
12	move was aborted due to an E-stop
13	move was aborted due to a controlled stop
14	invalid start mode or sync mode
15	invalid cam profile
16	invalid slave count
17	input value is out of range
18	cannot access time stamp or latched position data
19	data not available For example, if a <a href="#">MC_ReadParm</a> of FollowingError (1006) is programmed on a simulator axis for which no following error is available, an error 19 (data not available) is returned.
20	Motion engine is not running
21	Invalid ratio of velocity to acceleration, or acceleration to jerk. See <a href="#">Limitations on Acceleration and Jerk</a> for more details.
22	Too many profiles – the number of selected profiles is limited to 256.
23	Internal Error
24	Object already exists
25	Block needs to be called between <a href="#">MLMotionInit</a> and <a href="#">MLMotionStart</a>
26	The axis is owned by another group
27	The axis is already present in the group
28	No axes are mapped to the group
29	Invalid name. A name cannot be an empty string
30	Name is already used
31	The number of axes is too small

ErrorID	Description
32	The number of axes is too large
33	Invalid update rate
34	Invalid axis group
35	Invalid kinematic parameter (position, speed, velocity, acceleration, deceleration, or jerk)
36	Invalid floating point number
37	Invalid object index
38	Invalid number of positions in array
39	Position is not a valid number
40	Invalid coordinate system
41	Axis is not initialized
42	Array is not the correct size
43	A move cannot be made while the group is in a stopping state.
44	The axes group is not in the Disabled or Standby state
45	The axes group is in the Disabled state
46	The axis is not in this axes group
47	Duplicate points specified
48	Invalid radius specified
49	Collinear points were specified for the border points of a circle
50	Cannot construct a circle with specified parameters
51	Invalid circle mode
52	Invalid path choice for circular move
53	Invalid transition mode
54	The axis group does not have exactly two axes. As an example, transitions are only allowed on groups with two axes.
55	Invalid engine type for axis
56	An axis or its associated drive is in an error state.
57	Cannot execute function because both queues are not empty.
58	Lines are parallel
59	Not enough room for transitions
60	Abort mode not allowed for Circular Absolute moves
61	Invalid transition parameters
62	Transition not allowed with Abort Move.
63	The axes group is not in the Standby state.
64	The maximum number of axes has been created.
65	The axis is not powered on.
66	Error in data reported from drive.
67	The axes group is not in the ErrorStop state.
68	The axes group is still in Error State.
69	The axes group is not in Standby or ErrorStop state.
70	The fast input is in use.
71	Set position is in progress.



ErrorID	Description
72	Invalid data point count.
73	Position Axis and Trigger Axis cannot be different for position mode trigger
74	Invalid Fast Input trigger mode
75	Cannot simultaneously perform motion from different motion engines.
76	Cannot call this function block twice in the same cycle with the same value for execute input. This error may arise from two scenarios: <ul style="list-style-type: none"> <li>User application code was written in a way that the function block was called more than once in the same cycle</li> <li>PLC program is being overloaded causing the application to think the function block is called more than once in the same cycle</li> </ul>
77	An axis is in a controlled stop error state.
78	Invalid kinematic transform type specified.
79	Invalid kinematic transform parameter specified.
80	Axes group has no ACS to MCS transform specified.
81	The maximum number of transforms has been created.
82	Invalid transition between coordinate systems.
83	Invalid enabled physical axes count.
84	Operation is not valid when the axis is enabled
100	The 'Cam_Data' array is not large enough to hold the specified number of elements
101	Invalid master or slave scale. The scale cannot be less than zero
102	Element master or slave position is outside the range defined by 'Cam_Props'
103	A segment type was specified that is not supported by the value of the 'Options' argument
104	Master position of an element is too close to the master position of a previous element
106	Invalid profile ID
107	'CamDataCount' argument exceeds maximum array size of 20000 data points
108	Profile is currently in use
109	Attempting to build a profile already containing elements
110	Unable to allocate cam table
111	Too few cam elements specified
112	Cam elements are not sorted
113	Average segment velocity outside of range defined by start and end element velocities
200	First element's 'MasterIn' value not equal to zero
201	Last element's 'MasterIn' value does not equal value of 'InputScale'
202	Cannot modify the first element in the cam element table. 'SlaveOut' value is outside the output range specified by 'Cam_Props'.
203	Cannot modify the last element in the cam element table. 'SlaveOut' value is outside the output range specified by 'Cam_Props'.
204	Unknown cam error.

### 8.2.4.3 PLCopen Function Blocks - General Rules

The general rules for [PLCopen](#) are:

- [Input parameters](#)
- [Missing input parameters](#)
- [Output Exclusivity](#)
- [Output Status](#)
- [Sign Rules](#)
- [Error Handling Behavior](#)
- [Behavior of Done Output](#)
- [Behavior of CommandAborted Output](#)
- [Behavior of Busy Output](#)
- [Inputs Exceed Application Limits](#)
- [Output 'Active'](#)

#### 8.2.4.3.1 Input parameters

Unless specified otherwise in the function block's description, the input parameters are read with the rising edge of the Execute input.

The input parameters can be as follows:

- Function Blocks with **Execute**

These FBs will initiate execution on the rising edge of 'Execute'. Execution will continue until completion independent of the 'Execute' value (or values) following initiation. Once execution is complete, the function block can be rearmed or reengaged by introducing another rising edge of 'Execute'.

#### NOTE

**Dual-Core limitation:** Consecutive calls to the function block with repeating 'Execute' values within the same cycle will return an error code [76](#).

- Function Blocks with **Enable**

These FBs will continuously be executed every PLC cycle, as long as the Enable remains high.

- Function with **En**

This is very similar to ENABLE on Function blocks. But, as already explained in [Difference between Functions and Function Blocks](#), functions are expected to complete in one cycle.

#### 8.2.4.3.2 Missing input parameters

If any input parameter of a function block is missing (**open**), the compiler generates an error.

#### 8.2.4.3.3 Output Exclusivity

If the **Execute** input of a function block is set TRUE, either the **Busy**, **Done**, **Error**, or **CommandAborted** outputs must also be set TRUE. These outputs are mutually exclusive, meaning that a function block may have only one of them set TRUE at any time.

#### NOTE

The output functionality of the [MC\\_GearOut](#) function block is an exception to this exclusivity rule. The MC\_GearOut is done when the slave axis is disengaged from the master axis. Unlike most other motion function blocks, once done, the MC\_GearOut will remain busy and active until it is aborted by a different motion function block. The MC\_GearOut function block represents an exception to the exclusivity rule as the **Done** and **Active** outputs may be true at the same time.

#### 8.2.4.3.4 Output Status

The **Done**, **Error**, **ErrorID** and **CommandAborted** outputs are reset with the next rising edge of **Execute**.

If an instance of a function block receives a new **Execute** before it finishes (as a series of commands on the same instance), the function block does not return any feedback, like **Done** or **CommandAborted**, for the previous action.

#### 8.2.4.3.5 Sign Rules

**Velocity**, **Acceleration**, **Deceleration** and **Jerk** are always positive values. **Position** and **Distance** can be positive or negative.

#### 8.2.4.3.6 Error Handling Behavior

Two outputs deal with errors that can occur while executing a function block. These outputs are defined as follows:

- **Error**: the rising edge of **Error** informs you that an error occurred during the execution of the function block
- **ErrorID**: Error number.

**Done**, **InVelocity**, **InGear**, and **InSync** mean successful completion so these signals are logically exclusive to **Error**.

Instance errors do not always result in an axis error. Some bring the axis to **StandStill**.

#### 8.2.4.3.7 Behavior of Done Output

The **Done** output (as well as **InGear**, **InSync**) is set when the commanded action has been completed successfully.

With multiple function blocks working on the same axis in a sequence, the following applies: when one movement on an axis is interrupted with another movement on the same axis without having reached the final goal, **Done** of the first function block is not set.

When a motion command is executed, there are three possible outcomes:

1. It completes successfully. At that time, the **Done** output goes high.
2. It is aborted prior to completing by a subsequent motion command. At that time, the **CommandAborted** output goes high.
3. It encounters an error prior to completing or an invalid input is specified. At that time, the **Error** output goes high.

These outputs stay in this state until that motion function block is executed again. At that time, the **Done**, **CommandAborted** and **Error** outputs go low; and the **Busy** output goes high, provided all the inputs are valid.

#### 8.2.4.3.8 Behavior of CommandAborted Output

**CommandAborted** is set when a commanded motion is interrupted by another motion command. The reset-behavior of **CommandAborted** is like that of **Done**. When **CommandAborted** occurs, the other output signals such as **InVelocity** are reset.

#### 8.2.4.3.9 Behavior of Busy Output

The **Busy** output indicates that the function block is still working, with new output values to be expected.

**Busy** is SET at the rising edge of **Execute** and RESET when one of the outputs **Done**, **Aborted** or **Error** is set. It is recommended that this function block is kept in the active loop of the application program for at least as long as **Busy** is True, because the outputs can still change.

For one axis, several function blocks can be busy, but only one can be active at a time. Exceptions are **MC\_SuperImposed** and **MC\_Phasing**, where more than one function block related to one axis can be active.

#### 8.2.4.3.10 Inputs Exceed Application Limits

If a function block is commanded with parameters which result in a violation of application limits, the instance of the function block generates an error.

### 8.2.4.3.11 Output 'Active'

The **Active** output is set at the moment the function block takes control of the motion of the respective axis.

#### 8.2.4.3.11.1 Coordinated Motion

Use the following table to determine the state of the Busy and Active outputs when transitioning from one move to the next.

Function Block Output	First Move	Second Move
<b>Busy</b>	False	True
<b>Active</b>	False	False
<b>Done</b>	True <sup>1</sup>	False

### 8.2.4.3.12 List of Input Parameters

The input parameters are listed as follows:

- [Function Blocks with Execute](#)
- [Function Blocks with Enable](#)
- [Function with En](#)

#### 8.2.4.3.12.1 List of PLCopen function blocks with Execute

These FBs will initiate execution on the rising edge of 'Execute'. Execution will continue until completion independent of the 'Execute' value (or values) following initiation. Once execution is complete, the function block can be rearmed or reengaged by introducing another rising edge of 'Execute'.

#### NOTE

**Dual-Core limitation:** Consecutive calls to the function block with repeating 'Execute' values within the same cycle will return an error code [76](#).

Function Block	Description
MC_MoveAbsolute MC_MoveRelative MC_MoveAdditive MC_MoveSuperimp MC_MoveVelocity MC_Halt MC_CamIn MC_CamOut MC_GearIn MC_GearOut MC_GearInPos	A positive transition of this input requests to queue the move
MC_Phasing	A positive transition of this input requests to queue the phase shift move
MC_SyncSlaves MC_TouchProbe MC_AbortTrigger MC_SetPosition	A positive transition of this input causes this function block to execute
MC_WriteBoolPar MC_WriteParam	A positive transition of this input writes the specified parameter

<sup>1</sup>This output will be false if the move is interrupted by a MC\_GrpHalt, MC\_GrpStop, or if the second move's buffer mode is "Aborting".

Function Block	Description
MC_Reference	A positive transition of this input requests to queue the reference move and arm the reference trigger event(s)
MC_CamTblSelect	A positive transition of this input reads and initializes the specified profile
MC_Stop	A positive transition of this input initiates a stop move. While this input is held high, no other move can be queued for this axis

#### 8.2.4.3.12.2 List of PLCopen function blocks with Enable

These FBs will continuously be executed every PLC cycle, as long as the Enable remains high.

Function Block	Description
MC_ReadBoolPar MC_ReadParam	When this input is high, the specified parameter is read
MC_SetOverride MC_ReadActPos	When this input is high, the override factors is written
MC_ReadActVel	When this input is high, the axis's actual velocity is returned
MC_ReadAxisErr	When this input is high, the axis's error status is returned
MC_ReadStatus	When this input is high, the function block outputs is updated
MC_Power	If this input is high and the drive is currently disabled, this function block requests to close the servo loop and enable the drive. If this input is low and the drive is currently enabled, this function block requests to open the servo loop and disable the drive

#### 8.2.4.3.12.3 List of PLCopen functions with input parameter En

This is very similar to ENABLE on Function blocks. But, as already explained in [Difference between Functions and Function Blocks](#), functions are expected to complete in one cycle.

Function	Description
<a href="#">MC_CreatePLCAxis</a> <sup>MC_</sup> CreatePLCAxis	When this input is high, a PLCopen axis is created
MC_InitAxis	When this input is high, the specified axis is initialized
MC_EStop	When this input is high, an E-stop is generated for the specified axis
MC_ResetError	When this input is high, the specified axis's errors is reset
MC_AddSuperAxis	When this input is high, the specified axis is added to the superimposed axis list of the receiving axis.
MC_RemSuperAxis	When this input is high, the specified axis is removed from the superimposed axis list for the specified receiving axis.

#### 8.2.4.4 PLCopen State Machine

The following diagram normatively defines according to PLCopen the behavior of the axis at a high-level when multiple motion control function blocks are "simultaneously" activated. This combination of motion profiles is useful in building a more complicated profile or in handling exceptions within a program. In real implementations there can be additional states defined at a lower level.

The basic rule is that motion commands are always taken sequentially, even if the PLC has the capability of real parallel processing. These commands act on the state diagram of the axis.

The axis is always in one of the defined states (see diagram below). A change of state is reflected immediately when issuing the corresponding motion command (please note that the response time of 'immediately' is system dependent).

There are seven states defined:

1. Stand Still
2. Homing
3. Discrete Motion
4. Continuous Motion
5. Synchronized Motion
6. Stopping
7. Error Stop

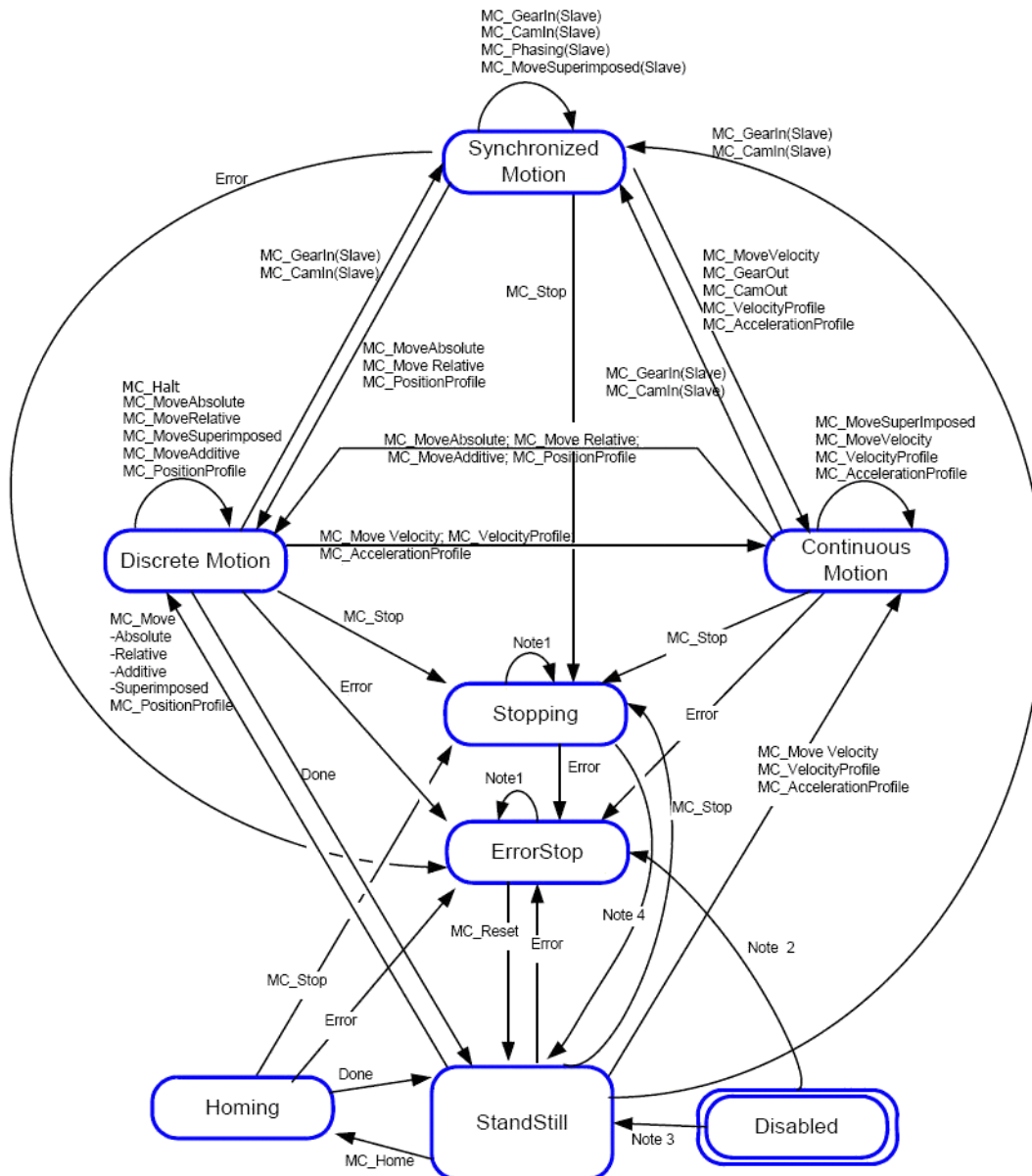


Figure 5-16: Motion State Machine (PLCopen)

**NOTE**

**Note 1:** In this state **ErrorStop** or **Stopping**, all function blocks can be called, although they are not executed, except MC\_Reset and Error which generate the transition to **StandStill** or **ErrorStop** respectively

**Note 2:** MC\_Power FB is called with Enable=TRUE and there is an error in the Axis

**Note 3:** MC\_Power FB is called with Enable=TRUE and there is **no** error in the Axis

**Note 4:** MC\_Stop.Done **and not** MC\_Stop.Execute

A normal procedure would start in **StandStill**. In this state the power can be switched on per axis (via the command MC\_Power). Also, you can access the **Homing** state (via the issue of the command Home per axis), which after normal completion returns to **StandStill**. From here you can transfer an axis to either **Discrete Motion** or **Continuous Motion**. Via the **Stopping** state you can return to **StandStill**. **ErrorStop** is a state to which the axis transfers in case of error. Via a (manual) Reset command, you can return to **StandStill**, from which the machine can be moved to an operational state again.

Please note that the States define the functionality of the function blocks. Function Blocks which are not listed in the State Diagram do not affect the state of the axis, meaning that, whenever they are called, the state does not change. They are:

- MC\_ReadStatus
- MC\_ReadAxisErr
- MC\_ReadParameter
- MC\_ReadBoolParameter
- MC\_WriteParameter
- MC\_WriteBoolParameter
- MC\_ReadActualPosition
- MC\_CamTableSelect

### State **Disabled**

The **Disabled** state describes the initial state of the axis. In this state, the movement of the axis is not influenced by the FBs. The axis feedback is operational.

If the MC\_PowerFB is called with Enable=TRUE while being in **Disabled**, this either leads to **Standstill** if there is no error inside the axis, or to **ErrorStop** if an error exists.

Calling MC\_Power with Enable=FALSE in any state, the axis goes to the state **Disabled**, either directly or via any other state. If a motion generating function block controls an axis while the MC\_Power FB with Enable=FALSE is called, the motion generating function block is aborted (CommandAborted).

**Disable** means power off without error.

### State **ErrorStop**

The intention of the **ErrorStop** state is that the axis goes to a stop, if possible. No further FBs are accepted until a reset has been done from the **ErrorStop** state. The transition Error refers to errors from the axis and axis control, and not from the function block instances. These axis' errors can also be reflected in the output of the function blocks "FB instances errors".

Issuing MC\_Home in any other state than **StandStill** goes to **ErrorStop**, even if MC\_Home is issued from the **Homing** state itself.

**ErrorStop** is valid as highest priority and applicable in case of an error. The axis can have either power enabled or disabled, and can be changed via MC\_Power. However, as long as the error is pending the state remains **ErrorStop**.

### From **StandStill** to **Stopping**

Calling the FB MC\_Stop in state **StandStill** changes the state to **Stopping** and back to **Standstill** when "Execute = FALSE". The state **Stopping** is kept as long as the input "Execute" is true. The "Done" output is set when the stop ramp is finished.

**StandStill** is power on without an error.

*State machine for **multi-axes** motion control*

The diagram is focused on a single-axis. The multi-axes function blocks (e.g. MC\_CamIn, MC\_GearIn or MC\_Phasing) can be looked at, from a state diagram point of view, as multiple single-axes all in specific states. For instance, the CAM-master can be in the state **Continuous Motion**. The corresponding slave is in the state **Synchronized Motion**. Connecting a slave axis to a master axis has no influence on the master axis.

## 8.2.4.5 Programming a Dual Core Controller

A dual-core controller operates slightly differently in order to take advantage of the extra performance provided by a multiple-core CPU. Consequently, there are some differences in programming a dual-core controller from programming a single-core controller. This page lists those differences.

### 8.2.4.5.1 Cached Data



A dual-core controller gains most of its performance by running PLC code and motion code at the same time. In order to avoid [race conditions](#), function blocks that read motion data will report the data that the motion code used or calculated during the last cycle.

*List of function blocks that return cached data:*

- MC\_AddSuperAxis
- MC\_ClearFaults
- MC\_EStop
- MC\_GrpReadBoolPar
- MC\_GrpReadParam
- MC\_GrpReadStatus
- MC\_ReadActPos
- MC\_ReadActVel
- MC\_GrpReadCmdPos
- MC\_GrpReadCmdVel
- MC\_GrpReadActAcc
- MC\_GrpReadActPos
- MC\_GrpReadActVel
- MC\_GrpReadError
- MC\_GrpWriteBoolPar
- MC\_GrpWriteParam
- MC\_MoveContVel
- MC\_Power
- MC\_ReadAxisErr
- MC\_ReadBoolPar
- MC\_ReadParam
- MC\_ReadStatus
- MC\_RemSuperAxis
- MC\_ResetError
- MC\_SetPos
- MC\_StopRegist
- MC\_WriteParam
- MLAxisAddTq
- MLAxisCmdPos
- MLAxisFBackPos
- MLAxisGenEN
- MLAxisGenIsRdy
- MLAxisGenPos
- MLAxisPipePos
- MLAxisReadActPos
- MLAxisReadGenStatus
- MLAxisReadTq
- MLAxisReadVel
- MLAxisReAlgnRdy
- MLAxisStatus
- MLAxisStop
- MLAxisTimeStamp
- MLBlkIsReady
- MLBlkReadOutVal
- MLCompCheck
- MLIntWriteOutVal
- MLMotionStatus
- MLPhaReadActPhase
- MLPmpStatus
- MLSyncWriteDeltaS
- MLTrigClearFlag

- [MLTrigIsTriggered](#)
- [MLTrigReadPos](#)
- [MLTrigReadTime](#)

#### 8.2.4.5.2 Calling Function Blocks Multiple Times in the Same Cycle

A given function block can usually be called multiple times without causing problems when using a single-core controller. For example, it is possible to call `MC_MoveVelocity` multiple times to start velocity moves on multiple axes. You would need to set the **Execute** input to **False** between every call to start a new velocity move. In doing so, you lose the ability to monitor the status of the velocity moves because you changed the axis parameter for each call, but if you were not concerned with monitoring the status of the velocity moves, you could call `MC_MoveVelocity` any number of times in a given cycle.

Function blocks that start a motion-related action need to avoid [race conditions](#) when using a dual-core controller. To do so the function blocks have to save argument data so that calculations and error detection can be determined during the next controller cycle. There are two consequences to this:

- With the possible exception of the `MulticoreFbExecuteDuplicateValueInCycle` error (error [76](#)), the **Error** and **ErrorID** outputs will report no error on the first cycle. The **Error** and **ErrorID** outputs will report their status on the next VM cycle.
- The function block cannot be called with the same **Execute** state twice in a single VM cycle. If a function block is called with the same **Execute** state twice in a single VM cycle, a `MulticoreFbExecuteDuplicateValueInCycle` error (error [76](#)) will be returned immediately.

*List of function blocks that start motion-related actions:*

- [MC\\_AbortTrigger](#)
- [MC\\_AddAxisToGrp](#)
- [MC\\_AxisSetDefaults](#)
- [MC\\_CamIn](#)
- [MC\\_CamOut](#)
- [MC\\_CamResumePos](#)
- [MC\\_CamStartPos](#)
- [MC\\_GearIn](#)
- [MC\\_GearInPos](#)
- [MC\\_GearOut](#)
- [MC\\_GrpDisable](#)
- [MC\\_GrpEnable](#)
- [MC\\_GrpHalt](#)
- [MC\\_GrpReset](#)
- [MC\\_GrpSetPos](#)
- [MC\\_GrpStop](#)
- [MC\\_Halt](#)
- [MC\\_MachRegist](#)
- [MC\\_MarkRegist](#)
- [MC\\_MoveAbsolute](#)
- [MC\\_MoveAdditive](#)
- [MC\\_MoveCircular](#)
- [MC\\_MoveContVel](#)
- [MC\\_MoveDirect](#)
- [MC\\_MoveLinear](#)
- [MC\\_MoveRelative](#)
- [MC\\_MoveSuperimp](#)
- [MC\\_MoveVelocity](#)
- [MC\\_Phasing](#)
- [MC\\_Reference](#)

- [MC\\_RemAxisFromGrp](#)
- [MC\\_SetPos](#)
- [MC\\_Stop](#)
- [MC\\_SyncSlaves](#)
- [MC\\_TouchProbe](#)
- [MC\\_UngroupAllAxes](#)

### 8.3 EtherCAT Motion Bus Concepts

To exchange data between the controller (master) and the devices (slaves), the KAS Runtime relies on the EtherCAT motion bus. This communication can be done in two modes: cyclic and non-cyclic (mailbox).

In **cyclic mode**, a single frame containing the data of all slaves (input and output) travels along all slaves and goes back to the master. Data is read and/or written "on the fly" by each slave.

Slave device input and output data definitions:

- Outputs are written by the master and read by the slave device
- Inputs written by the slave device and read by the master

#### EtherCATProcessImage

This cyclic frame is called the EtherCAT Process Image. It contains the Process Data, which is defined during network initialization.

The cyclic data is grouped in predefined blocks called Process Data Objects or [PDO](#).

#### NOTE

PDOs contain real-time cyclic data which is deterministic. Non-cyclic data is not deterministic and is defined by Service Data Objects ([SDO](#)).

#### References

- EtherCAT Specification V1.0 - refer to <http://www.ethercat.org> ( in Member Area - Downloads)
- Büttner, H.; Janssen, D.; Rostan, M. (2003), [EtherCAT - the Ethernet fieldbus](#), (PDF), PC Control Magazine 3: 14-19



#### 8.3.1 Functional Principle

Typical automation networks are characterized by short data-length per node, typically less than the minimum payload of an Ethernet frame. Using one frame per node per cycle leads to low bandwidth utilization and thus to poor overall network performance. EtherCAT therefore takes a different approach, called "processing on the fly" (for more details, refer to [EtherCAT Implementation](#)).

With EtherCAT, the Ethernet packet or frame is no longer received, and then interpreted and copied as process data at every node. Instead, the EtherCAT slave devices read the data addressed to them while the telegram passes through the device. Similarly, input data is inserted while the telegram passes through. The frames are only delayed by a fraction of a microsecond in each node, and many nodes - typically the entire network - can be addressed with just one frame.

#### 8.3.2 EtherCAT Features

##### Summary

EtherCAT is characterized by outstanding performance, very simple wiring, and openness to other protocols. EtherCAT sets new standards where conventional fieldbus systems reach their limits: 1000 I/Os in 30  $\mu$ s, optionally twisted-pair cable or optical fiber and, thanks to Ethernet and Internet technologies, optimum vertical integration. With EtherCAT, the costly Ethernet star topology can be replaced with a simple line structure - no expensive infrastructure components are required. Optionally, EtherCAT can also be wired in the classic way using switches, to integrate other Ethernet devices. Where other real-time Ethernet approaches require special connections in the controller, for EtherCAT, very cost-effective standard Ethernet cards suffice.

EtherCAT is versatile: Master to Slave, Slave to Slave and Master to Master Communication is supported (see figure below). [Safety](#) over EtherCAT is available. EtherCAT makes Ethernet down to the I/O level technically feasible and economically sensible. Outstanding features of this network include full Ethernet compatibility, Internet technologies (even in simple devices), maximum utilization of the large bandwidth offered by Ethernet, and outstanding real-time characteristics at low costs.

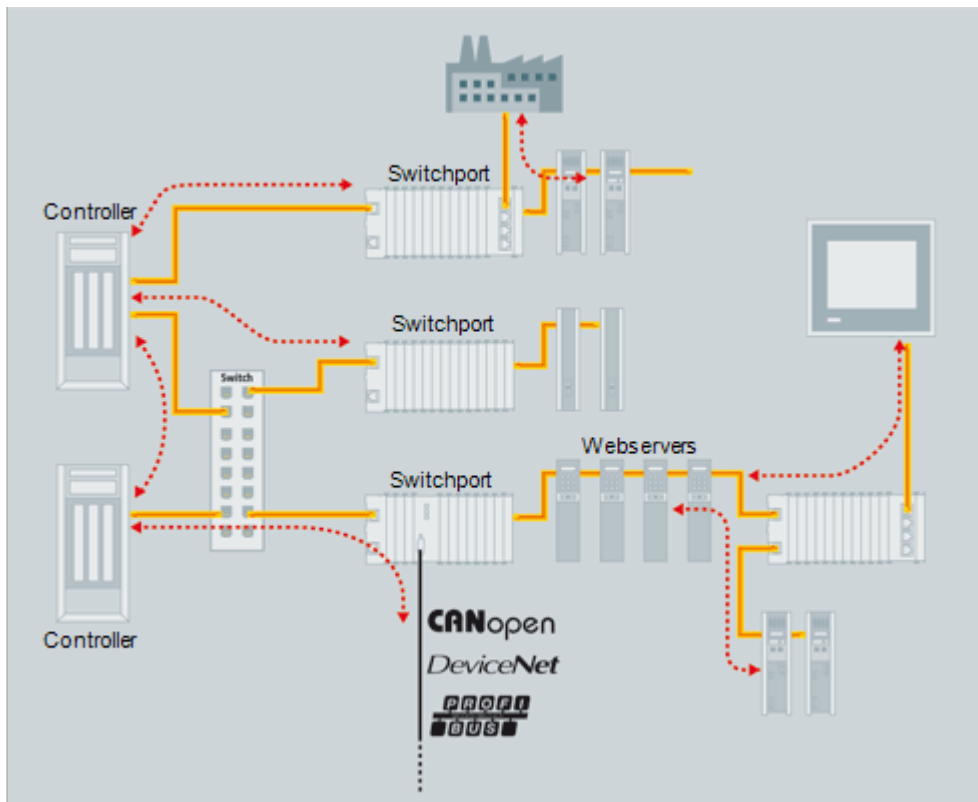
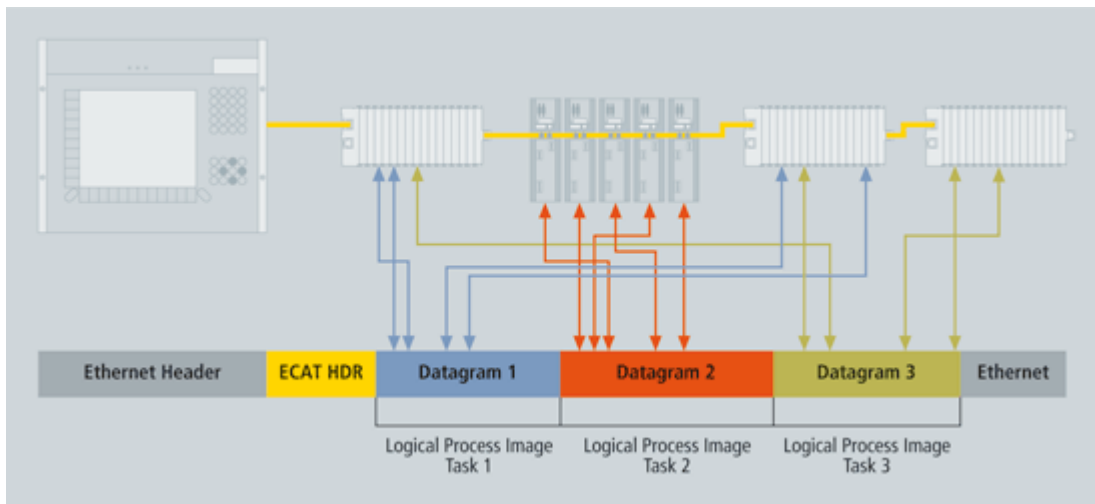


Figure 5-17: Versatile Network Architecture

### 8.3.2.1 Protocol

The EtherCAT protocol is optimized for process data and is transported directly within the standard IEEE 802.3 Ethernet frame using Ethertype 0x88a4. It can consist of several sub-datagrams, each serving a particular memory area of the logical process images, that can be up to 4 gigabytes in size. The data sequence is independent of the physical order of the nodes in the network, and addressing can be in any order. Broadcast, multicast and communication between slaves is possible and must be done by the master device. If IP routing is required, the EtherCAT protocol can be inserted into UDP/IP datagrams. This also enables any control with Ethernet protocol stack to address EtherCAT systems.



**Figure 5-18:** Process Data is Inserted in Telegrams

KAS supports **CANopen over EtherCAT** (CoE). It enables the advantages of EtherCAT in terms of transfer characteristics to be combined with proven, profile-specific drive functions.

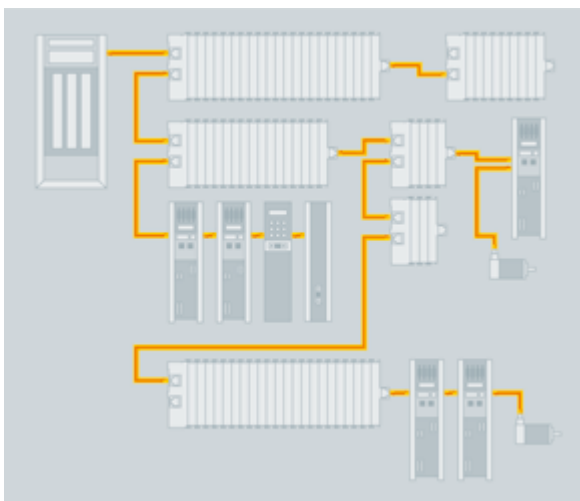
KAS also uses **File Access over EtherCAT** (FoE) protocols to provide options for efficiently exchanging firmware via the bus (see [Several Device Profiles and Protocols can coexist](#)).

### 8.3.2.2 Topology

Using full-duplex Ethernet physical layers, the EtherCAT slave controllers close an open port automatically and return the Ethernet frame if no downstream device is detected. Slave devices can have several ports. Using these features, EtherCAT can support almost any physical topology, such as line, tree or star. The bus or line structure known from the fieldbuses thus also becomes available for Ethernet.

The combination of line and branches or stubs is also possible: any EtherCAT device with three or more ports can act as a junction, and no additional switches are required. The classic switch-based Ethernet star topology can be used either with switches configured to forward traffic directly between ports, or with special slave devices: the switches are then located between the network master and the slave devices.

The special slave device assembly (remember standard slave devices don't have a MAC address) attached to one switch port together forms an EtherCAT segment, which is either addressed via its MAC address or via port-based VLANs. Since the 100BASE-TX Ethernet physical layer is used, the distance between any two nodes can be up to 100 m (300 ft). Up to 65535 devices can be connected per segment. If an EtherCAT network is wired in ring configuration (requiring two ports on the master device), it can provide cable redundancy.



**Figure 5-19:** Flexible Topology: Line, Tree or Star

KAS controllers support line, tree, and star topologies. As the KAS controllers (PCMM and AKD PDMM) have a single EtherCAT port, ring topologies and cable redundancy is not supported.

### 8.3.2.3 Distributed Clock (Synchronization)

A distributed clock is an EtherCAT feature that allows synchronization, with a reference clock, of all EtherCAT slaves and the master. This solves problems related to clock-shifting between the master and the devices.

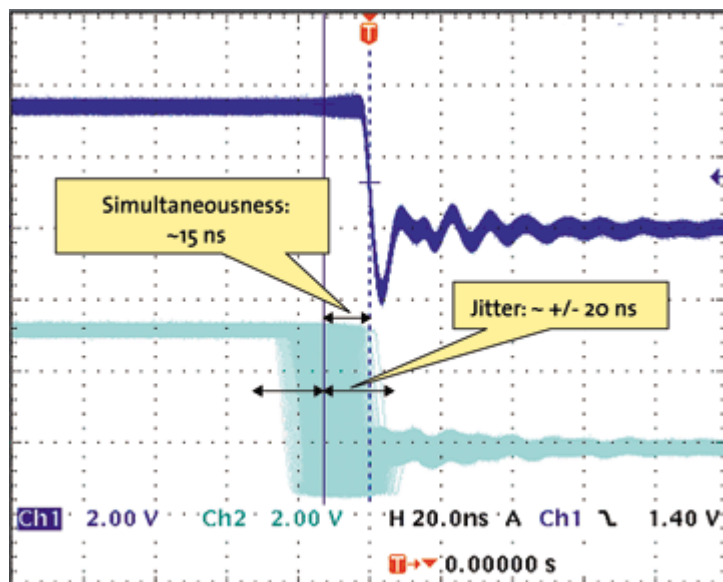
This mechanism also leads to very low jitter of significantly less than 1  $\mu$ s. Even if the communication cycle jitters, it is still compliant with the IEEE 1588 Precision Time Protocol standard.

Therefore, EtherCAT does not require special hardware in the master device and can be implemented in software on any standard Ethernet MAC, even without a dedicated communication coprocessor.

The typical process of establishing a distributed clock is initiated by the master by sending a broadcast to all slaves at a specific address. On reception of this message, all slaves latch the value of their internal clock twice, once when the message is received and once when it returns (remember EtherCAT has a ring topology). The master can then read all latched values and calculate the delay for each slave. This process can be repeated as many times as required to reduce jitter and to average out values. Total delays are calculated for each slave depending on their position in the slave-ring and are uploaded to an offset register. Finally the master issues a broadcast read-write on the system clock, which makes the first slave the reference clock and forcing all other slaves to set their internal clock appropriately with the now known offset.

To keep the clocks synchronized after initialization, the master or slave must regularly send out the broadcast again to counter any effects of speed difference between the internal clocks of each slave. Each slave has to adjust the speed of their internal clock or implement an internal correction mechanism whenever they have to adjust.

The system clock is specified as a 32-bit counter with a base unit of 1 ns starting at January 1st 2000, 0:00.

**Figure 5-20:** Synchronicity and Simultaneity

Scope view of two distributed devices with 300 nodes and 120 m of cable between them.

### 8.3.2.4 Performance

Short cycle times can be achieved because the host microprocessors in the slave devices are not involved in the processing of the Ethernet packets to transfer the process images. All process data communication is handled by the slave controller hardware. Combined with these features, this makes EtherCAT a high-performance distributed I/O system: Process data exchange with 1000 distributed digital I/O takes about 30 μs, which is typical for a transfer of 125 byte over 100Mb/s Ethernet. Data for and from 100 servo axes can be updated with up to 10 kHz. Typical network update rates are 1-30 kHz, but EtherCAT can be used with slower cycle times, too, if the DMA load is too high on your PC.

Process Data	Update Time
256 distributed digital I/O	11 μs = 0,01 ms
1000 distributed digital I/O	30 μs
200 analog I/O (16 bit)	50μs ↔ 20 kHz
100 Servo Axis, with 8 Bytes input and output data each	100 μs
1 Fieldbus Master-Gateway (1486 Bytes Input and 1486 Bytes Output Data)	150 μs

Table 5-4: EtherCAT Performance Overview

The communication with 100 servo axes is also extremely fast: every 100μs, all axes are provided with command values and control data and report their actual position and status. The [Distributed Clocks](#) technique enables the axes to be synchronized with a deviation of significantly less than 1 microsecond. And even at this pace, there is more than sufficient bandwidth for asynchronous communications such as TCP/IP, parameter download or diagnostic data upload.

### 8.3.2.5 Safety over EtherCAT

The protocol enhancement called Safety over EtherCAT ([FSoE](#)) enables safety-related communication and control communication on the same network. The safety protocol is based on the application layer of EtherCAT, with no influence on the lower layers. It is certified according to IEC 61508 and meets the requirements of Safety Integrity Level (SIL) 3.

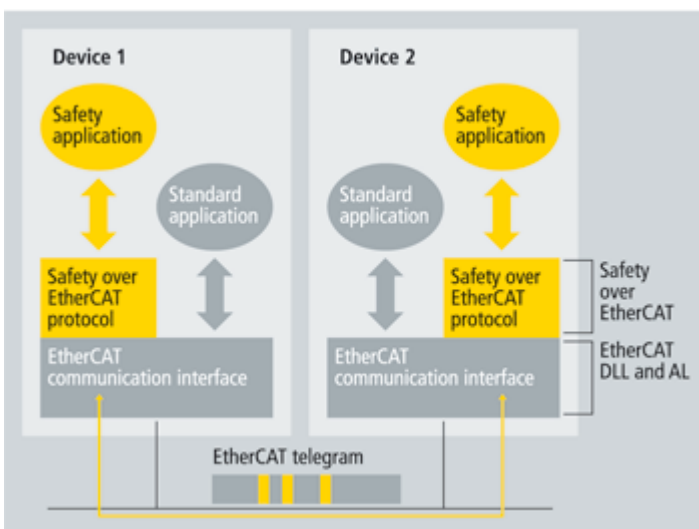


Figure 5-21: Safety over EtherCAT Software Architecture



### 8.3.2.6 Gateways

Gateway devices are available for the integration of existing fieldbus components (e.g., CANopen, DeviceNet) into EtherCAT networks. Also, other Ethernet protocols can be used in conjunction with EtherCAT: the Ethernet frames are tunneled via the EtherCAT protocol, which is the standard approach for Internet applications. The EtherCAT network is fully transparent for the Ethernet device, and the real-time characteristics are not impaired, since the master dictates exactly when the tunneled transfers are to occur and how much of the 100Mb/s media the tunneled protocols can use. Therefore, all Internet technologies can also be used in the EtherCAT environment.

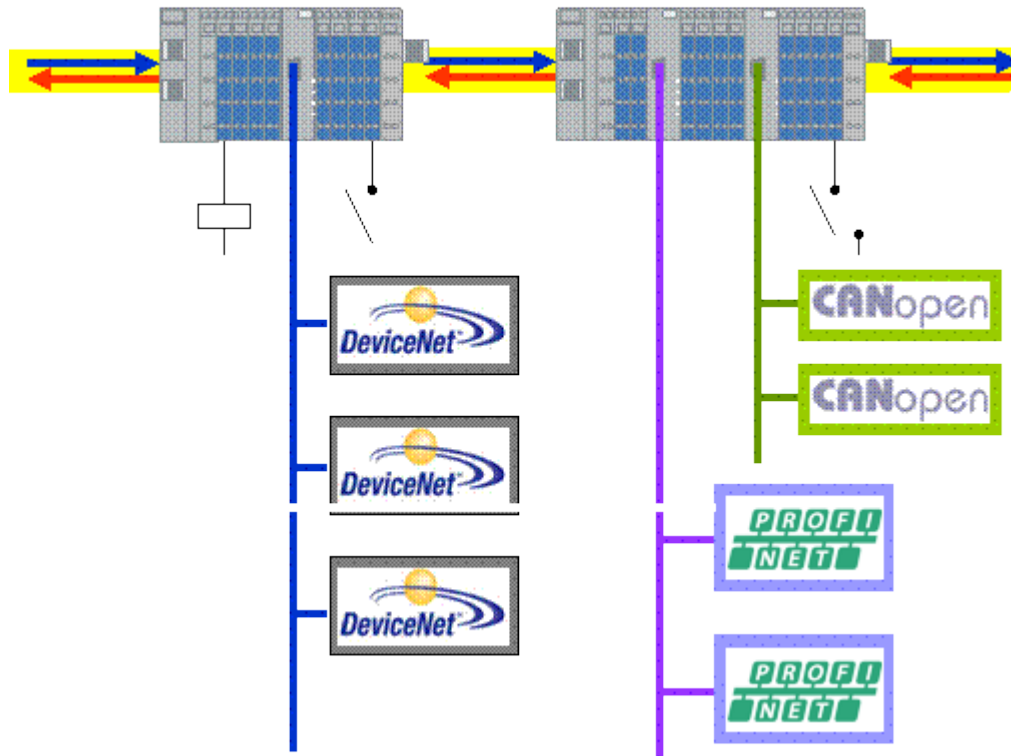


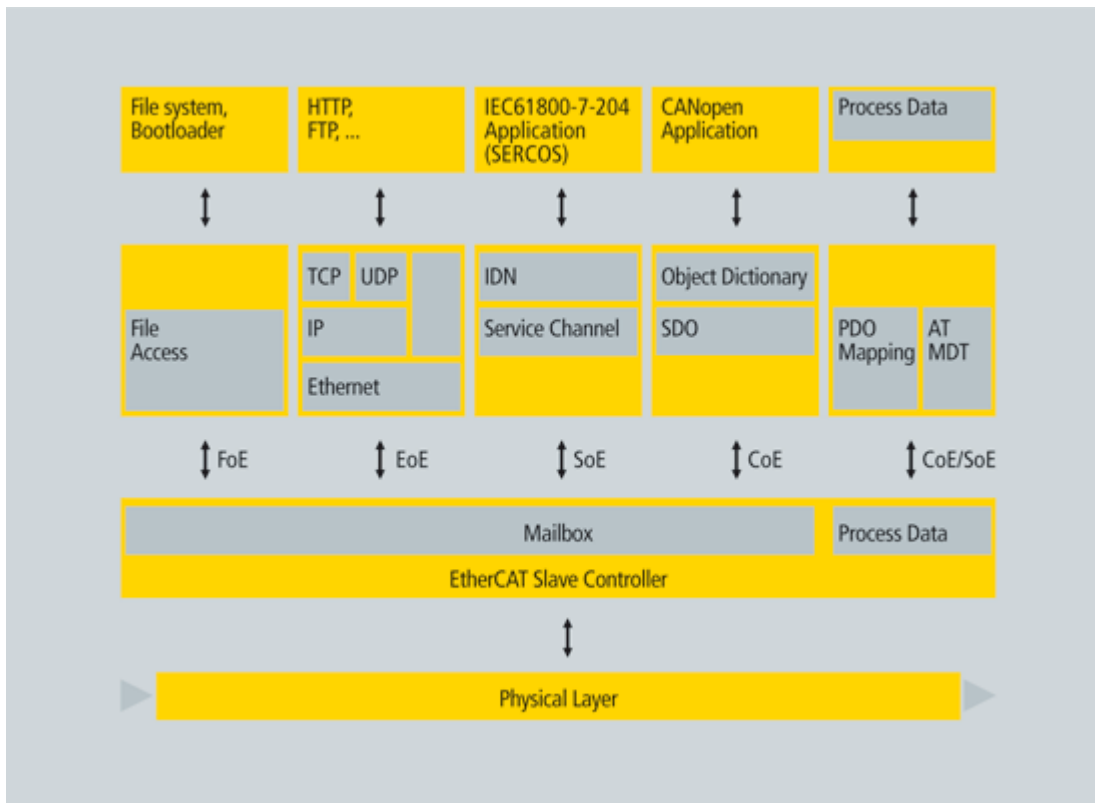
Figure 5-22: Fieldbus Gateway

### 8.3.2.7 Device profiles

The device profiles describe the application parameters and the functional behavior of the devices, including the device class-specific state machines. For many device classes, fieldbus technology already offers reliable device profiles, such as for I/O devices or drives. Users are familiar with these profiles and the associated parameters and tools. Therefore, no EtherCAT-specific device profiles have been developed for these device classes. Instead, simple interfaces for existing device profiles are offered. This greatly assists users and device manufacturers alike during the change from existing fieldbuses to EtherCAT.

#### *CANopen over EtherCAT (CoE)*

CANopen device and application profiles are available for a wide range of device classes and applications, ranging from I/O components, drives, encoders, proportional valves and hydraulic controllers to application profiles for plastic or textile machinery. EtherCAT can provide the same communication mechanisms as the familiar CANopen mechanisms: object dictionary, PDO (process data objects) and SDO (service data objects), and even the network management is comparable. EtherCAT can thus be implemented with minimum effort on devices equipped with CANopen. Large parts of the CANopen firmware can be re-used. Objects can optionally be expanded in order to account for the larger bandwidth offered by EtherCAT.



**Figure 5-23:** Several Device Profiles and Protocols can coexist

### 8.3.2.8 File Access over EtherCAT (FoE)

This very simple protocol, similar to TFTP, enables access to any data structure in the device. Therefore, standardized firmware upload to devices is possible, regardless of whether or not they support TCP/IP.

## 8.3.3 EtherCAT Implementation

The EtherCAT Technology was developed with very low cost devices in mind, like I/O terminals, sensors, and embedded controllers. EtherCAT only uses standard Ethernet frames according to IEEE 802.3. These frames are sent by the master device, and the slave devices extract and/or insert data on the fly. Thus EtherCAT uses standard Ethernet MACs, where they really make sense: in the master device. EtherCAT slave controllers are also used where such dedicated chips really make sense: in the slave device, where they handle the process data protocol in hardware and provide maximum real-time performance regardless of the local processing power or software quality.

### 8.3.3.1 Master Configuration

EtherCAT communicates a maximum of 1486 bytes of distributed process data with just one Ethernet frame. Therefore, unlike other solutions where the master device in each network cycle has to process, send and receive frames for each node, EtherCAT systems typically only need one or two frames per cycle for the entire communication with all nodes, so EtherCAT masters do not require a dedicated communication processor. The master functionality puts hardly any load on the host CPU, which can handle this task easily while processing the application program: so EtherCAT can be implemented without special or expensive active plug-in cards, just by using a passive NIC card or the on-board Ethernet MAC. Implementation of an EtherCAT master is very easy, particularly for small and medium-sized control systems and for clearly defined applications.

For example, a PLC with a single process image: if it does not exceed the 1486 bytes, cyclic sending of a single Ethernet frame with the cycle time of the PLC is sufficient (as shown in [Master-Implementation with one Process Image](#)). Because the header does not change at run-time, the

only thing required is that a constant header be added to the process image and that the result be transferred to the Ethernet controller.

The process image is already sorted, since with EtherCAT mapping does not occur in the master, but in the slaves - the peripheral devices insert their data at the respective points in the passing frame. This further unburdens the host CPU. It was found that an EtherCAT master entirely implemented in software on the host CPU uses less of its processing power than much slower fieldbus systems implemented with active plug-in cards; servicing the DPRAM of the active card alone puts more load on the host.

System [configuration tools](#) provide the network and device parameters (including the corresponding boot-up sequence) in a standardized XML format.

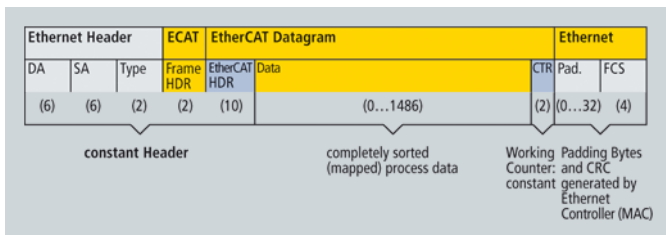


Figure 5-24: Master-Implementation with one Process Image

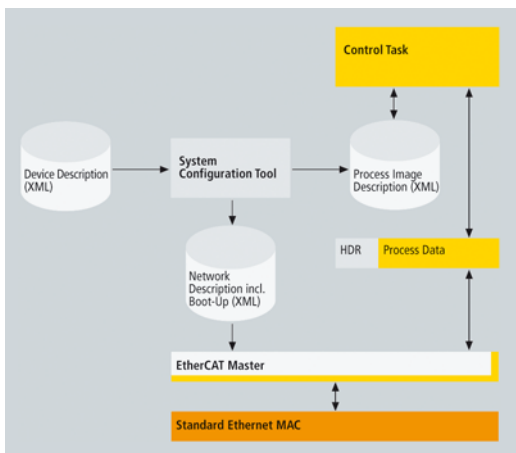


Figure 5-25: Structure of EtherCAT Master Implementation

### 8.3.3.1.1 ESI and ENI Files

The EtherCAT master uses the information from the ENI file to initialize and configure the EtherCAT network. The ESI files are provided by the vendor for each device. They contain information about the device functionality and its settings. The ESI files are used by the KAS IDE to generate the ENI file. The KAS controller’s EtherCAT master uses the ENI file for network initialization and configuration.

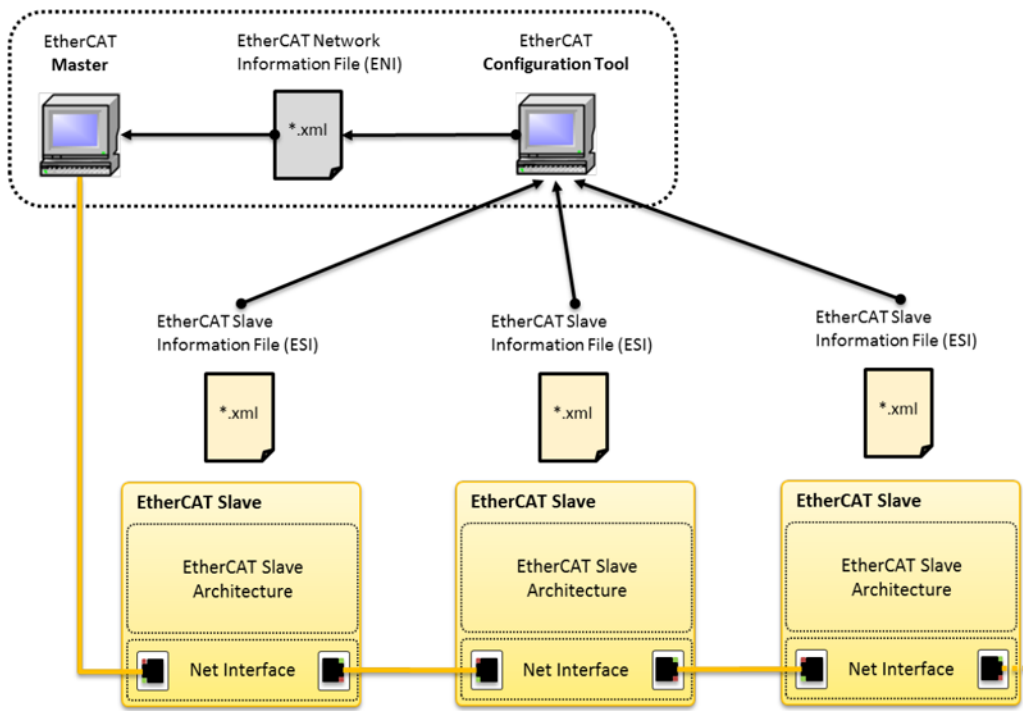


Figure 5-26: EtherCAT Network Architecture1

**ENI File** A network configuration file in XML format, the ENI file describes the network topology, the initialization commands for each device, and commands which have to be sent cyclically. The ENI file is provided to the master, which sends commands according to this file.

The KAS IDE creates the ENI file after a network discovery, which can be exported or imported. A scan and compile should be redone, if the network changes, in order to regenerate the ENI file.

**ESI File** A device description in XML format. This is a fixed file provided by the supplier of a given EtherCAT device. The ESI file contains information about the device's functionality and settings.

EtherCAT device vendors must provide an ESI file, which is used by the KAS IDE to compile the network information (e.g. process data structures, initialization commands) and create the ENI file.

**NOTE**

KAS uses EtherCAT Network Information (ENI) schema version 1.3 (May 20, 2009). For more details, see the ETG.2100 specification at [www.ethercat.org](http://www.ethercat.org).

**8.3.3.2 Slave Configuration**

A cost-effective EtherCAT slave controller (ESC) is used in the slave devices. With EtherCAT the slave does not need a microcontroller at all. Simple devices that get by with an I/O interface can be implemented only with the ESC and the RJ45 connector. The process data interface (PDI) to the slave application is a 32-bit I/O interface. This slave without configurable parameters needs no software or mailbox protocol. The EtherCAT State Machine is handled in the ESC. The boot-up information for the ESC comes out of the EEPROM that also supports the identity information of the slave. More complex slaves that are configurable have a host CPU on board. This CPU is connected to the ESC with an 8-bit or 16-bit parallel interface or via a serial connection.

**EtherCAT Slave Controller**

<sup>1</sup>Image courtesy of EtherCAT.org, [http://www.ethercat.org/pdf/english/etg2200\\_v2i0i1\\_slaveimplementationguide.pdf](http://www.ethercat.org/pdf/english/etg2200_v2i0i1_slaveimplementationguide.pdf)

The slave controllers typically feature an internal DPRAM and offer a range of interfaces for accessing this application memory:

- The SPI (serial peripheral interface bus) is intended particularly for devices with small process data quantity, such as analog I/O modules, sensors, encoders or simple drives.
- The parallel 8/16-bit microcontroller interface corresponds to conventional interfaces for fieldbus controllers with DPRAM interface. It is particularly suitable for more complex devices with larger data volume.
- The 32-bit parallel I/O interface is suitable for the connection of up to 32 digital inputs/outputs, but also for simple sensors or actuators operating with 32 data bits. Such devices do not need a host CPU at all (as shown in [Slave Hardware: FPGA with direct I/O](#)).

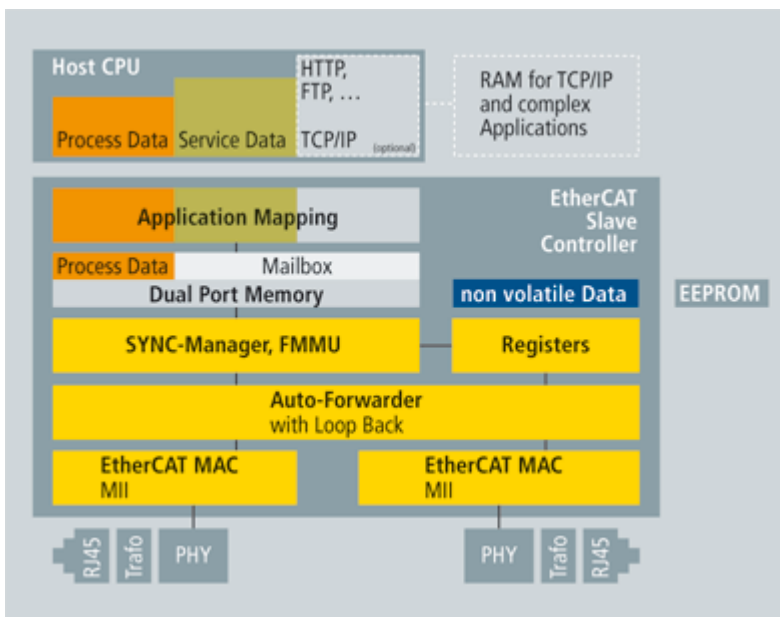


Figure 5-27: Slave Hardware: FPGA with Host CPU

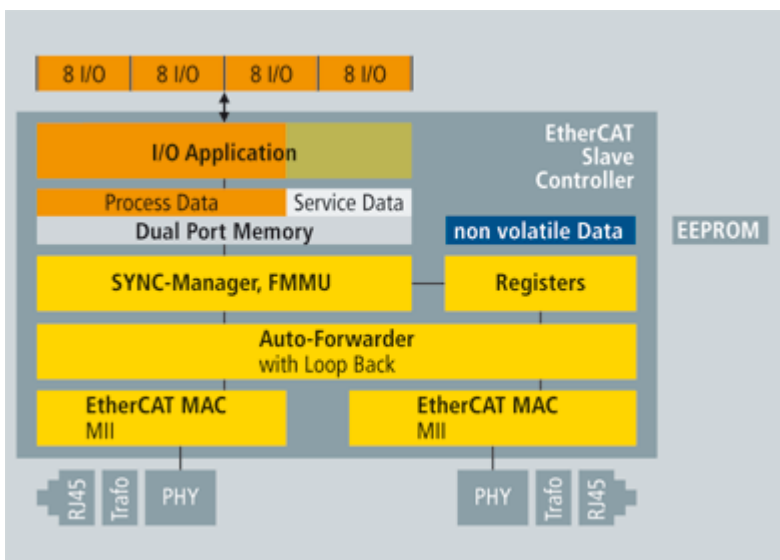


Figure 5-28: Slave Hardware: FPGA with direct I/O

### 8.3.3.3 EtherCAT State Machine

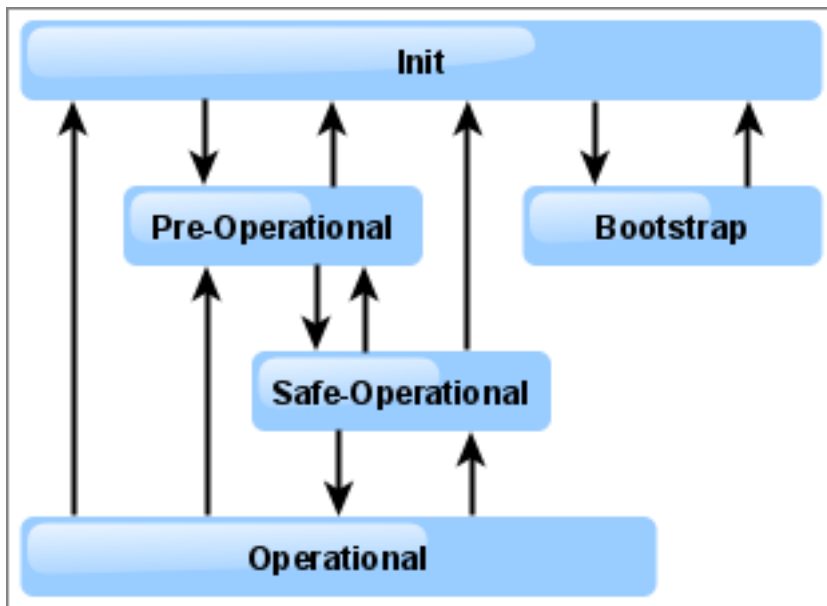


Figure 5-29: EtherCAT State Machine

Where the transitions are:

- from **Init** to **Pre-Operational (Pre-Op)**: Master configures the Sync Manager channels for Mailbox communication
- from **Pre-Op** to **Safe-Operational (Safe-Op)**: Master configures parameter using the Mailbox
- from **Safe-Op** to **Operational (Op)**: Master sends valid Outputs

The different states are:

- **Init**  
No communication on the Application Layer  
Master has access to the DL-Information registers
- **Pre-Operational (Pre-Op)**  
Mailbox communication on the Application Layer  
No Process Data communication
- **Safe-Operational (Safe-Op)**  
Mailbox communication on the Application Layer  
Process Data communication. Only Inputs are evaluated (Outputs in **Safe** state)
- **Operational (Op)**  
Inputs and Outputs are valid
- **Bootstrap**  
Recommended if firmware updates are necessary  
No Process Data communication  
Communication via Mailbox on Application Layer  
Only **FoE** protocol available

#### 8.3.3.4 PDOs for AKD, AKD2G, AKD-N, and S300/S700

The KAS Motion Engine interacts with the AKD, AKD-N, and S300/S700 drives through CANopen objects in the selected PDOs. Some of the ML and MC function blocks require specific CANopen objects in the PDO(s).

The default AKD and AKD-N PDO selection includes all of the CANopen objects needed by the motion engine and function blocks. The default S300/S700 PDOs include only the minimal CANopen objects required by the KAS motion engine.

The following tables identify which CANOpen objects are required by the motion engine and function blocks, and whether they are available in the default PDO for AKD, AKD-N, or S300/S700, and their corresponding AKD drive parameter. There are two types of PDOs:

- **RxPDO** — from Controller to Drive
- **TxPDO** — from Drive to Controller

**ⓘ IMPORTANT**

Check these tables to make sure the objects needed by the features your application uses are included in the AKD, AKD-N, and S300/S700 PDO selection.

**8.3.3.4.1 From Controller to Drive (RxPDO)**

*AKD Drive CANOpen Objects Available in the Default PDO*

Index - subindex	Object Name	Required	Associated ML FB	Associated MC FB	Associated AKD Parameter
0x20A4 - 0 or 0x2802 - 0	Latch control word	No	MLAxisCfgFastIn, MLAxisTimeStamp, all Trigger MLTrig FB	MC_TouchProbe, MC_AbortTrigger	CAP0.EN, CAP1.EN, CAP0.MODE, CAP1.MODE
0x3470 - 3	AOUT.VALUE (used by Onboard I/O mappings)	No	N/A	N/A	AOUT.VALUEU
0x6040 - 0	CANopen Control Word	Yes		MC_ClearFaults, MC_Power	
0x60B2 - 0 or 0x60F6 - 1	Additive torque value (Torque Feed Forward)	No	MLAxisAddTq	N/A <sup>1</sup>	IL.BUSFF
0x60C1 - 1 or 0x6062 - 0	Position demand value	Yes	Related to Axis pipe block positions (for more details, About Associated Data on Positions)	MC_ReadParam (ID 1)	PL.CMD
0x60FE - 1	Digital outputs (used by Onboard I/O mappings)	No	N/A	N/A	DOUTx.STATE

*AKD2G Drive CANOpen Objects Available in the Default PDO*

Index - subindex	Object Name	Required	Associated ML FB	Associated MC FB	Associated AKD Parameter
0x3601:1	Physical Outputs (used by Onboard I/O mappings)	No	N/A	N/A	DOUT#.STATE

<sup>1</sup>means Not Applicable

Index - subindex	Object Name	Required	Associated ML FB	Associated MC FB	Associated AKD Parameter
0x3800:2	AOUT1.Fieldbus Value (used by Onboard I/O mappings)	No	N/A	N/A	<a href="#">AOUT#.VALUE</a>
0x3801:2	AOUT2.Fieldbus Value (used by Onboard I/O mappings)	No	N/A	N/A	<a href="#">AOUT#.VALUE</a>
0x6040:0 / 0x6840:0	CANopen Control Word	Yes	Needed for motion engine axis to work properly	Needed for motion engine axis to work properly: <a href="#">MC_ClearFaults</a> , <a href="#">MC_Power</a>	
0x6070A:0 / 0x6870:0	Target Position Axis 1 / 2	Yes	Needed for motion engine to work properly; Related to Axis pipe block positions. For more details see <a href="#">About Associated Data on Positions</a>	Needed for motion engine axis to work properly: <a href="#">MC_ReadParam (ID 1)</a>	
0x60B8:0 / 0x68B8:0	Touchprobe function Axis 1 / 2	No	<a href="#">MLAxisCfgFastIn</a> , <a href="#">MLAxisTimeStamp</a> , all Trigger MLTrig FB	<a href="#">MC_TouchProbe</a> , <a href="#">MC_AbortTrigger</a> , <a href="#">MC_Reference</a> , <a href="#">MC_MachRegist</a> , <a href="#">MC_MarkRegist</a> , <a href="#">MC_StopRegist</a>	

**NOTE**

Axis 1 objects begin with 0x60, Axis 2 objects begin with 0x68.

*AKD-N Drive CANOpen Objects Available in the Default PDO*

Index - subindex	Object Name	Required	Associated ML FB	Associated MC FB	Associated AKD Parameter
0x20A4 - 0 or 0x2802 - 0	Latch control word	No	<a href="#">MLAxisCfgFastIn</a> , <a href="#">MLAxisTimeStamp</a> , all Trigger MLTrig FB	<a href="#">MC_TouchProbe</a> , <a href="#">MC_AbortTrigger</a>	CAP0.EN, CAP1.EN, CAP0.MODE, CAP1.MODE
0x6040 - 0	<a href="#">CANopen Control Word</a>	Yes		<a href="#">MC_ClearFaults</a> , <a href="#">MC_Power</a>	



Index - subindex	Object Name	Required	Associated ML FB	Associated MC FB	Associated AKD Parameter
0x60B2 - 0 or 0x60F6 - 1	Additive torque value (Torque Feed Forward)	No	MLAxisAddTq	N/A <sup>1</sup>	IL.BUSFF
0x60C1 - 1 or 0x6062 - 0	Position demand value	Yes	Related to <a href="#">Axis</a> pipe block positions (for more details, <a href="#">About Associated Data on Positions</a> )	MC_ReadParam (ID 1)	PL.CMD
0x60FE - 1	Digital outputs (used by Onboard I/O mappings)	No	N/A	N/A	DOUTx.STATE

*S300 and S700 Drive CANOpen Objects Available in the Default PDO*

Index - subindex	Object Name	Required	Associated ML FB	Associated MC FB	Associated AKD Parameter
0x20A4 - 0 or 0x2802 - 0	Latch control word	No	MLAxisCfgFastIn, MLAxisTimeStamp, all Trigger MLTrig FB	MC_TouchProbe, MC_AbortTrigger	CAP0.EN, CAP1.EN, CAP0.MODE, CAP1.MODE
0x6040 - 0	<a href="#">CANopen Control Word</a>	Yes		MC_ClearFaults, MC_Power	
0x60B2 - 0 or 0x60F6 - 1	Additive torque value (Torque Feed Forward)	No	MLAxisAddTq	N/A <sup>2</sup>	IL.BUSFF
0x60C1 - 1 or 0x6062 - 0	Position demand value	Yes	Related to <a href="#">Axis</a> pipe block positions (for more details, <a href="#">About Associated Data on Positions</a> )	MC_ReadParam (ID 1)	PL.CMD

**8.3.3.4.2 From Drive to Controller (TxPDO)**

*AKD Drive CANOpen Objects Available in the Default PDO*

Index - subindex	Object Name	Required	Associated ML FB	Associated MC FB	Associated AKD Parameter
0x6041 - 0	<a href="#">CANopen Status Word</a>	Yes	N/A	N/A	N/A

<sup>1</sup>means Not Applicable

<sup>2</sup>means Not Applicable

Index - subindex	Object Name	Required	Associated ML FB	Associated MC FB	Associated AKD Parameter
0x6063 - 0 or 0x6064 - 0	Position actual value	No	MLAxisFBackPos, MLAxisReadActPos	MC_ReadActPos	PL.FB
0x2050 - 0 or 0x35C9 - 0	Position actual value 2	No	MLAxisRead2ndFB  <b>NOTE</b> This FB is deprecated. We recommend reading the actual position of the axis which is assigned to the Sampler with this CoE object.	For a <b>Digitizing</b> axis: Secondary feedback can be read by reading the actual position of the axis which is assigned to the secondary feedback. Digitizing axes always use the second feedback for the Drive. KAS does not allow a digitizing axis on a drive which has not a servo axis already assigned	PL.FB (if DRV.CMDSOURCE = 1)
0x606C - 0	Velocity actual value	No	MLAxisReadVel	MC_ReadActVel	VL.FB
0x6077 - 0	Torque actual value	No	MLAxisReadTq	MC_ReadParam (ID 1016)	IL.FB
0x20A5 - 0 or 0x2901 - 0	Latch status word	No	MLAxisCfgFastIn, MLAxisTimeStamp, all Trigger MLTrig FB	MC_TouchProbe, MC_AbortTrigger	CAPx.STATE
0x20A6 - 0 or 0x2902 - 0	Latch position	No	MLAxisCfgFastIn, MLAxisTimeStamp, all Trigger MLTrig FB	MC_TouchProbe, MC_AbortTrigger	CAPx.T (for time) CAPx.PLFB (for position)
0x60FD - 0	Digital inputs (used by Onboard I/O mappings)	No	N/A	N/A	DIN.STATES

Index - subindex	Object Name	Required	Associated ML FB	Associated MC FB	Associated AKD Parameter
0x3470-4	AIN.VALUE (used by Onboard I/O mappings)	No	N/A	N/A	AIN.VALUE
0x60F4	Following error	No	MLAxisReadFEUU	MC_ReadParam (ID 1006)	PL.ERR

*AKD2G Drive CANOpen Objects Available in the Default PDO*

Index - subindex	Object Name	Required	Associated ML FB	Associated MC FB	Associated AKD Parameter
0x3780:1	AIN1.VALUE	No	N/A	N/A	
0x3781:1	AIN2.VALUE	No	N/A	N/A	
0x6041:0 / 0x6841:0	Statusword - Axis 1 / 2	Yes	Needed for the servo axis mapped to the drive to work properly	Needed for the servo axis mapped to the drive to work properly	
0x6064:0 / 0x6864:0	Position Actual Value - Axis 1 / 2	Yes	Needed for the servo axis mapped to the drive to work properly	Needed for the servo axis mapped to the drive to work properly	
0x6077:0 / 0x6877:0	Torque Actual Value - Axis 1 / 2	No	MLAxisReadTq	MC_ReadParam (ID 1016)	
0x60B9:0 / 0x68B9:0	Touch Probe Status Axis 1 / 2	No	MLAxisCfgFastIn, MLAxisTimeStamp, all Trigger MLTrig FB	MC_TouchProbe, MC_AbortTrigger, MC_Reference, MC_MachRegist, MC_MarkRegist, MC_StopRegist	
0x60D1:0 / 0x68D1:0	Touch Probe 1 Time Stamp positive value Axis 1 / 2	No	MLAxisCfgFastIn, MLAxisTimeStamp, all Trigger MLTrig FB	MC_TouchProbe, MC_AbortTrigger, MC_Reference, MC_MachRegist, MC_MarkRegist, MC_StopRegist	

Index - subindex	Object Name	Required	Associated ML FB	Associated MC FB	Associated AKD Parameter
0x60D4:0 / 0x68D4:0	Touch Probe 2 Time Stamp positive value Axis 1 / 2	No	MLAxisCfgFastIn, MLAxisTimeStamp, all Trigger MLTrig FB	MC_TouchProbe, MC_AbortTrigger, MC_Reference, MC_MachRegist, MC_MarkRegist, MC_StopRegist	
0x60E4:3	3rd additional position actual Value Axis 1	No	MLSmpConECAT	For a digitizing axis	
0x60E4:4	4th additional position actual Value Axis 1		MLSmpConECAT	For a digitizing axis	
0x60E4:5	5th additional position actual Value Axis 1		MLSmpConECAT	For a digitizing axis	
0x60F4:0 / 0x68F4:0	Following error actual value Axis 1 / 2	No	MLAxisReadFEUU	MC_ReadParam (ID 1006)	

**NOTE**

Axis 1 objects begin with 0x60, Axis 2 objects begin with 0x68.

*AKD-N Drive CANOpen Objects Available in the Default PDO*

Index - subindex	Object Name	Required	Associated ML FB	Associated MC FB	Associated AKD Parameter
0x6041 - 0	CANopen Status Word	Yes	N/A	N/A	N/A
0x6063 - 0 or 0x6064 - 0	Position actual value	No	MLAxisFBackPos, MLAxisReadActPos	MC_ReadActPos	PL.FB
0x606C - 0	Velocity actual value	No	MLAxisReadVel	MC_ReadActVel	VL.FB
0x6077 - 0	Torque actual value	No	MLAxisReadTq	MC_ReadParam (ID 1016)	IL.FB
0x20A5 - 0 or 0x2901 - 0	Latch status word	No	MLAxisCfgFastIn, MLAxisTimeStamp, all Trigger MLTrig FB	MC_TouchProbe, MC_AbortTrigger	CAPx.STATE

Index - subindex	Object Name	Required	Associated ML FB	Associated MC FB	Associated AKD Parameter
0x20A6 - 0 or 0x2902 - 0	Latch position	No	MLAxisCfgFastIn, MLAxisTimeStamp, all Trigger MLTrig FB	MC_TouchProbe, MC_AbortTrigger	CAPx.T (for time) CAPx.PLFB (for position)
0x60FD - 0	Digital inputs (used by Onboard I/O mappings)	No	N/A	N/A	DIN.STATES
0x60F4	Following error	No	MLAxisReadFEUU	MC_ReadParam (ID 1006)	PL.ERR

*S300 and S700 Drive CANOpen Objects Available in the Default PDO*

Index - subindex	Object Name	Required	Associated ML FB	Associated MC FB	Associated AKD Parameter
0x6041 - 0	CANopen Status Word	Yes	N/A	N/A	N/A
0x6063 - 0 or 0x6064 - 0	Position actual value	No	MLAxisFBackPos, MLAxisReadActPos	MC_ReadActPos	PL.FB
0x2050 - 0 or 0x35C9 - 0	Position actual value 2	No	MLAxisRead2ndFB	For a Digitizing axis: Secondary feedback can be read by reading the actual position of the axis which is assigned to the secondary feedback. Digitizing axes always use the second feedback for the Drive. KAS does not allow a digitizing axis on a drive which has not a servo axis already assigned	PL.FB (if DRV.CMDSOURCE = 1)
0x6077 - 0	Torque actual value	No	MLAxisReadTq	MC_ReadParam (ID 1016)	IL.FB

**NOTE**  
This FB is deprecated. We recommend reading the actual position of the axis which is assigned to the Sampler with this CoE object.

Index - subindex	Object Name	Required	Associated ML FB	Associated MC FB	Associated AKD Parameter
0x20A5 - 0 or 0x2901 - 0	Latch status word	No	MLAxisCfgFastIn, MLAxisTimeStamp, all Trigger MLTrig FB	MC_TouchProbe, MC_AbortTrigger	CAPx.STATE
0x20A6 - 0 or 0x2902 - 0	Latch position	No	MLAxisCfgFastIn, MLAxisTimeStamp, all Trigger MLTrig FB	MC_TouchProbe, MC_AbortTrigger	CAPx.T (for time) CAPx.PLFB (for position)
0x60F4	Following error	No	MLAxisReadFEUU	MC_ReadParam (ID 1006)	PL.ERR

### 8.3.3.4.3 Examples

Below are three examples where the PDO object is passed as an argument in the function block.

```
MLSmpConECAT ( PipeNetwork.SMP1, 1001, 16#2050, 0 );
```

Where:

- PipeNetwork.SMP1 is a Sampler block ID.
- 1001 is the EtherCAT address for the first device
- 16#2050 (0x2050) is the object index and 0 is the subindex for the AKD's "Position actual value 2" object.

```
MLCNVConnectEx ( PipeNetwork.CNV1, PipeNetwork.AXIS1, EC_ADDITIVE_TORQUE_VALUE, 0 ) ;
```

The argument is a constant based on the object index.

```
MLSmpConPNAxis ( PipeNetwork.SMP1, PipeNetwork.AXIS1, ML_SECOND_FEEDBACK_POSITION );
```

Where:

ML\_SECOND\_FEEDBACK\_POSITION specifies the secondary actual position from the Pipe Network Axis.

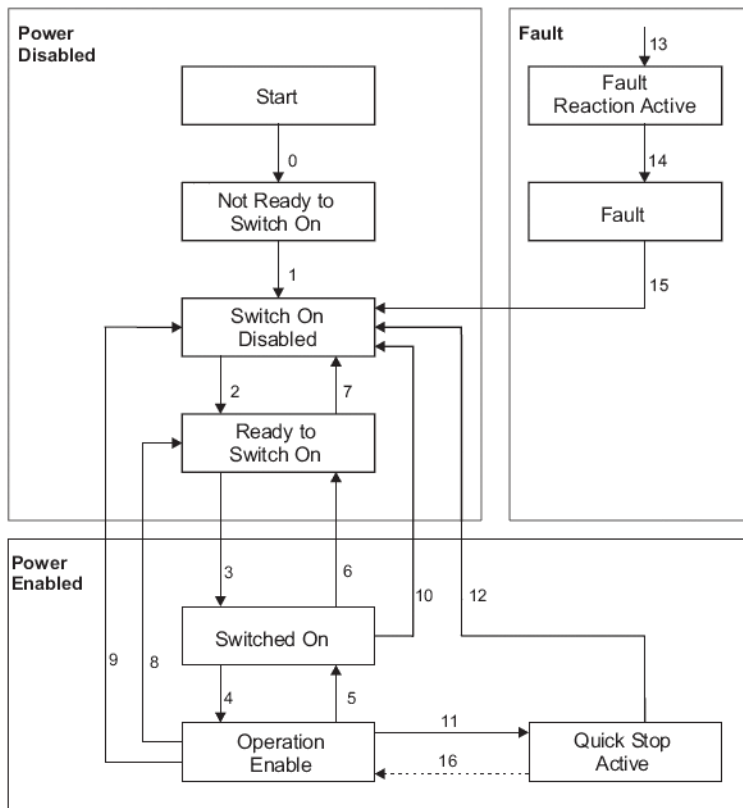
#### NOTE

Please note that the ML\_SECOND\_FEEDBACK\_POSITION constant has been deprecated and only AKD devices supported it. It is deprecated because the AKD2G supports more than one secondary feedback.

### 8.3.4 CANopen

#### 8.3.4.1 CANopen Status Machine

The states of the status machine can be revealed by using the [CANopen Status Word](#).



**Figure 5-30: CANopen Status Machine**

The start state is a pseudo-state indicating the start when the state machine is activated during the start-up sequence of the device drive's application software.

#### Status description

Status	Description
Not ready to switch on	The drive is not ready to switch on. The controller has not indicated readiness for service. The drive is still in the boot phase or in the fault status
Switch on disabled	The drive cannot be enabled via the EtherCAT interface; because for example there is no connection to a power source
Ready to switch on	The drive can be enabled via the control word. DC-link voltage can be switched on, parameters can be transferred, motion functions cannot be performed yet.
Switched on	The drive is enabled but the setpoints are not yet transferred from the EtherCAT interface. The drive is idle. DC-link voltage must be switched on, parameters can be transferred, but motion functions cannot be performed yet. Output stage is switched on (enabled). Operation Enable No fault present; output stage is enabled; motion functions are enabled.
Operation enabled	The drive is enabled and the setpoints are transferred from the EtherCAT interface. No fault present; output stage is enabled; motion functions are enabled.

Status	Description
Quick stop active	The drive has been stopped with the quick stop ramp; output stage is enabled; motion functions are not enabled.
Fault reaction active	A fault has occurred and the drive is stopped with the emergency stop ramp
Fault	A fault is active, and the drive has been stopped and disabled

Table 5-5: Status Description

### Transitions of the status machine

The drive device supports the transitions and actions as listed in the table below. The event initiates the transition. The transition is terminated after the action has been performed.

Transition	Event	Action
0	Automatic transition after power-on or reset application	Drive device self-test and/or self initialization has to be performed.
1	Automatic transition	Communication has to be activated.
2	Shutdown command from control device or local signal	None
3	Switch on command received from control device or local signal	The high-level power has to be switched on, if possible.
4	Enable operation command received from control device or local signal	The drive function has to be enabled and all internal setpoints cleared.
5	Disable operation command received from control device or local signal	The drive function has to be disabled.
6	Shutdown command received from control device or local signal	The high-level power has to be switched off, if possible.
7	Quick stop or disable voltage command from control device or local signal	None
8	Shutdown command from control device or local signal	The drive function has to be disabled, and the high-level power has to be switched off, if possible.
9	Disable voltage command from control device or local signal	The drive function has to be disabled, and the high-level power has to be switched off, if possible.
10	Disable voltage or quick stop command from control device or local signal	The high-level power has to be switched off, if possible.
11	Quick stop command from control device or local signal	The quick stop function has to be started.
12	Automatic transition when the quick stop function is completed and quick stop option code is 1, 2, 3 or 4, or disable voltage command received from control device (depends on the quick stop option code)	The drive function has to be disabled, and the high-level power has to be switched off, if possible.
13	Fault signal	The configured fault reaction function has to be executed.
14	Automatic transition	The drive function has to be disabled; the high-level power has to be switched off, if possible.



Transition	Event	Action
15	Fault reset command from control device or local signal	A reset of the fault condition is performed, if no fault exists currently on the drive device; after leaving the Fault state, the Fault reset bit in the control word has to be cleared by the control device.
16	Enable operation command from control device, if the quick stop option code is 5, 6, 7, or 8	The drive function has to be enabled.

Table 5-6: Transition Events and Actions

### 8.3.4.2 CANopen Control Word

The status machine for the control word corresponds to the CANopen status machine.

The control word indicates the received command controlling the state machine. It is only read during **Operational** status. The control commands allow the manipulation of the state of a drive by setting its control word. Such commands are built up from the logical combination of the bits in the control word and external signals (e.g. enable output stage).

Bits definition of the control word

Bit	Name
0	Switch on
1	Disable Voltage
2	Quick Stop
3	Enable Operation
4	Operation mode specific
5	Operation mode specific
6	Operation mode specific
7	Reset Fault (only effective for faults)
8	Pause/halt
9	reserved
10	reserved
11	reserved
12	reserved
13	Manufacturer-specific
14	Manufacturer-specific
15	Manufacturer-specific

Table 5-7: Bit Assignment in Control Word

The commands are coded as given in the table below.

Command	Bits of the control word					Transitions
	Bit 7	Bit 3	Bit 2	Bit 1	Bit 0	
Shutdown	0	X	1	1	0	2,6,8
Switch on	0	0	1	1	1	3
Switch on + enable operation	0	1	1	1	1	3 + 4 (Note)
Disable voltage	0	X	X	0	X	7,9,10,12
Quick stop	0	X	0	1	X	7,10,11
Disable operation	0	0	1	1	1	5
Enable operation	0	1	1	1	1	4,16
Fault reset	up	X	X	X	X	15

Table 5-8: Command Coding

Note: automatic transition to Enable operation state after executing SWITCHED ON state functionality.

Bits marked by an X are irrelevant.

### 8.3.4.3 CANopen Status Word

The status machine for the control word corresponds to the CANopen status machine.

The status word is only updated and written by the drive in **Safe-Op** and **Operational** states.

Bits definition of the status word

Bit	Name
0	Ready to switch on
1	Switched on
2	Operation enable
3	Fault
4	Voltage enabled
5	Quick stop
6	Switch on disabled
7	Warning
8	Manufacturer-specific (reserved)
9	Remote (always 1)
10	Target reached
11	Internal limit active
12	Operation mode specific (reserved)
13	Operation mode specific (reserved)
14	Manufacturer-specific (reserved)
15	Manufacturer-specific (reserved)

Table 5-9: Bit Assignment in Status Word

The bit combinations coding the following states are listed in the table below.

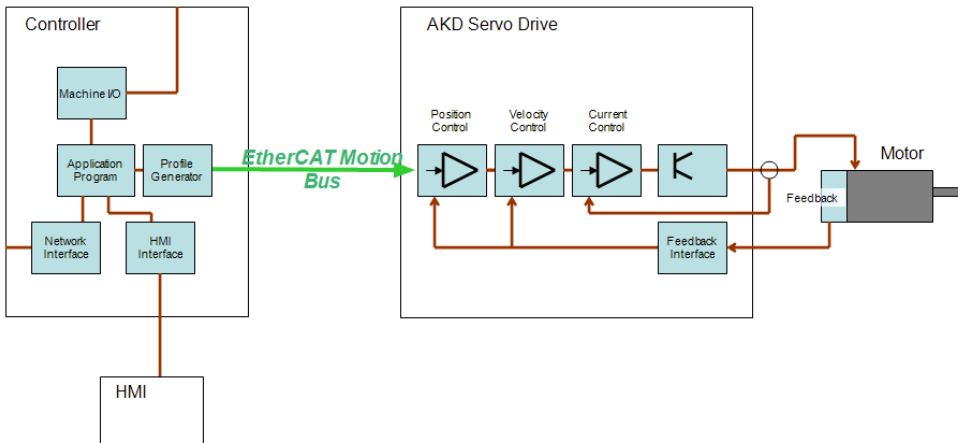
Status word MSB (15..12) (11 .. 8) (7 .. 4) (3 .. 0) LSB	State
xxxx xxxx x0xx 0000	Not ready to switch on
xxxx xxxx x1xx 0000	Switch on disabled
xxxx xxxx x01x 0001	Ready to switch on
xxxx xxxx x01x 0011	Switched on
xxxx xxxx x01x 0111	Operation enabled
xxxx xxxx x00x 0111	Quick stop active
xxxx xxxx x0xx 1111	Fault reaction active
xxxx xxxx x0xx 1000	Fault

Table 5-10: State Coding

Bits marked by an X are irrelevant

## 8.4 AKD Drive

The **servo loops** in a KAS system are located within the AKD Drive. The **profile generator** used for all the motion in your application is located in the controller (AKD PDMM or PCMM).



### 8.4.1 AKD Drive Configuration

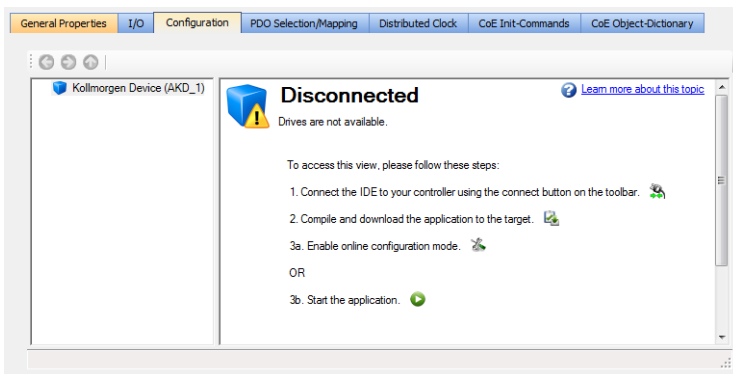
#### 8.4.1.1 Connection Modes

The AKD drive parameters can be configured when the KAS IDE is connected to the controller and operating in one of two functional modes:

- Online Configuration Mode
- Project Running on the Controller

#### Disconnected

If the controller is not operating in one of the above modes, then the AKD(s) configurations are not accessible. The Configuration tab will indicate the AKD is Disconnected:



#### Online Configuration Mode

In Online Configuration mode parameters are updated directly to the AKD2G or AKD with EtherCAT is in Pre-Operational mode. When you modify the value of a parameter, a command is sent to the drive and the corresponding parameter is updated.

#### Project Running on the Controller

When a project is running on the controller, parameters are updated directly to the AKD with EtherCAT in Operational mode. When you modify the value of a parameter, a command is sent to the drive and the corresponding parameter is updated. Beware that your PLC application is running and can also read or write parameters to the AKD.


**NOTE**  
An AKD drive cannot be deleted while it is connected.

**NOTE**  
The AKD drive leave [Online Configuration mode](#) and start an the application if you start an application while the drive is in that mode.

**NOTE**  
**Important!** Before leaving [Online Configuration mode](#) (either by clicking the Online Configuration button or clicking the Start button when the device is in Online Configuration mode) you should ensure the following conditions exist.

1. Drive is not enabled
2. Service motion is not enabled
3. Auto-tuning is not enabled

**Recommendation:** Clicking on the Disable button will force the above scenario.



**8.4.1.2 AKD Configuration According to EtherCAT State**

The drive configuration can only take place when EtherCAT is in the following state: [Pre-Op](#) or [Op](#).

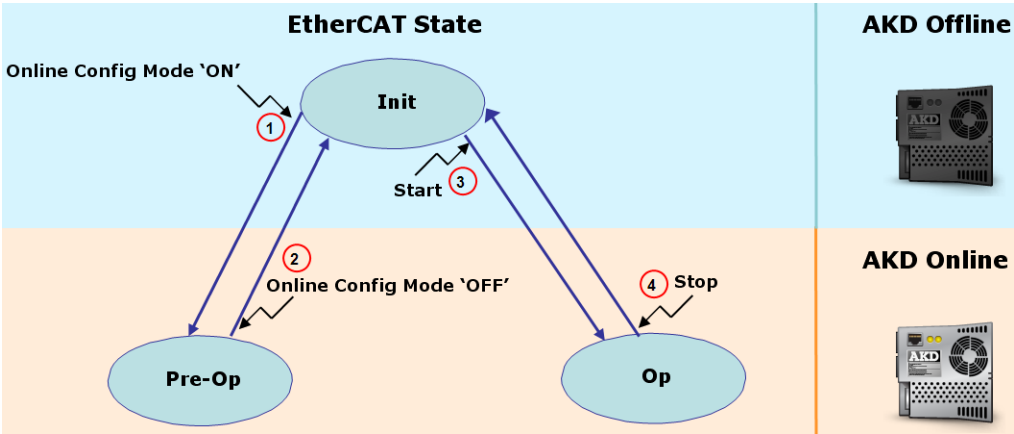


Figure 5-31: AKD Configuration According to EtherCAT State

Action	Name	Description
1	Online Configuration Mode "ON"	Online Configuration Mode (to see where you can access this button in the KAS IDE, see <a href="#">"Online Configuration Mode" on page 252</a> ) Sets the EtherCAT fieldbus to the Pre-Op state and allows AKD configuration (for procedure, see <a href="#">"Configure the AKD2G/AKD Drive" on page 176</a> )
2	Online Configuration Mode "OFF"	Sets the EtherCAT fieldbus to the Init state and does not allow AKD configuration. All AKD drives are disconnected from the IDE.

Action	Name	Description
3	Start the Project	Sets the EtherCAT fieldbus to the <a href="#">Operational</a> state <sup>1</sup> (to see where you can access this button, see " <a href="#">Menus and Toolbar Overview</a> " on <a href="#">page 241</a> ) This <a href="#">will start the application program running and allows</a> AKD configuration (with some restrictions for the views: Service Motion and Performance Servo Tuner)
4	Stop the Project	This step stops the application from running, sets the EtherCAT fieldbus to the Init state, and does not allow AKD configuration. All AKD drives are disconnected from the IDE.

Table 5-11: AKD Drive - List of Actions

---

<sup>1</sup>Depending on the number of AKD drives physically present in the EtherCAT network, the KAS IDE can slow down when getting data.

The KAS Runtime is **not concerned** with this limitation.

## 8.5 Tasking Model / Scheduling

In the KAS Runtime, both the Motion and Programmable Logic Controller (PLC) Programs are run every cycle. The cycle update time is set when configuring the EtherCAT motion bus (see [EtherCAT Master Settings Tab](#)).

The cycle time becomes effective only when the Motion Engine is started (i.e. when the PLC code initializes the Motion by calling the MLMotionIni function block), and the application runs on a controller.

The time base remains much longer than the cycle time as long as the Motion Engine is **not** yet started, or if the application runs on the KAS Simulator (for more details, [When the application runs on the KAS Simulator, the PLC execution rate is approx. 10 milliseconds. KAS Simulator cannot execute the PLC programs faster because Windows is not able to handle timing less than 10ms..](#)). In these cases, the PLC execution rate is approx. 10 milliseconds.

### 8.5.1 Priority Between Motion and PLC

The Motion computation is always executed each cycle, and occurs before executing the PLC programs application. The figure below shows the execution in the following order:

1. Motion command, position feedback from each axis, and other elements in the EtherCAT PDO object are sent and received on the EtherCAT motion bus (this includes servo drives and Remote I/O)
2. Motion engines are executed
3. I/O related to the PLC program are serviced (for more details, see ["EtherCAT Processing Time" on page 463](#))
4. PLC programs are executed
5. NVRAM variables are saved (for more details, see ["NVRAM Processing Time" on page 464](#))

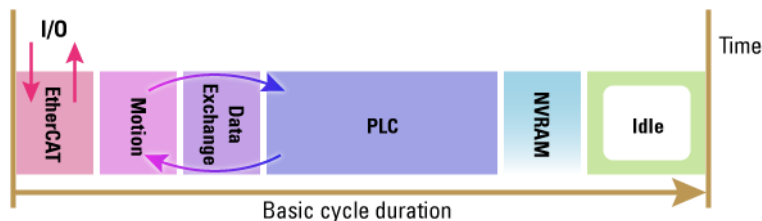


Figure 5-32: Priority Between Motion and PLC

#### NOTE

The Motion time (see figure above) must be shorter than the basic cycle duration at each cycle. This condition is checked at each cycle and if the cycle is overran, Kollmorgen Automation Suite generates a fatal error and the application execution is stopped.

#### 8.5.1.1 EtherCAT Processing Time

The EtherCAT frame is executed at the beginning of the cycle. During this period, all the values related to EtherCAT (see ["PDOs for AKD, AKD2G, AKD-N, and S300/S700" on page 446](#)) are exchanged, including:

- **Inputs** are read
- **Outputs** are set

Based on the I/O mapping to PLC variables (see ["Map Input and Output to Variables" on page 312](#)), the I/Os are updated before they are effectively used during the PLC period.

As a consequence, when the PLC variables set an Output, it is updated during the EtherCAT frame of the next cycle.

### 8.5.1.1.1 About Variation during the EtherCAT Processing

The EtherCAT period is subject to time variation along the cycles due to the following reasons:

- Some EtherCAT function blocks use asynchronous SDO communication, which is not deterministic.
- Some EtherCAT slave devices support mailbox protocols. The master cyclically reads the mailbox of the EtherCAT slaves (polling of mailbox is performed every 50 cycles and is spread on several cycles depending on the number of EtherCAT slaves)

See also the [FAQs](#) for information about SDO communication.

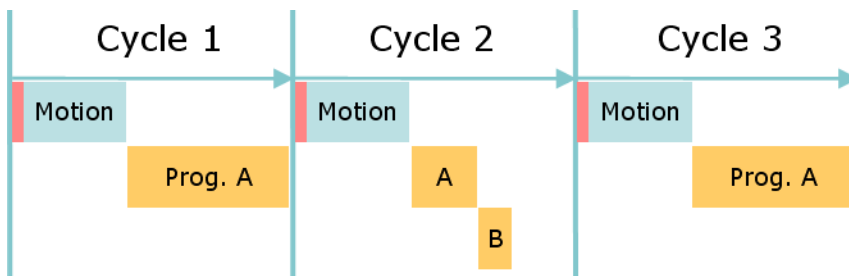
### 8.5.1.2 NVRAM Processing Time

Due to slow processing when saving Retain [Variables](#) to the NVRAM, this action is not performed each cycle. The save operation is performed in the background every 20 seconds (frequency increases to each 2 seconds when the application is running).

When executed during a cycle, it occurs after the PLC period.

### 8.5.1.3 What happens when a PLC Program is overrunning the Cycle Duration

Large application can require more than one cycle to completely execute all the PLC programs.



**Figure 5-33:** Application Overrunning the Basic Cycle

The figure above shows an example of an application with two PLC programs (A and B). It takes two cycles to execute all the code in the two programs.

- Cycle 1 executes most of Prog. A
- Cycle 2 finishes Prog. A and executes Prog. B, which is set to run with a cycle period of every other cycle.
- Cycle 3 starts executing Prog. A again

#### NOTE

Even if there is time left over in the cycle, execution of Prog. A does not start until the next cycle

An application overrun has no effect other than a short delay in the application execution. Execution of the real-time application is recovered as soon as the overload disappears.

#### ⚠ IMPORTANT

If Outputs are set when a program runs over several basic cycles, unexpected and potentially dangerous effects can happen.

#### NOTE

When running with the KAS Simulator, there is no overrunning because the cycle is extended to include all the PLC programs, right after the Motion computation.

See Also: [Priority Between PLC Programs](#)



### 8.5.2 Priority Between PLC Programs

In turn, PLC programs are assigned a priority. At times of heavy demand for processing time, the operating system serves programs with higher priority first.

For more details, see how to:

- Set the [PLC cycling](#)

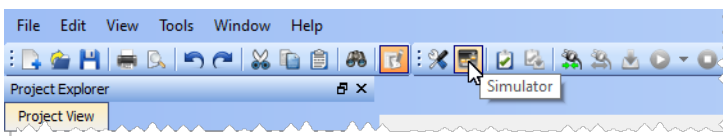
See Also: [Priority Between Motion and PLC](#)

## 9 Using the KAS Simulator



### 9.1 Start KAS Simulator

The **KAS Simulator** is launched from within the KAS IDE. Click on the KAS Simulator button in the toolbar.



Clicking the button will perform the following:

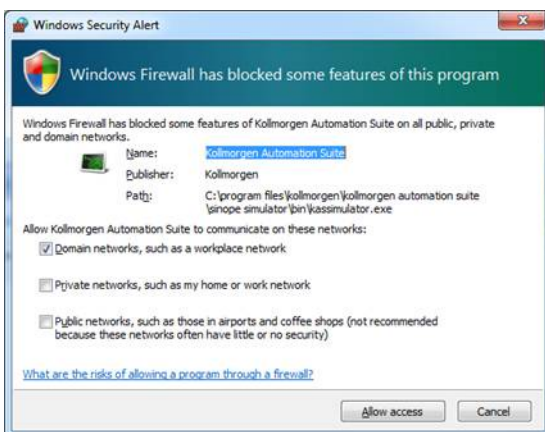
1. Change the KAS project to use KAS Simulator.
2. Launch the KAS Simulator.

#### **TIP**

Each version of KAS IDE has a matching version of Simulator which it will open. Only one instance of KAS Simulator may be run at a time.

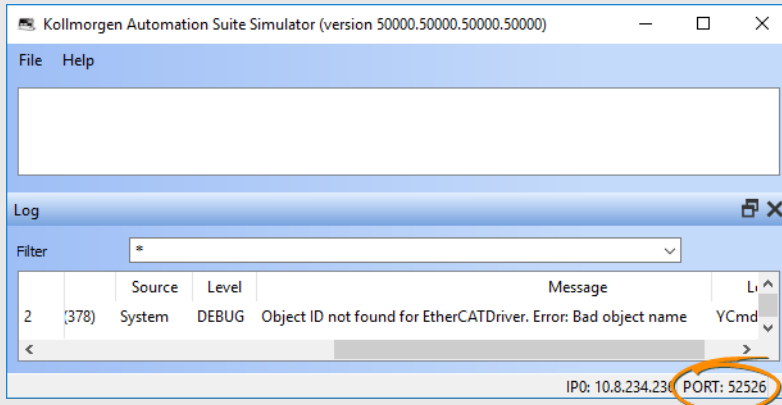
3. Connect the KAS IDE to the KAS Simulator.

The first time KAS Simulator is run it will attempt to open some TCP/IP ports to allow communication. Your system's firewall will detect this and prompt for an action. Allow the KAS Simulator to open the ports by selecting Unblock (Windows XP) or Allow Access (Windows 7).



**Figure 5-34:** Firewall alert dialog.**NOTE**

Simulator normally uses port 80 for the web server. If another service is using port 80, then Simulator will automatically assign a free port available to the webserver to listen to the incoming HTTP requests. The port number assigned for the Simulator web server will be displayed in the status bar.



- If an alternate port is assigned to the web server in the Simulator for HTTP communication, then the HTTP URL should also include the port number to access the web server. For example, in the above image the web server is listening to port 52526. To connect to the Simulator from the web browser use the URL: `http://127.0.0.1:52526`
- Depending on the applications running on the host computer, the alternate port may vary every time the simulator is started.

**TIP**

Only one instance of Simulator may be used at a time, but multiple instances of the KAS IDE may be used with the Simulator. A simple rule to remember is whatever instance is connected to Simulator, owns Simulator.

**9.1.1 KAS Runtime Log Window**

The KAS Runtime Log window provides a running display of activity related to the execution of the application. Items displayed include application startup and initialization information.

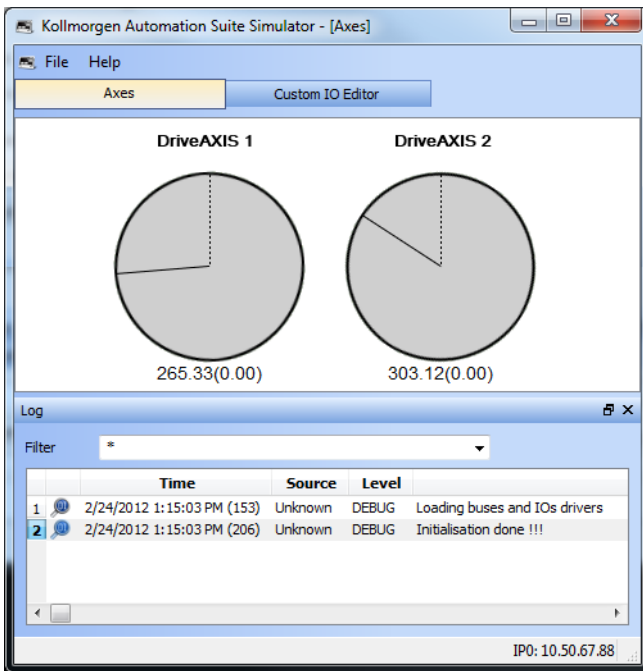


Figure 5-35: KAS Runtime Log Window

#### NOTE

Some of the steps performed during the initialization process can be specified in the **initscript.bin** file located under:  
 C:\Program Files (x86)\Kollmorgen\Kollmorgen Automation Suite\Sinope Simulator\Resources

See also [KAS Simulator log window](#)

## 9.2 Stop KAS Simulator

The **KAS Simulator** can be closed in several ways:

- from within the KAS IDE by clicking on the KAS Simulator button in the toolbar.
- by selecting File > Exit.
- by clicking on the Close button at the top right of the KAS Simulator.



Clicking the KAS Simulator button will perform the following actions.

1. Stop the project if it is running.
2. Disconnect the KAS IDE from the KAS Simulator.
3. Close the KAS Simulator.
4. Change the project back to using a PxMM as the controller.

Selecting File > Exit or the Close button will stop the project and disconnect the KAS IDE from the KAS Simulator, but the project will not reset the controller.

#### TIP

After the project is debugged using KAS Simulator, it can be downloaded to the real controller in production. This operation can be done simply by modifying the IP address of the device and then [Download the Application](#).

**TIP**

Only one instance of Simulator may be used at a time, but multiple instances of the KAS IDE may be used with the Simulator. A simple rule to remember is whatever instance is connected to Simulator, owns Simulator.

### 9.3 KAS Simulator Axes Tab

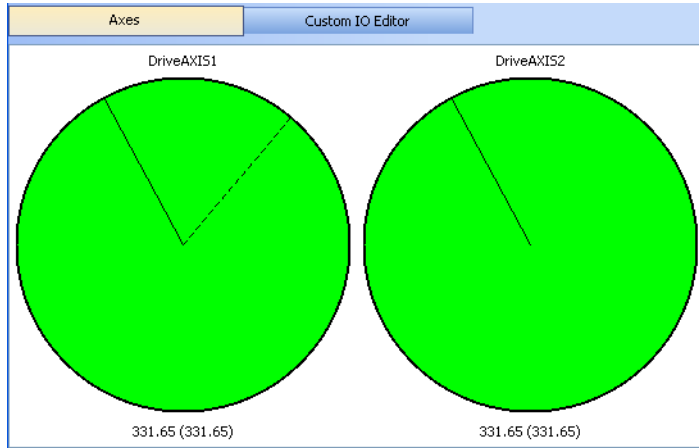


Figure 5-36: Axes Tab

The dashed line (or dotted line) represents the **Reference Position** in User units. When the is solid line (or normal line) visible, it represents the **Actual Position** in User units.

Below the disk, the reference position for the associated axis is represented in the following format:

**Range** value (**Modulo** value according to the periodicity)

As shown on the figure below, the **Error** command (in the contextual menu of the axis tab) is used to simulate an error on an axis (then you can see the impact on the HMI and implement counter-measures if necessary).

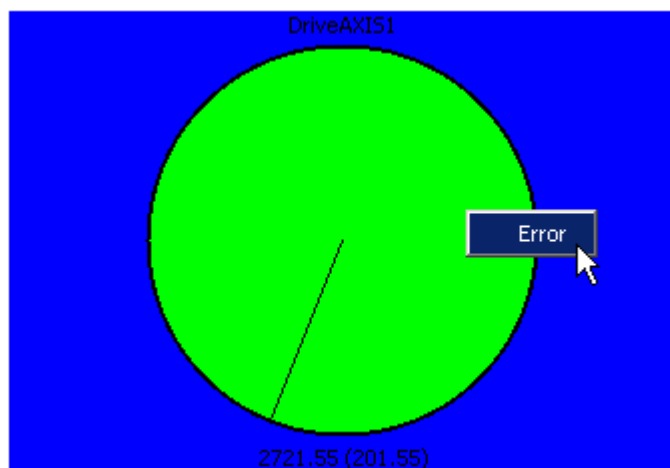


Figure 5-37: Set Axis in Error Mode

The drive becomes Red when it is set to **Error** . **Motion State Machine**

To deselect an axis already selected (blue rectangle), click on the white surrounded outside border of the axis tab.

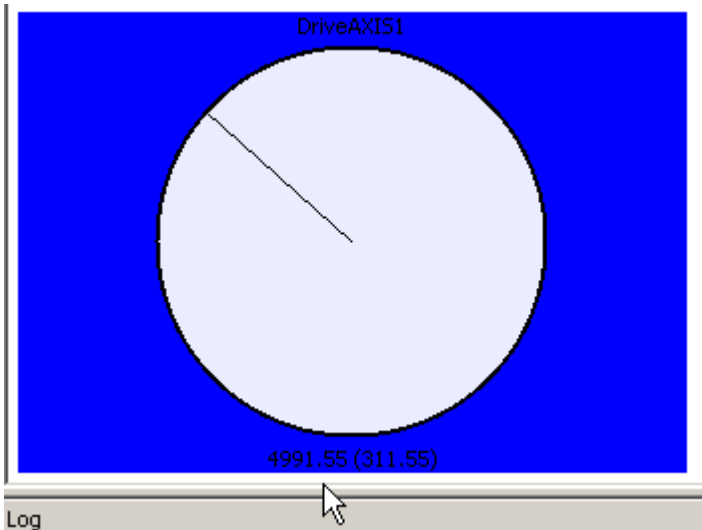


Figure 5-38: Deselect an Axis

## 9.4 Describing KAS Simulator Graphical User Interface

### 9.4.1 Windows Overview

#### 9.4.1.1 Main window

KAS Simulator main window contains:

- The **menu** bar (see call out 1)
- The **workspace** (2)
- The **Log** window (3)

In addition, the workspace displays the **Axes**.

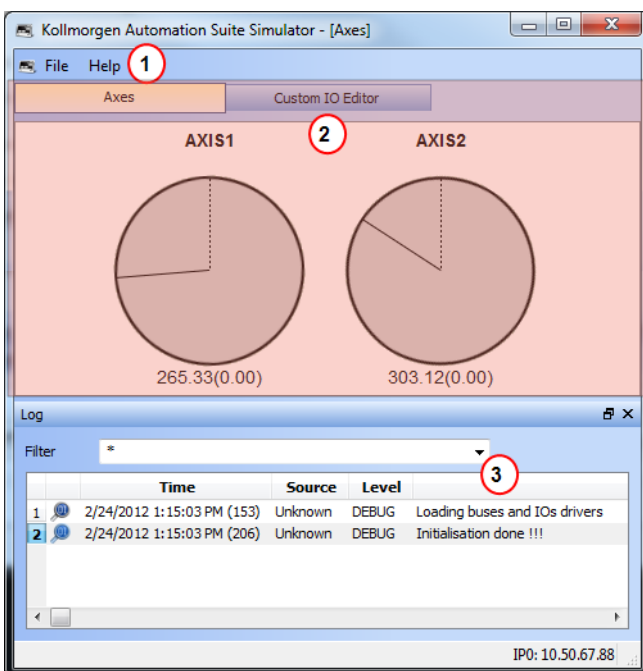


Figure 5-39: KAS Simulator Main Window

### 9.4.1.2 KAS Simulator log window

This Log window shows all log messages related to the KAS Simulator. Error and warning messages issued from the operating system, as well as [Printf Function](#) instructions, are also placed on this window.

Every log message includes the following:

- Timestamp
- Source
- Logging Level
- Message

## 9.5 KAS Simulator Menus Description

### 9.5.1 File Menu

Command	Description
<b>Start</b>	Start the application with the <a href="#">Retain Variables</a> .
<b>Cold Start</b>	Start the application with the initial settings
<b>Stop</b>	Stop the application
<b>Option</b>	Set parameters for the KAS Simulator application (see explanations below)
<b>Exit</b>	Leave KAS Simulator application

#### Option

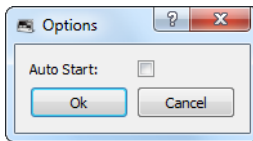


Figure 5-40: Options for KAS Simulator

Option	Description
<b>Autostart</b>	Autostart the application when KAS Simulator is launched. Autostart mode is recommended when a system is in production.

#### NOTE

Parameters are saved in the **Options.bin** file located under:  
 <user>\AppData\Local\Kollmorgen\KAS\Sinope Simulator\Resources\

### 9.5.2 Help Menu

Command	Description
<b>About</b>	Show version numbers and other <a href="#">Find KAS IDE Version</a> about KAS Simulator

## 10 Using the AKD PDMM or PCMM Controller



Tasks related to the controllers are:

- Configure parameters
- Start and stop your KAS application
- Update the firmware
- Reset to factory settings

Rebooting the controller, recovering the firmware, and resetting the controller may be performed from the device or, more conveniently, using the web server.

### NOTE

Please note that any reference to AKD PDMM or PCMM refers to both the 800MHz and the single- and dual-core 1.2GHz variants, unless otherwise noted.



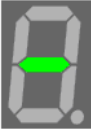







## 10.1 Booting the PDMM or PCMM

This topic explains the boot sequence for the controller, which is based on the RAM and the Flash memory.

The flash memory contains two images:

- Recovery image (4 Mb) contains QNX operating system and the KAS web server
- Regular image (9 Mb) contains QNX operating system, the KAS web server, and the KAS Runtime

### 10.1.1 Boot Sequence

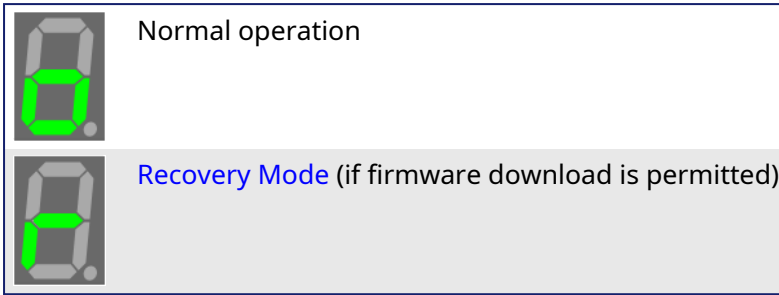
State	Display	Description
Hardware power on		Controller option card has power
Stage 0		Reached after the I <sup>2</sup> C is initialized
Stage 1		Reached after the DDR3 ram memory is initialized
Stage 2		Reached just after the RAM memory relocation At this point the boot is running in DDR3 RAM memory
Stage 3		Reached after the flash memory is initialized
Boot Startup Script		After all the previous steps, the startup script starts automatically.
QNX startup		Reached after the Boot startup script is finished
Sysinit		Reached after specific configuration parameters of the target are loaded, and after the network is started using the <a href="#">rotary switch</a> .

#### NOTE

The controller may be booted with or without an Ethernet cable attached.

When the controller is booted with a cable attached the configured IP address (depending upon the current position of the rotary switch) will be displayed in the 7-segment display (see [Display the IP Address of a PDMM or PCMM](#)). If the controller is started without a network connection then the IP address will not be displayed.

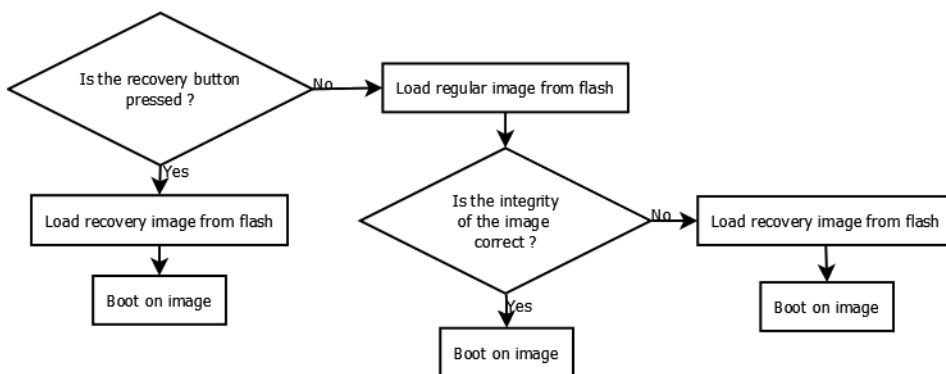
After the boot sequence is successful, the controller will be in one of two modes:



### 10.1.2 Boot Startup Script

After all the previous steps, the startup script starts automatically. The script first puts the 7-segment display into stage 4.

Before the controller boots up, the following flowchart applies:



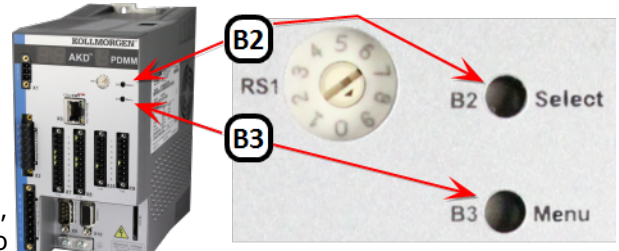
### 10.1.3 Booting from the Recovery Image

<b>Automatic Mode</b>	The boot from the recovery image is done automatically if the regular image is corrupted.	
<b>Manual Mode</b>	If the controller starts booting normally but freezes after the startup script (see image to the right), then you have to boot manually from the recovery image by pressing the recovery button (B2). See <a href="#">Booting from the Recovery Image</a> for more information.	

**NOTE**  
 The controller will display a "backward C" on the 7-segment display if the wrong firmware is installed. You must enter recovery mode to load the correct firmware file.

## 10.2 Working with the Hardware

In some cases, using the buttons on the controller may be preferable to using the web server. On the front of the controller there are two buttons, B2 and B3. B2 is above B3. These buttons may be used to enter Recovery Mode (see "Booting from the Recovery Image" on page 476), Display the IP Address of a PDMM or PCMM, stop and start the application, reset the control to factory settings (see "Reset the Controller to Factory Settings" on page 477), and backup/restore the firmware.



Result	Press and hold
B2	Recovery Mode
B3	Menu access

Table 6-1: B2/B3 button functionality at start-up

Press	Result
B2	Menu option selector
B3	Menu access

Table 6-2: B2/B3 button functionality while running

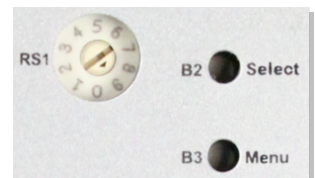
### 10.2.1 AKD PDMM or PCMM Memory

The AKD PDMM is equipped with ample memory to handle the most challenging programs.

Memory Type	Amount	Purpose
Flash	64 MB	Non-volatile memory
SD Card slot	2+ GB	Backup and Restore functionality as well as moving data. See <a href="#">SD Card Support</a> for more information.
DDR RAM	256 MB	everything else

### 10.2.2 PDMM or PCMM B3 Button Menu

The B3 "Menu" push-button will cycle through a list of menu items displayed on the 7-segment LED. Pressing and holding the B3 button during the boot sequence (before the Boot Startup Script runs) provides access to a menu of functions. Each B3 press will advance to the next menu item. The menu item will be displayed for 10 seconds. Press B2 to select the current menu item. If no button is pressed within the 10 seconds, the 7-segment display will return to Normal operation.



Functionality	Display	Notes
Display the IP	88.892.88.88	See <a href="#">Display the IP Address of a PDMM or PCMM</a>
Start the application	58.8.8.8	This will start the KAS Runtime.
Factory Reset	88.58.8.8	See <a href="#">Reset the Controller to Factory Settings</a>

Functionality	Display	Notes
Backup firmware to SD card		See <a href="#">Backup Using the B3 Button</a> .
Restore firmware from SD card		See <a href="#">Restore Using the B3 Button</a> .

Table 6-3: Application is not running

Functionality	Display	Notes
Display the IP		See <a href="#">Display the IP Address of a PDMM or PCMM</a>
Stop the application		This will stop the KAS Runtime.

Table 6-4: Application is running

**NOTE**

Please note that when selected, the **Start, Stop, Backup, Restore,** and **Reset** functions do not initiate immediately; they require confirmation. The 7-segment displays flashes a "y", prompting for confirmation. Pressing B2 confirms the function and the process begins. If the function is not confirmed within 10 seconds the action is canceled.



### 10.2.3 Display the IP Address of a PDMM or PCMM

The IP Address assigned to the controller can be shown on the 7-segment display. The IP may be displayed at boot and can be accessed from the [PDMM or PCMM B3 Button Menu](#). Note that there is a 5 second delay before this function may be used again.



Figure 6-1: Example of the IP sequence by the 7-segment display.

**NOTE**

The AKD PDMM or PCMM will not set (or show) an IP address without an attached network cable.

### 10.2.4 Booting from the Recovery Image

To enter recovery mode you must press and hold B2 during the boot sequence before the Boot Startup Script runs. If the system detects that the button is pressed then it will enter Recovery Mode. The 7-segment display will show a lower-case "r" as seen here.



While in Recovery Mode the AKD PDMM or PCMM will download the firmware from the recovery image. When the firmware is being written to the flash drive the 7-segment display will animate as seen below. Do not power-off the system during this process.



(repeats)

When the download is complete the controller will go into normal operation. If the download or write to flash fails the 7-segment display will display a numeric [error code](#).

## 10.2.5 Reset the Controller to Factory Settings

The AKD PDMM or PCMM can be manually triggered to perform a factory reset. The reset is performed using either of two methods:

- From the [File System Tab](#) of the web server while the drive is running. *This method is recommended due to its ease of use.*
- Selecting the function from the [PDMM or PCMM B3 Button Menu](#). This can be done during the boot sequence or while the drive is running.

After two seconds have expired (or longer if pressed during power-up), the 7-segment display on the control will change to an animation pattern indicating that the factory reset has started.

The following changes occur during a factory reset:

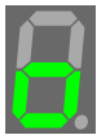
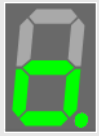
- Reset any application previously downloaded
- Reset IP address, Subnet and Gateway settings
- Reset retained variables
- Reset Auto-Start option
- Reset password to the default
- Reset shared directory settings

Some important facts to remember:

- Factory reset cannot be performed while an application is running.
- If the AKD PDMM or PCMM has just been powered up, the B3 button will have to be held down much longer than 2 seconds. In this case, hold down the button until the 7-segment display shows the [PDMM or PCMM B3 Button Menu](#).
- Factory reset will take about 4-5 minutes to complete and the 7-segment display on the controller will animate during this process. The control should not be turned off during this procedure.
- After the factory reset is complete, the control will be powered down and restarted automatically.

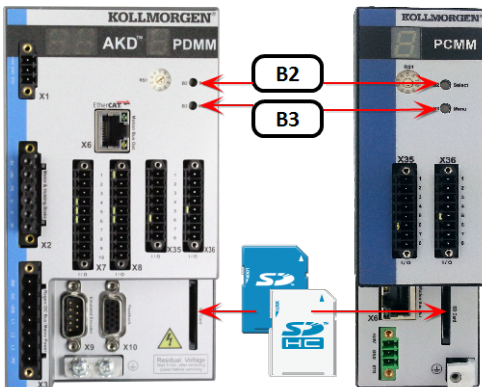
### 10.2.5.1 Resetting While the Drive is Running

A reset may be done any time after the controller is powered on and an application is not running. Please note that the reset will be ignored if an application is running on the AKD PDMM or PCMM.

	<b>Normal Operation</b>	Press the B2 button to access the menu and using B3 button scroll to the "reset" option in the menu. Press B2 to confirm the "reset" to factory defaults.
	<b>Program Running</b>	Reset to factory defaults is not permitted. The "reset" menu item is not available.

### 10.2.6 SD Card Support

The AKD PDMM and PCMM controllers support using an SD card for backup and restore functionality. This lets you manage the configuration, application and operation data. The controller has a SD card slot and push buttons (B2 and B3) which activate file transfers to and from a SD card.



Using the SD card provides an easy way to

- backup and restore a AKD PDMM or PCMM configuration
- store and retrieve an application, including source code
- store and retrieve user data from an application or PC

#### 10.2.6.1 Supported SD Card Formats

Format	File System	Capacity
SD (SDSC)	FAT16 <sup>‡</sup> or FAT32	2GB
SDHC	FAT32 <sup>‡</sup>	4-32GB

<sup>‡</sup> The default file system for the format.

#### ⚠ IMPORTANT

We recommend using an SD card which is rated to a 85°C max. operating temperature. Cards which do not meet this criteria should not be left in the device.

## 10.2.7 Backup and Restore a PDMM or PCMM

The SD card is used for backing up and restoring the data from a AKD PDMM or PCMM.

Backing up may be performed in either of two ways.

- Using the web server, which is recommended as it has functionality the button method does not offer, such as backing up the drive data. For more details, see [Backup & Restore from the Web Server](#).
- Using the [PDMM or PCMM B3 Button Menu](#) on the front of the controller. See [Backup Using the B3 Button](#) for more details.

### TIP

For a complete backup of both the controller and drive, use the Web Server backup.

Restoring information to the controller may be performed in either of two ways. Both methods restore the controller (and PDMM drive) with whatever was saved to the SD card.

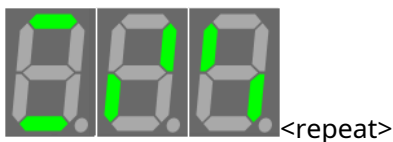
- Using the web server. For more details, see [Backup & Restore from the Web Server](#).
- Using the B3 button menu on the front of the controller. See [Restore Using the B3 Button](#) for more details.

Backup & Restore What?	To/From SD card via B2/B3 buttons	To/From SD card via Web Server
Controller: Firmware	✓	✓
Controller: Network Configuration	✓	✓
Controller: PLC Application	✓	✓
Controller: Retain Variables	✓	✓
EtherCAT Devices: Network Topology	✗	✓
EtherCAT Devices: AKD Drive Firmware	✗ <sup>‡</sup>	✓
EtherCAT Devices: AKD Drive Parameters	✗	✓

<sup>‡</sup> Restore only

### 10.2.7.1 About the data transfer

- The 7-segment display will show the chasing lights animation while the backup or restore is occurring.



- The Backup and Restore functions have an "all or nothing" behavior. If there is no SD card inserted, if there is not enough space on the card or if files are missing then nothing will be copied and the 7-segment display will show an error.
- If files already exist on the SD card (in the backup directory), then they will be deleted and replaced with the new AKD PDMM or PCMM backup configuration files. Likewise, the files on the controller will be replaced with the SD files.

**NOTE**

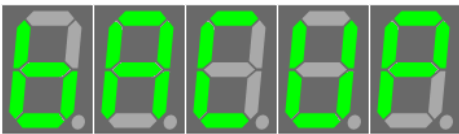
**Warning!** Do not modify the files on the SD card as this could result in the Restore function failing.

**TIP**

If you have multiple AKD PDMM or PCMM backup configurations, you will need to use one SD card per backup configuration.

**10.2.7.2 Backup Using the B3 Button**

One of the functions accessible from the B3 button is backing up the controller. The Backup function will store a copy of the AKD PDMM or PCMM's data on a SD card. This function is displayed on the 7-segment display as shown here ("bACUP"). Pressing B2 selects the function. This function does not initiate automatically, B2 must be pressed again to confirm the process.



The data that is backed up and copied to the SD card includes:

- PLC Application (including ECAT XML configuration, cam tables, etc.)
- AKD PDMM or PCMM configurations (auto-start and IP address)
- PLC application
- [Retain Variables](#)

**NOTE**

Controller Log files and EtherCAT devices are not copied to the SD card. They can be backed up using the [Backup Tab](#) in the Web Server.

**10.2.7.3 Restore Using the B3 Button**

One of the functions accessible from the B3 button is restoring the controller and PDMM's drive. The Restore function will restore and load files onto the controller from an SD card. This function is displayed on the 7-segment display as shown. Pressing B2 selects the function. This function does not initiate automatically. The 7-segment displays flashes a "y", prompting for confirmation. Pressing B2 again confirms the function and the data transfer begins. If the function is not confirmed within 10 seconds the action is canceled.



The data that is restored from the SD card includes:


- PLC Application (including ECAT XML configuration, cam tables, etc.)
- AKD PDMM or PCMM configurations (auto-start and IP address)
- PLC application
- [Retain Variables](#)



## 10.2.8 EtherCAT Devices Backup and Restore

The AKD PDMM or PCMM can backup/restore EtherCAT devices on an EtherCAT network. This feature is useful as a maintenance operation to replace any AKD or AKD2G drives in an operational machine. This feature reduces the manual steps for saving/loading each drive's firmware and parameters into a few simple automated steps. The Backup/Restore functionality is located in the controller's web server and is accessible from a web browser. For details about the web server see [Using the KAS Web Server](#), [Backup Tab](#), and [Restore Tab](#).

KAS Application
Settings
Backup & Restore
Diagnostics
Help




### Backup & Restore

Backup data is stored on the SD card. The following features are not accessible when a PLC application is running or Online Configuration Mode is active.

**EtherCAT devices backup:** Thu, 24 Oct 2019 18:57:44 UTC

Backup
Restore
Import/Export

### Controller



**Do you want to add the 1.2 GHz firmware to the backup (800MHz is already included)?**

YES  NO

Backup Controller

*Backup Controller is used to replicate a PCMM or the controller portion of a PDMM. The elements that are backed up are the firmware, network configuration, retained variables, and PLC application.*

*Notes:*

- \* This function cannot be performed while an application is running.
- \* **Backup Controller** takes several minutes to complete. Do not power off the controller once started.

### EtherCAT Devices

Backup EtherCAT Devices

*Backup EtherCAT Devices will save the network topology, the AKD and/or AKD2G drives' firmware, and the drives' parameters.*

*For EtherCAT devices backup operation, a matching AKD and/or AKD2G firmware package is required.*

- \* If a matching firmware package already exists on the controller, it will be used for future backups.
- \* If a matching firmware package does not exist on the controller, you will be asked to provide it.

<sup>(1)</sup> All firmware packages are available at [Kollmorgen.com](http://Kollmorgen.com)

See also [Backup and Restore a PDMM or PCMM](#) in the KAS IDE help.

### 10.2.8.1 EtherCAT Devices Backup

The Backup operation discovers all the devices on the EtherCAT network and stores the topology information, AKD/AKD2G firmware files, and AKD/AKD2G parameters to the controller's local storage.

Controller	Local Storage
AKD PDMM or PCMM	SD Memory Card (see <a href="#">SD Card Support</a> for more information)

Before starting a backup of the EtherCAT Devices, you will need:

- The AKD or AKD2G firmware files package. This package is included in the KAS software installation directory (\Program Files (x86)\Kollmorgen\Kollmorgen Automation Suite\Astrolabe\DrivesFW) or is available for download from the Kollmorgen [website](http://www.kollmorgen.com/en-us/website-resources/other/akd-software/) (www.kollmorgen.com/en-us/website-resources/other/akd-software/). You will be prompted with the expected file.
- An SD memory card must be in the SD slot if you are using an AKD PDMM or PCMM.
- All the EtherCAT network devices must be connected to the AKD PDMM or PCMM, and configured as necessary for machine operation.

#### TIP

Make sure the devices are in the order as expected. The topology information is stored in the backup and is used to check for identical topology during a restore function. The device order is not important, e.g. an AKD can be anywhere in the EtherCAT topology, but for restore to occur, the devices must be in the same order as when backed up.

- A PLC application downloaded to the AKD PDMM / PCMM, containing the EtherCAT device map.

#### NOTE

The PLC application cannot be running and the IDE must not be in [Online Configuration Mode](#). Please stop your PLC application or disable Online Configuration Mode before a Backup or Restore.

#### 10.2.8.1.1 Modular EtherCAT

The following steps should be taken to backup EtherCAT devices when the modular EtherCAT feature (see [Modular EtherCAT Concept](#) for more information) is used to remove devices to create a machine variant.

1. Run the KAS application which sets up the correct EtherCAT topology for the machine variant.
2. Stop the KAS application.

#### IMPORTANT

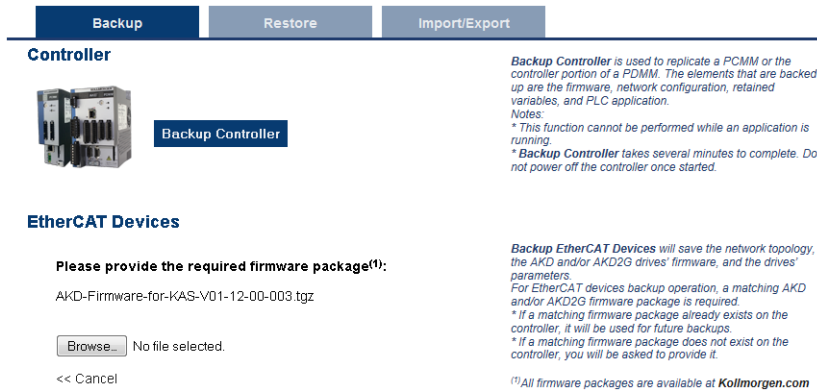
Do not power cycle or reboot the controller between steps 2 and 3.

3. Backup the EtherCAT devices (see [Backup Steps](#)).

### 10.2.8.1.2 Backup Steps

1. From the web server home page, click on the Backup & Restore tab:
2. Under the Backup tab, press the Backup EtherCAT Devices button.

The web server will ask you to choose one or more firmware package files and suggest the filename that matches the firmware version on your drives. Selecting the firmware package files is a one-time event. The controller will remember your selection and will not ask you again for future backups.



**Controller**

**Backup Controller**

**EtherCAT Devices**

Please provide the required firmware package<sup>(1)</sup>:

AKD-Firmware-for-KAS-V01-12-00-003.tgz

Browse... No file selected.

<< Cancel

**Backup Controller**

**Continue Backup EtherCAT Devices**

<< Cancel

**Backup Controller** is used to replicate a PCMM or the controller portion of a PDMM. The elements that are backed up are the firmware, network configuration, retained variables, and PLC application.

Notes:

- \* This function cannot be performed while an application is running.
- \* **Backup Controller** takes several minutes to complete. Do not power off the controller once started.

**Backup EtherCAT Devices** will save the network topology, the AKD and/or AKD2G drives' firmware, and the drives' parameters.

For EtherCAT devices backup operation, a matching AKD and/or AKD2G firmware package is required.

- \* If a matching firmware package already exists on the controller, it will be used for future backups.
- \* If a matching firmware package does not exist on the controller, you will be asked to provide it.

<sup>(1)</sup>All firmware packages are available at [Kollmorgen.com](http://Kollmorgen.com)

#### NOTE

AKD series drives have a firmware package file specific to their firmware version. AKD2G has a different firmware package file specific to its firmware version.

3. Press the Continue Backup EtherCAT Devices button. It will take a couple of minutes or longer, depending on the number of AKD and/or AKD2Gs in the system.



**Controller**

**Backup Controller**

**EtherCAT Devices**

**Continue Backup EtherCAT Devices**

<< Cancel

**Backup Controller** is used to replicate a PCMM or the controller portion of a PDMM. The elements that are backed up are the firmware, network configuration, retained variables, and PLC application.

Notes:

- \* This function cannot be performed while an application is running.
- \* **Backup Controller** takes several minutes to complete. Do not power off the controller once started.

**Backup EtherCAT Devices** will save the network topology, the AKD and/or AKD2G drives' firmware, and the drives' parameters.

For EtherCAT devices backup operation, a matching AKD and/or AKD2G firmware package is required.

- \* If a matching firmware package already exists on the controller, it will be used for future backups.
- \* If a matching firmware package does not exist on the controller, you will be asked to provide it.

<sup>(1)</sup>All firmware packages are available at [Kollmorgen.com](http://Kollmorgen.com)

When the backup is complete, the web server will indicate whether the backup was successful.

#### TIP

After the backup is complete, it is a good idea to export the backup to an off-site location for safe keeping. See [Import / Export Backup from Webserver](#).

### 10.2.8.2 EtherCAT Devices Restore

The Restore operation discovers the devices on the EtherCAT network and compares the physical topology information to the topology information stored in the backup. A Restore will detect the replaced AKD and/or AKD2G devices and restore them. Advanced users can manually select the specific AKD or AKD2G devices and restore them. The backup files containing the topology information, drive firmware file, and drive parameters are retrieved from the controller's local storage.

Controller	Local Storage
AKD PDMM or PCMM	SD Memory Card (see <a href="#">SD Card Support</a> for more information)

#### 10.2.8.2.1 Steps

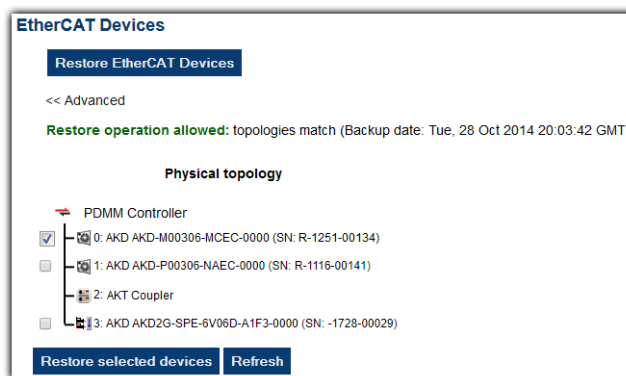
##### ⚠ IMPORTANT

The application will start immediately after the Restore operation is complete if the Autostart option is enabled on the controller. Be sure to Restore all of the replaced devices and the EtherCAT device order is correct. *Disable the Autostart option before commanding the Restore operation if you want to check the devices before starting the application.*

##### NOTE

The EtherCAT Devices Backup and Restore feature may be used in EtherCAT networks which have third party devices, but only the Kollmorgen AKD or AKD2Gs can be backup and restored. Specific configurations applied to third party devices with non-Kollmorgen tools have to be reapplied when the third party device is replaced.

1. From the web server home page, click on the *Backup & Restore* tab:
2. Under the *Restore* tab, you can choose to restore the replaced EtherCAT AKD and/or AKD2G devices or manually select the devices with the Advanced view.
  - To restore the replaced AKD or AKD2G devices, press the *Restore EtherCAT Devices* button. When the restore is complete, the web server will indicate whether it was successful.
  - To select the AKD or AKD2G devices and manually restore:
    1. Press the *Advanced* link.
    2. Select the devices you want to restore. The controller will identify the replaced drives and pre-select them for you.
    3. Press *Restore selected devices*.



When the restore is complete, the web server will indicate whether it was successful.

### 10.2.8.3 AKD Backup/Restore Compatibility


The replacement AKD or AKD2G must have the same model number as the device that was originally stored in the backup. The *Restore* operation compares the model numbers between the backup and the replacement AKD. The restore will not be allowed if they are not compatible.

**TIP**

The "AKD-Pxxxxx-NB" models are functionally compatible with, and may be replaced with "AKD-Pxxxxx-NC" devices provided the backup contains AKD firmware version v01-20-00-003 or later.

		Restore	
		NB	NC
Backup	NB	✓	✓
	NC	✓	✓

- To check the model number on your AKD or AKD2G, see the sticker on the side of the drive.
- To check compatibility between your replacement drive and the backup, replace the AKD or AKD2G on the EtherCAT network, and press the Scan network button.
  - The web server displays the physical topology and allows you to restore the selected drives if the backup and replacement drives are compatible.




**Backup & Restore**

Note that most of the following features are not available when PLC application is running or online config mode is active

**EtherCAT devices backup:** Mon, 29 Oct 2012 08:40:08 GMT

Backup
Restore
Import/Export

**Controller**



**Restore Controller**

*Restore Controller* is used to replicate a PCMM or the controller portion of a PDMM. The elements that are restored are the firmware, network configuration, retained variables, and PLC application.

Notes:

\* *Restore Controller* takes several minutes to complete. Do not power off the controller once started.

\* The controller is rebooted after a *Restore*.

**EtherCAT Devices**

**Restore EtherCAT Devices**

<< Advanced

**Restore operation allowed:** topologies match (Backup date: Tue, 07 Jul 2015 18:12:48 GMT)

**Physical topology**

Controller

- 0: AKD AKD-M00306-M1EC-D000 (SN: R-1432-00169)
- 1: AKD AKD-P00306-NAEC-0000 (SN: R-0934-00008)
- 2: Beckhoff Automation GmbH device 0x4602C22

Restore selected devices
Refresh

- The web server displays the backup and physical topology and indicates the non-compatible drives if the backup and replacement drive are not compatible

**TIP**

The serial number (SN) is displayed in the Physical topology web server view and on the sticker affixed to the AKD. You can use the serial number to match the actual hardware with its representation on the web server.

### 10.2.8.4 Import / Export Backup from Webserver

#### 10.2.8.4.1 Controller Backup Import / Export

Controller backup data may be saved to or written from a local drive or USB memory stick. This functionality is available when a controller backup file (see [Backup Tab](#)) exists on the device's SD card.

During the import process this function will replace an existing backup file without further prompting, but should the process be interrupted or fail, it will not overwrite the backup file.

#### 10.2.8.4.2 EtherCAT Devices Backup Import / Export

A network backup may be exported and imported. The export procedure saves a backup file to the computer running the web browser. The import procedure allows you to transfer a backup file onto the controller to be used later for restoring a previous configuration.

#### 10.2.8.4.3 Export Procedure

**NOTE**

The Export button is only displayed if an AKD or AKD2G backup is available.

1. Click on the "Export EtherCAT Backup" button. The browser starts transferring a backup file. Depending upon the browser being used, this may involve a prompt confirming that you wish to receive the file.
2. You may move the file to a different directory once the file is saved. The file may be renamed to help identify the backup file with the machine.

**10.2.8.4.4 Import Procedure**

1. Specify a backup file to import by clicking on the "Browse" button. This backup file will be used to replace the current backup on the controller.
2. After a backup file is specified, click on the "Replace EtherCAT Backup" button. This creates a backup on the controller with the data stored in the specified backup file. Any previously existing backup will be replaced. If the import fails, the previous backup will not be replaced.

**NOTE**

- The Replace Backup button is disabled until a backup file has been selected.
- On some browsers, the "Browse" button may be labeled "Choose File".

**10.2.8.5 EtherCAT Devices Backup/Restore Limitations**

- Only AKD and AKD2G drives are supported for backup/restore. Kollmorgen S300/S700 drives are not supported by backup/restore.
- AKD PDMM or PCMM system backup or restore is a two-step process:
  1. Backup or Restore the AKD PDMM or PCMM controller
  2. Backup or Restore the drives (including the AKD drive inside the AKD PDMM).
- Only a single backup is supported in the controller at one time.

**NOTE**

If you want to keep multiple backup configurations, you can perform the backup and then export it to a local PC or USB flash memory stick. Later, when you are ready to restore, you can import the specific backup file to the controller, and then perform the restore.

**TIP**

Beware that an import will overwrite any backup existing in the controller or AKD PDMM SD card. If you are using an AKD PDMM, the alternative method is to use a separate SD card for each backup/restore configuration.

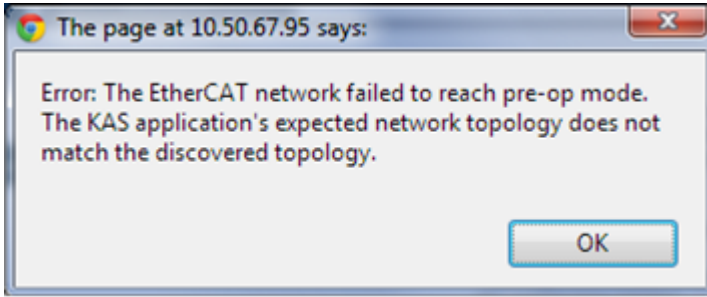
- The following conditions must be met before starting an EtherCAT device backup:
  1. The KAS application should be run, and then stopped.
  2. Do not power cycle or reboot the controller after stopping the KAS application.
- The [EtherCAT Devices Backup](#) and [EtherCAT Devices Restore](#) functions are not permitted while a PLC application is running or when the KAS IDE is in [Online Configuration Mode](#). Please stop your PLC application or disable Online Configuration Mode before a Backup or Restore.
- All drives on the EtherCAT network must have the same firmware version.
- The AKD firmware version must be 01-06-00-003 or higher.
- AKD and AKD2G firmware packages are available for all production releases.

**10.2.8.6 Troubleshooting EtherCAT Devices Backup/Restore**

The web server displays an error message if an EtherCAT Device backup or restore fails. The message describes the cause of the failure and a possible remedy. Please be sure to note any error message(s), as they will be helpful with remedying the problem.

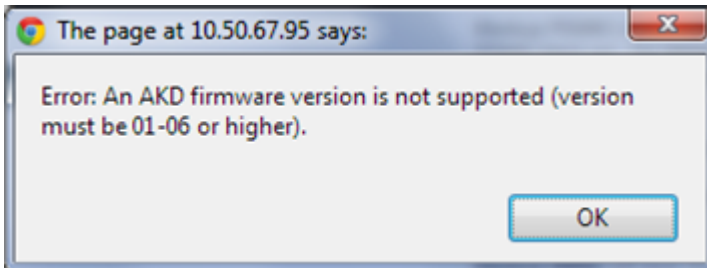
Described below are some common error messages and remedies. The message box format may appear differently depending on the web browser, but the message content is the same.

*During AKD/AKD2G backup, EtherCAT fails to reach pre-op mode:*



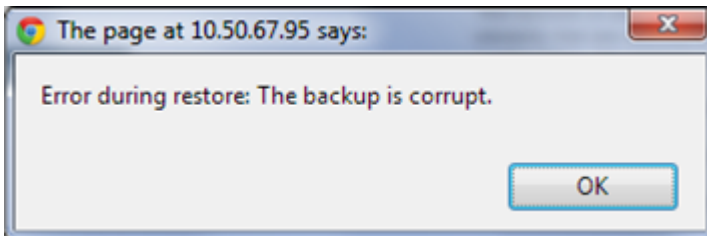
1. Using the IDE, open the EtherCAT view and scan the network.
2. Compare the nodes, their order, and types to the topology in your application.
3. After you identify the differences do one of the following:
  - Modify the application's devices to match the physical network.
  - Correct the physical network by adding/moving/removing nodes.

*During AKD/AKD2G backup, at least one drive has an unsupported firmware error for backup:*



1. Using the IDE, open the EtherCAT view.
2. Upgrade all of the AKDs drive firmware to at least version 01-06.

*During AKD/AKD2G restore, the backup file is corrupt:*

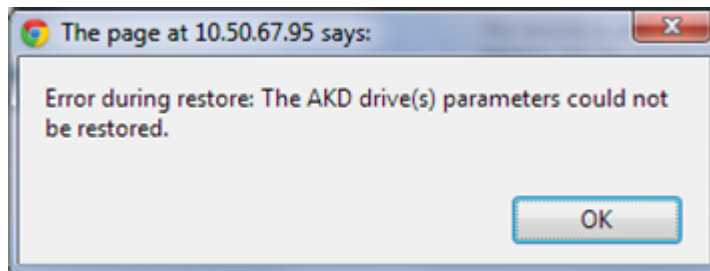


Before restoring AKD/AKD2G(s), the backup file must be valid. To correct a bad backup file on a controller, import a known good backup file from another source (local hard drive, network drive, USB flash stick, etc.). If you do not have a valid backup file, then you will need to manually configure the replacement drives by downloading firmware and modifying the drive's parameters using the AKD/AKD2G views in the IDE.

*Not able to restore AKD/AKD2Gs successfully.*

The error message will describe at which step the restore failed. For example, failing to restore parameters:





1. If an AKD/AKD2G restore fails and you have already verified the controller has a valid backup and the network topology is correct, then retry the Restore.
2. If you still cannot restore successfully after two or three attempts, check your network cables and try a different replacement drive(s). This test will isolate the problem to the specific drive(s) or the controller problem.
3. If you still cannot restore a replacement AKD/AKD2G, then you will need to manually configure the replacement AKD/AKD2Gs by downloading firmware and modifying the drive's parameters using the drive views in the IDE.

### 10.2.9 Configure Controller Onboard I/O

The procedure to define the local I/Os of the AKD PDMM or PCMM is very similar to the one for I/O slices. Note that the Onboard digital I/O is limited to a 1kHz update rate.

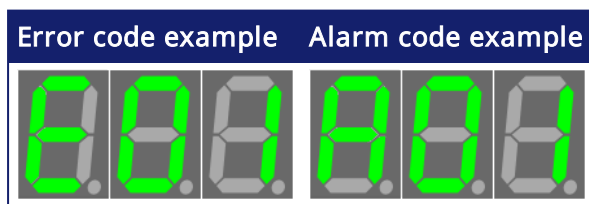
EtherCAT update rate	Onboard digital I/O update rate
500Hz	500Hz
1kHz	1kHz
2kHz	1kHz
4kHz	1kHz

For more details, refer to [Map Input and Output to Variables](#)

### 10.2.10 About Errors and Alarms

The AKD PDMM or PCMM controller will continuously display any error or alarm codes after booting, and not in recovery mode.

Only one error or alarm code will be displayed at a time. [Errors](#) have a priority over [Alarms](#) and the code with the highest priority will be displayed until it is cleared.



### 10.2.11 Using the KAS Web Server

Kollmorgen Automation Suite™ comes with a web server that allows you to perform the following operations:

- Read information about the controller (model, Runtime/firmware version, version of your KAS application)
- Interact with your application (Start and Stop your KAS application)
- View real and simulated axes
- Connect to a shared directory
- See all the log messages

- Upgrade the controller firmware ‡
- Change the IP address ‡
- View system diagnostics including storage space, memory and CPU temperature ‡
- Reset the controller to factory settings
- Backup/Restore the controller ‡
- Backup/Restore EtherCAT devices

‡ AKD PDMM and PCMM only

The web server may be accessed two ways:

1. Open a web browser and enter the controller's IP address.
2. From the Controller node in the Project tree in the KAS IDE.
  - Double-click the Controller node
  - Select "Access Webserver" from the right-mouse menu.

**NOTE**

If you do not know the IP address assigned to the controller:

1. Press B3 once. The 7-segment display will flash the letters I and P.
2. Press B2 to select the IP option. The 7-segment display will show the IP address.

The web server consists of the home page, and the [KAS Application](#), [Settings](#), [Backup & Restore](#), [Diagnostics](#) and [Help](#) tabs. The Help tab is a link which opens the Web Server manual.



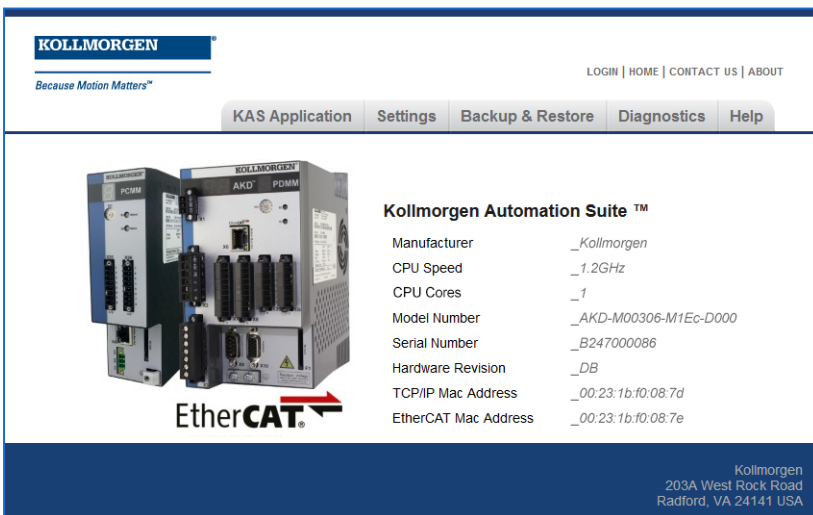
Figure 6-2: The Webserver Tabs as seen on an AKD PDMM webserver.

**TIP**

**Browser Requirements:** We recommend using [Mozilla Firefox](#), [Google Chrome](#), or [Microsoft Edge](#) or later for accessing the web server.

### 10.2.12 Web Server Home Page

To access the KAS web server home page, enter the controller's IP address.



This page provides an overview of the device including:

- Manufacturer
- CPU Speed ‡
- CPU Cores
- Model Number ‡

- Serial Number ‡
- Hardware Revision ‡
- TCP/IP MAC Address ‡ — a unique value associated with the TCP/IP network adapter that uniquely identifies the adapter on a LAN.
- EtherCAT MAC Address ‡ — a unique value associated with the EtherCAT network adapter that uniquely identifies the adapter on an EtherCAT network.

‡ AKD PDMM and PCMM only

#### NOTE

Please note that any reference to AKD PDMM or PCMM refers to both the 800MHz and the single- and dual-core 1.2GHz variants, unless otherwise noted.

### 10.2.12.1 Security

Some parts of the web server are locked in order to protect critical operations from unauthorized users. Simply log into the web server to enable access to the locked functions. See [User Authentication](#) for more information.

#### NOTE

Functions will not be locked if you access the web server through the IDE. Doing so automatically grants administrator access.

The functions which are locked are:

- KAS Application Tab
  - Start/Stop/Cold Start an application
  - "Clear User Data"
  - "Clear all Errors"
  - Configure 'Auto-start'
- Settings tab
  - Firmware upgrade -->'Choose File' &'upgrade' ‡
  - Reboot ‡
  - "Reset to Factory Settings"
  - SD card Format ‡
  - Change password
  - Change the network settings ( IP address ) ‡
- Backup & Restore
  - Backup Controller ‡
  - Backup & Restore AKD PDMM and PCMM only
  - Restore Controller ‡
  - Export Backup
  - Choose File & Replace Backup
- Diagnostics tab ‡
  - Reboot the controller
  - Clear Errors and alarms
  - Clear Crash dump

‡ AKD PDMM and PCMM only

#### 10.2.12.1.1 Timeout After Inactivity

To prevent misuse, if the webserver has been idle (no keyboard activity or mouse clicks) for 20 minutes, the user account will be automatically logged out. A dialog box will open to alert you that the session has timed out.

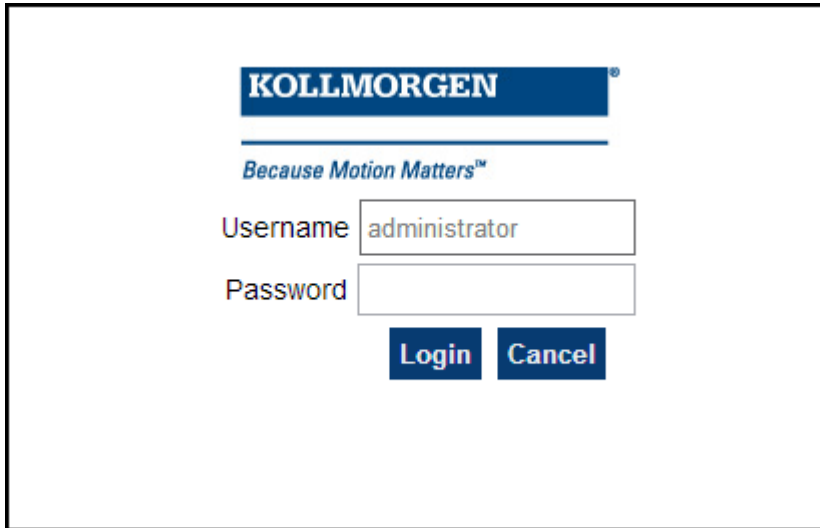
**NOTE**

The idle logout only occurs if you've logged into the webserver with a web browser. Logging in through the KAS IDE will never time out.

## 10.2.12.2 User Authentication

### 10.2.12.2.1 Logging In

Logging into the web server is required to prevent unauthorized access or changes. This is accessed from the **LOGIN** link at the top of all web server pages. Clicking the link brings up a form to enter user credentials.

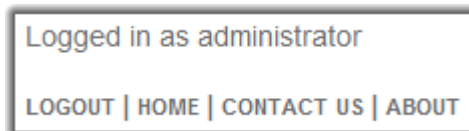


Enter the password to log in. The factory default password is `administrator`. This can be changed after logging in.

#### NOTE

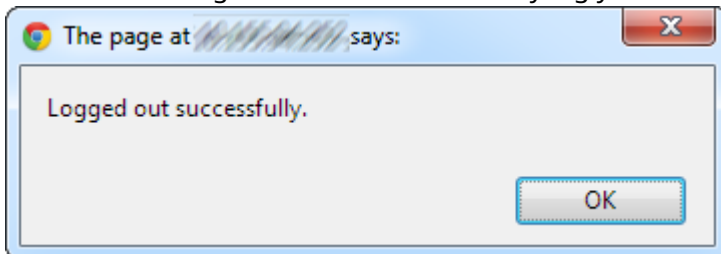
- As there is only one user Administrator, "administrator" is the default user name.
- You must re-enter the password each time you log in; the entered password is not stored in the login form.
- Accessing the webserver from the IDE automatically logs you in as *administrator*.

When you are successfully logged in, the user name will appear in the top-right corner of all web server screens.



### 10.2.12.2.2 Logging Out

After successfully logging in, the menu in the top right corner of the web server contains a link to **LOGOUT**. Clicking this link will immediately log you out of the web server, and informs you of this.




### 10.2.12.2.3 Changing the Password

The user password is managed from the User Account section of the Settings tab. See [User Account](#) for more information.

## 10.2.13 KAS Application

This tab allows you to:

- Display general information about your project that is currently loaded on the controller
- Start and stop the motion
- Display the Axes run by the controller from the [Axis](#) tab
- Manage log messages from the [Log Configuration](#) and [Log Data](#) tabs
- Display User Data present on the controller from the [User Data](#) tab
- Connect to a remote computer from the [Shared Directory](#) tab

Item	Description
<b>Version of KAS App</b>	<p>This label provides information about the name and version number of the application that is in the controller. The format is &lt;project_name&gt;:&lt;version&gt;.</p> <p>The application's source code may be downloaded to the local computer if it is present on the controller. This is accomplished by clicking on the download icon () . This icon is found by the name and version information and is only present when source code is available.</p>
<b>Status of KAS App</b>	The state of the application, <i>Started</i> or <i>Stopped</i> .
<b>Start</b>	Default mode (warm start) where the <a href="#">Retain Variables</a> are loaded at the application startup. They are <b>Not</b> re-initialized; whereas other variables are started with their initial values.
<b>Cold Start</b>	Use retain variables with their default values. Such starts occurs from time to time but are few.
<b>Stop</b>	Stop the application
<b>Auto-start</b>	<p>Select this option to automatically start the KAS application when the controller is powered up. The application will start using retained variables (a "warm start") after the controller has booted up.</p> <p>To change this setting, click the <b>Auto-start</b> checkbox to either activate or deactivate this option and click the <b>Apply</b> button. The control will use the new setting at the next power-up. This option is recommended when the system is in production.</p>
<b>Clear all errors</b>	Clicking this button will clear the error log for all axes.

### 10.2.13.1 Axis

You can view a visual representation of the motors from the Axis tab. The axis wheels are visible after your application is started. The following can be monitored from the display:

- Real and Simulated axes
- Actual position with solid line and actual position value
- Command position with the dotted line and (command position value) in parentheses
- Axis State: Powered-off , Powered-On, or Error as well as Simulated Powered Off and ON
- Identify the axes from the label, as defined by the axis name in your application
- Axis status or positions snapshot



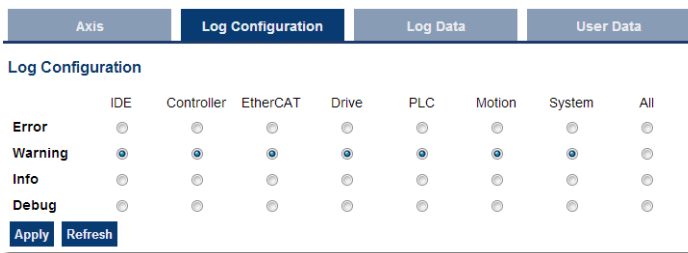
**Information that is available by clicking on the axis**

Title	Image (PN axis)	Image (PLCopen axis)	Image (digitizing axis)																																																																						
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #003366; color: white;"> <th colspan="2">AXIS1</th> </tr> </thead> <tbody> <tr> <td colspan="2" style="text-align: center;"><b>Axis status snapshot</b></td> </tr> <tr><td>Initialised</td><td style="color: green;">true</td></tr> <tr><td>Power ON</td><td style="color: red;">false</td></tr> <tr><td>Enable</td><td style="color: green;">true</td></tr> <tr><td>Found</td><td style="color: green;">true</td></tr> <tr><td>Configured</td><td style="color: green;">true</td></tr> <tr><td>Running</td><td style="color: red;">false</td></tr> <tr><td>Error</td><td style="color: red;">false</td></tr> <tr><td>Simulated</td><td style="color: red;">false</td></tr> <tr><td>Connected</td><td style="color: red;">false</td></tr> <tr><td>Warning</td><td style="color: red;">false</td></tr> <tr><td>Stopping</td><td style="color: red;">false</td></tr> <tr><td>Stopped</td><td style="color: red;">false</td></tr> </tbody> </table>	AXIS1		<b>Axis status snapshot</b>		Initialised	true	Power ON	false	Enable	true	Found	true	Configured	true	Running	false	Error	false	Simulated	false	Connected	false	Warning	false	Stopping	false	Stopped	false	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #003366; color: white;"> <th colspan="2">AXIS1</th> </tr> </thead> <tbody> <tr> <td colspan="2" style="text-align: center;"><b>Axis positions snapshot</b></td> </tr> <tr><td>Actual</td><td>174.220505</td></tr> <tr><td>Current</td><td>0.000000</td></tr> <tr><td>Feedback</td><td>174.220505</td></tr> <tr><td>Generator</td><td>0.000000</td></tr> <tr><td>Pipe offset</td><td>0.000000</td></tr> <tr><td>Pipe</td><td>0.000000</td></tr> <tr><td>Power ON Delta Offset</td><td>0.000000</td></tr> <tr><td>Reference</td><td>0.000000</td></tr> <tr><td>Zero offset</td><td>0.000000</td></tr> </tbody> </table> <p>See the <a href="#">Axis Pipe Block</a> section of the KAS IDE documentation for information about the axis parameters.</p>	AXIS1		<b>Axis positions snapshot</b>		Actual	174.220505	Current	0.000000	Feedback	174.220505	Generator	0.000000	Pipe offset	0.000000	Pipe	0.000000	Power ON Delta Offset	0.000000	Reference	0.000000	Zero offset	0.000000	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #003366; color: white;"> <th colspan="2">PLCOpenAxis2</th> </tr> </thead> <tbody> <tr> <td colspan="2" style="text-align: center;"><b>Axis positions snapshot</b></td> </tr> <tr><td>Actual</td><td>3141.856728</td></tr> <tr><td>Command</td><td>3141.856728</td></tr> <tr><td>Normal</td><td>3141.856728</td></tr> <tr><td>Phase</td><td>0.000000</td></tr> <tr><td>Super imposed</td><td>0.000000</td></tr> </tbody> </table> <p>See <a href="#">Axis Parameters</a> in the <a href="#">PLCopen Function Blocks - Overview</a> section of the KAS IDE documentation for information about the axis parameters.</p>	PLCOpenAxis2		<b>Axis positions snapshot</b>		Actual	3141.856728	Command	3141.856728	Normal	3141.856728	Phase	0.000000	Super imposed	0.000000	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #003366; color: white;"> <th colspan="2">Axis3</th> </tr> </thead> <tbody> <tr> <td colspan="2" style="text-align: center;"><b>Axis positions snapshot</b></td> </tr> <tr><td>Actual</td><td>0.000000</td></tr> </tbody> </table>	Axis3		<b>Axis positions snapshot</b>		Actual	0.000000
AXIS1																																																																									
<b>Axis status snapshot</b>																																																																									
Initialised	true																																																																								
Power ON	false																																																																								
Enable	true																																																																								
Found	true																																																																								
Configured	true																																																																								
Running	false																																																																								
Error	false																																																																								
Simulated	false																																																																								
Connected	false																																																																								
Warning	false																																																																								
Stopping	false																																																																								
Stopped	false																																																																								
AXIS1																																																																									
<b>Axis positions snapshot</b>																																																																									
Actual	174.220505																																																																								
Current	0.000000																																																																								
Feedback	174.220505																																																																								
Generator	0.000000																																																																								
Pipe offset	0.000000																																																																								
Pipe	0.000000																																																																								
Power ON Delta Offset	0.000000																																																																								
Reference	0.000000																																																																								
Zero offset	0.000000																																																																								
PLCOpenAxis2																																																																									
<b>Axis positions snapshot</b>																																																																									
Actual	3141.856728																																																																								
Command	3141.856728																																																																								
Normal	3141.856728																																																																								
Phase	0.000000																																																																								
Super imposed	0.000000																																																																								
Axis3																																																																									
<b>Axis positions snapshot</b>																																																																									
Actual	0.000000																																																																								

Additionally, if an axis is in error, the error can be cleared by clicking the text below the axis title.

### 10.2.13.2 Log Configuration

You can configure the log to filter the messages that are displayed. Each **source** can be set with its own **level**.



Each message has one of the following levels, with importance in descending order: Error > Warning > Info > Debug

#### **TIP**

##### How to Choose the Appropriate Level?

When a level is set for a source, only messages with the same or higher importance are recorded. For example, if a source is set to WARNING, then all messages with levels WARNING, ERROR and CRITICAL are recorded (DEBUG and INFO messages are discarded).

Therefore, DEBUG is the most verbose and ERROR is the least verbose level. Filtering is quicker with less verbose levels, due to the number of messages.

#### **NOTE**

Critical messages are always recorded. Therefore, the Critical level is not visible.



#### Source

Source	Apply to...
IDE	Win32 applications: the KAS IDE and the KAS Runtime Server (also called the KAS Runtime Front-end)
Controller	For the KAS Runtime items: Drivers, IOEngine, SinopEngine...
EtherCAT	For all kinds of EtherCAT items: Motion bus, I/Os
Drive	Messages from Kollmorgen drives
PLC	For application engineers to create custom log within the PLC programs (similar to printf)
Motion	Messages coming from the Motion engines: PLCopen, Pipe network or VM
System	For common API and libraries. Also includes messages issued from the operating system.

#### Level

Level	Icon	Description
DEBUG		Any information logged for development purpose. You may safely ignore this log.
INFO		Information status of the current process. You may safely ignore this log.
WARNING		System is stable but the KAS IDE warns that an unexpected event can occur. You can ignore this log.



Level	Icon	Description
ERROR		The application does not behave as expected but the processes remain stable.
CRITICAL		Application crashes or becomes unstable. Data is corrupted. At this point the application behavior can be unpredictable.

### 10.2.13.3 Log Data

KAS log files may be viewed from the Log Data tab. These messages can help describe the current state of the system and to help identify any operation errors encountered when developing your system. A AKD PDMM or PCMM will display as many as 10 files.

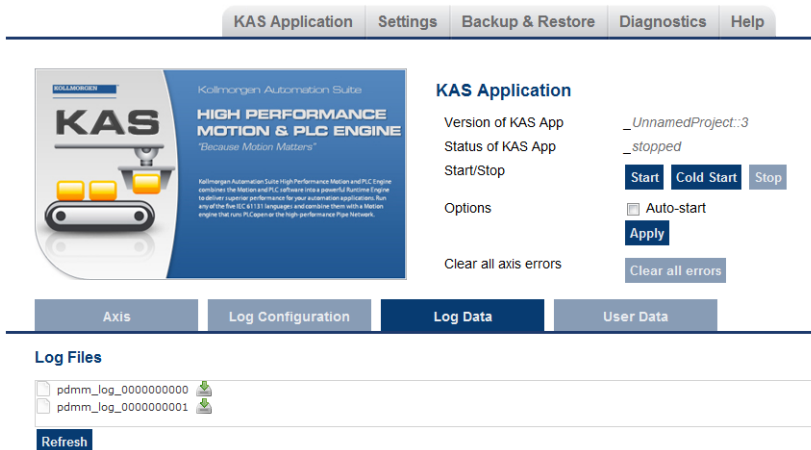


Figure 6-3: Example of log files displayed from a AKD PDMM or PCMM webserver.

Clicking on a listed log file will open it in your web browser. The log file may be downloaded by clicking on the green download icon next to the log entry. The default name is the same as the file's name. If you try to open a file that no longer exists, the message *"/logfiles/<selected file name> not found."* Refresh your browser window and try again.

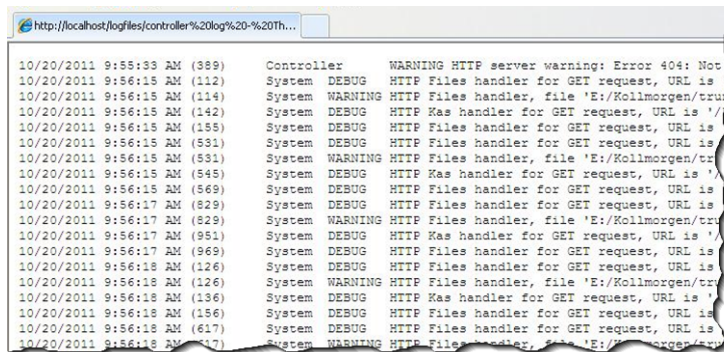


Figure 6-4: Example of a log file's content, displayed in a browser.

**NOTE**  
 Log data is collected and updated every 15 seconds on a AKD PDMM or PCMM. A new log file will be created when the current file is full. You may need to wait for up to 15 seconds for a log to show up in the list. The AKD PDMM and PCMM log files both have the same file naming scheme (pdmm\_log\_XXXXXXXXXX).

#### 10.2.13.3.1 Log Message Content

Every log message in the table has the following information:

Field	Description
<b>Time</b>	Time when the log was recorded with the format: DD-MMMM-YY hh:mm:ss (millisecond)
<b>Source</b>	Identifies a software or hardware component issuing the messages. Each source is <a href="#">configured</a> with a specific Level.
<b>Level</b>	Each message has one of the following levels with importance in ascending order: DEBUG > INFO > WARNING > ERROR > CRITICAL
<b>Message</b>	Text of the message issued from the source

Table 6-5: Log Messages - List of Field

**TIP**

Log messages is an important source of information when you are troubleshooting your project. When reporting an issue to [Support](#), copy/paste the logs in your report.

**10.2.13.3.2 PDMM and PCMM Log Files**

Logs generated on a AKD PDMM or PCMM are stored in flash memory at `/mount/flash/log`. The files are stored in a rotating pool consisting of a maximum of 10 files. The files have a maximum size of 200 kilobytes each; the most amount of space the log files will consume is 2 MB. Once an "eleventh" file is created the earliest file is flushed to make room for the new file.

The AKD PDMM and PCMM generated log levels can be controlled form the KAS IDE and Web Server. From the IDE, the log levels can be filtered in the configuration window in the *Logs and Information* tab.

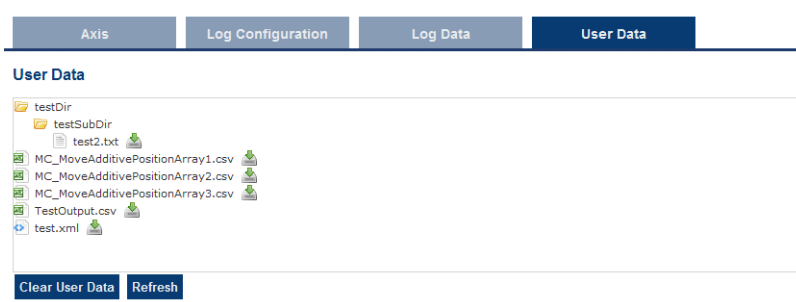
**10.2.13.3.2.1 Log File Naming Convention**

The logs have the naming format `pdmm_logs_n` where *n* is a value ranging from 000000000 to 4294967295, which is the maximum value a 32-bit location can store.

As an example, when the files are first created they will be named `pdmm_logs_000000000`, `pdmm_logs_000000001`, `pdmm_logs_000000002` and so on. The file that will be created after `pdmm_logs_4294967295` is `pdmm_logs_000000000`. The naming gets reset and continues.

**10.2.13.4 User Data**

This tab lists any user-generated files or folders found in the AKD PDMM's, or PCMM's flash memory. Clicking a folder will display the folders contents. Clicking on the green download icon will immediately download the file.



The **Clear User Data** button will erase all of the files in the user data folder.

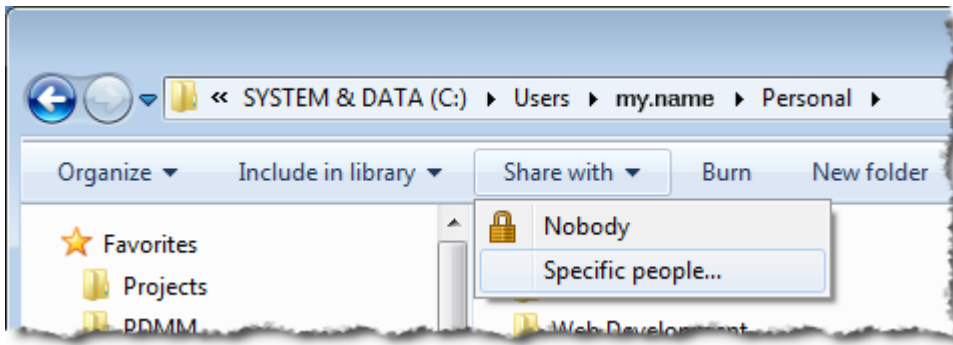
### 10.2.13.5 Shared Directory

This tab provides the ability to connect to and mount a shared directory on a remote computer (Windows® 7 or better, and non-Windows). This allows a program to read and parse instructions for motion. Once the shared directory is mounted, the contents may be navigated in the **Shared Network Directory Viewer** section of this tab. The following fields must be set to connect to the share.

<b>IP Address</b>	Enter the IP Address of the computer that has a shared directory.
<b>Domain Name</b>	The network domain of the computer that is sharing the directory. This field may or may not be necessary.
<b>NetBIOS Name</b>	The unique name that identifies the computer on the network. For Windows computers, this is typically the computer name.
<b>Shared Directory Name</b>	Enter the name of the shared directory as defined on the remote computer. The contents of this directory will be displayed in the viewer.
<b>User Name</b>	The name of a user that is allowed access to the shared directory. If the user name is that of a domain user (instead of a local user), then the computer that hosts the shared folder must be connected to the domain in order to validate the name.
<b>User Password</b>	The password of a user that is allowed access to the shared directory.
<b>Windows File System</b>	Indicate whether the remote computer is Windows or a different operating system. This is used to automatically set file permissions.
<b>Mount Status</b>	This field indicates whether the directory <b>Is Mounted</b> or <b>Is Not Mounted</b> .
<b>Apply</b>	Pressing this button will mount the share and save the configuration to flash memory so it can be automatically mounted at power-on. This button is disabled when the application is running and when the user is not logged in.
<b>Clear</b>	Pressing this button will clear the above fields, unmount the share, and clear the configuration from flash memory. This button is disabled when the application is running and when the user is not logged in.
<b>Refresh</b>	Refresh the contents of the fields and the Shared Directory Viewer.

**Example of how to share a directory from Windows:**

1. In Windows Explorer, navigate to the directory you wish to share.
2. Select **Share with** from the Toolbar.



3. Using the resulting window, select the users and/or groups who that should have access to the directory.

**TIP**

We strongly recommend consulting with your IT department for setting up and connecting to a shared directory. With differences between operating systems and the variability of security settings, this can become challenging.

**NOTE**

SMB1 must be enabled to use the Shared Directory feature with a Windows file system. Please review the [KDN article "Shared Drive in PDMM or PCMM Application Fails to Mount"](#) for information on configuring SMB1 in Windows.

See also [Shared Directory Path Conventions](#).

### 10.2.14 Web Server Settings

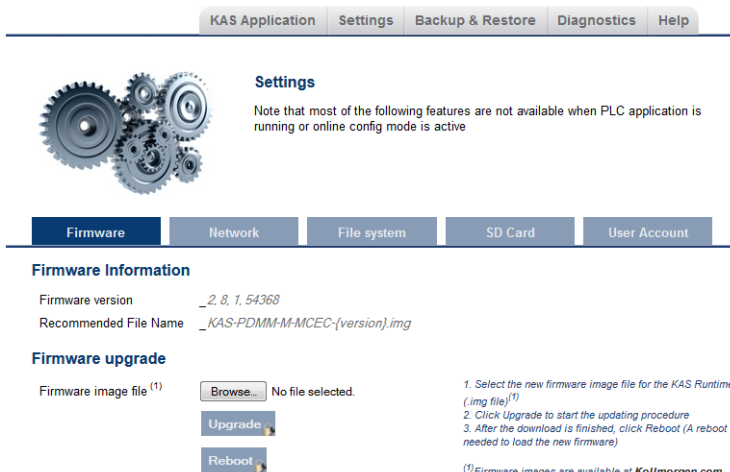
This section allows you to:

- Display and update the firmware for the KAS Runtime‡
- Display the network settings and modify the IP address ‡
- Reset the control to factory settings
- Access the [SD Card Actions ‡](#)
- Access the [User Account](#) to change the password.

‡ AKD PDMM and PCMMonly

#### 10.2.14.1 Firmware Tab (AKD PDMM and PCMM Only)

This tab displays the firmware version. Additionally, you may upgrade the firmware from this tab.



### 10.2.14.1.1 Upgrading the Firmware

This section describes how to upgrade the AKD PDMM or PCMM's firmware using the web server. This operation downloads the KAS Runtime and its version number to the on-board flash memory in the controller.

The firmware files are IMG files with the following name format:

`KAS-PCMM-M- {model-code} - {software-revision}.img`

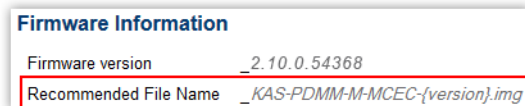
`KAS-PDMM-M- {model-code} - {software-revision}.img`

The *model-code* reflects the CPU speed.

Model	CPU Speed	Cores
MCEC	800 MHz	single
M1EC	1.2 GHz	single
M2EC	1.2 GHz	dual

1. Download the latest drive firmware and/or runtime firmware from [Kollmorgen.com](http://Kollmorgen.com).
2. Open the controller's web server in your web browser by entering its IP address.
3. Select the **Settings** tabbed-page.
4. In the **Firmware** pane, click the **Choose File** button to select the new firmware image file for the KAS Runtime.

The recommended file is displayed in the **Current Information** section, as seen below.



5. Click **Upgrade** to start the update procedure.

#### **TIP**

If the Upgrade button is disabled, log into the webserver. Click on **Login** at the top of the web page and enter the password. See [User Authentication](#) for more information about logging in.

A message and a [throbber](#)<sup>1</sup> are shown across the web page, indicating that maintenance is in progress. The device's 7-segment display will animate chasing lights.

<b>Successful upgrade</b>	<p>A message similar to the following is shown upon a successful firmware upgrade:</p> <p><b>Upload of firmware KAS-PDMM-M-MCEC-2.10.0.54368.img successful.</b></p> <p><b>Please reboot the unit in order to boot on the new firmware, and once reboot is performed, press CTRL+F5 in your web browser to force a page refresh.</b></p>
<b>Incompatible firmware</b>	<p>An error message similar to the following will be displayed if the wrong firmware file was downloaded:</p> <p><b>The file provided is not compatible with this device.</b></p> <p><b>The file name should be...</b></p> <p><b>"KAS-PDMM-M-MCEC-{version}.img"</b></p>

6. After the download is complete, click **Reboot** (for more details on the boot sequence, refer to [Booting the PDMM or PCMM](#)).

<sup>1</sup>An animated element which indicates something is in progress. Interactions should wait until the process is complete.

A message and a throbber are shown over the web server while the reboot is in progress. The login session will no longer be valid when the reboot is complete. The web server will display a message to indicate the user has been logged out.

**NOTE**

This step is not necessary if the controller automatically reboots during the upgrade (previous step).

7. Press **CTRL+F5** to force the web browser to refresh the page.

**! IMPORTANT**

Do not try to refresh the web page until firmware upgrade is done.

**10.2.14.1.2 Recovery Mode (PDMM or PCMM Only)**

If the AKD PDMM or PCMM detects a problem in the firmware, it displays an "r" on the 7-segment display and will automatically enter Recovery Mode. Recovery Mode provides the ability to select and upgrade a firmware image file containing the KAS Runtime image on the AKD PDMM or PCMM. In the rare case when Recovery Mode cannot be automatically accessed, pressing and holding B2 at boot will force the AKD PDMM or PCMM to boot into Recovery Mode.

**! IMPORTANT**

**Warning!** The Recovery Mode allows any firmware image file to be loaded into the AKD PDMM's or PCMM's flash memory. The Recovery Mode does NOT check the selected firmware file to verify its compatibility with the hardware model. Check that the "Recommended File Name" matches the selected firmware file. If an incompatible firmware file is loaded into the AKD PDMM's or PCMM's flash memory, the AKD PDMM or PCMM will fail to boot into the Runtime image and will fail to automatically boot into Recovery Mode. To recover from this situation requires manually booting into Recovery Mode. For more details, see [Booting from the Recovery Image](#).

### 10.2.14.2 Network Tab (PDMM or PCMM Only)

The contents of this tab display the current rotary switch position of the AKD PDMM or PCMM and its MAC address. Additionally, you may manually change the AKD PDMM's or PCMM's IP address.

Figure 6-5: Example of a AKD PDMM or PCMM with a manually defined IP address

#### 10.2.14.2.1 About the Rotary Switch

The rotary switch on the AKD PDMM or PCMM can be set on a position from 0 to 9.

Switch Position	Description
<b>Position 0</b>	The drive tries to get an IP address from a DHCP server. If the DHCP fails, then the AKD PDMM or PCMM uses AutoIP to get a usable IP address.
<b>Position 1</b>	The default custom static IP address, 192.168.0.101 or a custom IP address.
<b>Position 2-9</b>	The AKD PDMM or PCMM is pre-configured with static IP addresses ranging from 192.168.0.102 (Position 2) to 192.168.0.109 (Position 9).

#### **TIP**

If a DHCP server is not present, the drive will assume an Automatic Private IP Address of the form 169.254.x.x

#### 10.2.14.2.2 Change the IP Address

To connect and use your AKD PDMM or PCMM within your computer network, you may configure its IP address by using the web server as follows:

1. Open the controller's web server in your internet browser
2. Select the **Settings** tabbed-page
3. In the **Network** pane, set static IP address according to the position defined via the rotary switch
  - If the rotary switch is set to Position 1 you may use the default custom address or set a value in the Manual IP Address fields.
4. Configure the Manual IP Address
5. Configure the subnet mask (default is 255.255.255.0)
6. (Optional) Configure the gateway address if the AKD PDMM or PCMM is outside your local network
7. Click **Apply**
8. Click **Reboot**

#### **TIP**

It is recommended to not use leading zeroes when entering an IP address. The webserver automatically configures numbers with leading zeroes as octal values. For example, use **10.1.1.10** instead of **010.001.001.010**.

### 10.2.14.3 File System Tab

This section contains a button which allows you to reset the control to the factory settings. The steps to reset the controller vary slightly based on the platform.

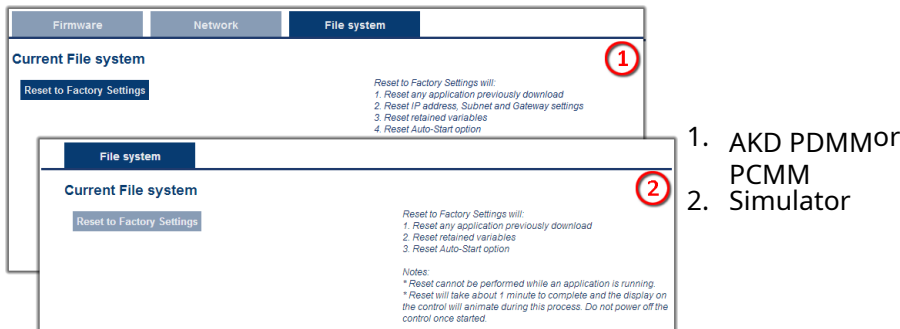


Figure 6-6: File System tab on an AKD PDMM or PCMM web server, and when using Simulator.

#### 10.2.14.3.1 Reset to Factory Settings

When this button is pressed, the control will be reset to factory default settings. The user is prompted to confirm this action before the function is performed.

The following changes occur during factory reset:

- Reset any application previously downloaded
- All log files and user data files are erased
- Reset the IP address, Subnet, and Gateway settings ‡
- Reset any retained variables
- Reset the Auto-Start option
- Reset the password to default

‡ AKD PDMM and PCMM only

Notes about the reset:

- The factory reset cannot be performed while an application is running. The "Reset to Factory Settings" button is disabled while an application is running.
- The factory reset will take 4-5 minutes to complete and the 7-segment display on the control will animate during this process. The control should not be turned off during this procedure.
- After the factory reset is complete, the controller will be powered down and restarted automatically.
- This webpage will not update during the reset procedure and can be closed.
- AKD PDMM or PCMM Only: After the controller is restarted, the IP address of the control may change based on the controls rotary switch. If the rotary switch is at position 0, the same IP address as before should be assigned to the control. If the rotary switch is set to 1-9, a pre-configured IP address will be defined and must be taken into account when trying to reconnect to the webpage using a web browser.

### 10.2.14.4 SD Card Tab (PDMM or PCMM Only)

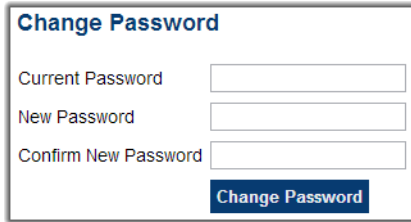
#### 10.2.14.4.1 SD Card Actions

The *Format* function formats the SD card as FAT32, erasing all data from the card. This function cannot be performed while an application is running.

### 10.2.14.5 User Account



To change the password you must enter the current password and the new password twice.



The screenshot shows a web form titled "Change Password". It contains three input fields: "Current Password", "New Password", and "Confirm New Password". Below the "Confirm New Password" field is a blue button labeled "Change Password".

The new password must meet the following conditions:

- It must be 6-20 characters long
- It may not contain semicolons (;), ampersands (&), spaces, quotes (' and "), slashes (/ and \), or the number sign(#).

#### 10.2.14.5.1 I forgot my password

Should this happen, you can set a new password from the IDE.

1. Open the webserver from the IDE.
2. Click on the Settings tab.
3. Click on the User Account tab.
4. Enter and confirm the new password.

This allows you to create a new password without entering the current one.

## 10.2.15 Backup & Restore from the Web Server

These functions are used to replicate a AKD PDMM or PCMM (*Backup* and then *Restore*) to or from the SD card. The elements that are backed up or restored are the firmware, the network configuration, the [Retain Variables](#), and the PLC application.

- These functions cannot be performed while an application is running.
- *Restore* and *Backup* take several minutes to complete. Do not power off the control once started.
- The Controller is rebooted after a *Restore*.

The AKD PDMM and PCMM are available with different CPU variants (800 MHz, 1.2 GHz single core, and 1.2 GHz dual core). The AKD PDMM and PCMM models have the same functionality, but due to the CPU speed differences an application designed for the 1.2 GHz model may not execute as expected on the 800 MHz model. Additionally, an application designed for a single core CPU model may not execute as expected on the dual core model. The Restore will not allow a higher speed model to be replaced with a lower speed model. The Restore will allow a lower speed model to be replaced with a higher speed model, if the Backup contains the model specific Runtime firmware file.

Backup File Source	800 MHz Restore	1.2 GHz Single Core Restore	1.2 GHz Dual Core Restore
800 MHz Backup	Yes	Yes - If 1.2 GHz Runtime file is included in the Backup	No
1.2 GHz Single Core Backup	No	Yes	No
1.2 GHz Dual Core Backup	No	No	Yes

### TIP

This section provides an overview of the backup and restore processes. For a deeper discussion, see [EtherCAT Devices Backup and Restore](#) in the KAS IDE help.

### 10.2.15.1 Backup Tab

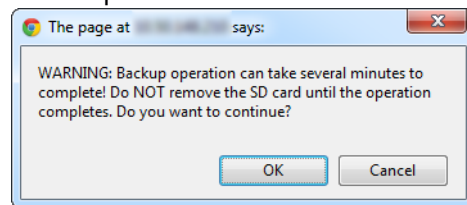
## Backup Controller

This function is used to replicate a pCMM or the controller portion of the AKD PDMM. Clicking the button will save the data to the SD card.

The 1.2 GHz single core Runtime firmware file is optional. If provided, an 800 MHz single core model can be replaced with a 1.2 GHz single core model. If the 1.2 GHz single core Runtime firmware file is not included, then an 800 MHz single core model can only be replaced with another 800 MHz single core model.



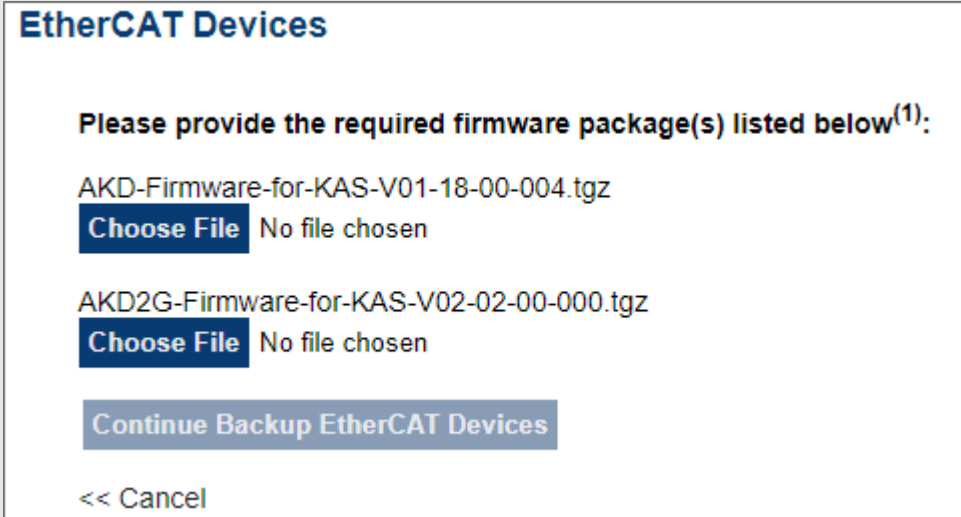
After starting the Backup, an alert will be presented to confirm the backup should proceed.



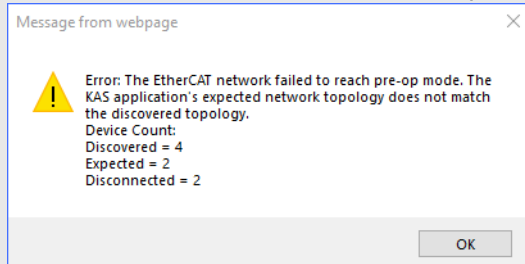
**Backup EtherCAT Devices**

This function replicates the network topology as well as the AKD, AKD PDMM, AKD-C, AKD-N, and AKD2G drives' unique ID, firmware and data. To accomplish the backup, a copy of the firmware package is required. There are several possible scenarios upon clicking this button.

- If an archived copy of the same firmware package is on the controller as is used on the drives then clicking the button will start the backup.
- If a copy of the firmware package cannot be found, you will be prompted to browse for one or more. A link to the Kollmorgen website is provided; all firmware packages can be found on the site.



- If an archive is found on the controller but it does not match the network configuration then you will be prompted to browse for one or download one from the website.
- The following message will be displayed if the EtherCAT network topology discovered during backup does not match the topology expected by the KAS application. The message displays the number of devices that were discovered, the number that were expected, and how many are disconnected.

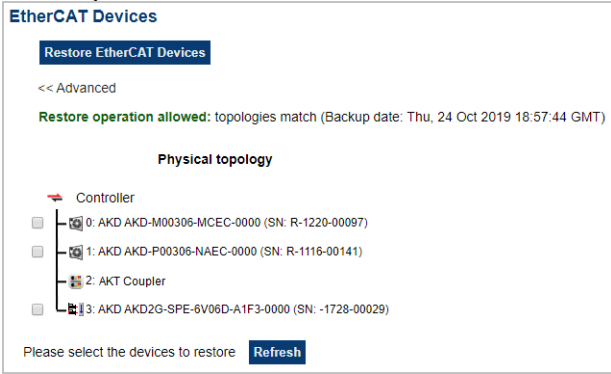


For all other EtherCAT devices, such as couplers, IOs, ServoStar drives, and third-party devices, only the unique ID is backed up.

**TIP**

Files on the website are saved in ZIP format. You must unzip the download to access the TGZ file.

### 10.2.15.2 Restore Tab

<b>Restore Controller</b>	This function restores a AKD PDMM or PCMM's firmware, network configuration, retained variables and PLC application from the SD card.
<b>Restore EtherCAT Devices</b>	This function automatically restores the data and firmware of any replaced AKD or AKD2G drives, and the station alias of other EtherCAT devices.
<b>Advanced</b>	<p>Clicking this button creates a map of the network and it's components. If a valid firmware file is found and the current topology matches the backup file you may select the device(s) that need to be restored. Clicking <i>Restore selected devices</i> will restore the drive's firmware.</p> 

### 10.2.15.3 Import/Export

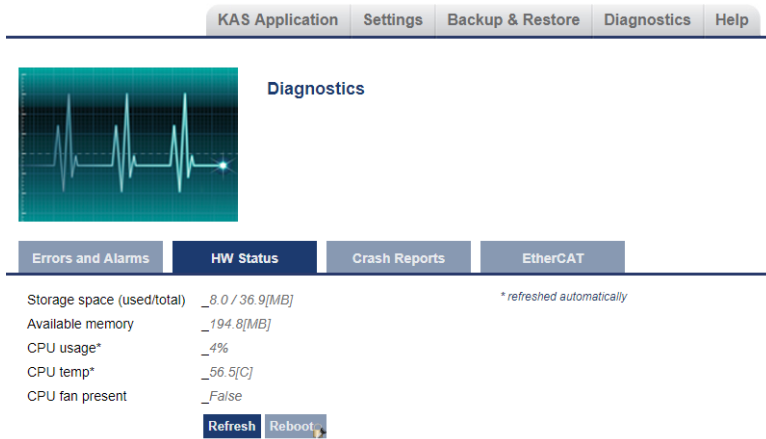
These functions allow you to save a copy of the EtherCAT configurations backup file to a computer and Import the backup file so it may be used for restore functions.



<b>Export Backup</b>	This button allows you to save a backup of the EtherCAT configuration to a computer.
<b>Choose File</b>	This button allows you to select a previously exported backup file to be imported.
<b>Replace Backup</b>	This button imports the selected backup file, replacing any existing backup.

### 10.2.16 Diagnostic (PDMM and PCMM Only)

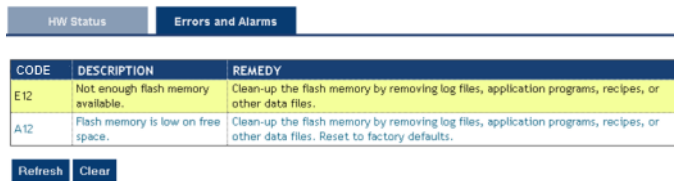
This page displays information about the hardware status (storage space, memory and CPU temperature) and errors and alarms.



#### 10.2.16.1 Errors and Alarms

Any controller errors or alarms generated by the system will be shown here and on the 7-segment display. A common error or alarm is due to the flash memory being full. This is often caused by heavy use of the PLC Advanced File function blocks.

The **Refresh** button updates the list. The **Clear** button will remove the contents of this tab. Please note that some errors or alarms are only cleared by powering off and restarting the AKD PDMM or PCMM.



See [Errors and Alarms](#) for a complete list of codes.

#### TIP

Axis errors can be seen in the KAS Application *Axis* tab.

#### 10.2.16.2 Hardware Status

<b>Storage Space</b>	The diagnostic displays both the used and total available amount of storage space in megabytes (MB). <b>Used</b> is the amount of file space currently being used by all files in flash memory. <b>Total</b> is the total amount of file space available for files in flash memory.
<b>Available Memory</b>	This field displays the amount of RAM memory available on the Controller.
<b>CPU usage</b>	This field displays the current load on the CPU. For dual-core systems, this is the average of the two cores. If the load goes over 90%, the field turns red.
<b>CPU Temp</b>	This field displays the temperature of the CPU in Celsius. If the CPU temperature is greater than the CPU warning limit, the temperature background color will be changed to yellow. If the CPU temperature is greater than the CPU critical temperature, the temperature background color will be changed to red. The normal operating range is 0-125°C. For dual-core systems, this value is the average of the two cores.

<b>CPU Fan Present</b>	This field is either <b>True</b> or <b>False</b> , depending upon if there is a CPU fan present in the controller.
<b>Refresh</b>	Clicking this button will refresh the Hardware Status information.
<b>Reboot</b>	Clicking this button will reboot the web server.

**! IMPORTANT**

Do not try to refresh the web page until the server has rebooted.

**10.2.16.3 Crash Reports**

The files shown on this tab are reports of the process that failed if there is a crash. These files (GZ archives) may be sent to Kollmorgen for analysis.

Errors and Alarms | HW Status | **Crash Reports** | EtherCAT

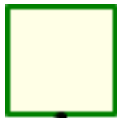

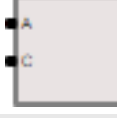



Crashdump Files


sinope\_3\_00\_0\_73351\_0.core.gz

Clear Crashdump Refresh

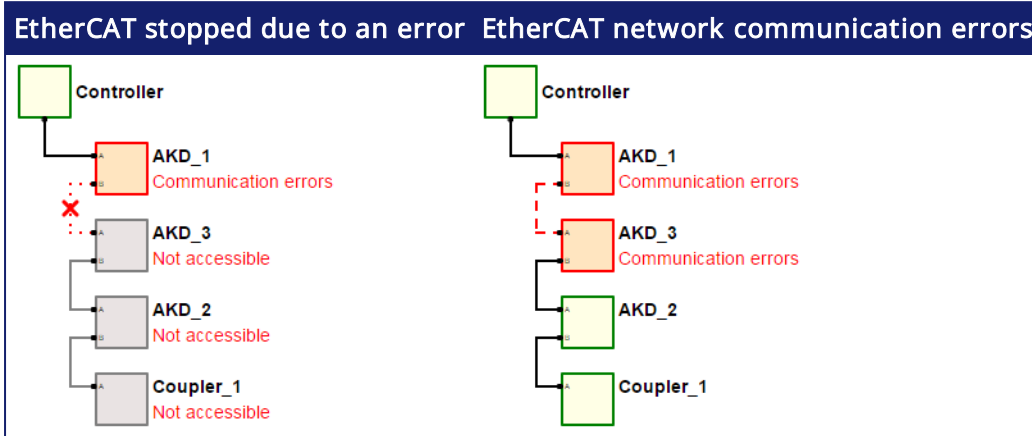
**10.2.16.4 EtherCAT Diagnostics**

This tab is used to identify failed or degraded connections between nodes, (e.g. Controller, drives, and I/O), and non-accessible nodes or cables. The tab's content includes a visualization of the network topology, and it's last-known state. The tab contents are updated using the **Refresh** button.

Element	Description	Meaning
	Green Box	This is a node with normal communication.
	Red Box	This is a node that is experiencing communication errors.
	Gray Box	This is a node which cannot be reached, but was known to exist.
	Black Line	This cable is working normally.
	Gray Line	This cable is not accessible.
	Red Dashed Line	This cable is experiencing degraded communication

Element	Description	Meaning
	Red X in Dotted Line	The connection between the nodes is broken.

Following are examples of error situations when the application is running.



### 10.3 Using SSH

AKD PDMM and PCMM controllers can be accessed using SSH. SSH is used to perform maintenance operations that are not available through the web interface. Additionally, Kollmorgen Application Engineers may need to use SSH to access the controller to help resolve customer issues.

#### ⚠ IMPORTANT

To ensure security, it is critical that the login password be changed from the default value. See [Change Controller's Password Via SSH](#) for information.

#### 10.3.1 Logging In To A Controller Via SSH

An SSH client is required to log in to a controller via SSH.

1. Configure the SSH client using the following settings.

Setting	Value
Host Name or IP Address	(IP Address of the controller)
Connection Type	SSH
Port	22

2. Enter the user name and password at the login prompt.
  - Default User Name: `root`
  - Default Password: `administrator`

#### NOTE

Please note that the root password and the webserver's password are separate; one does not affect the other.

#### A List of Some SSH Clients

Kollmorgen does not recommend any particular program. Following are some common SSH clients, for your convenience.



- [PuTTY](http://www.putty.org/) - <http://www.putty.org/>
- [MobaXterm](http://mobaxterm.mobatek.net/) - <http://mobaxterm.mobatek.net/>
- [TeraTerm](https://tssh2.osdn.jp/index.html.en) - <https://tssh2.osdn.jp/index.html.en>
- [Terminals](https://terminals.codeplex.com/) - <https://terminals.codeplex.com/>

#### ⓘ IMPORTANT

To ensure security, it is critical that the login password be changed from the default value. See [Change Controller's Password Via SSH](#) for information.

### 10.3.2 Change Controller's Password Via SSH

Following is the method for changing the controller's root password via SSH.

1. Log on through SSH
2. At the prompt type "passwd" and press Enter.
3. Enter the new password at the two prompts.

**Example of changing the root password.**

```
# passwd
changing password for root
New password:
Retype new password:
#
```

### 10.3.3 How to Restore the Controller's Default Password and/or SSH Settings

There are two methods to restore the controller's default root password and SSH settings. The first method, restoring the directory via SSH, is recommended if it is possible.

- Restore the `/mount/flash/etc` directory.

This method should be used if logging in via SSH is possible. This will restore the `/mount/flash/etc` directory to the default state. Following are the shell commands to accomplish this.

```
# rm -r /mount/flash/etc
# shutdown
```

The controller will load with the default settings upon restarting.

- [Reset the Controller to Factory Settings](#) using one of two manual methods.

#### ⓘ IMPORTANT

Performing a factory reset of the controller should be considered very carefully as it will reset everything including the application, the IP address, any retain variables, passwords, and shared directory settings.

#### ⓘ IMPORTANT

To ensure security, it is critical that the login password be changed from the default value. See [Change Controller's Password Via SSH](#) for information.

# 11 Tools

## 11.1 PipeNetwork Editor

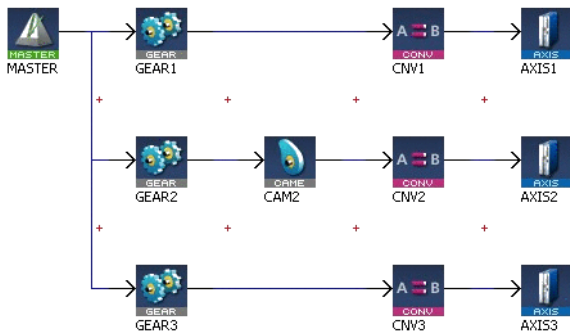


Figure 6-7: PipeNetwork Structure

The PipeNetwork Editor is a graphical tool dedicated to the description of the motion part of the application ([PipeNetwork Concept](#)).

Functions of the PipeNetwork Editor are accessed via context sensitive menus.

When the PipeNetwork Editor is used, an ST file containing all the calls to the Motion Library is automatically generated during compilation, and based on the graphical description of the Pipe Network.

*PipeNetwork Editor is optional*

Although strongly recommended, the PipeNetwork Editor is optional: you can use it to graphically create a Pipe Network or you can decide to manually instantiate Pipe and Pipe Blocks by calling the appropriate functions in the Pipe Library directly from the IEC 61131-3 editors (SFC, FBD, ST, IL, FFLD).

### Grid

The layout of the editor is grid oriented, which means that items (except the comments) are placed in the middle of a rectangular area called a grid unit.

#### NOTE

Comments are not centered in the grid unit but merely placed at the cursor position.

### 11.1.1 Inserting Pipe Blocks or Comments

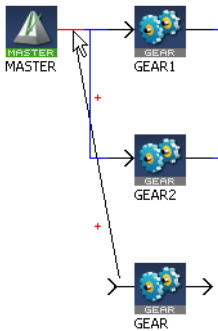
To insert Pipe Blocks or comments, right-click on a free grid unit and choose the corresponding command in the contextual menu.

### 11.1.2 Inserting Connections

Connections are simply inserted by clicking on an [adequate<sup>1</sup>](#) point and dragging the mouse to another adequate point. For more details, refer to [Adding Motion](#).

Connections are drawn between an input and an output port of two different Pipe Blocks. Connections can be drawn from input to output ports or vice-versa.

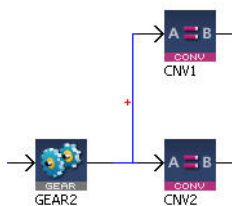
<sup>1</sup>As explained below, an adequate point depends on the type of the connection



**Figure 6-8:** Pipe Network - Create a Link

When you try to connect two Pipe Blocks, the editor highlights the target port in **red** when the connection is allowed.

*Relation type for output-input is 1-n*



**Figure 6-9:** Pipe Block - Relation Type for Output-Input

One output can be connected to several inputs, but one input can only be connected to one output.

### 11.1.3 Connect a Comment to a Pipe Block

Connections are drawn between the text area of the comment (title bar is reserved for moving the comment) and the Pipe Block icon.

#### NOTE

The connection cannot be drawn from the Pipe Block to the comment. Allowed target is not highlighted.

### 11.1.4 Pipe Network Editor Controls

#### Basic Functions

This section details the basic moving and editing functionality with Pipe Network blocks.

*Edit Pipe Blocks or Comments*

To edit Pipe Blocks or comments, double-click an item to open its **Property** dialog box

#### NOTE

You can also access the property dialog box of an item through its contextual menu.

*Move Pipe Blocks*

Pipe Blocks are moved by dragging their center. When dragging a Pipe Block, a colored shadow is shown under the Pipe Block indicating where the Pipe Block is dropped. When the shadow fills out a complete grid unit, the Pipe Block is placed in this grid unit.

*Move Connections*

You can move an end-point of a connection from one item to another. To do this, select the connection and drag an end-point to a new target.

*Move Comments*

You can drag-and-drop a comment by selecting its title bar.

*Remove Pipe Blocks, Comments and Connections*

Select one or several items (Pipe Blocks, comments or connections) and choose **Delete Selection** in the menu.

**NOTE**  
You can select several items by clicking on them while pressing either the Ctrl or Shift keys.

*Insert rows and columns*

When the shadow does not fill out a whole grid unit, but is squeezed between two grid units, a row or column is inserted before placing the Pipe Block in the newly created grid unit. When the Pipe Block is dropped on the crossing point of four grid units, a row and a column are inserted simultaneously.

**NOTE**  
You cannot drop a Pipe Block into a grid unit which is already occupied by a Pipe Block or a comment.

*Remove Rows and Columns*

It is not possible to remove rows or columns. If a row or column has been inserted by error, click the **UNDO** icon in the toolbar (**Ctrl+Z**).

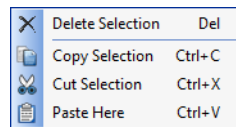
**Multiple Selections**

This section details selecting, moving, copying, pasting, and deleting multiple Pipe Network blocks.

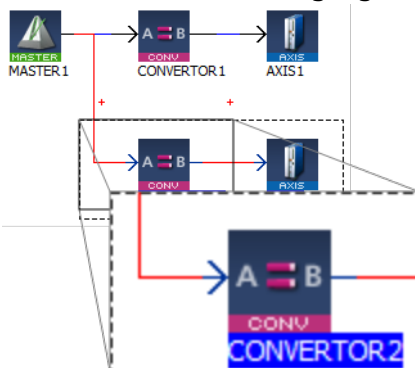
*Selecting multiple Pipe Network blocks.*

Multiple blocks may be selected by either:

- Ctrl-clicking on individual blocks and connectors
- Click-dragging an area to select an area of blocks and connectors



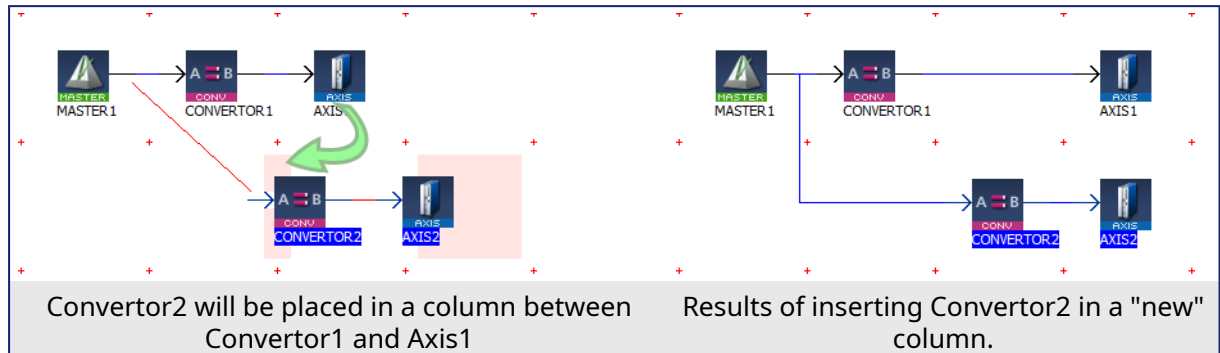
Selected blocks will be highlighted and connectors will be red.



**TIP**  
Be sure to select the connectors between blocks. If the connectors are not selected, any copied blocks will be unconnected.

*Moving multiple Pipe network blocks.*

Once selected, multiple Pipe Network blocks and connectors can be moved as a group. Simply click on a block and drag it to its new location. Light red rectangles show where the blocks will be placed. Narrow rectangles represent that a block will be placed between two current blocks, which will be moved to accommodate the repositioned blocks.



*Copying multiple Pipe Network blocks.*

Selected blocks and connectors may be copied by pressing Ctrl-C, or by using the right-mouse menu.

*Pasting multiple Pipe Network blocks.*

Selections may be pasted in one of two ways.

- Right-clicking in the Pipe Network Editor and selecting Paste Here.
- Clicking in an empty location will place the blocks starting at that spot.
- Clicking on an existing block will place the copied blocks on top of the selection. You will need to move the pasted blocks to see the entire network.
- Pressing Ctrl-V then clicking in the location you want the blocks to be placed.

#### **TIP**

Copied connectors will only be pasted when they are connected to a copied block.

*Deleting multiple Pipe Network blocks.*

Selected blocks may be deleted using either of two methods.

- Pressing the **Delete** key.
- By right-clicking on a selection and choosing **Delete Selection**.

### 11.1.5 Plug/Unplug Channels

Right-click on a Pipe Block to plug/unplug a channel of the Softscope. For more details, refer to [How to Plug Motion Variables](#).

## 11.2 Cam Profile Editor

### 11.2.1 About the Cam Profile Editor

To open the cam profile editor in a new tab of the workspace, you have to double-click on the profile in the [Project Explorer](#).

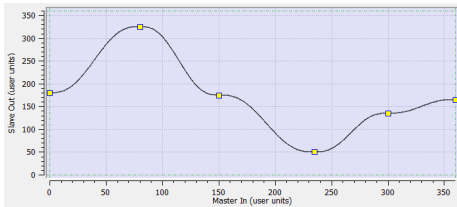


Figure 6-10: A cam profile

The cam profile editor enables you to create and/or modify a profile definition that describes the position evolution of the cam. This evolution is displayed in a 2D graphical format.

You can add, delete, or modify cam elements which consist of points and lines. Based on those elements and some constraints, the KAS IDE calculates a complete cam shape. See [Cam Profile Segment Overview](#) for more information on the segment types.

**Master/Input** (X-Axis) and **Slave/Output** (Y-axis) coordinates can be specified to define the position.

In addition to the position, it is also possible to visualize the velocity, acceleration, and jerk diagrams.

#### 11.2.1.1 Windows Overview

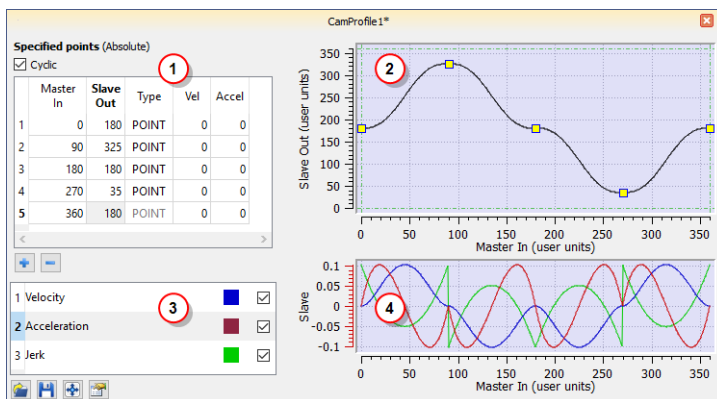


Figure 6-11: Cam Profile Editor Main Window

The cam profile editor contains four distinct parts separated by splitters:

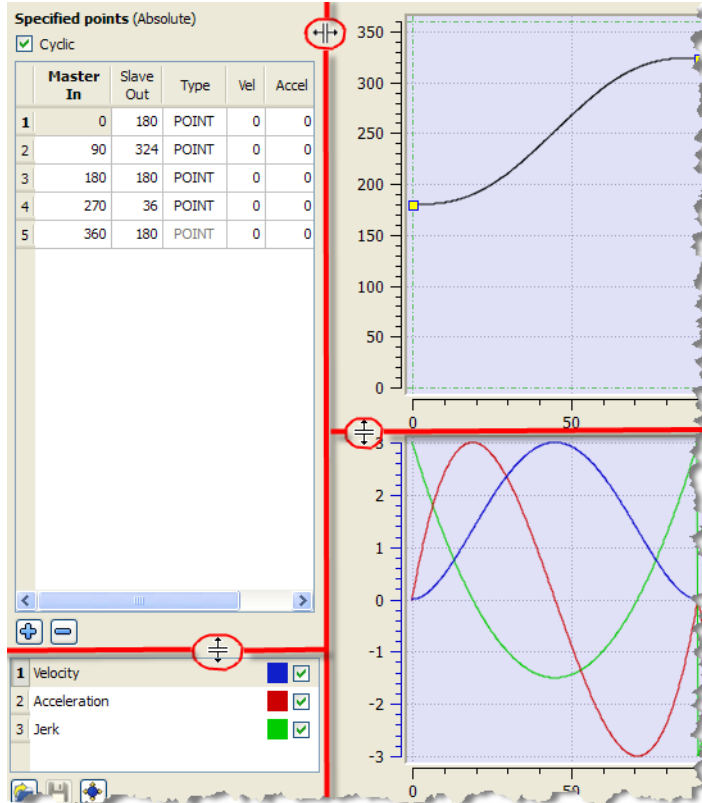
1. The [cam table](#) (see call out **1**) displays each element and allows editing of the cam.
2. The Graphical Area for the [cam profile](#) **2**  
The upper graph displays a graphical representation of the cam elements
3. The [Curve Selection and Color Table](#) **3** allows you to select which plots (velocity, acceleration and jerk) are displayed
4. The Graphical Area for [Curves](#) **4**  
The lower graph displays a graphical representation of the velocity, acceleration and jerk plots

**TIP**

Undo (Ctrl+Z) and Redo (Ctrl+Y) operations are available for any changes you make to the cam profile.

Splitters allow you to resize each part.

*Improve your display with the splitters*

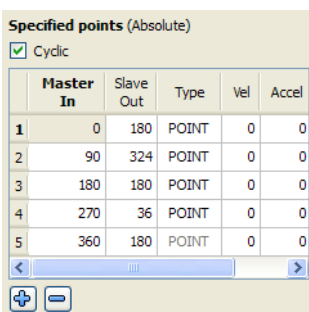


**NOTE**

The tables and the graphs are separated by a vertical splitter so that you can completely hide the tables to increase the graphical area.

For more information on cam profiles see [Adding Cam Profiles](#) and [Profiles](#).

**11.2.2 Cam Profile Editor's Cam Table**



**Figure 6-12: Cam Table**

When a new profile is created, the cam profile contains five points by default.

**NOTE**

These points could be different from those in the figure above, depending on the offsets and amplitudes specified in the cam profile (see ["Create Cam Profiles" on page 198](#)) dialog box.

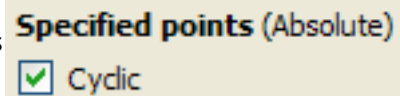
Column	Description
<b>Master/In</b>	The time is located in the <b>Master/In</b> column. It is the X-axis of the cam profile graph
<b>Slave/Out</b>	The position is located in the <b>Slave/Out</b> column. It is the Y-axis of the cam profile graph
<b>Type</b>	The <b>Type</b> column defines whether this element is a point or a line. If the element is a line, In/Out specifies the start point of the line. The next element in the table defines the end of the line. The last element type in the table cannot be changed, since a line cannot exist as the last element.
<b>Vel</b>	The <b>Velocity</b> of the current element (first derivative)
<b>Accel</b>	The <b>Acceleration</b> of the current element (second derivative)

Table 6-6: Cam Editor - Table Parameters

### About Cyclic Cam Element

If the *Cyclic* check box is selected, the cam profile is executed cyclically. This means that, when the axis attached to this cam runs continuously, the same profile is executed again. In this case, the first and last element must have the same **Vel** and **Accel** values.

Therefore, changing the **Vel** or **Accel** value of the first or last elements automatically changes the other elements' value.

**NOTE**

When *Cyclic* is first turned on, the Vel/Accel values will automatically be copied from the first element to the last element when they do not match. A warning dialog is displayed to inform you that this change has happened. This alert can be suppressed until KAS is closed.

There are some combinations of points and lines where *Cyclic* will automatically be turned off. If this occurs, the cyclic checkbox label will be changed to *Cyclic (automatically turned off)*. The following changes to the profile will automatically turn off cyclic:

1. The first element has been changed from a point to a line. If needed, cyclic can manually be turned back on which will affect the velocity of the last element.
2. The next to last element has been changed from a point to a line and now both first and next to last elements are lines. Cyclic will be disabled and will only be re-enabled when the first and next to last elements are not lines.
3. The first element is a line and the first element is moved. If needed, cyclic can manually be turned back on which will affect the velocity of the last element.
4. The first element is a line and the second element is moved. If needed, cyclic can manually be turned back on which will affect the velocity of the last element.
5. The first element is a line and the last elements velocity (or slope line) has changed. If needed, cyclic can manually be turned back on which will change the velocity setting just made.

#### 11.2.2.1 Modifying an Element using the Cam Table

You can modify a cam element by clicking in the **Master/Input**, **Slave/Output**, **Vel**, or **Accel** column and typing in a new value. For **Type**, refer to [Modifying the Type of a Cam Element](#).

The graphs are updated automatically when an element changes.

Some rules apply to the value entered:



- The **Master/Input** value must lie between adjacent **Master/Input** points
- The **Master/Input** value of the first and last point cannot change. These values are determined by the profile properties X offset and X amplitude
- The **Slave/Output** value must lie between the Y offset and Y amplitude set in the profile properties

If an entered value is invalid (due to the interpolation calculation), it is superseded with the original value without any error message.

#### About interpolation

The section between two consecutive cam elements is automatically calculated by a fifth order polynomial algorithm.

Modification of one cam element only affects the two adjacent segments.

#### 11.2.2.2 Modifying the Type of a Cam Element

The type of element can either be a point or a line. The element type can be modified by double-clicking in the **Type** column of an element and then clicking on the down arrow. A list of choices is displayed as shown. Select the type of element from the list.

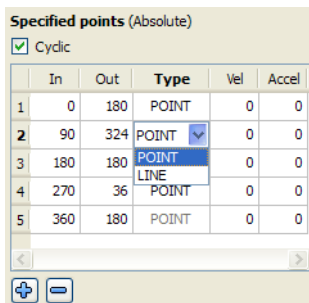


Figure 6-13: Modifying an Element Type

#### 11.2.2.3 Cam Table Contextual Menu

Right-clicking on an entry in the cam table displays a contextual menu.

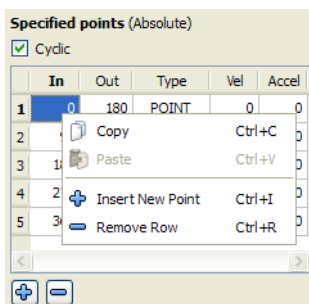



Figure 6-14: Cam Table Contextual Menu

Command	Shortcut	Description
<b>Copy</b>	Ctrl+C	Copy data from the selected cell in the clipboard
<b>Paste</b>	Ctrl+V	Paste the data from the clipboard into the selected cell
<b>Insert New Point</b>	Ctrl+I	Inserts a new row in the cam table above the highlighted entry. This command is described in <a href="#">Adding a Point to the Cam Table</a>
<b>Remove Row</b>	Ctrl+R	Deletes the row that contains the highlighted entry. This command is described in <a href="#">Remove a Point from the Cam Table</a>

### 11.2.2.4 Adding a Point to the Cam Table

You can add a point to the cam table using one of the following methods:

- Use the menu in the cam table (shown in [Cam Table Contextual Menu](#))
- Click the button  located below the cam table
- Use the menu in the cam profile graph

All of these methods displays the **Add New Point** dialog box:

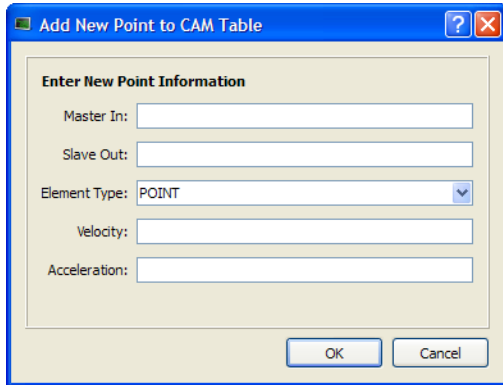


Figure 6-15: Add New Point

Field/Command	Description
<b>Master In</b>	The X value of the new point
<b>Slave Out</b>	The Y value of the new point
<b>Element Type</b>	POINT or LINE
<b>Velocity</b>	The velocity of the new point (first derivative)
<b>Acceleration</b>	The acceleration of the new point (second derivative)
<b>OK</b>	Accept the entry and verify if the point can be added.
<b>Cancel</b>	Cancel the dialog box – no point is added.

Table 6-7: Cam Editor - New Point Parameters

When you click OK, a check is performed to see if the point can be added to the cam profile. If not, an error dialog box is displayed.

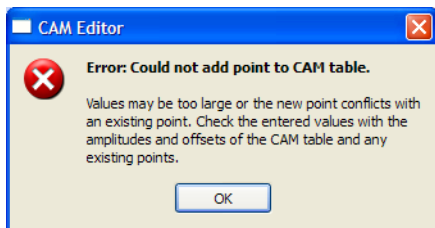


Figure 6-16: Cam Table Contextual Menu


If no problem is found, the point is added to the cam table and the graphical plots are updated.

#### NOTE

A new point cannot be inserted above the first element in the cam table.

### 11.2.2.5 Remove a Point from the Cam Table

You can remove a point from the cam table with one of the following methods:

- Use the menu in the cam table (shown in [Cam Table Contextual Menu](#))
- Click the button  located below the cam table
- Use the menu in the cam profile graph

The selected point is removed without prompting.

#### NOTE

The first and last points cannot be removed.

### 11.2.3 Cam Profile Graph

The upper graph displays the points and lines specified in the cam table along with the calculated curve. It also allows you to add, delete or modify a cam element.

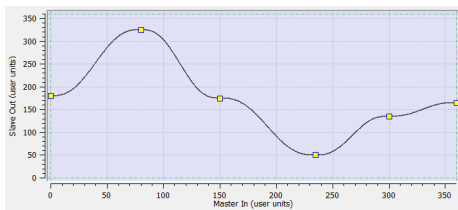



Figure 6-17: Cam Profile Graph

Points and endpoints of lines are displayed as yellow squares (■) in the graph. The profile offset and amplitude specified in the cam profile editor (see ["Create Cam Profiles" on page 198](#)) are displayed with a green dashed rectangle. The yellow squares are always contained within the green dashed rectangle (although calculated points can extend outside it).

#### 11.2.3.1 Modifying an Element

You can modify the profile by moving point with the mouse as follows:

1. Move the mouse over a yellow square (the cursor becomes  indicating that the point can be selected)
2. Click to select the point and hold down the mouse button (left-click). When you move the mouse, the point follows the cursor (note that graphical curves and In/Out values are dynamically updated)

In addition, when a point is selected, a slope line is drawn over the point. This line is dashed purple with two additional grips (◆) attached to it. The slope line can be used to change the velocity of the selected point.

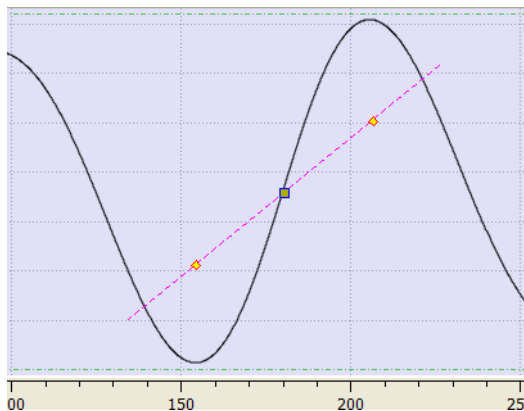


Figure 6-18: Cam Profile Graph - Slope Line

You can change the velocity of the selected point as follows:

1. Move the mouse over a slope grip (◆). The cursor changes to an open hand (☞)
2. Click to select the grip and hold down the mouse. The cursor changes to a closed hand (☜)
3. When you move the mouse, the slope line follows the cursor, rotating about the selected point and causing the velocity of the selected point to change. (Note that graphical curves and Vel value are dynamically updated)

### 11.2.3.2 Cam Profile Graph Contextual Menu

A right-click on the cam profile graph displays a contextual menu.

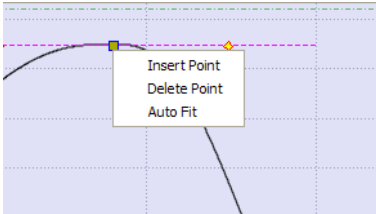


Figure 6-19: Cam Profile Graph - Contextual Menu

Command	Description
Insert Point	Inserts a new point at the X-Y location of the cursor
Delete Point	Deletes the highlighted point If the mouse is not near enough to a point, no point is highlighted and this command remains grayed-out
Auto Fit	Adjusts the zoom and pan settings so that the entire graph is displayed in the graphical area

### 11.2.3.3 Zoom In and Out

In the cam profile graph, you can zoom in or out as follows:

1. Move the cursor in the graphical area
2. Turn the mouse wheel forward or backward


The current cursor becomes the center point of the zoom function and the area under the cursor remains stationary on the graph.

### 11.2.3.4 Panning

In the cam profile graph, you can also pan (or move) in any direction as follows:

1. Click on any part of the graph (but not on a yellow square) and hold down the mouse button (left-click)
2. Move the mouse to move the graph accordingly

### 11.2.3.5 Restoring Zoom and Pan

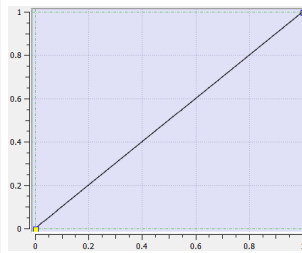
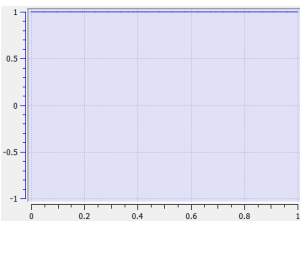
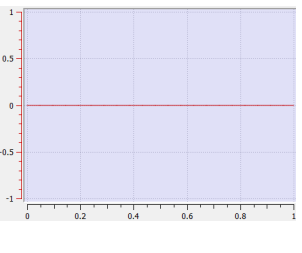
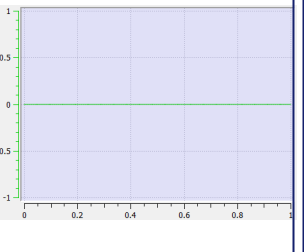
To restore the zoom and pan settings, so the entire curve is displayed in the graphical area, click on the Auto Fit button  or select the Auto Fit command in the cam profile graph menu.

## 11.2.4 Cam Profile Segment Overview

### Point Segment Type

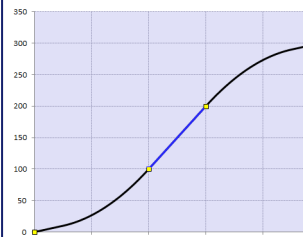
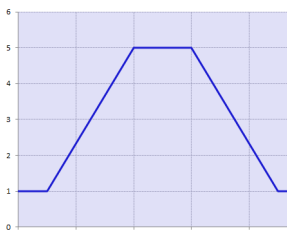
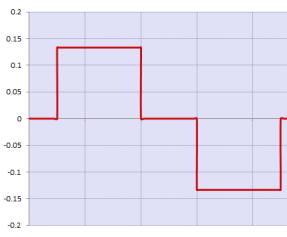
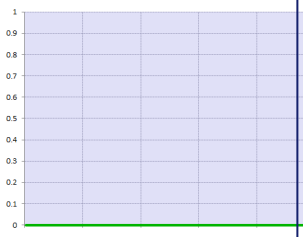
<b>Supported by</b>	KAS IDE Profile Editor, <a href="#">MLProfileBuild</a> (default option)		
<b>Continuous velocity</b>	✓		
<b>Continuous acceleration</b>	✓		
<b>Interpolation method</b>	5 <sup>th</sup> order polynomial. $f(x) = Ax^5 + Bx^4 + Cx^3 + Dx^2 + Ex + F$		
<b>Advantages</b>	With only a few segments, this type can be used to define profiles with continuously changing accelerations. For example, sinusoidal profiles can be emulated with 6 to 12 point segments.		
<b>Disadvantages</b>	One must specify the velocity and acceleration at the endpoints for each segment. It is difficult to use the point segment type to define constant acceleration or constant velocity segments.		
<b>Segment Curve</b>	<b>Velocity</b>	<b>Acceleration</b>	<b>Jerk</b>

### Line Segment Type

<b>Supported by</b>	KAS IDE Profile Editor, <a href="#">MLProfileBuild</a>		
<b>Continuous velocity</b>	<b>X</b>		
<b>Continuous acceleration</b>	<b>X</b>		
<b>Interpolation method</b>	Linear function. $f(x) = Ax + B$		
<b>Advantages</b>	This segment type easily defines constant-velocity segments. It can be used in very large numbers to define profiles of any size without having to specify the velocity or acceleration at the segment endpoints.		
<b>Disadvantages</b>	Profiles can result in discontinuous velocities.		
<b>Segment Curve</b>	<b>Velocity</b>	<b>Acceleration</b>	<b>Jerk</b>
			

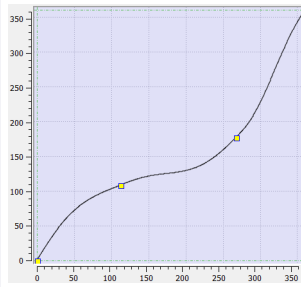
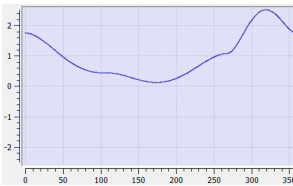
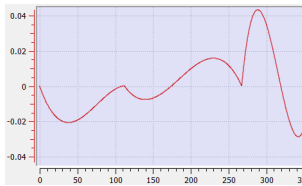
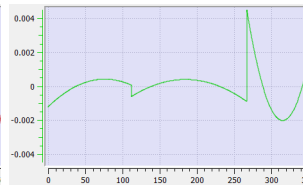
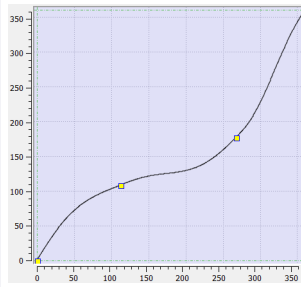
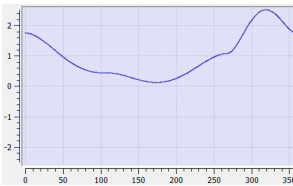
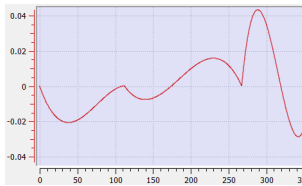
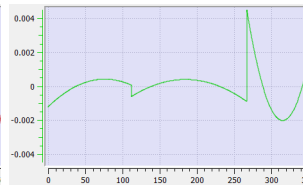
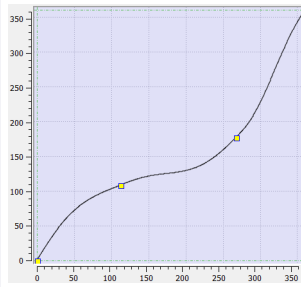
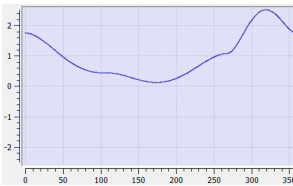
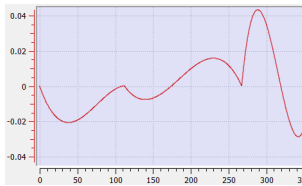
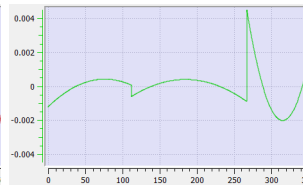
### Parabolic Segment Type

<b>Supported by</b>	KAS IDE Profile Editor, <a href="#">MLProfileBuild</a> (parabolic option)
<b>Continuous velocity</b>	✓
<b>Continuous acceleration</b>	✓
<b>Interpolation method</b>	Linear function: $f(x) = Ax + B$ and a 2 <sup>nd</sup> order polynomial: $f(x) = Cx^2 + Dx + E$
<b>Advantages</b>	This segment type is used to define constant acceleration portions of a profile. This minimizes the peak acceleration needed to move from one cam point to another. This can be useful when the motors cannot support the accelerations used by other segment types.
<b>Disadvantages</b>	Acceleration is discontinuous which can lead to additional electrical stress on the drives and motors.

Segment Curve	Velocity	Acceleration	Jerk
			

In the example shown, the blue line represents the linear (constant velocity) part of the segment, while the black lines represent the parabolic (constant acceleration) parts of the segment.

### Spline Segment Type

<b>Supported by</b>	KAS IDE Profile Import. The points are created by separate software and is imported into KAS IDE project.								
<b>Continuous velocity</b>	✔								
<b>Continuous acceleration</b>	✔								
<b>Interpolation method</b>	3 <sup>rd</sup> order polynomial. $f(x) = Ax^3 + Bx^2 + Cx + D$								
<b>Advantages</b>	With only a few segments, this segment type can be used to define profiles with continuously changing velocities. Only the positions of the master and slave need to be specified. This produces smoother profiles than using line segments.								
<b>Disadvantages</b>	Since only positions are specified, the user has less control over the velocities and accelerations that occur throughout the profile.								
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #1a3d54; color: white;"> <th style="padding: 5px;">Segment Curve</th> <th style="padding: 5px;">Velocity</th> <th style="padding: 5px;">Acceleration</th> <th style="padding: 5px;">Jerk</th> </tr> </thead> <tbody> <tr> <td style="text-align: center; padding: 5px;">  </td> <td style="text-align: center; padding: 5px;">  </td> <td style="text-align: center; padding: 5px;">  </td> <td style="text-align: center; padding: 5px;">  </td> </tr> </tbody> </table>		Segment Curve	Velocity	Acceleration	Jerk				
Segment Curve	Velocity	Acceleration	Jerk						
									



### 11.2.5 Curve Selection and Color Table

Velocity (first derivative), acceleration (second derivative) and jerk (third derivative) plots are displayed in the lower graph. If the element is a line, the velocity is constant and acceleration is 0.

With the check boxes in the Curve selection table shown in figure below, you can select or clear each individual curve to be displayed.

When a curve is selected (see blue highlighted row in figure below), the Y-scale of the Curves graph is adjusted to display the Y-scale of the selected curve. Also, the color of the 'tick' line of the scale is changed to match the color code of the selected curve.

1	Velocity	<input checked="" type="checkbox"/>
2	Acceleration	<input checked="" type="checkbox"/>
3	Jerk	<input checked="" type="checkbox"/>

Figure 6-20: Curve Selection Table

#### 11.2.5.1 How to change color

You can change the color of a plot as follows:

1. Double-click on a colored square shown in the Curve Selection Table to open the color selection dialog box

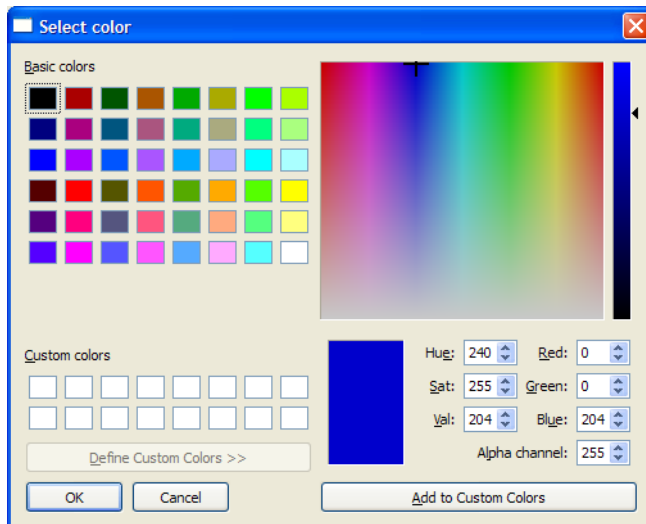


Figure 6-21: Standard Color Selection

2. Click on an existing color square to select it, or specify the numerical values for a color. (You can also move the black indicator on the right side until the desired color appears in the large colored rectangle)

### 11.2.6 Curves Graph

Velocity (the first derivative), acceleration (the second derivative) and jerk (the third derivative) curves are displayed in the lower graph. All plots are displayed by default.

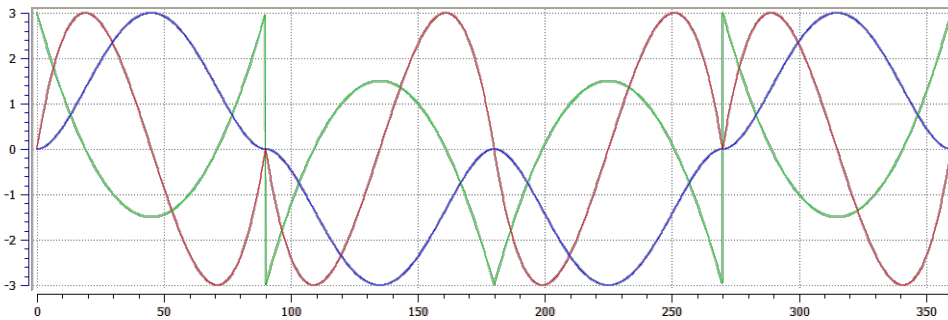


Figure 6-22: Curves Graph

With the check boxes in the Curve selection table shown in [Curve Selection Table](#), you can select or clear each individual curve that you want to be displayed.

The Y-scale of the Curves graph is adjusted to display the Y-scale of the selected curve in the Curve Selection Table. The color of the Y-axis scale "tick" lines is also changed to match the color code of the selected curve.

Y axis	Unit	Description
Vel	Units/Time	Being the rate of change of position, the velocity is the ratio between the slave and master derivatives
Accel	Units/Time <sup>2</sup>	Rate of change of velocity with time
Jerk	Units/Time <sup>3</sup>	Rate of change of acceleration; more precisely, the derivative of acceleration with respect to time

In general the numbers relate to how the Y-axis positions (Cam Output) change with respect to the X-axis positions (CAMinput).

The zoom and pan functions, when performed on the cam profile graph, are duplicated in the Curves graph.

Zoom and pan functions are not available when the cursor is in the curves graph.

### 11.2.7 Reload, Save, Auto Fit, and Properties Buttons

The following buttons are provided:

Icon	Description
<b>Reload</b>	Reload the saved profile. If unsaved changes have been made to the profile, a dialog box asks you to confirm that you want to discard the changes.
<b>Save</b>	Save a modified profile.
<b>Autofit</b>	Adjusts the zoom and pan settings so that the entire graph is displayed in the graphical area.
<b>Properties</b>	Open the Cam Profile Properties dialog box to modify the Master/Input and Slave/Output Offset and Scale values

Table 6-8: Cam Editor - List of Icons

### 11.2.8 Import Cam Profile

The KAS IDE can import cam profiles from several sources.

- Cam profiles (.CAM file format) which were exported from another KAS project.
- User-defined cam profiles (in .CSV format) as described below. Upon importing, the .CSV file is converted to a .CAM file.

Row	Syntax
1	CYCLIC;YES;
2	TABLE_BEGIN;;
3	0;0;SPLINE
4	X;Y;SPLINE
:	X;Y;SPLINE
N	1000;1000;SPLINE
N+1	TABLE_END;;

Each row from 4 to N specifies the successive points that are part of the cam profile. The X and Y coordinates can be specified as floating-point values with sufficient digits after the decimal point (example: 995.2514255). To be valid, a CSV file must have **at least 4 spline segments** in it. See [Cam Profile Segment Overview](#) for more information on the segment types.

#### NOTE

Splines are the only segment type that is currently supported when importing CSV data into KAS IDE. Other segment types are supported by importing and exporting .CAM files.

When a CSV file is imported the X, Y values are normalized with respect to maximum X, Y values present in the CSV file. The normalized X, Y values are scaled with respect to Master/Input scale and Slave/Output scale. They are added with Master/Input Offset and Slave /Output Offset respectively and will be displayed in the Specified points (Absolute) section of the cam profile.

#### Example:

CSV file X,Y Values:

```
0;0;SPLINE
100;111;SPLINE
200;222;SPLINE
300;333;SPLINE
```

Max Value in CSV is

```
300;333;SPLINE
```

Normalized values:

```
0; 0;
0.3333333333333333; 0.3333333333333333;
0.6666666666666667; 0.6666666666666667;
1;1;
```

Offset:

```
10      20
```

Scale:

```
300      360
```

Value displayed in profile:

```
10;20;
109.99999999999999; 139.99999999999989;
210.00000000000009; 260.0000000000011;
310;380;
```

### 11.2.8.1 About the Import

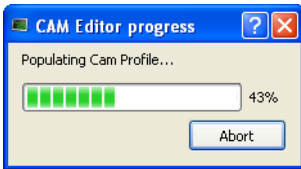
A quick validation is performed when the profile is first imported.

- The profile name is checked and if it is in use you are promoted to provide a new name.
- The data format is validated and we check to see if the profile can be compiled without error.

If cyclic is on and the Vel/Accel values of the first element do not match the Vel/Accel of the last element, the first elements Vel/Accel will be copied to the last elements. A warning message will be posted to the log if this change takes place.

### 11.2.8.2 When Displaying the Imported Cam Profile

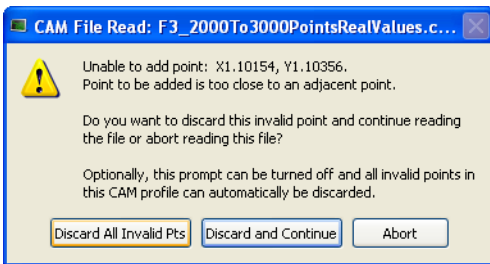
When you display an imported profile a dialog box indicates the progression of the import process.



Click the **Abort** button to abort the process, then a default cam profile is created.

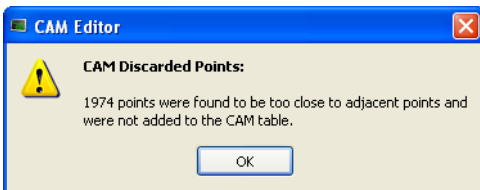
### 11.2.8.3 About Invalid Data

When you display a CAM profile where two points are too close, a dialog box indicates the error.



Click the **Discard All Invalid Pts** button to discard all additional invalid points found in this cam profile.

A summary is displayed when the process is finished.



## 11.3 Soft Oscilloscope

The soft oscilloscope (commonly known as "softscope" or "scope") is a tool which allows you to view, in a two-dimensional graph, one or more variables' evolution (vertical axis) across the time (horizontal axis).

The scope (as seen in [Figure 6-23: Example of using the Oscilloscope](#)) has a set of channels where each can acquire the evolution of a value. A value may be the feedback position of an axis, the speed of a machine, or anything else that can be measured with the oscilloscope probes. For details on how to assign a variable to a channel, see ["Plugging Probes, Mapping Variables"](#) on page 540.



**Figure 6-23:** Example of using the Oscilloscope

The soft oscilloscope is a child window in the workspace which has two visually distinct parts:

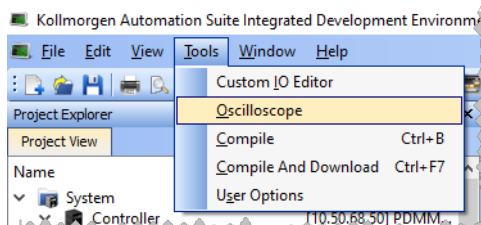
- The [Control Panel](#) (1) enables you to assign variables to a channel and to change the settings of the soft oscilloscope (including those of the channels)
- The [Graphical Area](#) (2) shows the traces (or variable values) acquired by the channels
- The control panel and the graph are separated by a splitter which allows you to change the size of the two panes.

#### **TIP**

You can hide the control panel by dragging the splitter all the way to the left. To show the control panel again, click and drag the splitter to the right.

### 11.3.1 Access the Soft Oscilloscope

In order to access the softscope, select the **Oscilloscope** command from the **Tools** menu.



**Figure 6-24:** Accessing the Scope

### 11.3.2 About OpenGL

For the Graphical Area, the scope uses **OpenGL** for performance reasons. If you encounter problems in the quality of drawings, we suggest that you consider the following points before contacting our support desk:

*Check that your graphical card driver is up-to-date.*

Newer drivers often fix the rendering bugs of OpenGL.

*Disable some optimizations on the Display hardware acceleration*

Open your operating system's **Display Properties**<sup>1</sup>. In the **Settings** tab, click the **Advanced** button, then select the **Troubleshoot** tab. If **Hardware acceleration** is set to full, try to disable some optimizations. This procedure has proven to be useful in particular with cursor drawing problems that appear when the user performs high-zooming operations (the cursor can indicate a value which is out of the trace).

*Change the settings of your graphic card*

Open the manufacturer-specific settings of your graphic card. If there are some settings related to **Performance and quality**, try to set them to **quality** (but not high quality) instead of performance, at least for the program **kaside.exe**. This solves many drawing problems that occur when zooming a lot in the graph.

*Display a given amount of samples, according to the refresh rate*

If your channels have acquired a large number of samples, and the refreshing of the graph does not occur frequently enough, do not display all samples at the same time either by:

- Hiding some less useful channels (use the [visibility](#) property)
  - Reducing the time-base and/or restricting the time-frame in the time position.
- In any cases, this action does not stop acquisition or lose your acquired samples.

#### **!** IMPORTANT

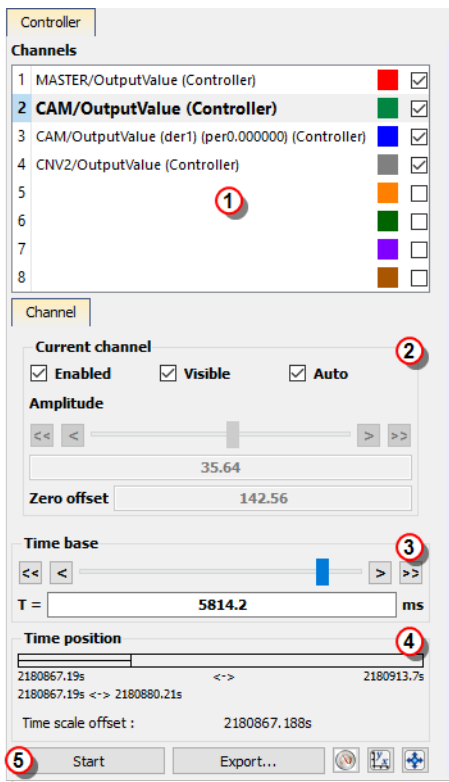
Disabling most or all OpenGL accelerations is compensated by an increase in CPU consumption. It can lead to a point where the soft oscilloscope is not very usable when limited hardware is trying to display loads of samples.

### 11.3.3 The Control Panel

As shown in [Figure 6-25: Oscilloscope Control Panel](#), the control panel consists of the following items:

---

<sup>1</sup>The Properties command is accessible in the contextual menu on your desktop (you can also access the Display from the Windows Control Panel)

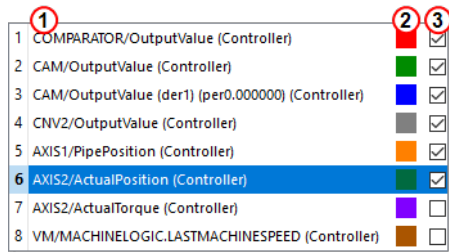


1. The **Channels** list
2. The **Current channel** property
3. The **Time-base**
4. The **Time position**
5. Five buttons

**Figure 6-25:** Oscilloscope Control Panel

1. The **Channels** list

This lists the eight available channels, and the variables assigned to each.



**Figure 6-26:** Scope Control Panel - Channels

For each channel, it displays:

- The **name** of the associated variable 1
- The **color** of the associated trace in the graph with a color icon 2
- The **visibility** of the associated curve with a check box. 3 This lets you quickly show/hide each captured trace.

**TIP**

You can change the color of a curve by double-clicking on its color icon, and its visibility by clicking on its check box.

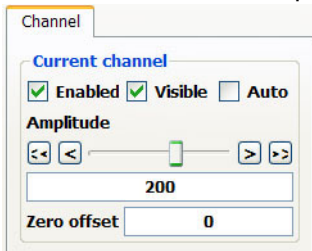
**TIP**

Right-click on a channel to map, unmap, or edit the color of a variable's trace.

When selecting a channel in the channels list, it is superimposed on the existing traces, and some related information are displayed on the left and lower sides of the graph.

### 2. The **Current Channel**

This section details the properties of the currently selected channel.



Properties	Description
<b>Enabled</b>	A channel has to be enabled to acquire the samples sent by its associated probe
<b>Visible</b>	A channel has to be visible to be drawn on the graph. Even if not visible, it continues to acquire the samples sent by its associated probe
<b>Auto</b>	A channel in auto mode automatically adapts its amplitude ( <a href="#">unit/division<sup>1</sup></a> ) and zero offset in order to be able to display all its samples. Setting the auto mode disables the possibility of changing the Amplitude and the Zero offset (see <a href="#">Setting Scale</a> for more details about scaling)
<b>Amplitude</b>	Allows you to control the amplitude (unit/division) of the channel. The buttons and slider change the amplitude according to a logarithmic scale. The dialog box allows a more precise definition of the value
<b>Zero offset</b>	The curve is vertically shifted so that this value is located halfway through the graph height

Table 6-9: Scope - Current Channel Properties

#### NOTE

On some special devices, some more tabs that are specific to extra configurations appear in this widget. For example, S300 device provides trigger functionality, so an additional tab is displayed for the trigger configuration.

### 3. The **time-base**

This enables you to set the speed at which all the lines for each channel are drawn, and is calibrated in milliseconds per division.

Its usage is similar to the Amplitude property described in the above section. The time-base can always be changed, even during sampling (see also [Time Scale](#)).

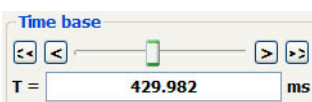


Figure 6-27: Scope Control Panel - Time-base

<sup>1</sup>The term refers to the time-base value for the X-axis and to the amplitude value for the Y-axis. For example, if the user sets a time-base of 10ms and an amplitude of 1, each division in the soft oscilloscope grid corresponds to a time of 10ms for the X-axis and an amplitude of 1 for the Y-axis.



To setup the time-base properly, the total measurement duration and the required time resolution have to be taken in account.

#### 4. The **time position**

This enables you to change the time-frame of the acquired samples shown on the graph. It is composed of:

- A single **horizontal line** representing all the acquired samples with start and stop timings
- A **rectangle** representing only the time slot of the acquired samples, which is displayed in the graphical area (the time-frame) with timings:



**Figure 6-28:** Scope Control Panel - Time Position

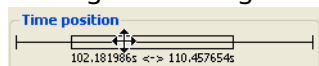
#### NOTE

The acquisition of samples is limited to 100'000 cycles (ie. 100 s when cycle time is set to 1000  $\mu$ s, and 25 s when cycle time is set to 250  $\mu$ s). When you reach this limit:

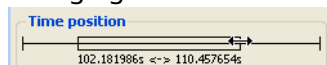
- The first data that are added to the queue are the first data to be removed (FIFO queue)
- The start timing increases

You can change the time slot with the mouse by:

- moving the rectangle



- changing the size



The **Time Scale Offset** is the time value of the first sample the graph when plotting is started. Using this as an offset, the time axis is always started at 0 seconds. To get the actual time value of any sample, add the time scale offset to the Time axis value.

$$\text{Actual Sample Time Value} = \text{Time Scale Offset} + \text{Time axis value}$$

*How to set the time-frame?*

When clicking anywhere on the horizontal line, the time-frame is centered on the clicked point. It is also possible to move the time-frame by clicking on its rectangle part and dragging.

You can resize the time-frame in a user-friendly manner by clicking on its left or right ends and dragging.


#### NOTE

During acquisition the time position item is disabled and displays the progression of acquisition.

#### 5. The five buttons

At the bottom of the controls are five buttons:

1. The **Start/Stop** button  allows you to start or stop the acquisition of samples. When starting acquisition, all previous samples are lost.
2. The **Export...** button  allows you to save the acquisition data in a CSV file. For more details, [The Graphical Area](#).

3. The **Plug Trace Times Channels** button  allows you to quickly load and display the four following channels

- **Channel 1: Cycle Jitter** (in  $\mu\text{s}$ )

When the motion is started, the current cycle time remains constant on an average of several cycles, and equal to the EtherCAT cycle time which is a constant value (1000, 500 or 250  $\mu\text{s}$ ). The Cyclejitter is due to EtherCAT transmissions that can vary in a particular cycle (see call out **1**).

The channel 1 of the scope monitors the time difference between the expected Cycle Time and the actual Cycle Time. (see figure below).

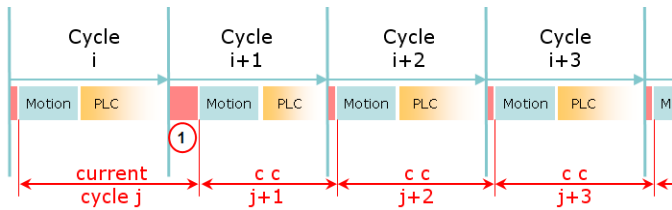


Figure 6-29: Cycle Time Calculation

- **Channel 2: Motion execution time** (microseconds)
- **Channel 3: PLC execution time** (microseconds)
- **Channel 4: Real Time Margin** (microseconds) This channel monitors the available execution time (Cycle Time Period x Controller Core Count - EtherCAT network execution time - MotionExecTime - PLCProgExecTime). The measurement is based on CPU usage, which provides the most accurate measurement. The measurement is updated each cycle period.

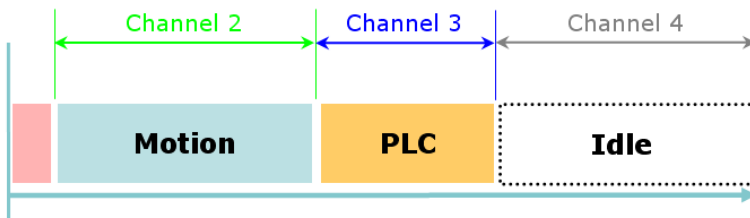
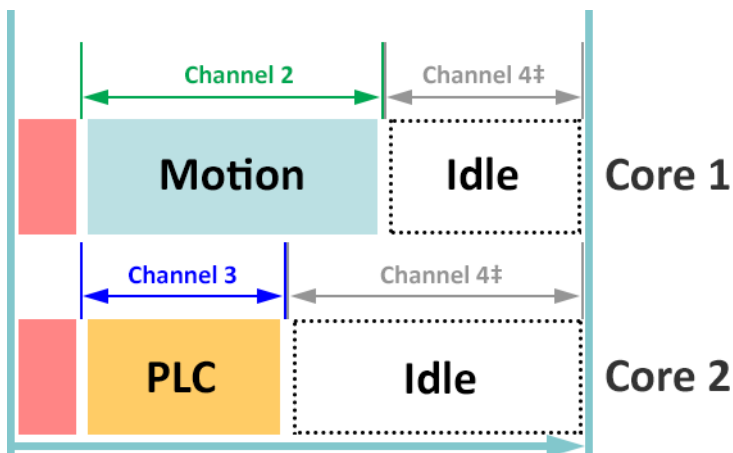


Figure 6-30: Motion, PLC and Real Time Margin Time Calculations





‡ Channel 4 is the total amount of time across all CPU cores.

**Figure 6-31:** Multiple Core Controller, PLC and Real Time Margin Time Calculations.

#### NOTE

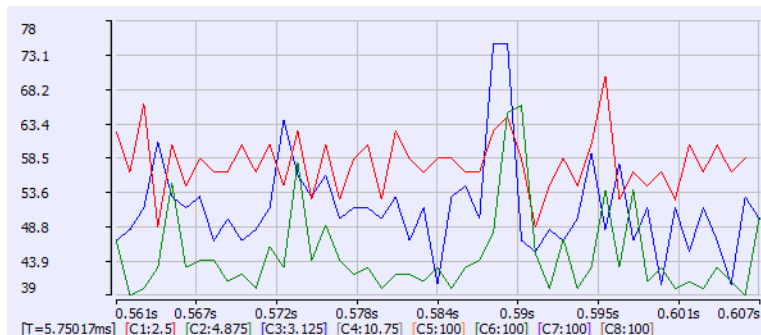
This feature is **not** relevant with the KAS Simulator. The MotionExecTime and PLCProgExecTime traces will be visible with simulated values. The CycleJitter and RealTimeMargin will always remain at zero with the simulator.

For more explanations, refer to [Tasking Model / Scheduling](#)

4. The **graduations** button  displays or removes the axis graduations of the graphical area. By graduations are visible by default.
5. The **autofit** button  changes the time-frame of the graph and amplitudes and zero offsets of channels so that they all fit entirely into the graphical area.

### 11.3.4 The Graphical Area

The graph displays a subset of the collected data: the **time-frame**.



**Figure 6-32:** Example of the traces generated in the Oscilloscope.

To better view and analyze the data, the graph has the following features:

- **Graduations** are displayed on the left and lower sides of the graph
- Information concerning the time-frame of the graph and the amplitude of channels also appears at the bottom of the graph. The current channel amplitude is underlined and the coordinates of the nearest collected sample are displayed
- It is possible to **zoom** in the graph using various methods (for more details, see [Trace Zoom Feature](#))
- It is possible to **move** the contents of the graph within the time-base (for more details, [Trace Zoom Feature](#))

#### NOTE

Moving the contents is possible **only** when the acquisition is stopped.

See [How Do I Export the Oscilloscope Data?](#) for information on capturing the trace data.

### 11.3.5 Traces

The trace is the resulting graph of the variable's evolution against time, with the more distant past on the left and the more recent past on the right.

#### NOTE

The acquisition of samples is limited to 100'000 cycles (ie. 100 s when cycle time is set to 1000  $\mu$ s, and 25 s when cycle time is set to 250  $\mu$ s). When you reach this limit:

- The first data that are added to the queue are the first data to be removed (FIFO queue)
- The start timing increases

### 11.3.6 Plugging Probes, Mapping Variables

A probe is a virtual measurement point that can be connected to a variable.

Three types of variables can be mapped:

1. Pipe Block variable
2. IEC 61131-3 variable, which is any other variable.
3. PLCOpen axis values.

#### NOTE

Your application **must be connected and running** to plug a variable to the channel.

You can connect a probe to a variable in one of the following ways:

- from the Softscope, see [Mapping a Variable to the Oscilloscope](#)
- from the Dictionary, see [Plugging a probe from the Dictionary](#)
- from the Pipe Network, [Plugging a probe from the Pipe Network](#)

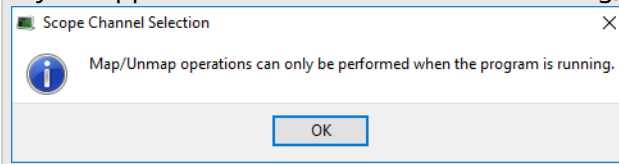
#### 11.3.6.1 Mapping a Variable to the Oscilloscope

A probe (variable) can be directly plugged or mapped from within the Oscilloscope:

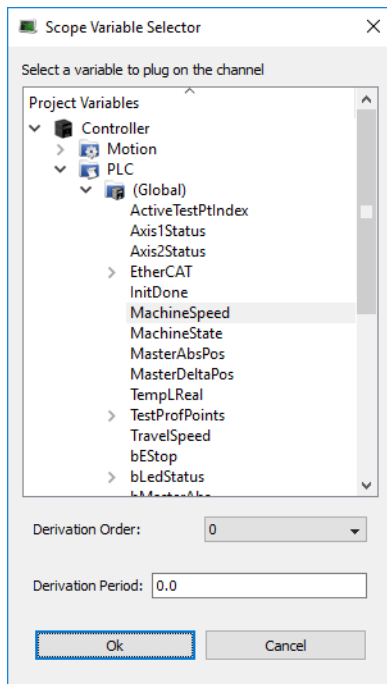
1. Right-click on any channel (already mapped or not) in the channels list and select the **Map** option to open the **Scope Variable Selector**.

#### NOTE

If your application is not connected and running, the following message is displayed.



2. Navigate through the available variables and select the one you want to connect to the channel



**Figure 6-33:** Using the Variable Selector to choose a variable to be mapped in the Oscilloscope

#### NOTE

The **Scope** Variable Selector contains only the PLC variables that are eligible for the softscope (i.e. BOOL, INT, SINT, DINT, LINT, UINT, USINT, UDINT, ULINT, BYTE, WORD, DWORD, LWORD, TIME and LREAL, as long as they are not in a **UDFB instance**).

Additionally, only a subset of variables are displayed (e.g. ActualVelocity 1 is not visible), when in Simulated mode.

For more details on:

- Axis pipe block positions, [About Associated Data on Positions](#)
  - PLCopen Axis positions, [Axis Positions Data](#)
3. (Optional) Set the **Derivation Order**.
  4. (Optional) Set the **Derivation Period**. The value entered should be either 0.0 (no modulo) or the Modulo Period, e.g. 360.0.

If the selected Derivation Order is greater than zero, the Derivation Period of the selected signal can be used to remove rollover spikes in the derivative value if the variable is of a periodic nature as the result of "modulo" behavior.

You can also disconnect a probe as follows:

*Unmap a variable(s) from the Scope*

In order to unmap a variable, or unplug a probe:

- Right-click on the channel you wish to remove from the channels list and select the **Unmap** command in the menu to disconnect the probes.
- Select the **Unmap all Oscilloscope Channels** command from the right-mouse menu in the Dictionary or the PipeNetwork.

#### 11.3.6.2 Plugging a probe from the Dictionary

A probe (variable) can be directly plugged or mapped to the Oscilloscope from the Dictionary:

---

<sup>1</sup>The measured value is the instant velocity of the axis in RPM\*1000. Note that you can see some oscillations because it is an instant velocity, not an average velocity.

1. In the **Dictionary**, right-click on the variable
2. In the menu, select the **Map to Oscilloscope channel ...** command

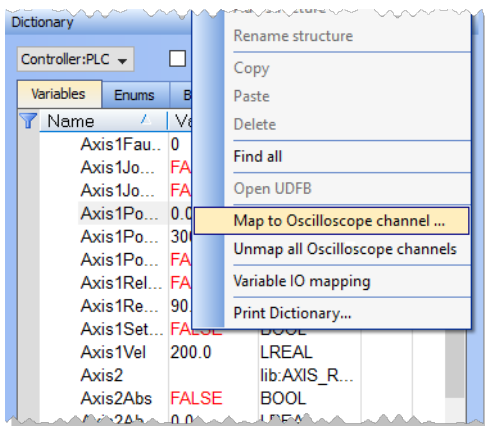


Figure 6-34: Plugging a Probe from the Dictionary

**NOTE**  
 This command is enabled if the **type** of variable is eligible for the softscope (i.e. BOOL, INT, SINT, DINT, LINT, UINT, USINT, UDINT, ULINT, BYTE, WORD, DWORD, LWORD, TIME and LREAL, as long as they are not in a **UDFB instance**).

3. Define the probe parameters.  
 There is a slight difference depending upon whether you selected a PipeNetwork probe, or a PLC probe. With PipeNetwork variables you will need to select which Data point is being used, while PLC variables may require you to navigate within the group to find the correct item.

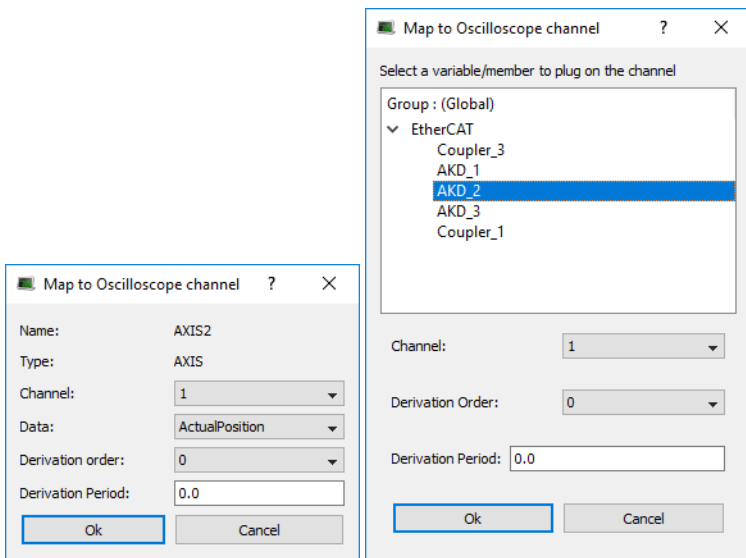


Figure 6-35: Associating a PipeNetwork (left) and PLCopen (right) variable to a Channel

Field	Description
<b>Name</b>	Variable's name
<b>Type</b>	Variable's type
<b>Channel</b>	Channel's number where the variable has to be plugged
<b>Data</b>	Desired variable information to show (the list depends on the type of Pipe Block. )

Field	Description
<b>Derivation order</b>	Performs a derivation of the measurement of the selected variable. If this value is different from 0, the derived value of the selected order is shown on the selected channel
<b>Derivation Period</b>	Specifies the modulo period for a periodic variable to remove spikes in the display of derivative orders greater than zero. The value entered should either be 0.0 (No Modulo) or the Modulo Period (eg. 360.0).

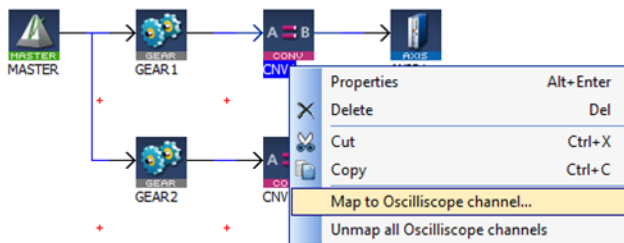
**NOTE**

In order to enable the **Plug on channel...** dialog box, the KAS IDE must be connected to the device first!

### 11.3.6.3 Plugging a probe from the Pipe Network

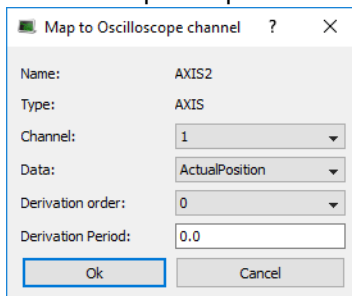
A probe (variable) can be directly plugged or mapped to the Oscilloscope from the Pipe Network:

1. Right-click on a Pipe Block
2. Select **Plug on channel...** in the menu



**Figure 6-36:** Plugging a Probe from the Pipe Network

3. Define the probe parameters.



Field	Description
<b>Name</b>	Variable's name
<b>Type</b>	Variable's type
<b>Channel</b>	Channel's number where the variable has to be plugged
<b>Data</b>	Desired variable information to show (the list depends on the type of Pipe Block. )
<b>Derivation order</b>	Performs a derivation of the measurement of the selected variable. If this value is different from 0, the derived value of the selected order is shown on the selected channel

Field	Description
<b>Derivation Period</b>	Specifies the modulo period for a periodic variable to remove spikes in the display of derivative orders greater than zero. The value entered should either be 0.0 (No Modulo) or the Modulo Period (eg. 360.0).

**NOTE**

In order to enable the **Plug on channel...** dialog box, the KAS IDE must be connected to the device first!

### 11.3.7 Setting Scale

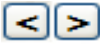
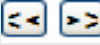
The soft oscilloscope graph is divided into 8 units for the horizontal time scale (X-axis) and 8 units for the amplitude (Y-axis). These units can be user-defined by using the configuration panels described below.

**NOTE**

**Unit per ulision:** the term refers to the time-base value for the X-axis and to the amplitude value for the Y-axis. For example, if the user sets a time-base of 10ms and an amplitude of 1, each ulision in the soft oscilloscope grid corresponds to a time of 10ms for the X-axis and an amplitude of 1 for the Y-axis.

#### Time Scale

The time scale can be configured with the **Time-base** configuration panel. The default value is 100ms/unit with the limits being 0.1ms to 25,000ms. The new value can be entered by hand directly in the text field or by using the buttons:

Buttons	Description
	Used to divide / multiply the time-base by 2 (performing a <b>division</b> corresponds to a <b>zoom in</b> while performing a <b>multiply</b> corresponds to a <b>zoom out</b> )
	Used to divide / multiply the time-base by 10

The base time unit is 1 ms.

**TIP**

You can also modify the time scale by scrolling the mouse wheel with the cursor located in the graphical area.

#### Variable Scale

Variable scaling is done by modifying the [amplitude](#) and [offset](#) value of a channel.

The variable scale can be configured in different places:

- The [Current channel](#) control panel.
- The [Edit all channels](#) dialog.

**TIP**

You can also modify the variable scale by pressing down the Ctrl key while scrolling the mouse wheel with the cursor located in the graphical area.

**NOTE**

The changes affect only the selected channel.

### 11.3.8 Trace Zoom Feature



The zoom feature is used to magnify or reduce a portion of a trace. Two zoom modes are available:

<b>Time zoom</b>	Used to expand/collapse the time-base in order to have a better view of the signal evolution through time. This zoom operation updates the time-base value.
<b>Amplitude zoom</b>	Used to have a better view of a part of a signal. This zoom operation updates the amplitude & zero offset value

The zoom operations can be done:

- By modifying the corresponding values by hand
- By using the mouse wheel

For more details on setting the amplitude, zero offset and time-base values, refer to [Setting Scale](#).

### Mouse Shortcuts

Action	Result
Scrolling up the mouse wheel	Expands the time-base value
Scrolling down the mouse wheel	Collapses the time-base value
Pressing the Ctrl key while scrolling up	Makes the amplitude value greater
Pressing the Ctrl key while scrolling down	Makes the amplitude value smaller

#### NOTE

When performing an amplitude zoom, the zero offset is automatically set by the cursor position.

### 11.3.9 How Do I Export the Oscilloscope Data?

To copy the trace data into a CSV file:

1. Display the softscope
2. Ensure the channels you want to export are [Enabled](#) and [Visible](#)
3. **Start** the data collection
4. Wait for the probe data you want to save to be collected
5. **Stop** the data collection
6. Click the **Export...** button
7. Select where you want to save the CSV file
8. Click the **Save** button

*A tip about waiting for all channels to be recording before capturing Scope data.*

Channels start recording a few cycles after the previous channel, based on resources. To see all channels at same time you need to wait until data is being sent from all channels. If you start the capture at the first cycle the exported data will not line up. The first table below shows aligned data of the channels from starting the program. Note how it takes several cycles for the data to be captured. The second table shows how his data would appear in a CSV file. Note how the time columns do not match. Waiting for all channels to be recording will eliminate this.

Time	Channel 1	Time	Channel 2	Time	Channel 3	Time	Channel 4
0	499.9996						
1	499.9996						
2	499.9996						
3	499.9996						

Time	Channel 1	Time	Channel 2	Time	Channel 3	Time	Channel 4
4	499.9996	4	-31.543				
5	499.9996	5	-31.843				
6	499.9996	6	-32.1411				
7	499.9996	7	-32.4428				
8	499.9996	8	-32.7429				
9	499.9996	9	-33.0429	9	-534.543		
10	499.9996	10	-33.343	10	-531.843		
11	499.9996	11	-33.6431	11	-535.143		
12	499.9996	12	-33.9428	12	-535.443		
13	499.9996	13	-34.2428	13	-535.742		
14	499.9996	14	-34.5429	14	-536.043		
15	499.9996	15	-34.843	15	-536.343	15	-459.028
16	499.9996	16	-35.143	16	-536.643	16	-458.728
17	499.9996	17	-35.4431	17	-536.943	17	-458.428
18	499.9996	18	-35.7428	18	-537.242	18	-458.128
19	499.9996	19	-36.0429	19	-537.542	19	-457.829

Table 1: Scope output is aligned to show actual results

Time	Channel 1	Time	Channel 2	Time	Channel 3	Time	Channel 4
0	499.9996	4	-31.543	9	-534.543	15	-459.028
1	499.9996	5	-31.843	10	-531.843	16	-458.728
2	499.9996	6	-32.1411	11	-535.143	17	-458.428
3	499.9996	7	-32.4428	12	-535.443	18	-458.128
4	499.9996	8	-32.7429	13	-535.742	19	-457.829
5	499.9996	9	-33.0429	14	-536.043	20	-459.028
6	499.9996	10	-33.343	15	-536.343	21	-458.728
7	499.9996	11	-33.6431	16	-536.643	22	-458.428
8	499.9996	12	-33.9428	17	-536.943	23	-458.128
9	499.9996	13	-34.2428	18	-537.242	24	-457.829
10	499.9996	14	-34.5429	19	-537.542	25	-459.028
11	499.9996	15	-34.843	20	-535.742	26	-458.728
12	499.9996	16	-35.143	21	-536.043	27	-458.428
13	499.9996	17	-35.4431	22	-536.343	28	-458.128
14	499.9996	18	-35.7428	23	-536.643	29	-457.829
15	499.9996	19	-36.0429	24	-536.943	30	-459.028
16	499.9996	20	-35.143	25	-537.242	31	-458.728
17	499.9996	21	-35.4431	26	-537.542	32	-458.428
18	499.9996	22	-35.7428	27	-536.943	33	-458.128
19	499.9996	23	-36.0429	28	-537.242	34	-457.829

Table 2: Actual output to CSV file for Scope data captured from the start

You can now import the data into Microsoft Excel.

**NOTE**

The Export operation is possible even when acquisition of samples is in progress. But in that case, the latest exported data are the data collected when you have defined the CSV file.

**NOTE**

The acquisition of samples is limited to 100 s when the cycle time is set to 1000  $\mu$ s (respectively 50 s with 500  $\mu$ s, and 25 s with 250  $\mu$ s)

*About the CSV file format*

Each channel takes 2 columns: one for the **time** and the other for the **value**. This allows exporting channels with different **time-base**.

The **List separator** and the **Decimal symbol** are hard-coded (they are not bind to the regional settings)

- List separator is comma (,)
- Decimal symbol is dot (.)

**TIP**

If your regional settings are different, then you have to specify explicitly those two characters in Microsoft Excel to correctly import the CSV file

**11.3.10 Practical Application: Using Trace Time To Measure CPU Load**

To determine the overall controller CPU usage, look at the HW Status tab on the Diagnostics page of the controller's web server. If the **CPU usage** is less than 90% then the CPU load (both Real Time and Non-Real Time) is okay. If the **CPU usage** is 90% or higher then the CPU is too heavily loaded and should be reduced by simplifying the application or reducing the CycleTime update rate.

**NOTE**

This page assumes that a single-core controller is being used. The technique to measure the CPU load with dual-cores is the same, but the load limits change. Please refer to [The Control Panel](#) for more information about the trace times.

HW Status	Errors and Alarms
Storage Space (used/total)	_ 7.0 / 36.9[MB]
Available Memory	_ 204.7[MB]
CPU usage*	<b>97%</b>
CPU Temp*	_ 51.9[C]
CPU Fan Present	_ False
<input type="button" value="Refresh"/> <input type="button" value="Reboot"/>	

The IDE Oscilloscope trace times can be used to analyze the application performance on a controller or programmable drive. This section describes some techniques you can use to interpret the trace times to examine the real-time performance.

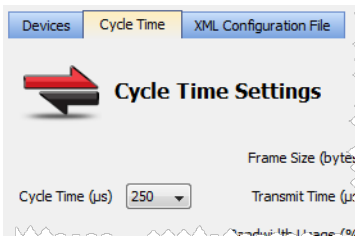
There are two major parts to consider when evaluating total performance:

**Real Time**      EtherCAT + Motion Engine + PLC program

**Non-Real Time** everything else (the background tasks)

The Oscilloscope trace times provide a very good tool to examine the Real Time response. Although it doesn't provide the complete system picture, it is a good place to start. It can provide some indication about the Non-Real Time load, but the best indicator is the overall **CPU usage** and the Controller Log messages.

First, you will want to know the Cycle Time for your system. From the **Project View**, select the **EtherCAT** view and the **EtherCAT Master Settings Tab** tab. The update period for the system in this example is set to 250 microseconds.

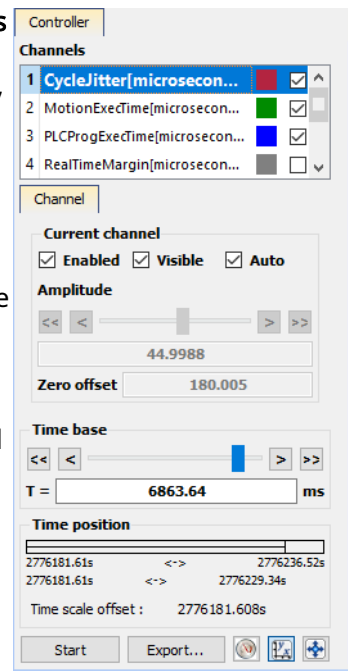


The "Trace Times" traces are enabled by pressing the **Plug Trace Times channels** button in the Oscilloscope view when your application program is running. This button automatically configures the Channels, as seen here.

### 11.3.10.1 Collect some data by pressing the "Start" button

The first thing to do is to collect data during the normal application operation, particularly once the system has reached a steady state. Press **Start** and let the data collect for a few seconds and then press the **Stop** button.

The first traces to examine are the "MotionExecTime" and "PLCProgExecTime". Configure the **Amplitude** and **Zero offset** so you can see both traces easily. Below are some recommended values based on several Cycle Time values.

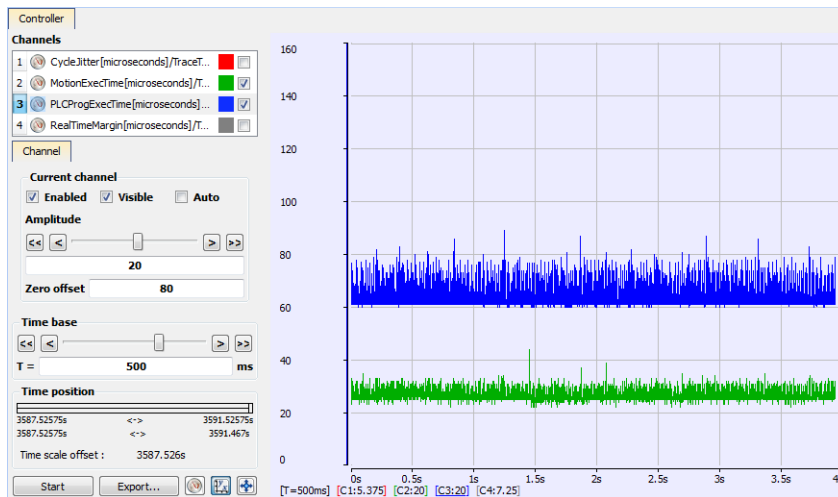


Cycle Time	Amplitude	Zero Offset
250ms	20	80
500ms	40	160
1000ms	80	320

#### **TIP**

Unchecking the "CycleJitter" and "RealTimeMargin" traces is useful so they don't clutter the view.

The following example has a Cycle Time of 250 microseconds. The "MotionExecTime" average is about 27 microseconds and the "PLCProgExecTime" average is about 68 microseconds.



The average time for the MotionExecTime + PLCProgExecTime is 95 ( $27 + 68 = 95$ ), which is about 38% of the cycle ( $95 / 250$ ). This is a good value.

#### NOTE

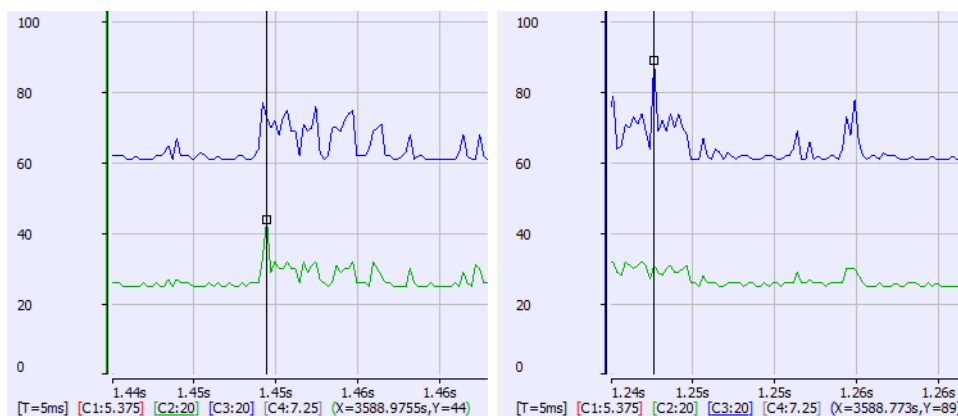
For multi-core controllers, we want to investigate the percentage of cycle time for each MotionExecTime and PLCProgExecTime. In the above example, MotionExecTime is about 11% of the cycle, and PLCProgExecTime is about 27% of the cycle. These are both good values.

#### 11.3.10.2 Check the peak times

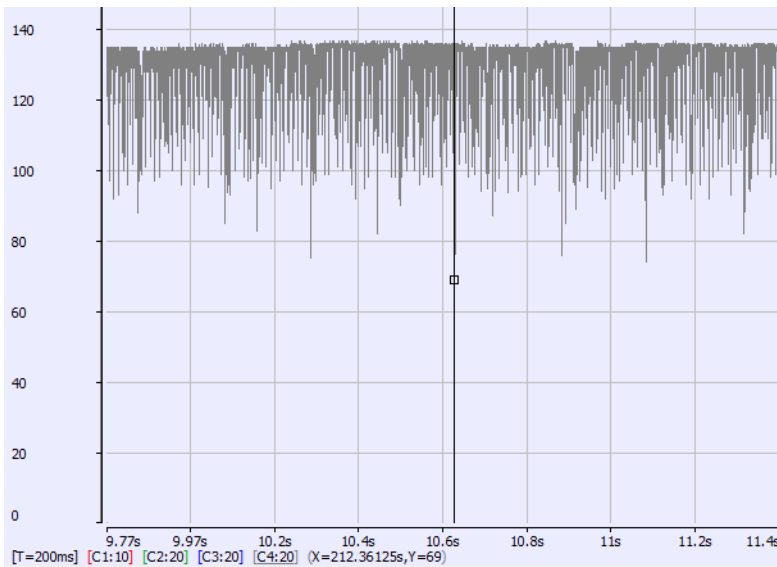
The next step is to examine the spikes. We will examine the "MotionExecTime", "PLCProgExecTime", "RealTimeMargin" and "CycleJitter" traces.

1. Reduce the **Time** base and move the traces left or right with the mouse while holding the left mouse button.
2. Position the cursor to measure the peak.

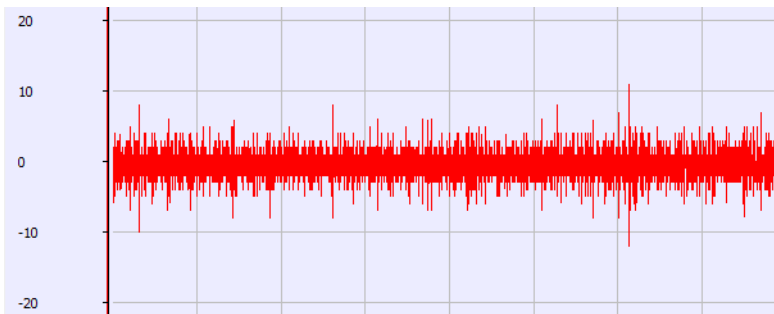
In this example the "MotionExecTime" peak is 44 and the "PLCProgExecTime" peak is 89. This is reasonable.



For the "RealTimeMargin" peaks configure the **Amplitude** and **Zero offset** so you can see the trace near zero. In this example the minimum peak (closest to zero) is 69 microseconds. This provides a 28% ( $69 / 250$ ) Real Time margin which is good.

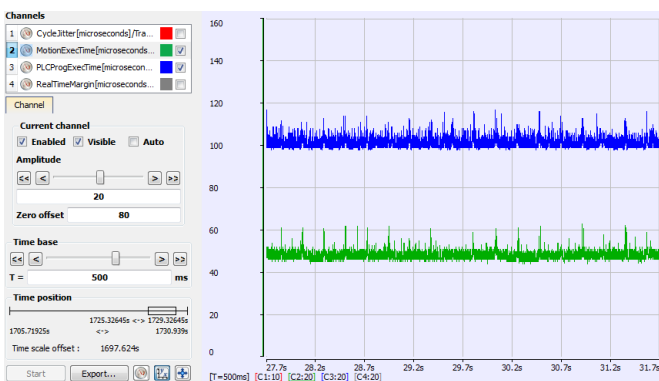


For the "CycleJitter" trace configure the **Amplitude** and **Zero offset** so you can see the trace *centered* at zero. This trace is not too interesting unless a system is misbehaving. A jitter of +/-15 microseconds is acceptable.



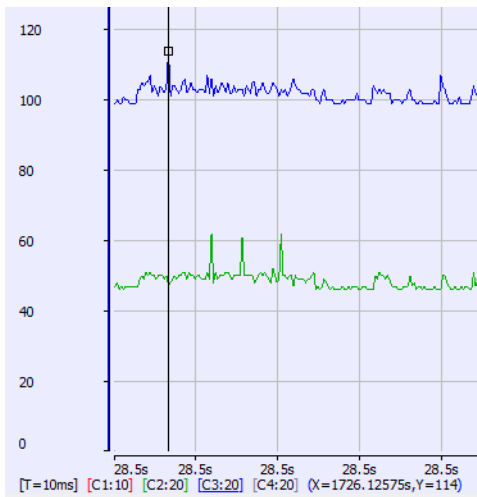
### 11.3.10.3 Heavily Loaded CPU Example

Here is an example of an application that is heavily loading a AKD PDMM or PCMM with the EtherCAT Cycle Time = 250 microseconds. Using the techniques described in [Practical Application: Using Trace Time To Measure CPU Load](#), examine the "MotionExec" and "PLCProgExec" times first:



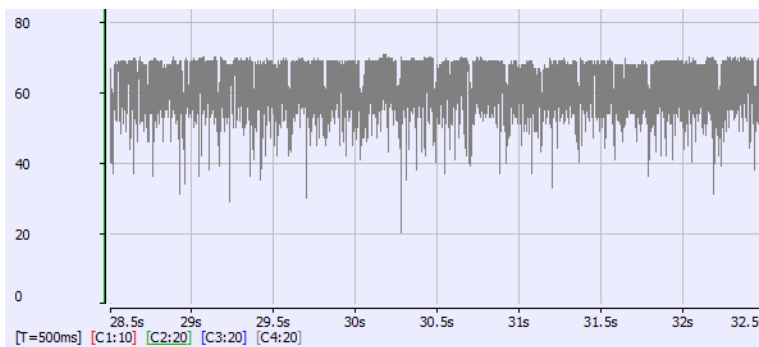
The average MotionExec + PLCProgExec = 50 + 105 = 155 microseconds. This is about 62% (155 / 250) of the cycle time.

Take a look at the peaks:



This shows the MotionExec at 62 microsec and the PLCProgExec at 114; there is not much time left over.

Check the "RealTimeMargin":



Notice the minimum time is 20 microseconds or 8% Real-Time margin (20 / 250). This is not a comfortable margin for deterministic Real-Time performance.

Checking the Controller log we see that the Virtual Machine (PLCProgExec) is missing a cycle occasionally:

Time	Level	Source	Category	Message
744	Warning	4/10/2012 10:37:21 AM (083)	Motion	WARNING The Virtual Machine missed 1 cycle(s) of PLC execution.
745	Warning	4/10/2012 10:37:22 AM (154)	Motion	WARNING The Virtual Machine missed 1 cycle(s) of PLC execution.
746	Warning	4/10/2012 10:37:22 AM (654)	Motion	WARNING The Virtual Machine missed 1 cycle(s) of PLC execution.
747	Warning	4/10/2012 10:37:23 AM (154)	Motion	WARNING The Virtual Machine missed 1 cycle(s) of PLC execution.
748	Warning	4/10/2012 10:37:23 AM (654)	Motion	WARNING The Virtual Machine missed 1 cycle(s) of PLC execution.
749	Warning	4/10/2012 10:37:24 AM (154)	Motion	WARNING The Virtual Machine missed 1 cycle(s) of PLC execution.
750	Warning	4/10/2012 10:37:24 AM (583)	Motion	WARNING The Virtual Machine missed 1 cycle(s) of PLC execution.
751	Warning	4/10/2012 10:37:25 AM (083)	Motion	WARNING The Virtual Machine missed 1 cycle(s) of PLC execution.
752	Warning	4/10/2012 10:37:25 AM (583)	Motion	WARNING The Virtual Machine missed 1 cycle(s) of PLC execution.
753	Warning	4/10/2012 10:37:26 AM (083)	Motion	WARNING The Virtual Machine missed 1 cycle(s) of PLC execution.

Lastly, take a look at the overall **CPU load**. At 88% usage there's not much CPU bandwidth available.

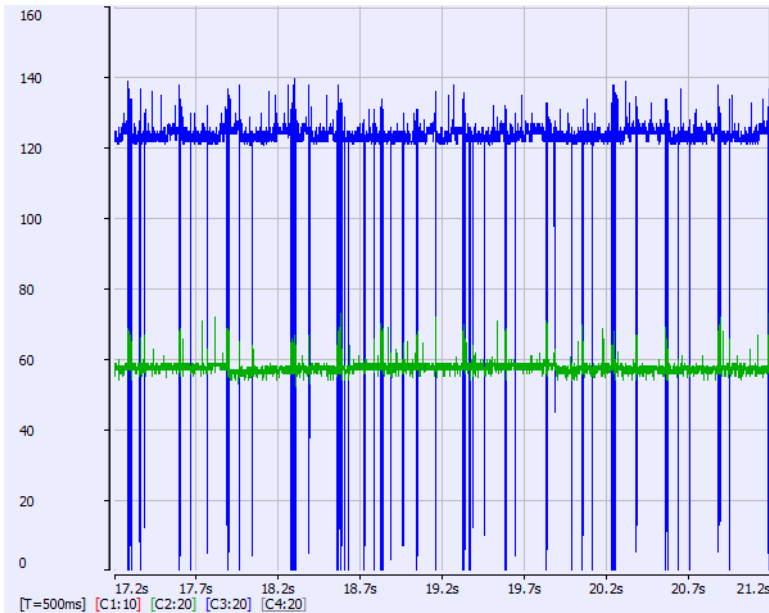
HW Status
Errors and Alarms

Storage Space (used/total)	_ 7.0 / 36.9[MB]
Available Memory	_ 203.5[MB]
CPU usage*	_ 88%
CPU Temp*	_ 50.0[C]
CPU Fan Present	_ False

Refresh
Reboot

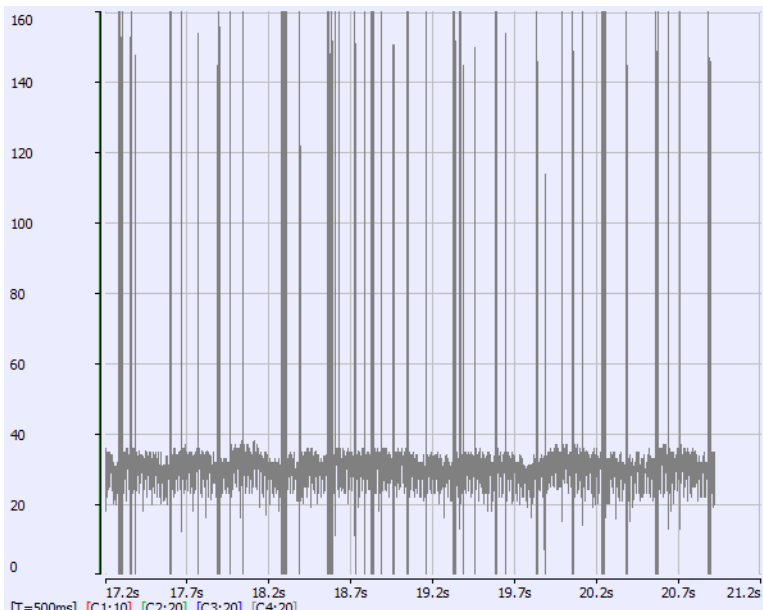
### 11.3.10.4 Over Loaded CPU Example

Now, let's take a look at an example of an application that is overloading a AKD PDMM or PCMM with the EtherCAT Cycle Time = 250 microseconds. Using the techniques described above, examine the "MotionExec" and "PLCProgExec" times first:



The average MotionExec and PLCProgExec times are  $57 + 125 = 182$  or 73% ( $182 / 250$ ) of the Cycle Time. Notice the big spikes on the PLCProgExec?

Next, look at the "RealTimeMargin":



There are many cycles with zero real-time margin. Notice the big spikes? This is a degraded case.

The Controller log confirms the missing VM cycles and an A23 alarm:

992	⚠	4/10/2012 10:52:26 AM (876)	Motion	WARNING	The Virtual Machine missed 68 cycle(s) of PLC execution.
993	⚠	4/10/2012 10:52:27 AM (376)	Motion	WARNING	The Virtual Machine missed 40 cycle(s) of PLC execution.
994	⚠	4/10/2012 10:52:27 AM (876)	Motion	WARNING	The Virtual Machine missed 104 cycle(s) of PLC execution.
995	⚠	4/10/2012 10:52:28 AM (376)	Motion	WARNING	The Virtual Machine missed 64 cycle(s) of PLC execution.
996	⚠	4/10/2012 10:52:28 AM (876)	Motion	WARNING	The Virtual Machine missed 70 cycle(s) of PLC execution.
997	⚠	4/10/2012 10:52:29 AM (376)	Motion	WARNING	The Virtual Machine missed 30 cycle(s) of PLC execution.
998	⚠	4/10/2012 10:52:29 AM (620)	Controller	WARNING	UserInfo : Alarm A23 : CPU is heavily loaded
999	⚠	4/10/2012 10:52:29 AM (876)	Motion	WARNING	The Virtual Machine missed 54 cycle(s) of PLC execution.
1000	⚠	4/10/2012 10:52:30 AM (376)	Motion	WARNING	The Virtual Machine missed 47 cycle(s) of PLC execution.

Lastly, the overall **CPU load** is 99%. Clearly this application is overloading the CPU:



HW Status	Errors and Alarms
Storage Space (used/total)	_7.0 / 36.9[MB]
Available Memory	_202.9[MB]
CPU usage*	<b>99%</b>
CPU Temp*	_49.9[C]
CPU Fan Present	_False
	<a href="#">Refresh</a> <a href="#">Reboot</a>

## 11.4 Human-Machine Interface Editor

This chapter covers the tools you can use to design your HMI panels.

- The [Kollmorgen Visualization Builder](#) to control your application
- The [internal Control Panel editor](#) to debug your application with the KAS Simulator

### 11.4.1 Using Kollmorgen Visualization Builder



To work with Kollmorgen Visualization Builder, do as follows:

- Tag the PLC variables you want to export and map with the HMI (for more details, refer to [Map Variables to HMI](#))
- Compile your project to generate the Modbus mapping file
- Create a KVB project 1 within the KAS IDE, and open it
- [Design Your HMI Panel with KVB](#)
- Save and close KVB

#### NOTE

**Important!** Be sure to use "Save" and not "Save As". The KVB is self-contained within the KAS archive and the Save As function moves the KVB out of the archive.

- Save your KAS project

#### NOTE

When you [create the KVB panel](#) with the KAS IDE, all the creation and mapping procedure is done automatically after compiling your project. So you can directly go to [Design Your HMI Panel with KVB](#).

#### ⚠ IMPORTANT

Be aware that as soon as you change the PLC variables exported for the HMI, the mapping file must be re-imported in Kollmorgen Visualization Builder to have an up-to-date version.

### Related Documents

For further information on Kollmorgen Visualization Builder, refer to the following manual:

KVB Guide	Description
<a href="#">Kollmorgen Visualization Builder™ Quick Start Guide</a>	Quick Start that covers the most important points to install and use Kollmorgen Visualization Builder, in order to configure HMI Panels and PC operated control applications.

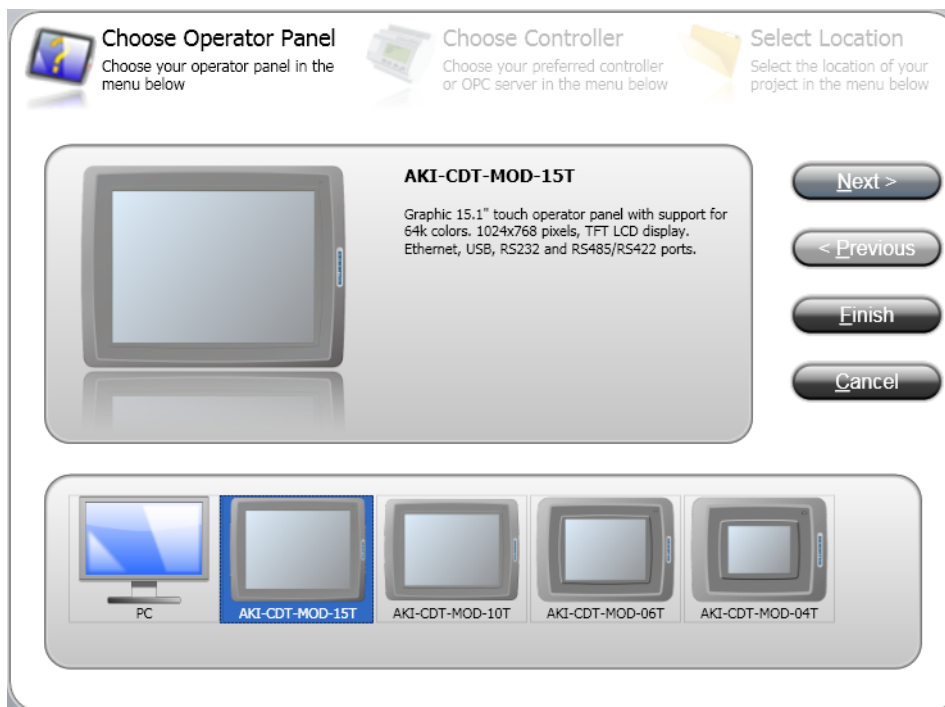
<sup>1</sup>There is no built-in feature to import/export KVB projects

KVB Guide	Description
<a href="#">Kollmorgen Visualization Builder™ User Manual</a>	Contains all the content to help you with Kollmorgen Visualization Builder.

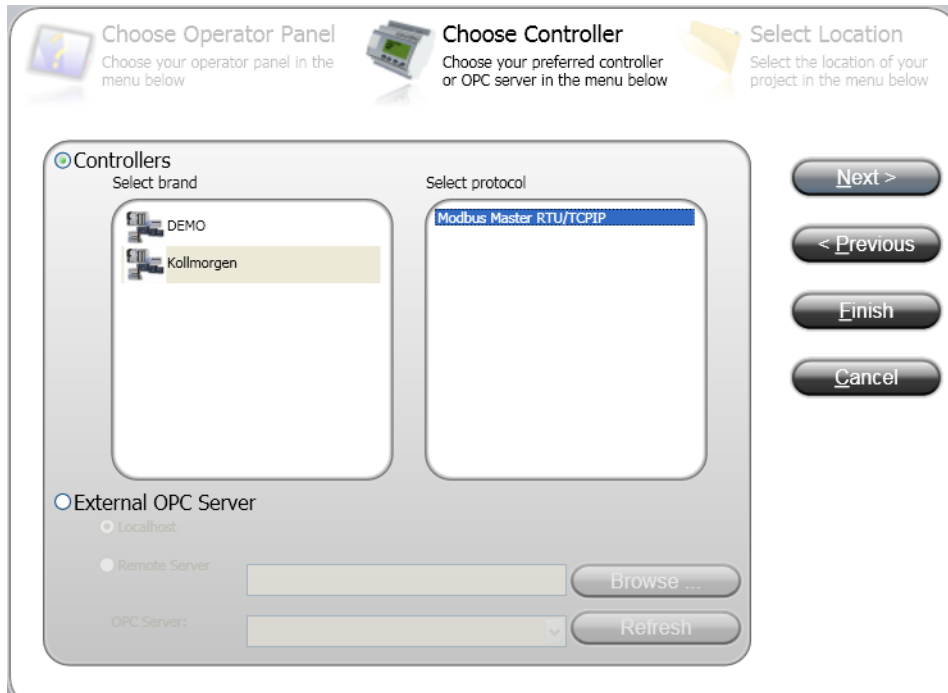
#### 11.4.1.1 Create A New KVB Controller

This procedure is applicable when you use Kollmorgen Visualization Builder externally.

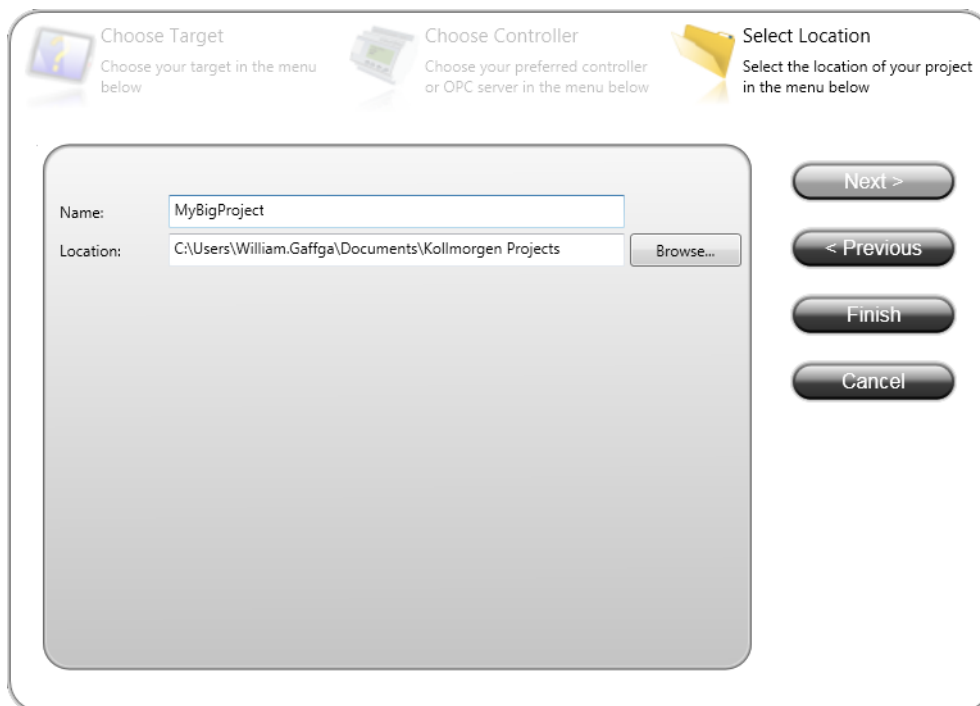
- After choosing to create a new project, select the type of operator panel to be used



- On the next dialog, select the **Kollmorgen** controller with the Modbus protocol, then click the **Next** button



- Enter the name of the project and where you want to create the project. Then click the **Finish** button



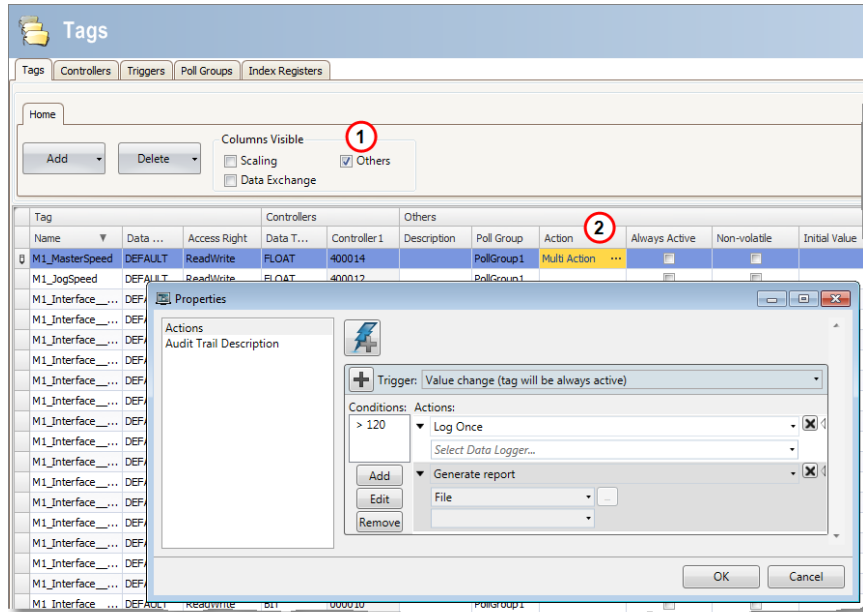
#### 11.4.1.2 Import Variables Into The Project

When you open the Kollmorgen Visualization Builder with your KVB panel (by double-clicking the KVB panel from the [project explorer](#)) all the variables tagged into the [Dictionary](#) at the time of compiling are automatically imported into Kollmorgen Visualization Builder. Once the file is imported, all PLC variables are available for use within Kollmorgen Visualization Builder.

**TIP**

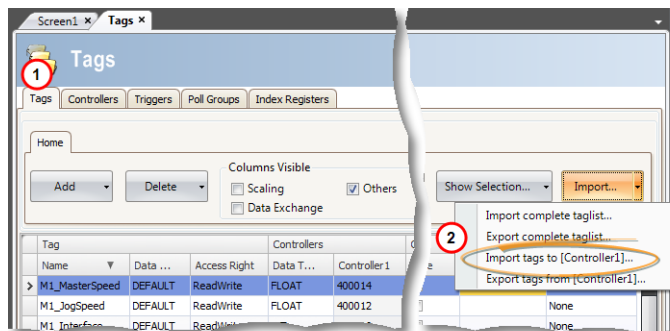
To reimport variables, (if they were changed or more were added), simply ensure they are tagged as "KVB" in the Variables list, recompile your KAS project, and open the KVB from the KAS IDE.

1. Select **Others** to display the Action column
2. You can edit the tag actions

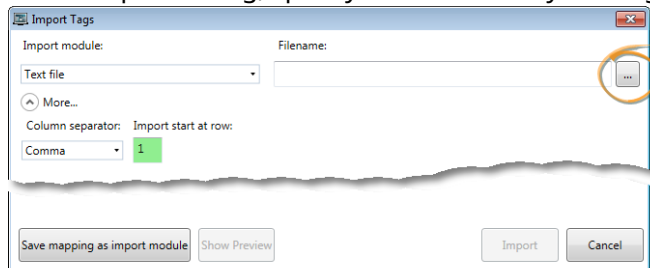


To manually export/import the variables (tags) of your project use the import procedure as follows:

- Select the **Tags** tab
- Click the arrow of the **Import** button, then select **Import tags to [Controller1]...** in the drop-down menu

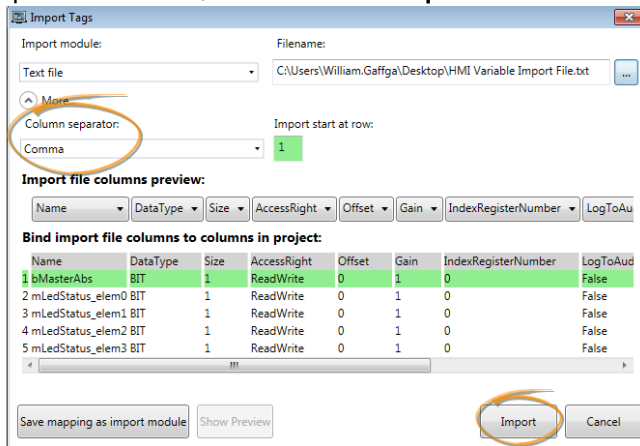


- In the import dialog, specify the filename by clicking the ... button

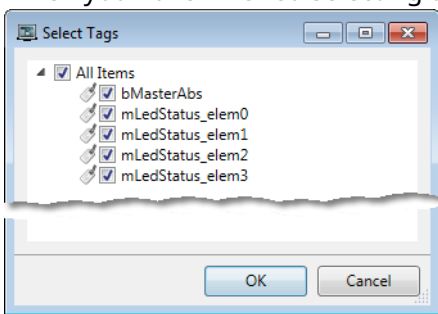


- Then use the open file dialog to find the **.txt** file
- Once the file is specified, click the **Open** button

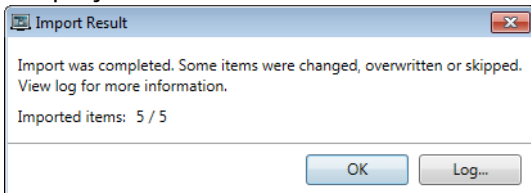
- Back in the import dialog, make sure the **Column separator** is set for Comma, and leave all options selected, then click the **Import** button



- Specify which tags (variables) you want to import. To select all tags, click the **Select All** option. When you have finished selecting the tags, click the **OK** button



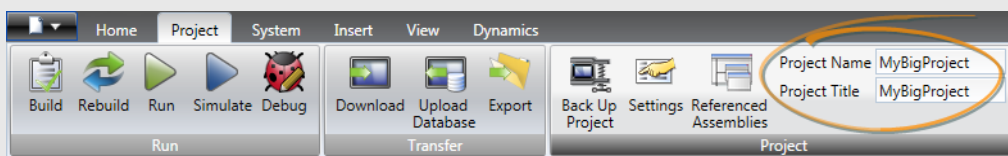
- You are notified of how many items are successfully imported. Click the **OK** button to return to the project.



### 11.4.1.3 Design Your HMI Panel with KVB

#### ! IMPORTANT

Do not modify **Project Name** and **Title** to keep consistency between Kollmorgen Visualization Builder and the KAS IDE.



#### 11.4.1.3.1 Add Object

You can drag-and-drop predefined objects from the library to the screen. The library is located in the **Home** tab of Kollmorgen Visualization Builder.

#### 11.4.1.3.2 Customize Object

Select an object and click the **General** tab to customize:

- its settings in the Settings section
- its style to a different template in the Style section

#### 11.4.1.3.3 Map Variable to the Object

In the **General** tab, you can set the Variable or Tag that maps to the current object in the Tag/Security section.

#### NOTE

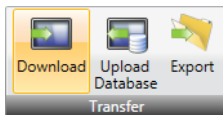
Click the **F1** key to open the Kollmorgen Visualization Builder online help (or use the Help button in the ribbon tab heading)

#### IMPORTANT

Be aware that as soon as you change the PLC variables exported for the HMI, the mapping file must be re-imported in Kollmorgen Visualization Builder to have an up-to-date version.

#### 11.4.1.4 Download the KVB Panel

To download your panel you have to use the Project ribbon in Kollmorgen Visualization Builder that contains the **Transfer** command . In order to download you must be connected directly to the HMI panel through an Ethernet cable. As the IP address is already defined (for more details, [Configure the Controller](#)), nothing special has to be done before transferring your panel to the graphic operator terminal.



Button	Description
<b>Download</b>	The Download control sends the project to an HMI-panel or a controller with an Kollmorgen Visualizer RT installation, in the network environment.
<b>Upload Database</b>	The Upload Database command will collect the database from an HMI-panel in the network environment to a file on the development PC. The panel will pause during the upload, and will be started automatically afterwards.
<b>Export</b>	The Export command builds and exports a copy of the project and all needed files to transfer the project to another location.

#### NOTE

- If you transfer your project on a USB stick, place it in the USB port of the AKI panel while it is booting up.
- For more details, refer to the online help in Kollmorgen Visualization Builder.

### 11.4.2 Design the Control Panel with the Internal Control Panel Editor

This section details the Controls and Properties used to define the Control Panel when you need to debug your application, as well as the procedure for [Mapping Variables to the Control Panel](#) controls.

#### 11.4.2.1 Create Control Panel

Control Panel are managed in the Project Explorer and can be created as follows:

1. In the Project Explorer, right-click the Controller item to open the menu
2. Select the **New Control Panel** command
3. Right-click on the newly created item and select the **Rename** command to change its name
4. Double-click the new Control Panel to open it in the graphical editor

### 11.4.2.2 Use the Control Panel control library

- Select a control in the Libraries toolbox (Controls tab) and drag-and-drop it in your Control Panel.

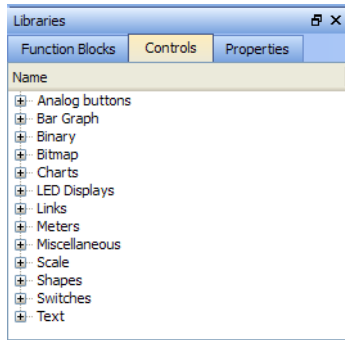


Figure 6-37: Control Panel Control Library

For an exhaustive list of controls, refer to [Graphic Objects](#).

### 11.4.2.3 Edit the Control panel

When a control is selected, you can change its properties (displayed in the Libraries toolbox) by double-clicking the **Value**.

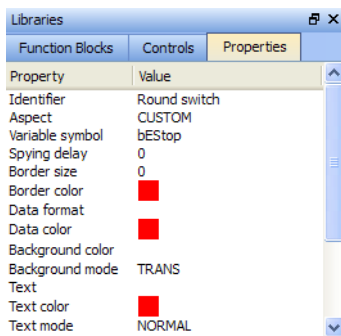


Figure 6-38: Control Panel Control Properties

For an exhaustive list of properties, refer to [Graphic Objects Properties](#)

#### NOTE

- You can perform multi-selection with the mouse (all the controls that are even partly inside the selection area are selected)
- You can add controls to your selection either with the **Ctrl** or **Shift** keys
- You can use Arrow keys to move the Control Panel page Up, Down and sideways.
- You can use Shift + Arrow keys to move the selected Control up-down and sideways

#### TIP

To duplicate all the selection, hold down **Ctrl** and click the right mouse button while performing your move operation (do not forget to release the mouse button first, before the **Ctrl** key).

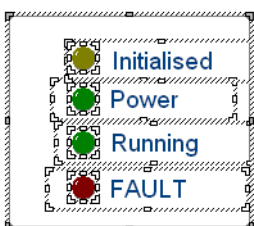
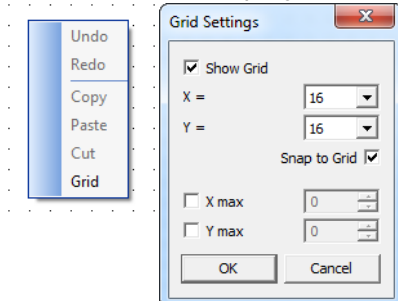




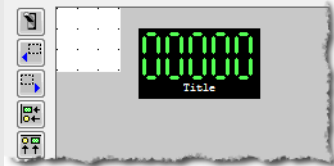
Figure 6-39: Control Panel - Selection of Controls

### 11.4.2.3.1 HMI Grid Settings

Right-clicking in the Control Panel's graphical editor provides access to the Grid Settings by selecting Grid. This may also be accessed by pressing Ctrl-G. The settings allow you to control the appearance of the grid as well as forcing objects to "snap" to the grid. Settings are per panel and are saved with the project.



Parameter	Description
Show Grid	Display or hide the grid in the Control Panel
X= / Y=	Space between each grid dot in X/Y directions
Snap to Grid	Controls will be forcibly aligned with the grid when they are placed/moved
X max / Y max	Set a maximum area in which the grid is shown. The editor is grayed out beyond these dimensions. Please note that objects may still be placed within the grayed out area as it is merely a visual aid.



### 11.4.2.4 Mapping Variables to the Control Panel

*How do I define a variable for PLC programs?*

To link your Control panel with the PLC programs, some controls contain a property called **Variable symbol**

Libraries		
Function Blocks	Controls	Properties
Property	Value	
Identifier	Round switch	
Aspect	CUSTOM	
Variable symbol	bEStop	
Spying delay	0	
Border size	0	

Figure 6-40: Map variables to a Control Panel control

To map the variable:

1. Select the variable in the Dictionary toolbox

2. Move it to the control to be linked in the Control panel editor using drag-and-drop

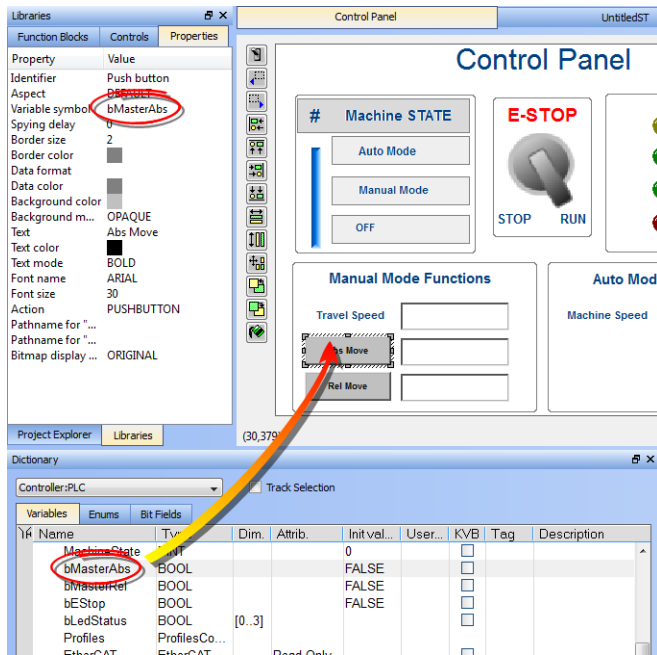


Figure 6-41: Map Variables to a Control Panel Control in the Graphical Editor

The Variable symbol is automatically updated in the **Properties** tab.

**NOTE**

A warning will be generated when the program is compiled if the control is mapped to a variable which is not in the Dictionary. Double-clicking the warning will highlight the control object and open the editor so the variable can be defined.

**11.4.2.5 Graphic Objects**

Below are available basic objects you can insert in your graphics:

**11.4.2.5.1 Basic Shapes**



A collection of basic drawings is available. Each object can be either static, or linked to a variable used to enable its visibility (show/hide).

**Properties**

- Aspect (shapes)
- Background color
- Background mode
- Border color
- Border size
- Color when not connected
- Data format
- Direction (basic shapes)
- FALSE color
- Font name
- Font size
- Identifier

- Spying delay
- Text
- Text color
- Text mode
- TRUE color
- Variable symbol

#### 11.4.2.5.2 Bitmaps

Bitmap file (BMP, GIF, JPG) can be inserted in the graphic area.

**Properties:**

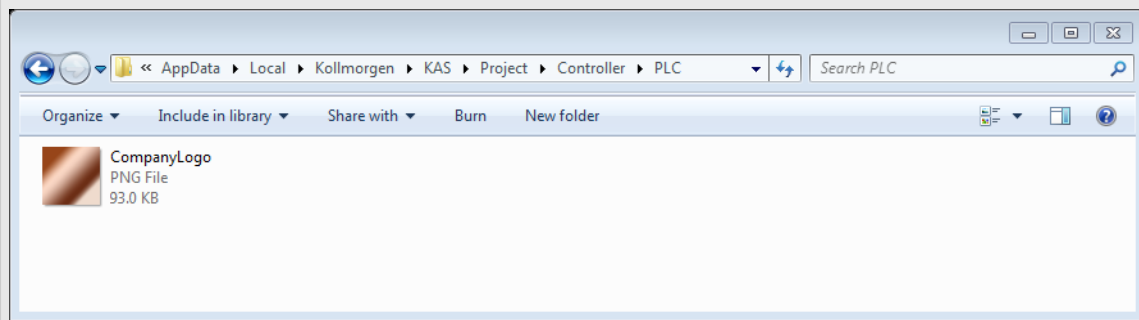
- Background color
- Background mode
- Bitmap display mode
- Bitmap pathname
- Border color
- Border size
- Border style
- Font name
- Font size
- Identifier
- Text
- Text color
- Text mode

#### NOTE

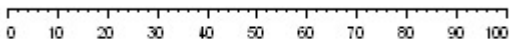
Large bitmaps are time-consuming during animation and can lead to poor performance, mainly if they have the "STRETCH" display mode or the "TRANS" (transparent) background mode.

#### TIP

Graphics need to be made portable with a project, otherwise the KAS IDE will attempt to point to the original location of the graphic files. To make graphics portable, copy them to `c:\Users\[user folder]\AppData\Local\Kollmorgen\Project\Controller\PLC`. After this the graphics are a part of the .KAS file.



#### 11.4.2.5.3 Scales



Scales are static drawings representing an X or Y axis, generally used to document other objects such as trend charts or bargraphs.

**Properties:**

- Background color
- Background mode

- Border color
- Border size
- Border style
- Color when not connected
- Direction (scale)
- Font name
- Font size
- Identifier
- Maximum value
- Minimum value
- Nb divisions (main)
- Nb divisions (small)
- Placement (scale)
- Scale color
- Text
- Text color
- Text mode

#### 11.4.2.5.4 Text boxes



Static, animated or edit text boxes are available for displaying / forcing variables. For edit boxes at runtime, double-click on the object to enter the value and then hit ENTER to validate the input.

#### Properties:

- Action (text)
- Background color
- Background mode
- Border color
- Border size
- Border style
- Data format
- Font name
- Font size
- Identifier
- Spying delay
- Text
- Text color
- Text mode
- Variable symbol

#### 11.4.2.5.5 Switches and 2-state displays



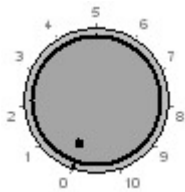
Buttons, switches and 2-state displays are used for control or display of a Boolean variable.

#### Properties:

- Action (switch)
- Aspect (switches)
- Background color
- Background mode
- Bitmap display mode
- Bitmap for "FALSE" state

- Bitmap for "TRUE" state
- Border color
- Border size
- Data color
- Data format
- Font name
- Font size
- Identifier
- Spying delay
- Text
- Text color
- Text mode
- Variable symbol

#### 11.4.2.5.6 Analog buttons



Analog buttons are used for setting the value of an integer or real variable. The mouse is used for setting the value.

#### Properties:

- Background color
- Background mode
- Border color
- Border size
- Border style
- Data color
- Data format
- Font name
- Font size
- Identifier
- Maximum value
- Minimum value
- Scale color
- Spying delay
- Text
- Text color
- Text mode
- Variable symbol

#### 11.4.2.5.7 Bar Graphs



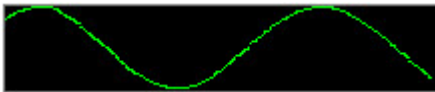
Bargraphs are rectangles filled according to the value of an analog variable. Bargraphs can be horizontal or vertical.

#### Properties:

- Background color
- Background mode
- Border color
- Border size

- Border style
- Data color
- Data format
- Direction (bar graph)
- Font name
- Font size
- Identifier
- Maximum value
- Minimum value
- Spying delay
- Text
- Text color
- Text mode
- Variable symbol

#### 11.4.2.5.8 Charts

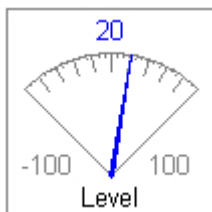


Charts enable the tracing of a variable as with an oscilloscope.

#### Properties:

- Aspect (trend charts)
- Background color
- Background mode
- Border color
- Border size
- Border style
- Data color
- Data format
- Font name
- Font size
- Identifier
- Maximum value
- Minimum value
- Nb of points (trends)
- Spying delay
- Text
- Text color
- Text mode
- Variable symbol

#### 11.4.2.5.9 Analog meters



Analog meters provide a graphical display of an analog value.

#### Properties:

- Background color
- Background mode

- Border color
- Border size
- Border style
- Data color
- Data format
- Font name
- Font size
- Identifier
- Link
- Maximum value
- Minimum value
- Nb divisions (main)
- Nb divisions (small)
- Scale color
- Spying delay
- Text
- Text color
- Text mode
- Variable symbol

## Sliders



Sliders are used for entering an analog value with a horizontal or vertical mouse driven cursor.

### Properties:

- Background color
- Background mode
- Border color
- Border size
- Border style
- Data color
- Data format
- Direction (slider)
- Font name
- Font size
- Identifier
- Maximum value
- Minimum value
- Scale color
- Spying delay
- Text
- Text color
- Text mode
- Variable symbol

### 11.4.2.5.10 Digital meters



Digital meters (digits) display the value of a variable with the same aspect as a digital clock.

### Properties:

- Aspect (digits)
- Background color
- Background mode
- Border color
- Border size
- Border style
- Data color
- Data format
- Font name
- Font size
- Identifier
- Maximum value
- Minimum value
- Spying delay
- Text
- Text color
- Text mode
- Variable symbol

#### 11.4.2.5.11 Links

### [Back to main page](#)

Links are mouse-driven hyperlinks that are used as shortcuts to open another graphic document. Using links enables the design of multi-page animated applications.

#### Properties:

- Background color
- Background mode
- Border color
- Border size
- Border style
- Font name
- Font size
- Identifier
- Link
- Text
- Text color
- Text mode

#### 11.4.2.5.12 Connection status

Connection status is a box actuated with the current status of the connection and the connected run-time application. It is mainly dedicated to diagnostic.

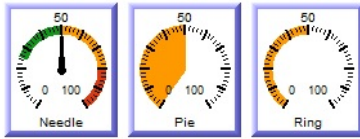
#### Properties:

- Background color
- Background mode
- Border color
- Border size
- Border style
- Data color
- Data format
- Font name
- Font size
- Identifier
- Spying delay
- Text



- Text color
- Text mode

#### 11.4.2.5.13 Gauges



Analog view meter.

#### Properties:

- Background color
- Background mode
- Border color
- Border size
- Border style
- Data color
- Data format
- Font name
- Font size
- Identifier
- Maximum value
- Minimum value
- Nb divisions (main)
- Nb divisions (small)
- Needle color
- Spying delay
- Title
- Title color
- Title placement
- Variable symbol

#### Other Properties:

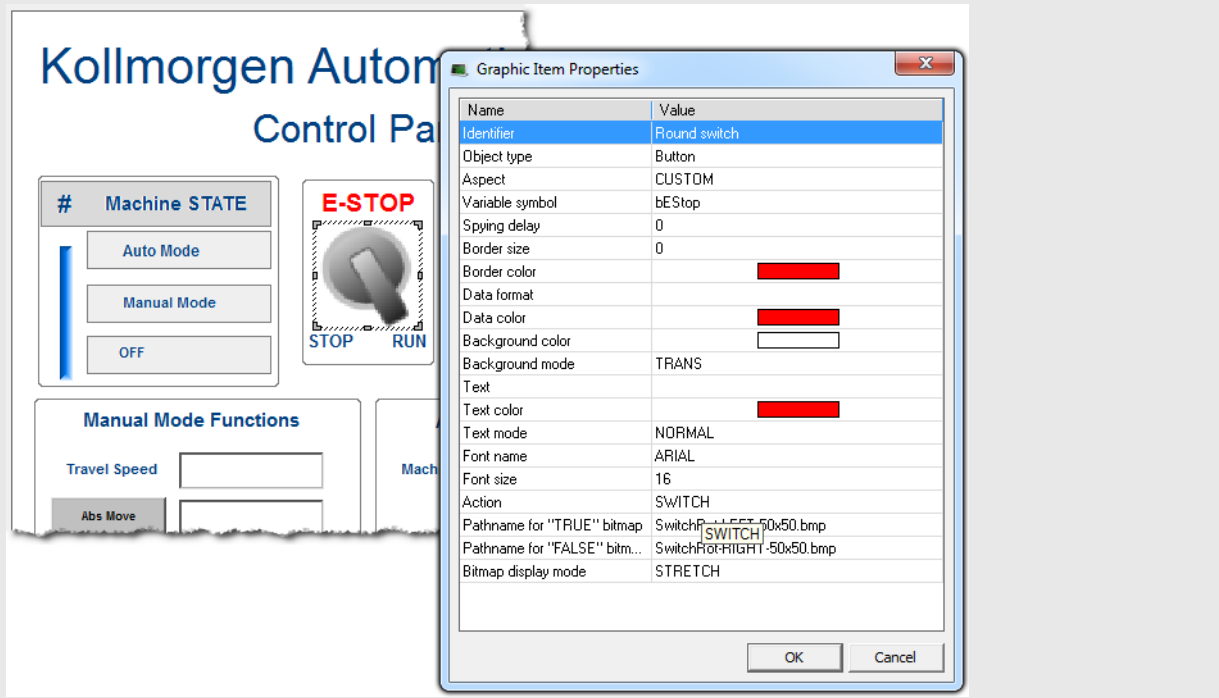
- Needle aspect: defines how the dynamic area must be shown (see bitmaps)
- Background bitmap: optional bitmap to be displayed as background
- Needle X/Y position: start position from the center of the image (percents)
- Needle size: needle with (percents)
- Needle angle: angles for min/max values in degrees
- Green color: color for the "low" range
- Green to: maximum "low" value
- Yellow color: color for the "medium" range
- Yellow to: maximum "medium value
- Red color: color for the "high" range
- Red to: maximum "high" value
- Value placement: where the text value should be drawn
- Set Gain/offset

#### 11.4.2.6 Graphic Objects Properties

This page details all possible properties for graphic objects. Refer to the [list of available objects](#) for further information on which property is used for which object.

**TIP**

Double-clicking an object will open the Graphic Item Properties dialog box, which lists the properties particular to that object.



Name	Description of Value
<b>Action (switch)</b>	Indicates the possible mouse action for switches. The following values are possible: STATIC = no mouse action PUSHBUTTON = the variable is forced to TRUE when pressed and to FALSE when released SWITCH = the status of the variable is inverted when the button is pressed ONESHOTBUTTON = same as switch, but the display continues to appear released
<b>Action (text)</b>	Indicates the possible mouse actions for text boxes. The following values are possible: STATIC = no mouse action EDIT = double-click opens an edit box for entering the variable value
<b>Aspect (digits)</b>	This property indicates the type of drawing for a digital meter. Possible aspects are: DEFAULT = plain drawing BEZEL = all segments have a 3D effect
<b>Aspect (shapes)</b>	This property indicates the type of basic shape to be drawn. Possible aspects are: CYLINDER = a 3D like cylinder ELLIPSE = an ellipse HALFELLIPSE = one half of an ellipse GATE = a simple vector drawing for a valve RECTANGLE = a rectangle ROUNDRECT = a rectangle with rounded corners TRIANGLE = a triangle
<b>Aspect (switches)</b>	This property indicates the type of switch to be drawn. Possible aspects are: DEFAULT = a standard Windows-like push button CUSTOM = a button with TRUE and FALSE drawings defined with bitmaps

Name	Description of Value
<b>Aspect (trend charts)</b>	This property indicates the type of drawing for a trend chart. Possible aspects are: POINT = only relevant dots are drawn LINE = lines are drawn from point to point HISTO = histogram style
<b>Background color</b>	This property indicates the color used for filling the background of the object. In case of a bitmap, it specifies the color that must not be drawn if the TRANS (transparent) background mode is specified.
<b>Background mode</b>	This property indicates whether the background of the object must be filled or not. If this property is OPAQUE, then the background is filled with the specified background color. If this property is TRANS (transparent) then the background is not filled. Transparent drawing mode can be useful in the case of overlapping objects.  <b>ⓘ IMPORTANT</b> Specifying the TRANS (transparent) mode for large bitmaps is time-consuming and will affect the real-time performances of graphic updates.
<b>Bitmap display mode</b>	For bitmap-based objects, this property indicates whether the attached bitmap must keep its original aspect or be stretched to the actual size of the object. Possible values are: ORIGINAL = keep the original aspect of the bitmap (cut if too large) STRETCH = stretch or shrink the bitmap for fitting the actual size of the graphic object  <b>ⓘ IMPORTANT</b> Large bitmaps with "STRETCH" display mode are time-consuming during animation and can lead to poor performance.
<b>Bitmap for "FALSE" state</b>	For two-state objects having the "CUSTOM" aspect, this property specifies the pathname of the bitmap to be displayed when the value of the attached variable is FALSE (or zero for analogs). BMP, GIF and JPG formats are supported. If no directory is specified, the specified file name is searched: <ul style="list-style-type: none"> <li>• in the project folder</li> <li>• in the "\BITMAP" folder of the KAS IDE</li> </ul>
<b>Bitmap for "TRUE" state</b>	For two-state objects having the "CUSTOM" aspect, this property specifies the pathname of the bitmap to be displayed when the value of the attached variable is TRUE (or not zero for analogs). BMP, GIF and JPG formats are supported. If no directory is specified, the specified file name is searched: <ul style="list-style-type: none"> <li>• in the project folder</li> <li>• in the "\BITMAP" folder of the KAS IDE</li> </ul>
<b>Bitmap pathname</b>	For bitmaps, this property specifies the pathname of the bitmap to be displayed. BMP, GIF and JPG formats are supported. If no directory is specified, the specified file name is searched: <ul style="list-style-type: none"> <li>• in the project folder</li> <li>• in the "\BITMAP" folder of the KAS IDE</li> </ul>
<b>Border color</b>	This property indicates the color of the border drawn around the object.
<b>Border size</b>	This property indicates the width of the border drawn around the object, expressed as a number of pixels. If this property is 0, then no border is drawn.

Name	Description of Value												
<b>Border style</b>	This property indicates the possible 3D effect used for drawing the border around the object. Possible values are: FLAT = no 3D effect 3DUP = depressed 3D effect 3DDOWN = pressed 3D effect 3D = default 3D effect												
<b>Color when not connected</b>	For shapes, this property indicates the color used for filling shapes when no variable is attached to the graphic object.												
<b>Data color</b>	This property indicates the color used to represent the value of a connected variable within the object (for example the filled part of a bar graph).												
<b>Data format</b>	<p>If defined, this property indicates that the value of the connected variable must be displayed on the graphic object. You must specify for this property a format string that indicates how the data will be formatted.</p> <p><b>⚠ IMPORTANT</b></p> <p>The "<b>text</b>" property is ignored when a data format is specified.</p> <p>Format string has the same format as the famous "printf" function of "C" language. It can include static characters together with one of the following possible pragmas that specify the value:</p> <ul style="list-style-type: none"> <li>%s = default formatting according to IEC syntax</li> <li>%d = integer (decimal)</li> <li>%X = hexadecimal</li> <li>%g = floating point</li> <li>%.nf = decimal real (<i>n</i> is the number of displayed decimal digits)</li> </ul> <p>Below are some examples:</p> <table border="1"> <thead> <tr> <th>Format</th> <th>Value</th> <th>Displayed string</th> </tr> </thead> <tbody> <tr> <td>%d</td> <td>12.3</td> <td>12</td> </tr> <tr> <td>Var = %g meters</td> <td>1.2</td> <td>Var = 1.2 meters</td> </tr> <tr> <td>%.2f</td> <td>1.12345</td> <td>1.12</td> </tr> </tbody> </table> <p><b>NOTE</b></p> <p>Only one % pragma can be used in a string.</p>	Format	Value	Displayed string	%d	12.3	12	Var = %g meters	1.2	Var = 1.2 meters	%.2f	1.12345	1.12
Format	Value	Displayed string											
%d	12.3	12											
Var = %g meters	1.2	Var = 1.2 meters											
%.2f	1.12345	1.12											
<b>Direction (bar graph)</b>	For bar graphs, this property indicates the growing direction: to the left, to the right, to the top or to the bottom.												
<b>Direction (basic shapes)</b>	For oriented shapes such as triangles, half ellipses or cylinders, this property indicates the direction of the drawing; to the left, to the right, to the top or to the bottom.												
<b>Direction (scale)</b>	For scales, this property indicates the direction of the axis. If LEFT, the minimum value is on the left side. If RIGHT, the minimum value is on the right side.												
<b>Direction (slider)</b>	For slider, this property indicates whether the slider is horizontal (RIGHT) or vertical (TOP).												
<b>FALSE color</b>	For shapes, this property indicates the color used for filling shapes when the attached variable has the FALSE state, or zero for analogs.												
<b>Font name</b>	This property indicates the name of the character font used for drawing texts in the graphic object.												

Name	Description of Value
<b>Font size</b>	This property indicates the size of the character font used for drawing texts in the graphic object. The size is expressed as a percentage of the actual height of the object. Maximum possible value is 100. This ensures that the ratio is kept when the object is resized.
<b>Identifier</b>	You can freely attach a text identifier to each graphic object inserted in a document. Identifiers are useful for arranging overlapped objects as they appear in the "Z-order" list.
<b>Link</b>	This property indicates the name of the target .GRA animated document for shortcuts. If no directory is specified in the link, then the file is searched in the project folder.
<b>Maximum value</b>	For analog animated objects (meters, bar graphs or trends) this property indicates the maximum possible value that can be displayed. For static scales, it indicates the value of the highest mark.
<b>Minimum value</b>	For analog animated objects (meters, bar graphs or trends) this property indicates the minimum possible value that can be displayed. For static scales, it indicates the value of the lowest mark.
<b>Nb divisions (main)</b>	For objects including a graphic scale, this property indicates the number of main division marks to be drawn in the scale.
<b>Nb divisions (small)</b>	For objects including a graphic scale, this property indicates the number of small division marks to be drawn in the scale, between each main division mark.
<b>Nb of points (trends)</b>	For trend charts, this property indicates the maximum number of stored points. If the width of the object (in pixels) is less than this number, then oldest points are not visible.
<b>Needle color</b>	For gauges, this is the color of the needle.
<b>Placement (scale)</b>	For scales, this property indicates the location of the scale within the object rectangle: on the left, on the right, on the top or at the bottom.
<b>Scale color</b>	For objects including a graphic scale, this property indicates the color used for drawing the axis, the division marks and corresponding values of the scale.
<b>Spying delay</b>	It is the minimum period for actuating the value of the connected variable, expressed as a number of milliseconds. If the delay is not specified or equal to 0, refresh is done as fast as possible.
<b>Text</b>	If defined, this property indicates the text to be displayed on the graphic object. <b>ⓘ IMPORTANT</b> This property is ignored when a <b>data format</b> is specified.
<b>Text color</b>	This property indicates the color used for inserting texts in the graphic object.
<b>Text mode</b>	This property indicates the font effect used for drawing texts in the graphic object. Possible values are: HIDE = text is not displayed NORMAL = normal font BOLD = bold text ITALIC = italic text UNDERLINE = underlined text
<b>Title</b>	For gauges, this is the text shown in the tile
<b>Title color</b>	For gauges, this is the color of the title text.
<b>Title placement</b>	For gauges, this is the position of the title within the gauge.

Name	Description of Value
<b>TRUE color</b>	For shapes, this property indicates the color used for filling shapes when the attached variable has the TRUE state, or non zero for analogs.
<b>Variable symbol</b>	It is the full name of the application variable connected to the graphic object. In case of a local variable, its symbol must be prefixed with the parent program name, separated with "/". Example: "MyProg/MyVar".

### 11.4.2.7 Operate the Control Panel

The Example program has a default control panel built-in to make it easy to start an application. Perform the following steps to operate the control panel:

1. Double-click on **Control Panel** in the Project Explorer to open the form

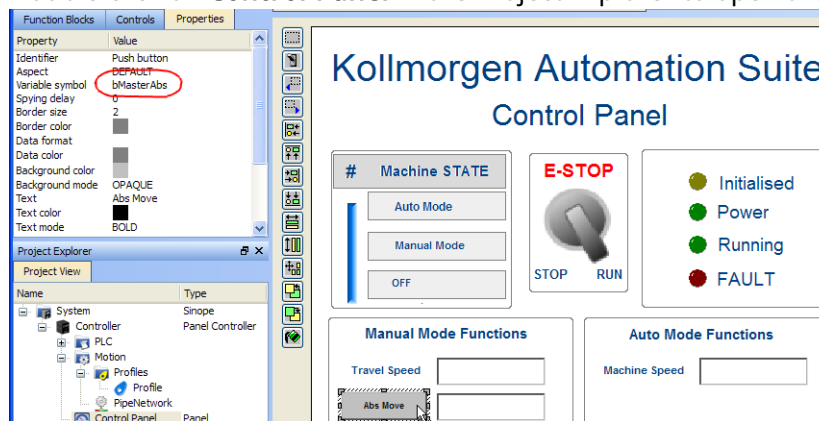
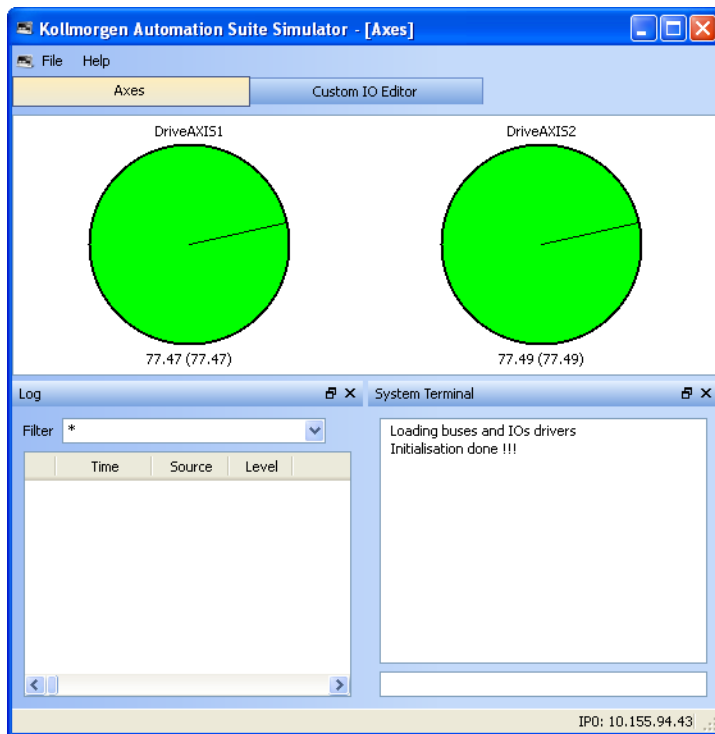


Figure 6-42: Control Panel

2. Start by moving the vertical slider bar to select the Machine STATE as **Manual Mode**
3. In the Manual Mode Functions area, double-click the text box for the Travel Speed
4. Enter the numeric value for the Travel Speed and press Enter

#### 11.4.2.7.1 About KAS Simulator Display

The KAS Simulator displays the status and position of the axes. It also displays the log messages.





**Figure 6-43:** Display of KAS Simulator

You can continue to use the Control Panel to:

- Experiment with the controls and observe the simulated output
- Perform an absolute move by entering a position in the text box
- Perform a relative move

#### 11.4.2.8 Exiting Simulation Mode

To exit Simulation mode, do as follows:

1. Click the Stop Device button 
2. Click the Disconnect Device button 

#### NOTE

For additional information about Kollmorgen Automation Suite, see the following documentation:

- Getting Started
- User Manual
- Technical Reference - PLC Library
- Technical Reference - Motion Library
- Online Help

## 12 Advanced Topics

### 12.1 Coordinated Motion

Coordinated motion in KAS IDE is discussed in several locations and manners.

- [Overview](#) — this section helps you to understand the concepts behind Coordinated Motion and the terminology associated with Coordinated Motion.
- [Design Coordinated Motion](#) — this section helps you to get started quickly with coordinated motion by walking you through the steps of setting up a project that uses coordinated motion.
- [Functions and Function Blocks](#) — this is the reference section for function block parameters.

#### NOTE

The coordinated motion library supports coordinated motion for up to 128 axes. Hardware limitations may impose a lower limit for most applications.

#### 12.1.1 Overview

This section provides an overview of Coordinated Motion, including general concepts you will need to understand to use Coordinated Motion.

- [Coordinated Motion Terminology](#)
- [Group State Diagrams](#)
- [Coordinate Systems](#)

[Create a Linear or Circular Coordinated Motion Application.](#)

##### 12.1.1.1 Coordinated Motion Terminology

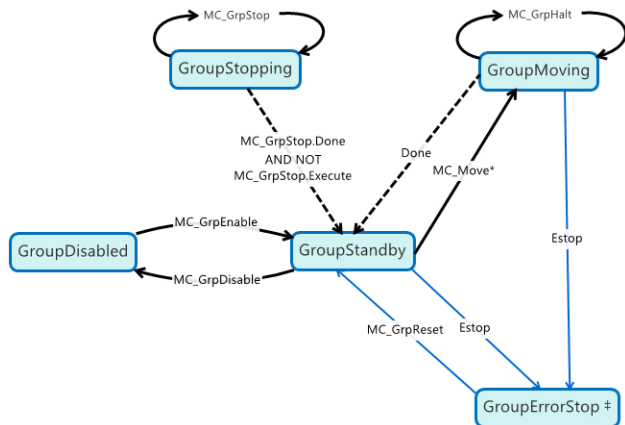
Term	Definition
<b>ACS</b>	Axes Coordinate System. The system of coordinates related to the physical motors and the single movements caused by the single drives.
<b>Blending</b>	A way that consecutive function blocks cooperate in the transition from the first to the next.
<b>Contour Curve</b>	Inserted curve that modifies the original path. It is the resulting curve after blending
<b>Coordinate system</b>	The reference system in which a coordinate or path is described
<b>Corner deviation</b>	The shortest distance between the programmed corner point and the contour curve
<b>Corner distance</b>	Distance of the start point of the contour curve to the programmed target point.
<b>Direction</b>	The orientation components of a vector in space. (Note: this is different from the MC_Direction input as used in part 1).
<b>Drive</b>	A unit controlling a motor via the current and timing in its coils
<b>Group-FB</b>	The set of function blocks that can work on a group of axes



Term	Definition
<b>MCS</b>	<p>Machine Coordinate System. The system of coordinates that is related to the machine. A Cartesian coordinate system with the origin in a fixed position relative to the machine (the origin is defined during the machine setup).</p> <p>Sometimes called "World Coordinate System" or "Base Coordinate System". (Note: with Cartesian build machines, MCS is a Cartesian Coordinate system and may be identical to ACS, or mapped via a trivial transformation). The coordinate system from the physical multiple axes ACS is linked to the MCS via a kinematic transformation (forward and backward conversion). The MCS represents an imaginable space with up to 6 dimensions.</p>
<b>Motor</b>	An actuator focused to a movement, converting electrical energy in a force or torque.
<b>Orientation</b>	The rotational components of a vector in space.
<b>Path</b>	Set of continuous positions and orientation information in multi-dimensional space Geometrical description of a space curve that the TCP of an axesgroup moves along.
<b>PathData</b>	Description of a path which can include additional information like velocity and acceleration.
<b>PCS</b>	<p>Product Coordinate System or Program Coordinate System. The PCS is based on the MCS typically by shifting and maybe rotating the MCS. The Zero point of the PCS is related to the product and can be changed during runtime by the program. The real work piece can have a rotation or shift to the MCS coordinate system or even might be moving relative to the MCS coordinate system. By specifying a trajectory in PCS one is able to describe the trajectory independent from the machine situation. To map these two worlds (MCS to PCS and vice versa), a Cartesian or cylindrical transformation is normally done.</p>
<b>Position</b>	<p>Position means a point in space which is described by different coordinates. Depending on the used system and transformation it can consist of up to 6 dimensions (coordinates) meaning 3 Cartesian coordinates in space and 3 coordinates for the orientation.</p> <p>In ACS there can be even more than 6 coordinates.</p> <p>If the same position is described in different coordinate systems the values of the coordinates are different.</p>
<b>Scara</b>	A special kinematic for robot or handling applications.
<b>Speed</b>	Speed is the absolute value of the velocity without direction.
<b>Synchronization</b>	Combines an axis or axes group (as slave) with an axis as master in order that the slave executes its path with synchronization to the progress of the master, meaning linked to a one dimension source for synchronization.
<b>TCP</b>	Tool Center point, the point in the machine that is commanded to move, typically the center or the head of the tool. It can be described in different coordinate systems.
<b>Tracking</b>	Is characterized by an axis group that follows with its movement the movement of another axis group.

Term	Definition
<b>Trajectory</b>	Time dependent description of the path the TCP of an axes group moves along. Additionally to the geometrical description of the space curve, time dependent state variables like velocity, acceleration, jerk, forces etc. are specified.
<b>Velocity</b>	For a group of axes this means: <ul style="list-style-type: none"> <li>• in ACS the velocities of the different axes</li> <li>• in MCS and PCS it provides the velocity of the TCP</li> </ul>

### 12.1.1.2 Group State Diagrams



‡ - A limitation exists where a single axis ErrorStop condition will not change the AxesGroup state to GroupErrorStop. (PLCopen Part 4, Section 3.1)

### 12.1.1.3 Coordinate Systems

There are three different coordinate system (CS) types:

- Machine (MCS)
- Axes (ACS)
- Product/Program (PCS)

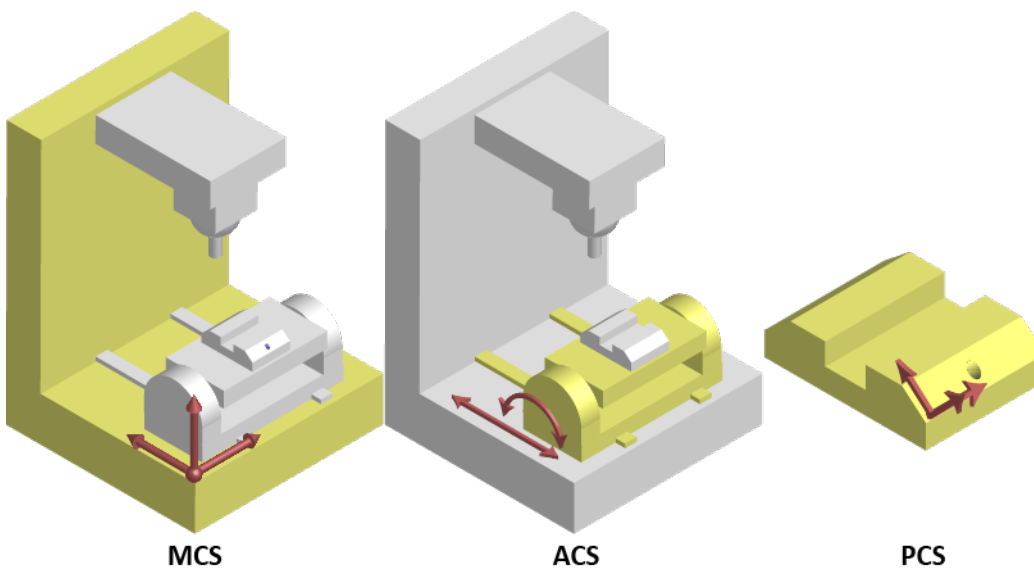
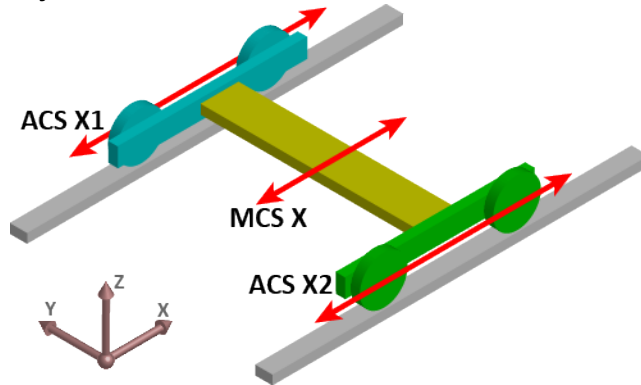


Figure 7-1: Examples of CS types on a machine and part.

Many coordinated moves may be done in a choice of coordinate systems. The differences between the types are offsets and possibly transformational algorithms to convert between the different systems, which ultimately control the actual axes on a piece of machinery.

For example, the X-axis of a Machine CS is meant to command a pair of Axes CS axes (X1 and X2) which together form a gantry. The relative movement of the MCS X axis would be added to both ACS axes. The two ACS axes can also be commanded independently for minor alignment adjustments.



#### **TIP**

The Product Coordinate System is often rotated and/or offset from the Machine Coordinate System.

## 12.1.2 Design Coordinated Motion

This section discusses how to create a coordinated motion application, including adding coordinated motion to existing applications.

For more information on Coordinate Motion and the associated functions and function blocks see:

- [Coordinated Motion](#) in the Advanced Topics section
- [Coordinated Motion Function Blocks](#)

### 12.1.2.1 Create a Linear or Circular Coordinated Motion Application

A Coordinated Motion application can be created in one of two ways:

- Use a Coordinated Motion template to create a new application. Two Coordinated Motion templates are currently available.
  - The first template controls two PLCopen axes in coordinated motion.
  - The second template controls two PLCopen axes in coordinated motion plus a third independent Pipe Network axis.
- Modify an existing application to include coordinated motion functions. When modifying an existing application, axes need to be grouped to define the axes that will be active when performing coordinated motion on that group. More information about Axes Groups can be found in the section [What are Axes Groups?](#).

#### **NOTE**

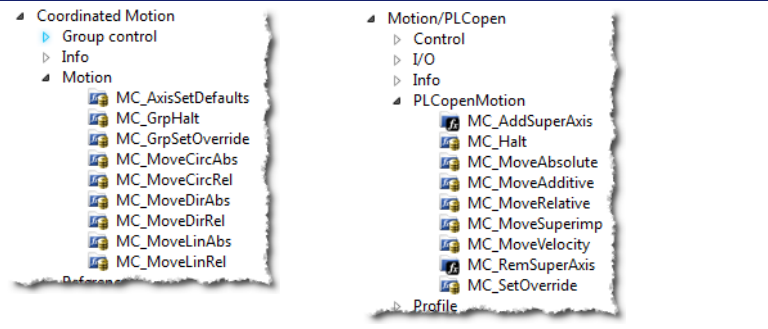
Coordinated motion can only be performed with PLCopen axes. Pipe Network axes do not support this feature, although Pipe Network axes can be moved independently from coordinated motion groups. Any synchronization between coordinated motion and Pipe Network axes must be performed by the PLC application.

Related axes are "grouped" in an axes group. Coordinated motion is then performed on an axes group. For more information see [What are Axes Groups?](#).

**TIP**

There are two vital concepts to remember when making interpolated motion.

- Interpolated motion requires creating a motion group that results in a second group coordinate system.
- Group coordinate system positions are only affected by group motion, and non-group coordinate system positions are only affected by non-group motion.

**Group motion commands Non-Group motion commands****NOTE**

Typically axes that become part of a motion group are first homed using non -group function blocks to establish a home or starting position for the group motion.

Typically, the following set of function blocks should be called before executing coordinated motion.

1. Call [MLMotionInit](#) (BasePeriod) to initialize the motion engine. Base period is specified in microseconds.

```
MLMotionInit(1000.0); // 1000 µSec -> 1 mSec
```

2. Call [MC\\_CreateAxesGrp](#) (Enable, GroupName, UpdateRate, MaxNumberOfAxes, AxesGroupRef) to create a Coordinated Motion Axes Group

**NOTE**

[MC\\_CreateAxesGrp](#) needs to be called between [MLMotionInit\(\)](#) and [MLMotionStart\(\)](#).

```
Inst_MC_CreateAxesGrp(TRUE, 'GROUP1', 6, 2, Group1_ref);
```

In the example above, the axes group name is 'GROUP1', the update rate is 1 mSec (specified by '6') and the maximum number of axes that can be added to the group is 2. The group reference variable 'Group1\_ref' will be used in future coordinated motion function block calls to reference this newly created group.

3. Call [MC\\_InitAxesGrp](#) (Enable, AxesGroup, VelLimit, AccLimit, DecLimit, JerkLimit) to initialize the path limits for velocity, acceleration, deceleration, and jerk.

```
Inst_MC_InitAxesGrp(TRUE, Group1_ref, 100.0, 300.0, 300.0, 1000.0);
```

In the example above, the kinematic limits for axes group 'Group1\_ref' will be set. The velocity limit will be set to 100.0 user units/second, acceleration and deceleration limits will be set to 300.0 user units/second<sup>2</sup> and jerk will be set to 1000.0 user units per second<sup>3</sup> (Jerk will be supported in a future release).

- Call [MC\\_CreatePLCAxis](#) (AxisName, BusInterface, BusAddress, AxisNumber, AxisType, UserUnits, FeedbackUnits, Rollover, UpdateRate) to create a Coordinated Motion Axis. This function needs to be called for each Coordinated Motion Axis wanted in the application.

**NOTE**

MC\_CreatePLCAxis needs to be called between MLMotionInit() and MLMotionStart().

```
Inst_MC_CreateAxis(TRUE, 'CoordAxis1', 'EtherCATDriver', 1001,
CoordAxis1_AxisNum, 0, 360, 1048576, 0, 6);
Inst_MC_CreateAxis(TRUE, 'CoordAxis2', 'EtherCATDriver', 1002,
CoordAxis2_AxisNum, 0, 360, 1048576, 0, 6);
```

In the example above:

- Two axes are created and are named 'CoordAxis1' and 'CoordAxis2'.
  - The bus interface for both is 'EtherCATDriver'.
  - The address of the drive on the bus is 1001 and 1002.
  - The axis numbers are set with variables CoordAxis1\_AxisNum and CoordAxis2\_AxisNum which is set to an integer value between 1 and 256. Each axis number is unique.
  - The axis type for both, '0', indicates a servo axis.
  - The user units are 360, which is the 'user unit' portion of the 'user unit/feedback' ratio.
  - The feedback units are 1048576, which is the 'feedback' portion of the 'user unit/feedback' ratio.
  - The rollover position for both, '0' indicates no rollover.
  - The update rate for both, '6', indicates a 1mSec update rate.
- Call [MLMotionStart](#) () to start the Motion and the motion bus driver. This also initializes the EtherCAT network to operational mode.

```
MLMotionStart();
```

- Call [MC\\_AddAxisToGrp](#) (Execute, AxesGroup, Axis, IdentInGroup) for each axis to be added to the group.

```
Inst_MC_AddAxisToGrp(TRUE, Group1_ref, CoordAxis1_ref, 0);
Inst_MC_AddAxisToGrp(TRUE, Group1_ref, CoordAxis2_ref, 1);
```

In the example above, we are adding two axes, CoordAxis1 and CoordAxis2, to the group referenced by 'Group1\_ref'. The axes are stored in the IdentInGroup positions 0 and 1. Note that when the group was created, it was specified that no more than 2 axes will be part of this group. Therefore, valid IdentInGroup locations are 0 and 1.

- Call [MC\\_Power](#) (Enable, Axis, EnablePositive, EnableNegative, BufferMode) for each Coordinated Motion Axis to enable the drive and close the servo loop.

```
Inst_MC_Power1(TRUE, CoordAxis1_ref, TRUE, TRUE, 0);
Inst_MC_Power2(TRUE, CoordAxis2_ref, TRUE, TRUE, 0);
```

In the example above, drives CoordAxis1\_ref and CoordAxis2\_ref will be enabled and the position loop will be closed. Note that parameters 'TRUE, TRUE, 0' are place holders for future use and are not currently used.

- Call [MC\\_GrpEnable](#) (Execute, AxesGroup) to change the state of the Coordinated Motion Axis Group from GroupDisabled to GroupStandby and allow motion to be performed on the

group.

```
Inst_MC_GrpEnable(TRUE, Group1_ref);
```

In the example above, 'Group1\_ref' state will be changed from GroupDisabled to GroupStandby. The group must be in GroupStandby in order to perform motion.

- For the examples that follow, we want to set the current location of the axes in the group to 0, 0. This can be done by calling `MC_GrpSetPos` (Execute, AxesGroup, Position[], Relative, CoordSystem, BufferMode)

```
PosAbs[1] := 0;
PosAbs[2] := 0;
Inst_MC_GrpSetPos(TRUE, Group1_ref, PosAbs, 0, MC_COORDINATE_SYSTEM_ACS, 0);
```

In the example above, the axis positions of 'Group1\_ref' will be set to 0, 0. 'PosAbs' specifies the position for each axis in the group. 'Relative' input, '0', uses 'PosAbs' to set the absolute position. The coordinate system is set to ACS. The buffer mode, '0', is a placeholder for future use and is not currently used.

**NOTE**  
No motion will be performed when this function block is executed.

- Optional:* To Add more axes to the group, modify the above code in the following way:
  - In Step 2: Update the MaxNumberOfAxes input argument so that the group can handle the desired number of axes.
  - In Step 4: Create the additional axes that will added to the group.
  - In Step 6: Add the additional axes to the group.
  - In Step 7: Power on the additional axes.
  - In Step 9: You will need to increase the size of the PosAbs array so it matches the number you used in step 2, and set the position of the additional axes to zero.

After the above function calls have been made, we can start coordinated motion moves.

[Performing a Linear Move](#)

[Performing a Circular Move](#)

**12.1.2.1.1 What are Axes Groups?**

Related axes are grouped in an Axes Group to support interpolation. Axes Groups are accessed via the type AXES\_GROUP\_REF. The following image shows the relationships between the different CSs and groups.

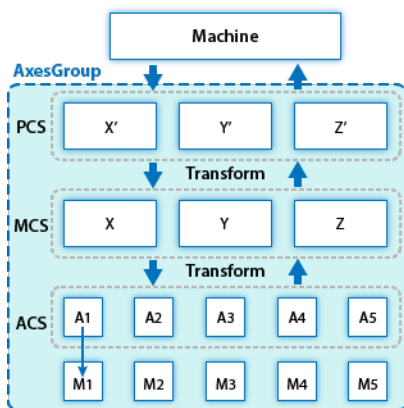


Figure 7-2: Overview of AxesGroup

The Axes Group, shown in blue above, provides the interface to the user of the group of axes. To access the relevant coordinate system, the relevant function blocks have an input Coordinate System which supports the three levels ACS, MCS, and PCS.

Parameters in the AxesGroupRef can include remaining time and remaining distance before target position (or velocity or equal) is reached.

#### 12.1.2.1.2 Performing a Linear Move

Linear moves can be programmed using absolute or relative positions using the following function blocks:

- [MC\\_MoveLinAbs](#) which commands interpolated linear movement on an axes group to the specified absolute positions.
- [MC\\_MoveLinRel](#) which commands interpolated linear movement on an axes group to the specified relative positions.

Prior to performing any coordinated moves, some setup is needed (see ["Create a Linear or Circular Coordinated Motion Application" on page 579](#)). Once these steps have been performed, a linear move can be performed.

In the following examples, two linear moves will be performed. The first move is an absolute linear move that goes from (0, 0) to (100, 200). The second move is a relative linear move that goes a distance of (-75, 50) from the end of the first move. The BufferMode input is set to 'Buffered', meaning this move will wait for the first move to complete before it begins executing.

- **To Perform an Absolute Linear Move**

Call [MC\\_MoveLinAbs](#) (Execute, AxesGroup, PositionArray, Velocity, Acceleration, Deceleration, Jerk, CoordSystem, BufferMode, TransitionMode, TransitionParameter). PositionArray is an array of absolute end positions containing one position for each axis in the group. The inputs velocity, acceleration, deceleration, and jerk establish the maximum values for the move.

In this example, PosArrayAbs[0] represent the x-axis and PosArrayAbs[1] represent the y-axis.

```
PosArrayAbs[0] := 100;
PosArrayAbs[1] := 200;
TransParam[0] := 0;
TransParam[1] := 0;

Inst_MC_MoveLinRel(TRUE, Group1_ref, PosArrayAbs, MaxVel, MaxAcc,
MaxDec, 0, MC_COORDINATE_SYSTEM_ACS, 1, 0, TransParam);
```

In the example a linear move will be performed on axis group 'Group1\_ref'.

- PosArrayAbs contains the absolute end points of the axes in the group. The axis stored in position 0 (IdentInGroup) of the group will be moved to 100.0. The axis stored in position 1 of the group will be moved to 200.0.
- The maximum velocity is specified by variable MaxVel and is specified in 'user units/sec'.
- The maximum acceleration and deceleration are specified by variables MaxAcc and MaxDec and are specified in 'user units/sec<sup>2</sup>'.
- The maximum jerk is currently not supported and can be set to a value of 0.
- The coordinate system is ACS
- The BufferMode is set to 1, indicating the move is buffered. For more information about buffer modes, see the [Buffer Modes](#) overview.
- The TransitionMode is set to 0, indicating no transition mode will be used. For more information about transition modes, see the [Transition Between Moves](#) section.

- The TransParam array is required and the contents can be set to 0 since the transition mode is not being used. There has to be one array entry for each axis in the group.

#### • To Perform a Relative Linear Move

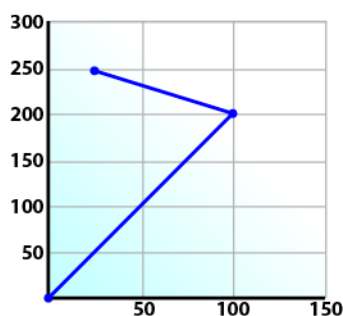
Call `MC_MoveLinRel` (Execute, AxesGroup, Distance, Velocity, Acceleration, Deceleration, Jerk, CoordSystem, BufferMode, TransitionMode, TransitionParameter). The Distance input is an array of distances, one distance for each axis in the group. The inputs velocity, acceleration, deceleration, and jerk establish the maximum values for the move.

In this example, `DistArrayRel[0]` represent the x-axis and `DistArrayRel[1]` represent the y-axis.

```
DistArrayRel[0] := -75.0; // Start pt 100 - rel 75 -> 25 absolute
end pt
DistArrayRel[1] := 50.0; // Start pt 200 + rel 50 -> 250 absolute
end pt
TransParam[0] := 0;
TransParam[1] := 0;

Inst_MC_MoveLinRel(TRUE, Group1_ref, DistArrayRel, MaxVel, MaxAcc,
MaxDec, 0, MC_COORDINATE_SYSTEM_ACS, 1, 0, TransParam);
```

In the example above, all the variables have the same meaning as the absolute linear example except `DistArrayRel`. `DistArrayRel` contains the relative distance to move for each axis in the group. The axis stored in position 0 (`IdentInGroup`) of the group will be moved a distance of -75.0. The axis stored in position 1 of the group will be moved a distance of 50.0.



#### • To Perform a Linear Move With More Than Two Axes

##### NOTE

The dimensionality of the move is determined by the number of axes mapped to the group. This implies that a group which could hold a maximum of three or more axes will do two dimensional moves if it only has two valid axes mapped to it.

In order to perform higher dimensional moves, additional axes must be added to the group. The steps to do this are detailed in [Create a Linear or Circular Coordinated Motion Application](#).

After the additional axes are added perform the following steps.

1. From within the Dictionary, update the array size of the variable being passed (`PosArrayAbs` and `DistArrayRel` in the examples above) to the Position input so that its length matches the maximum number of axes allowed in the group.
2. Set the desired values for the additional axes in the now larger position arrays.

##### 12.1.2.1.3 Performing a Circular Move

Circular moves can be programmed using absolute or relative positions using the following function blocks:



- [MC\\_MoveCircAbs](#) which commands interpolated circular movement on an axes group to the specified absolute positions.
- [MC\\_MoveCircRel](#) which commands interpolated circular movement on an axes group to the specified relative positions.

Prior to performing any coordinated moves, some setup is needed (see ["Create a Linear or Circular Coordinated Motion Application" on page 579](#)). Once these steps have been performed, a circular move can be performed.

In the following examples, two circular moves will be performed. The first move is an absolute circular move that goes from (0, 0) to (90, 90). CircMode specifies that the aux point (0, 180) will be crossed during the paths start to end. The second move is a relative circular move whose end point is (90, 90) from the end of the first move. In this move, CircMode specifies that the aux point (0, 90) is the relative center of the circle. The BufferMode input is set to 'Buffered', meaning this move will wait for the first move to complete before it begins executing.

- **To perform an Absolute Circular Move:**

Call [MC\\_MoveCircAbs](#) (Execute, AxesGroup, CircMode, AuxPoint[], EndPoint[], PathChoice, Velocity, Acceleration, Deceleration, Jerk, CoordSystem, BufferMode, TransitionMode, TransitionParameter).

In this example, PosCircAuxAbs[0] and PosCircEndAbs[0] represent the x-axis. PosCircAuxAbs[1] and PosCircEndAbs[1] represent the y-axis.

```
PosCircAuxAbs[0] := 0; // A point on the circle that is crossed
on the
PosCircAuxAbs[1] := 180; // path from start to end point.
PosCircEndAbs[0] := 90; // Absolute end point.
PosCircEndAbs[1] := 90;

Inst_MC_MoveCircAbs(TRUE, Group1_ref, MC_CIRC_MODE_BORDER,
PosCircAuxAbs, PosCircEndAbs, MC_CIRC_PATHCHOICE_CLOCKWISE, MaxVel,
MaxAcc, MaxDec, 0, MC_COORDINATE_SYSTEM_ACS, MC_BUFFER_MODE_
BUFFERED, MC_TRANSITION_MODE_NONE, TransParam);
```

In the example a circular move will be performed on axis group 'Group1\_ref'.

- CircMode is defined as MC\_CIRC\_MODE\_BORDER. This mode indicates that the AuxPoint array input will indicate a point on the circle which is crossed on the path from the starting point to the end point. See [Circular Moves Diagrams](#) for more information on CircMode movement options.
- The AuxPoint array, 'PosCircAuxAbs', defines an absolute point on the circle which is crossed on the path from the starting point to the end point. The contents of this array are determined by the CircMode variable, MC\_CIRC\_MODE\_BORDER.
- The EndPoint array, 'PosCircEndAbs', contains the absolute end point for each axis in the group. The absolute end point of the axis stored in position 0 (IdentInGroup) of the group will be 90.0. The absolute end point of the axis stored in position 1 of the group will be 90.0.
- PathChoice is only relevant when CircMode is set to MC\_CIRC\_MODE\_CENTER. In this case, this parameter is not used.
- The maximum velocity is specified by variable MaxVel and is specified in 'user units/sec'.
- The maximum acceleration and deceleration are specified by variables MaxAcc and MaxDec and are specified in 'user units/sec<sup>2</sup>'.
- The maximum jerk is currently not supported and can be set to a value of 0.
- The coordinate system is ACS

- The BufferMode is set to MC\_BUFFER\_MODE\_BUFFERED, indicating the move is buffered. For more information about buffer modes, see the [Buffer Modes](#) overview.
- The TransitionMode is set to MC\_TRANSITION\_MODE\_NONE, indicating no transition mode will be used. For more information about transition modes, see the [Transition Between Moves](#) section.
- The TransParam array is required. The TransParam array is a 2-element array containing the corner distance and velocity for the transition. Transitions are not used in this example and therefore the contents can be set to 0.

- **To perform a Relative Circular Move:**

Call `MC_MoveCircRel` (Execute, AxesGroup, CircMode, AuxPoint[], EndPoint[], PathChoice, Velocity, Acceleration, Deceleration, Jerk, CoordSystem, BufferMode, TransitionMode, TransitionParameter).

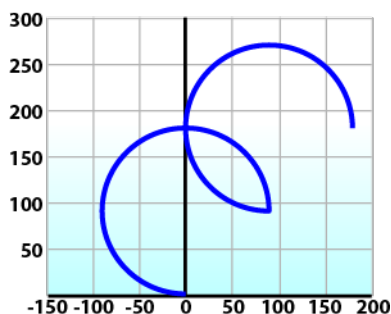
In this example, PosCircAuxRel[0] and PosCircEndRel[0] represent the x-axis. PosCircAuxRel[1] and PosCircEndRel[1] represent the y-axis.

```
PosCircAuxRel[0] := 0; // Relative center of the circle.
PosCircAuxRel[1] := 90;
PosCircEndRel[0] := 90; // Relative end point.
PosCircEndRel[1] := 90; // Start pt 90,90 + rel 90,90 -> 180,180
absolute end pt

Inst_MC_MoveCircRel(TRUE, Group1_ref, MC_CIRC_MODE_CENTER,
PosCircAuxRel, PosCircEndRel, MC_CIRC_PATHCHOICE_CLOCKWISE, MaxVel,
MaxAcc, MaxDec, 0, MC_COORDINATE_SYSTEM_ACS, MC_BUFFER_MODE_
BUFFERED, MC_TRANSITION_MODE_NONE, TransParam);
```

In the example all the variables have the same meaning as the circular absolute example except:

- CircMode is defined as MC\_CIRC\_MODE\_CENTER. This mode indicates that the AuxPoint array input will indicate the center point of the circle. See [Circular Moves Diagrams](#) for more information on CircMode movement options.
- The AuxPoint array, 'PosCircAuxRel', defines the relative center point of the circle. The contents of this array are determined by the CircMode variable, MC\_CIRC\_MODE\_CENTER.
- The EndPoint array, 'PosCircEndRel', contains the relative end point for each axis in the group. The relative end point of the axis stored in position 0 (IdentInGroup) of the group will be 90.0. The relative end point of the axis stored in position 1 of the group will be 90.0.
- PathChoice is relevant when CircMode is set to MC\_CIRC\_MODE\_CENTER. In this case, PathChoice is MC\_CIRC\_PATHCHOICE\_CLOCKWISE which specifies the direction of the path.

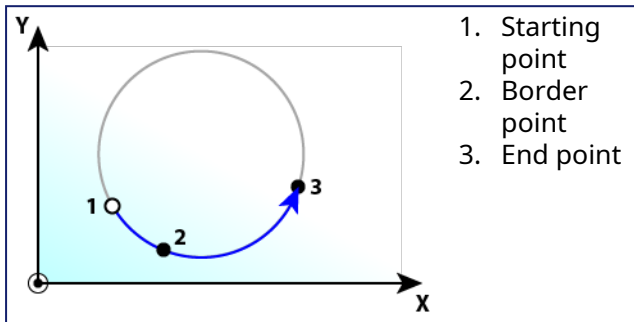


### 12.1.2.1.3.1 Circular Moves Diagrams

CircMode = BORDER

The user defines the end point and a border point (= input 'AuxPoint') on the sector of the circle which the machine will traverse. For Relative mode, both points are defined relative to the starting point.

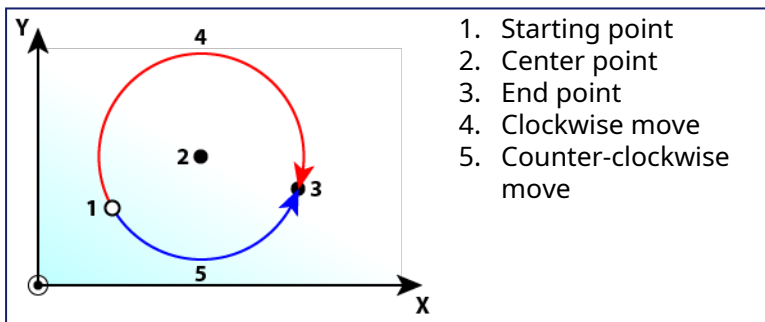
<b>Advantages</b>	<ul style="list-style-type: none"> <li>The border point can usually be reached by the machine, i.e. it can be taught.</li> </ul>
<b>Disadvantages</b>	<ul style="list-style-type: none"> <li>Restricted to angles &lt; 360° in one single command.</li> </ul>



**CircMode = CENTER**

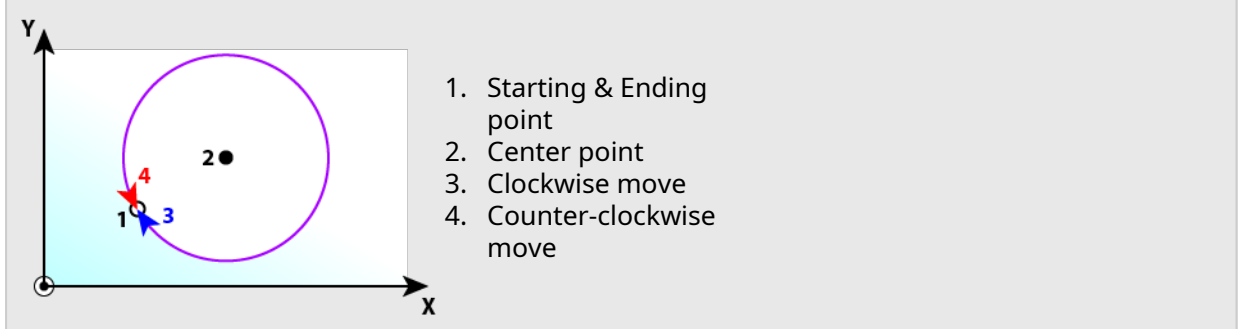
The user defines the end point and center point (= input 'AuxPoint') of the circle. The input 'PathChoice' defines clockwise or counter-clockwise motion. For Relative mode, both points are defined relative to the starting point.

<b>Advantages</b>	<ul style="list-style-type: none"> <li>Full 360° moves are possible.</li> </ul>
<b>Disadvantages</b>	<ul style="list-style-type: none"> <li>Cannot perform zero-distance moves.</li> <li>Over-determination of the circle equation.</li> </ul>



**NOTE**

A 360° move will be performed if the end point is the same as the start point.

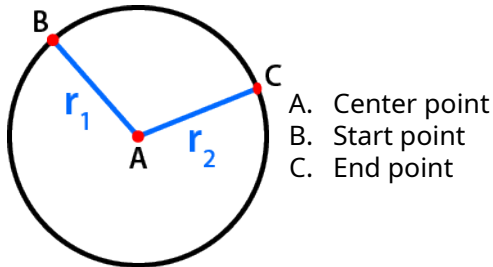


**12.1.2.1.3.2 Precision Requirements for Circular Move Input Parameters**

The input parameters to [MC\\_MoveCircAbs](#) and [MC\\_MoveCircRel](#) are validated when constructing a circle. The distance of the center point from the start and end points is checked. Ideally, the two distances will be the same but calculation errors or input data precision can cause the two

distances to be slightly different. If validation fails then PLCopen error 50 ("Cannot construct a circle with specified parameters", see "PLCopen Function Block ErrorID Output" on page 423) will be returned from the function block. Input parameters are validated using the methods described below.

Here is a circle we want to create. The distances from the center to the start and end point are measured as  $r_1$  and  $r_2$  respectively.



There are two methods for validating the circle parameters, the default method and a method using a custom tolerance.

#### NOTE

If MC\_GRP\_PARAM\_CIRCLE\_TOLERANCE is set to zero (0), the default validation method will be used.

#### Default Validation Method

The difference between the two measurements must not be larger than one part in 100,000. If all positions are specified to six significant digits then this requirement will be met.

Assuming that  $r_2$  is greater than  $r_1$ , we can write  $r_2$  in terms of  $r_1$  and a small deviation value named  $\epsilon$ :

$$r_2 = r_1(1 + \epsilon)$$

If  $\epsilon$  exceeds a value of 10<sup>-5</sup>, then the validation will fail and PLCopen error 50 will be returned from the function block.

#### Examples of Default Validation

- If the desired circle has a radius of 50 user units, then the center must be specified with a precision of 0.0005 user units.
- If the desired circle has a radius of 2,000 user units, then the center must be specified with a precision of 0.02 user units.
- If the desired circle has a radius of 500,000 user units, then the center must be specified with a precision of 5 user units.

#### Validation with Custom Tolerance

A custom tolerance can be specified via the group parameter MC\_GRP\_PARAM\_CIRCLE\_TOLERANCE (see "Axes Group Parameters" on page 417). This parameter represents the amount of error allowed, measured in user units. The parameter can be set by using MC\_GrpWriteParam and can be read via MC\_GrpReadParam.

Assuming that  $r_2$  is greater than  $r_1$ , we can write  $r_2$  in terms of  $r_1$  and a deviation value named  $\Delta$ :

$$r_2 = r_1 + \Delta$$

The validation will fail if  $\Delta$  exceeds the value of the group parameter MC\_GRP\_PARAM\_CIRCLE\_TOLERANCE and PLCopen error 50 will be returned from the function block.

#### Examples of Validation using a Custom Tolerance

- If r1 has a value of 50 user units and the circle tolerance parameter is .1 user units, then r2 must be in the range of 49.9 to 50.1 user units.
- If r1 has a value of 2000 user units and the circle tolerance parameter is 1 user units, then r2 must be in the range of 1999 to 2001 user units.

**TIP**

Use LREAL variables and LREAL versions of math functions when calculating the desired circle parameters inside a KAS application. The LREAL versions of functions usually have an 'l' at the end of their name. For example, the LREAL version of cos is cosl. This will help avoid errors.

**12.1.2.1.3.3 How to perform a complete circular move**

A full circle may be performed using the following procedure.

1. Call either the [MC\\_MoveCircAbs](#) or [MC\\_MoveCircRel](#) function block.
2. Set **CircMode** to Center (`MC_CIRC_MODE_CENTER`).
3. Set the **EndPoint** to be the same as the start point.
4. Repeat as necessary for multiple rotations.

**NOTE**

`MC_CIRC_MODE_BORDER` cannot be used because it is limited to angles <360°.

**12.1.2.2 Blending Between Moves**

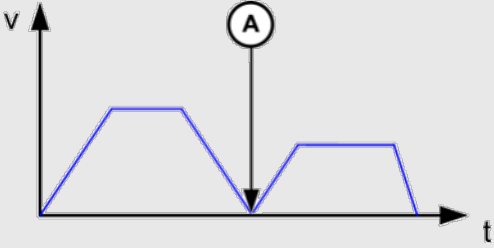
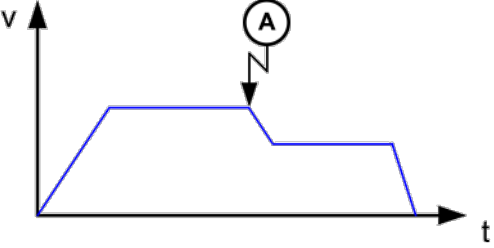
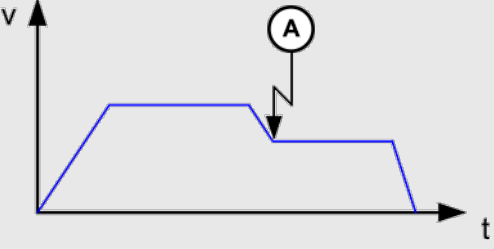
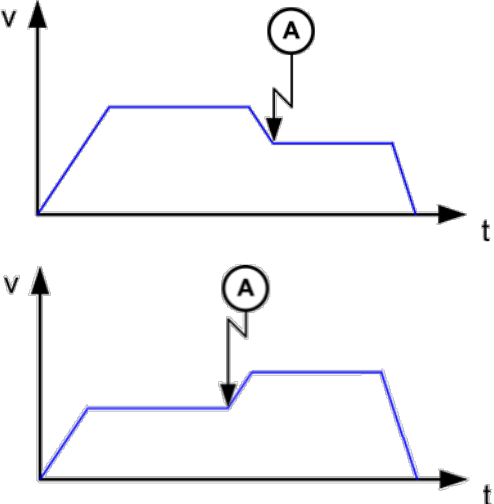
Some Coordinated Motion Function Blocks have a `BufferMode` input parameter. Possible buffer modes include:

- `MC_BUFFER_MODE_ABORTING = 0`
- `MC_BUFFER_MODE_BUFFERED = 1`
- `MC_BUFFER_MODE_BLENDED_PREVIOUS = 2`
- `MC_BUFFER_MODE_BLENDED_NEXT = 3`
- `MC_BUFFER_MODE_BLENDED_LOW = 4`
- `MC_BUFFER_MODE_BLENDED_HIGH = 5`

When the current and next motion function blocks are blended (2 through 5 above), the axes group will not stop between motions. The velocity will be blended according to the specified blending mode.

In addition, the Function Block `TransitionMode` parameter can be set to provide a smooth circular arc between moves. Transition modes are described in [Transition Between Moves](#).

Blend Type	Profile	Description
<code>MC_BUFFER_MODE_ABORTING</code>		<p>The current move stops execution immediately. The next move becomes active and begins to accelerate or decelerate from the current velocity to its programmed velocity.</p> <p>In the image to the left the second function block starts execution at point "A".</p>

Blend Type	Profile	Description
<b>MC_BUFFER_MODE_BUFFERED</b>		<p>The current move completes as was originally specified, coming to a stop. When the current move reaches its endpoint, the next move becomes active and begins to accelerate to its programmed velocity.</p> <p>In the image to the left the second function block starts execution at point "A".</p>
<b>MC_BUFFER_MODE_BLENDED_PREVIOUS</b>		<p>The current move stays at its programmed velocity to its endpoint. When the current move reaches its endpoint, the next move becomes active and begins to accelerate or decelerate to its programmed velocity.</p> <p>In the image to the left the second function block starts execution at point "A".</p>
<b>MC_BUFFER_MODE_BLENDED_NEXT</b>		<p>The current move will begin to accelerate or decelerate so that it reaches the next move's programmed velocity at the time the current move reaches its endpoint.</p> <p>In the image to the left the second function block starts execution at point "A".</p>
<b>MC_BUFFER_MODE_BLENDED_LOW</b>		<p>The velocity is blended to the lowest velocity of the two moves.</p> <ul style="list-style-type: none"> <li>• If the current move's velocity is lower than the next move's velocity, <b>Blending Previous</b> will be used.</li> <li>• If the next move's velocity is lower than the current move's velocity, <b>Blending Next</b> will be used.</li> </ul> <p>In the images to the left the second function block starts execution at point "A".</p>

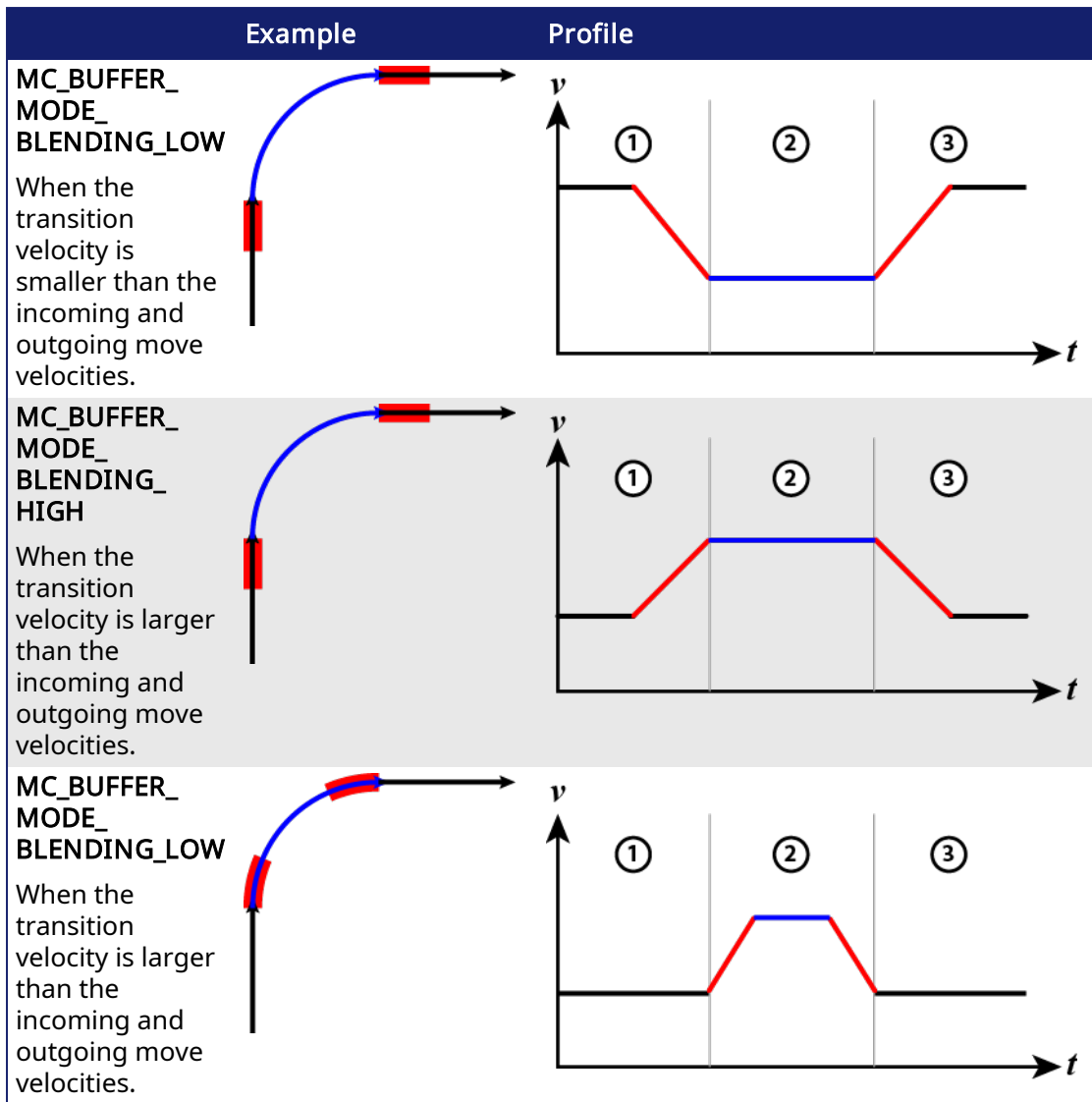
Blend Type	Profile	Description
MC_BUFFER_MODE_BLENDED_HIGH		<p>The velocity is blended to the highest velocity of the two moves.</p> <ul style="list-style-type: none"> <li>If the current move's velocity is higher than the next move's velocity, <b>Blending Previous</b> will be used.</li> <li>If the next move's velocity is higher than the current move's velocity, <b>Blending Next</b> will be used.</li> </ul> <p>In the images to the left the second function block starts execution at point "A".</p>

### 12.1.2.3 Blending with Transitions

When blending is specified when a transition is specified, then the blending mode is used to blend velocities of path segments when the path changes from the incoming segment to the transition segment and from the transition segment to the outgoing segment.

The most common blending mode choices for applications are listed below showing the velocity profile and where acceleration occurs on the path. In the examples, blue denotes the transition arc, red denotes where the acceleration occurs, 1 indicates the incoming move, 2 indicates the transition, and 3 indicates the outgoing move.

Example	Profile
<p>MC_BUFFER_MODE_BLENDED_BUFFERED</p>	
<p>MC_BUFFER_MODE_BLENDED_HIGH</p> <p>When the transition velocity is smaller than the incoming and outgoing move velocities.</p>	



#### 12.1.2.4 Transition Between Moves

A transition mode must be specified when a new move is appended to a move that is already in progress. Different transition parameters may be required, depending on the transition mode. This characterizes the contour of the transition segment.

The supported transition modes are:

- No Transition ("TMNone")
- Corner Distance ("TMCornerDistance")

Transition Mode	Number of Transition Parameter Array Elements	Transition Parameter Index	Transition Parameter Name	Transition Parameter Description	Units
TMNone	0				

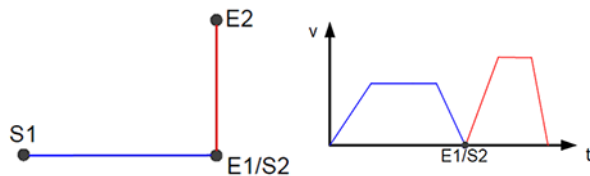


Transition Mode	Number of Transition Parameter Array Elements	Transition Parameter Index	Transition Parameter Name	Transition Parameter Description	Units
TMCornerDistance	2	0	Corner Distance	Distance to the corner of the deviation and the return point from the original contour	User units
		1	Velocity	The velocity value for the transition segment	User units per second

Table 7-1: Transition Mode Parameters

#### 12.1.2.4.1 No Transition (“TMNone”)

"Insert no transition contour segment."



Motion Path

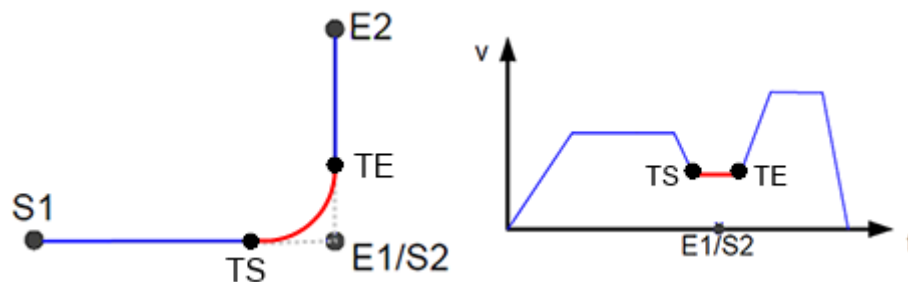
Motion Velocity Profile

The motion blocks are not modified and no transition curve is inserted using this mode. This is the only possible transition mode for the “Buffered” [buffer mode](#).

No transition parameters are used for this transition mode.

#### 12.1.2.4.2 Corner Distance (“TMCornerDistance”)

"Transition with given corner distance."



Motion Path

Motion Velocity Profile

The corner distance transition mode is specified using `TMCornerDistance` as the transition mode for a motion block.

TransitionParameter Index	Name	Description
0	Corner Distance	Distance to the corner of the deviation and the return point from the original contour.
1	Velocity	The velocity value of the transition segment.

Corner Distance transitions are handled differently, depending upon whether the connecting moves are lines or arcs, and all of the possible combinations, (line-line, arc-arc, line-arc, arc-line).

- Line-to-Line transitions will shorten the next move by the corner distance. See [Line to Line Transitions](#) for more information.
- Line-to-Arc and Arc-to-Line transitions shorten the linear move. See [Line-to-Arc and Arc-to-Line Transitions](#) for more information.
- Arc-to-Arc transitions will shorten the arc with the larger radius by the corner distance. See [Arc-to-Arc Transitions](#) for more information.

#### 12.1.2.4.3 Related Functions

[MC\\_MoveCircAbs \(Function Block\)](#)

[MC\\_MoveCircRel \(Function Block\)](#)

[MC\\_MoveLinAbs \(Function Block\)](#)

[MC\\_MoveLinRel \(Function Block\)](#)

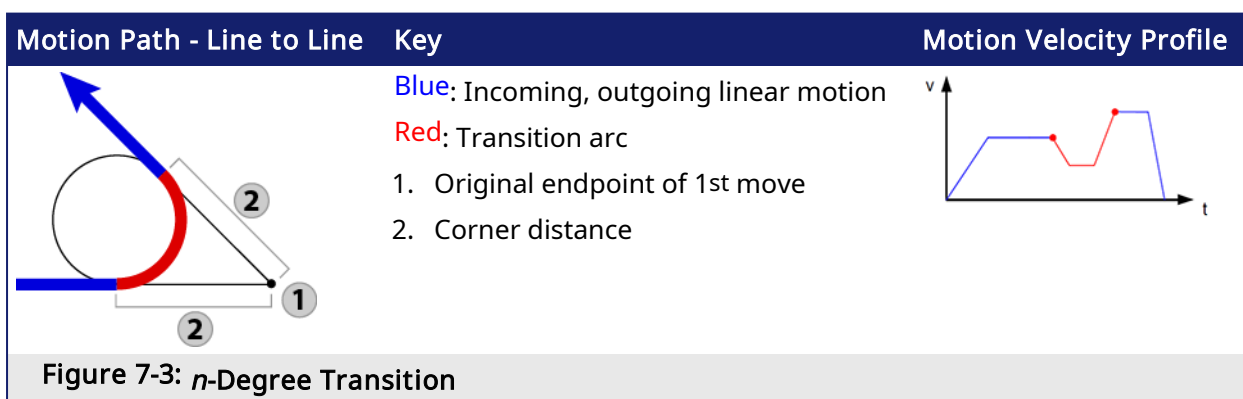
#### 12.1.2.4.4 Line to Line Transitions

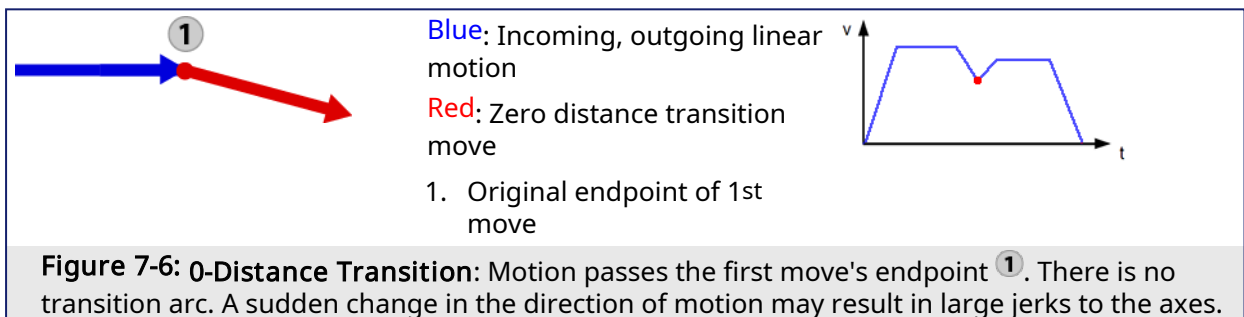
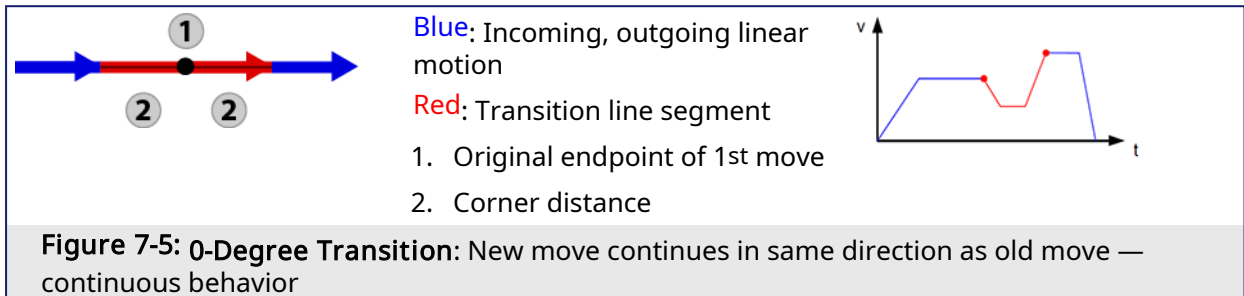
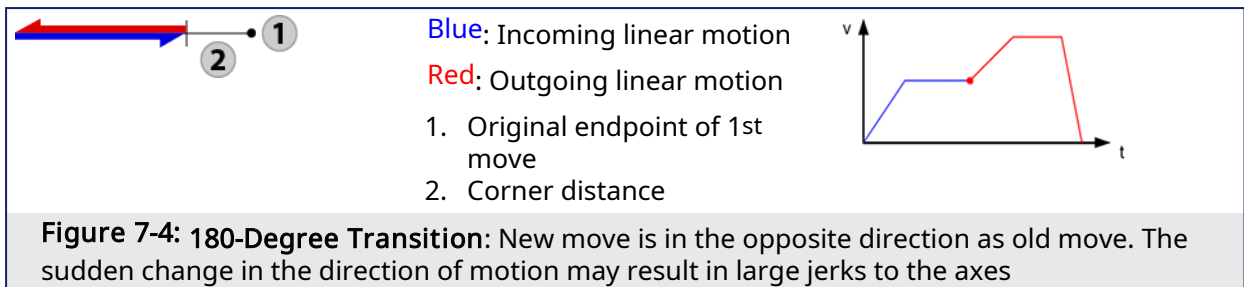
When both moves are linear the current and next moves are shortened using this transition mode. The amount is specified by the corner distance. A circular arc connects the two moves (except as noted below), allowing for a smooth transition (see [n-Degree Transition](#) below). The circular arc that connects the two moves derives its velocity from the transition parameter and the acceleration and deceleration values are derived from the next move.

#### NOTE

There are three special cases for line to line transitions:

- Zero-degree transitions:  
The path will remain unchanged but a linear transition move with the specified transition velocity will be inserted.
- 180-degree transitions:  
The current move will be shortened by the corner distance when the next move travels in the opposite direction.  
*If the buffer mode is "Buffered" then the path velocity will go to zero. All other buffer modes may cause a large acceleration and jerk.*
- Zero-distance transitions:  
A zero distance transition move will be inserted, which will only affect blending.  
*If the buffer mode is "Buffered" then the path velocity will go to zero. All other buffer modes may cause a large acceleration and jerk.*





#### 12.1.2.4.5 Line-to-Arc and Arc-to-Line Transitions

With this transition mode, the linear move is shortened when one move is linear and the other is circular. The amount is specified by the corner distance. A circular arc connects the two moves (except as noted below), allowing for a smooth transition (see [n-Degree Transition](#) below). The circular arc that connects the two moves derives its velocity from the transition parameter and the acceleration and deceleration values are derived from the next move.

#### NOTE

There are three special cases for Line-to-Arc and Arc-to-Line transitions:

- Tangent transitions:  
The linear move will be shortened by the corner distance and a linear transition move will be inserted to cover the distance that was removed.
- Intersection transitions:  
This will only when the arc intersects the line corner distance away from the point where the line and the arc meet. The transition move will be zero distance.  
*If the buffer mode is "Buffered" then the path velocity will go to zero. All other buffer modes may cause a large acceleration and jerk.*
- Zero-distance transitions:  
A zero distance transition move will be inserted, which will only affect blending.  
*If the buffer mode is "Buffered" then the path velocity will go to zero. All other buffer modes may cause a large acceleration and jerk.*

Motion Path - Line-to-Arc / Arc-to-Line	Key	Motion Velocity Profile
	<p><b>Blue:</b> Incoming, outgoing linear and circular motion</p> <p><b>Red:</b> Transition arc</p> <ol style="list-style-type: none"> <li>1. Original start/endpoint of linear move</li> <li>2. Corner distance</li> </ol>	
<b>Figure 7-7: <math>n</math>-Degree Transition</b>		
	<p><b>Blue:</b> Incoming, outgoing linear and circular motion</p> <p><b>Red:</b> Transition line segment</p> <ol style="list-style-type: none"> <li>1. Original start/endpoint of linear move</li> <li>2. Corner distance</li> </ol>	
<b>Figure 7-8: Tangent Transition: Line is tangent to the arc</b>		
	<p>Dashed: Original shape</p> <p><b>Blue:</b> Incoming, outgoing linear and circular motion</p> <p><b>Red:</b> Zero distance transition</p> <ol style="list-style-type: none"> <li>1. Original endpoint of 1st move</li> <li>2. Corner distance</li> </ol>	
<b>Figure 7-9: Intersection Transition: The line intersects the arc a "corner distance" away from the beginning of the new move.</b>		
	<p><b>Blue:</b> Incoming, outgoing linear and circular motion</p> <p><b>Red:</b> Zero distance transition</p> <ol style="list-style-type: none"> <li>1. Original endpoint of 1st move</li> </ol>	
<b>Figure 7-10: 0-Distance Transition: Special behavior for 0-distance transitions.</b>		

### 12.1.2.4.6 Arc-to-Arc Transitions

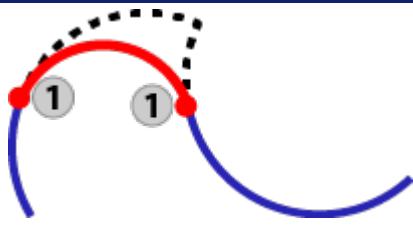
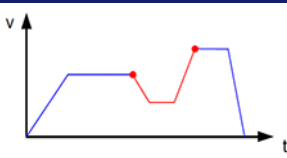
In arc-to-arc transition mode the transitions are handled as follows.

1. The arc with the larger radius is shortened by an arc length equal to the corner distance.
2. Using this point, a transition move is calculated such that it is tangent to both arcs.
3. The arc with the smaller radius is shortened to the point where it is tangent with the transition move.

The transition move that connects the two original arcs derives its velocity from the velocity transition parameter and the acceleration and deceleration values are derived from the next move.

There are six special cases for Arc-to-Arc transitions.

Transition Type	Description
<b>Zero-distance Transitions</b>	A zero-distance transition move is inserted when the corner distance is zero, which will only affect blending. The path velocity will go to zero if the buffer mode is "Buffered". All other buffer modes may cause a large acceleration and jerk.
<b>Intersection Transitions</b>	This will occur when the arcs intersect at two locations and the corner distance is equal to the arc length between the two intersections on the larger arc. The transition move will be zero distance. The path velocity will go to zero if the buffer mode is "Buffered". All other buffer modes may cause a large acceleration and jerk.
<b>Tangent Transitions with Opposite Direction</b>	A portion of the arc with the larger radius will be replaced by an arc whose length is equal to the corner distance. This will not affect the path, but will affect blending. If the arcs have the same radius, the incoming arc will be treated as having a larger radius.
<b>Line Segment Transitions</b>	For certain values of corner distance, the transition arc has an infinite radius and a line segment is used instead.
<b>Same Circle, Same Direction Transitions</b>	A transition arc with a length of twice the corner distance will be added if both arcs lie on the same circle and are in the same direction. This will not affect the path, but will affect blending.
<b>Same Circle, Opposite Direction Transitions</b>	The arcs will be shorted by an arc length equal to the corner distance and a zero distance transition will be inserted if both arcs lie on the same circle and are in the opposite direction. The path velocity will go to zero if the buffer mode is "Buffered". All other buffer modes may cause a large acceleration and jerk.

Case	Motion Path - Line-toArc / Arc-to-Line	Key	Motion Velocity Profile
<b>n-degree, arcs in opposite direction</b>		<p><b>Blue:</b> Incoming and outgoing circular motion</p> <p><b>Black:</b> "Trimmed" path</p> <p><b>Red:</b> Transition arc</p> <p><b>1:</b> Start / endpoint of arc move</p>	

Case	Motion Path - Line-to-Arc / Arc-to-Line	Key	Motion Velocity Profile
<p><math>n</math>-degree, arcs in same direction</p>		<p><b>Blue:</b> Incoming and outgoing circular motion</p> <p><b>Black:</b> "Trimmed" path</p> <p><b>Red:</b> Transition arc</p> <p>1: Start / endpoint of arc move</p> <p>2: Corner distance</p>	
<p>Zero-distance</p>		<p><b>Blue:</b> Incoming and outgoing circular motion</p> <p>1: Original Start / Endpoint of arc move</p>	<p>user units/sec</p>
<p>Intersection</p>		<p><b>Blue:</b> Incoming and outgoing circular motion</p> <p><b>Black:</b> "Trimmed" path</p> <p>1: Intersection of the two arcs</p>	
<p>Tangent arcs, opposite direction</p>		<p><b>Blue:</b> Incoming and outgoing circular motion</p> <p><b>Red:</b> Transition arc</p> <p>1: Original start / end point of moves</p> <p>2: Start / Endpoint of arc move</p>	

Case	Motion Path - Line-to-Arc / Arc-to-Line	Key	Motion Velocity Profile
Line segment		<p><b>Blue:</b> Incoming and outgoing circular motion</p> <p><b>Red:</b> Line Segment</p> <p>1: Start / Endpoint of transition move</p>	
Same circle, same direction		<p><b>Blue:</b> Incoming and outgoing circular motion</p> <p><b>Red:</b> Transition arc</p> <p>1: Start / Endpoint of arc move</p> <p>2: Original start/end point</p>	
Same circle, opposite direction		<p><b>Blue:</b> Incoming and outgoing circular motion</p> <p>1: Original Start / Endpoint of arc move</p> <p>2: Start / Endpoint of arc move</p>	

Figure 7-11: Examples of Arc-to-Arc Transitions

### 12.1.2.5 What Does MC\_GrpHalt Do?

**MC\_GrpHalt** (Execute, AxesGroup, Deceleration, Jerk) performs a controlled motion stop of all axes in a coordinated motion group. [Differences between MC\\_GrpHalt and MC\\_GrpStop.](#)

#### 12.1.2.5.1 MC\_GrpHalt Application Example

The following example demonstrates a linear coordinated move with a starting point of (0,0) and an ending point of (200, 0). The first and second **MC\_GrpHalt** commands are called while the linear coordinated move is still moving to the endpoint. As noted in the oscilloscope, the second call to **MC\_GrpHalt** aborts the first **MC\_GrpHalt** as seen by the new deceleration rate. The path velocity reaches zero (approximately (60, 0)) before the linear coordinated move reaches its end point value.

#### NOTE

For demonstration purposes it is assumed that the axes and group have been properly setup and configured. Example steps to setup coordinated motion are listed in the section [Create a Linear or Circular Coordinated Motion Application.](#)

```
1: // Perform Linear ABSOLUTE move start (0,0) end (200,0)
PosAbs[0]:= 0;
```

```

PosAbs[1] := 200;
PosAbs[2] := 0;

pathVelocity := Inst_MC_GrpReadCmdVel2.PathVelocity;

Inst_MC_MoveLinAbs( TRUE, Group1_ref, PosAbs, Velocity, Acceleration,
Deceleration, Jerk,
    MC_COORDINATE_SYSTEM_ACS, MC_BUFFER_MODE_ABORTING, MC_TRANSITION_
MODE_NONE, TransParam);
Inst_TON3( true, t#600ms );    //Allow for the move to reach path velocity
before calling MC_GrpStop
Inst_TON3.Q THEN
    MC_MoveCounter := MC_MoveCounter + 1;
    Inst_TON3(false, t#100ms);
END_IF;

2: //Perform a halt on the group
halt_deceleration := 50.0;

pathVelocity := Inst_MC_GrpReadCmdVel2.PathVelocity;

Inst_MC_GrpHalt(TRUE, Group1_ref, halt_deceleration, default_jerk);

Inst_TON3( true, t#200ms );    //Allow for first halt deceleration rate to
be captured on the scope
IF Inst_TON3.Q THEN
    MC_MoveCounter := MC_MoveCounter + 1;
    Inst_TON3(false, t#100ms);
END_IF;

3: // Perform a second halt increasing the deceleration value.
// The second call to MC_GrpHalt will abort the first MC_GrpHalt
halt_deceleration := 200.0;

pathVelocity := Inst_MC_GrpReadCmdVel2.PathVelocity;

Inst_MC_GrpHalt1(TRUE, Group1_ref, halt_deceleration, default_jerk);

Inst_TON3( true, t#200ms );
IF ((Inst_TON3.Q) and (Inst_MC_GrpHalt1.Done Or Inst_MC_GrpHalt1.Error))
THEN
    Inst_TON3(false, t#100ms);
    Inst_MC_MoveLinAbs( FALSE, Group1_ref, PosAbs, default_velocity,
default_acceleration, default_deceleration, default_jerk,
    MC_COORDINATE_SYSTEM_ACS, MC_BUFFER_MODE_ABORTING, MC_
TRANSITION_MODE_NONE, TransParam);
    Inst_MC_GrpHalt1(FALSE, Group1_ref, default_deceleration, default_
jerk);
END_IF;

```

When MC\_GrpHalt is called in the example above, the current move will be aborted and a controlled motion stop will be applied to axes group 'Group1\_ref'. The deceleration value, 'halt\_deceleration' is set to 50.0 user units/sec<sup>2</sup> on the first call and 200.0 user units/second<sup>2</sup> on the second call. Jerk is currently not supported.

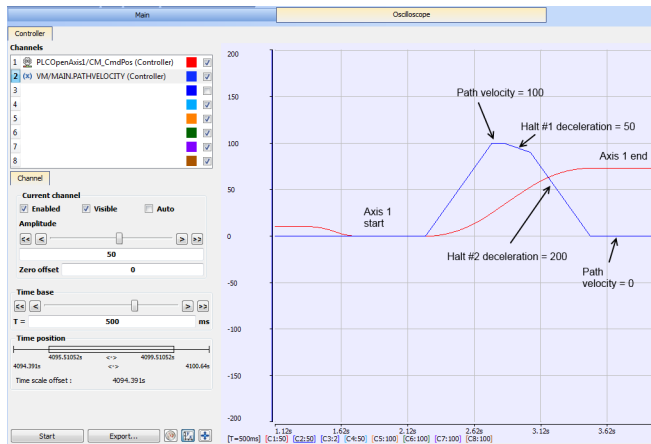
When MC\_GrpHalt is called, the deceleration value from the function block is applied to the path velocity until it reaches zero when the MC\_GrpHalt command is issued during a coordinated motion



move. The group state is "GroupMoving" while the coordinated move is decelerating. The group state goes to "GroupStandBy" once the path velocity reaches zero. Any coordinated moves in the buffer are flushed, and new coordinated moves can be queued up upon completion of the MC\_GrpHalt command.

#### NOTE

This behavior is different than the MC\_GrpStop command. For differences between MC\_GrpStop and MC\_GrpHalt, see "Differences between MC\_GrpHalt and MC\_GrpStop" on page 604.



**Figure 7-12:** Oscilloscope Representation of linear coordinated move with a MC\_GrpHalt command called twice

#### Exceptions:

- The deceleration rate from the MC\_GrpHalt function block is only applied to the absolute and relative coordinated motion moves. Direct coordinated motion moves use the default deceleration value as defined by the AxisRef.
- A MC\_GrpHalt cannot occur if a group is not enabled.
- A MC\_GrpHalt command may be aborted by another MC\_GrpHalt command, a MC\_GrpStop command or a MC\_Move\* command.
- When there are two coordinated motion moves (*Active* and *Next*) in the queue, and the path velocity does not reach zero before the end of the *Active* move, the path velocity will continue to reach zero during the *Next* move.
- The deceleration rate will be increased if there is only one *Active* coordinated move and the path velocity cannot reach zero before the endpoint. This will prevent overshooting the endpoint.
- A MC\_GrpHalt command does not prevent single axis motion from being performed.

#### 12.1.2.6 What Does MC\_GrpStop Do?

MC\_GrpStop (Execute, AxesGroup, Deceleration, Jerk) performs a controlled motion stop of all axes in a coordinated motion group. [Differences between MC\\_GrpHalt and MC\\_GrpStop](#).

##### 12.1.2.6.1 MC\_GrpStop Application Example

The following example demonstrates a linear coordinated move starting point of [0,0] and ending at point of [200, 0]. MC\_GrpStop is called while the linear coordinated move is still moving to the endpoint. As noted in the oscilloscope, the path velocity reaches zero (approximately [100,0]) before the linear coordinated move reaches its end point value.

Linear Move Parameters	MC_GroupStop Parameters
Velocity = 100	stop_velocity = 75
Acceleration = 200	

## Linear Move Parameters MC\_GroupStop Parameters

Deceleration = 200

Jerk = 0

### NOTE

For demonstration purposes it is assumed that the axes and group have been properly setup and configured. Example steps to setup coordinated motion are listed in the section [Create a Linear or Circular Coordinated Motion Application](#).

```

Inst_MC_GrpReadCmdPos( TRUE, Group1_ref, MC_COORDINATE_SYSTEM_ACS,
CmdPositionArray );
Inst_MC_GrpReadCmdVel2( TRUE, Group1_ref, MC_COORDINATE_SYSTEM_ACS,
VelocityArray);
CASE MC_MoveCounter OF

0: // Enable the group
Inst_MC_GrpEnable(TRUE, Group1_ref);
Inst_TON3( true, t#1500ms ); // Allow for turning on the scope
IF ((Inst_TON3.Q) and (Inst_MC_GrpEnable.Done OR Inst_MC_
GrpEnable.Error)) THEN

    Inst_TON3(false, t#1s);
    IF (Inst_MC_GrpEnable.Error) THEN
        PrintMessage( LEVEL_INFO (*DINT*), 'MC_GrpEnable failed -
ErrorID: ' + any_to_string(Inst_MC_GrpEnable.ErrorID));
    END_IF;

    Inst_MC_GrpEnable(FALSE, Group1_ref);
    MC_MoveCounter := MC_MoveCounter + 1;
END_IF;

1: // Perform Linear ABSOLUTE move start (0,0) end (200,0)
PosAbs[0] := 0;
PosAbs[1] := 200;
PosAbs[2] := 0;

pathVelocity := Inst_MC_GrpReadCmdVel2.PathVelocity;

Inst_MC_MoveLinAbs( TRUE, Group1_ref, PosAbs, Velocity, Acceleration,
Deceleration, Jerk,
    MC_COORDINATE_SYSTEM_ACS, MC_BUFFER_MODE_ABORTING, MC_TRANSITION_
MODE_NONE, TransParam);

Inst_TON3( true, t#600ms ); //Allow for the move to reach path velocity
before calling MC_GrpStop
IF Inst_TON3.Q THEN
    MC_MoveCounter := MC_MoveCounter + 1;
    Inst_TON3(false, t#100ms);
END_IF;

2: //Perform a stop on the group
stop_deceleration := 75.0;

pathVelocity := Inst_MC_GrpReadCmdVel2.PathVelocity;

```

```

Inst_MC_GrpStop(TRUE, Group1_ref, stop_deceleration, default_jerk );

Inst_TON3( true, t#200ms );
IF ((Inst_TON3.Q) AND (Inst_MC_GrpStop.Done Or Inst_MC_GrpStop.Error))
THEN
  IF Inst_MC_GrpStop.Error THEN
    PrintMessage( LEVEL_INFO (*DINT*), 'Step '+any_to_string(MC_
MoveCounter)+'MC_GrpStop ERROR. ErrorID('+any_to_string(Inst_MC_
GrpStop.ErrorID)'), Description:'+MC_ErrorDescription(any_to_int(Inst_
MC_GrpStop.ErrorID)));
  END_IF;

  Inst_TON3(false, t#100ms);

  Inst_MC_MoveLinAbs( FALSE, Group1_ref, PosAbs, default_velocity,
default_acceleration, default_deceleration, default_jerk,
MC_COORDINATE_SYSTEM_ACS, MC_BUFFER_MODE_ABORTING, MC_TRANSITION_
MODE_NONE, TransParam);

  Inst_MC_GrpStop(FALSE, Group1_ref, stop_deceleration, default_jerk
);
END_IF;

```

When MC\_GrpStop is called in the example above, the current move will be aborted and a controlled motion stop will be applied to axes group 'Group1\_ref'. The deceleration value, 'stop\_deceleration', is set to 75.0 user units/sec<sup>2</sup> and is applied to the path velocity until it reaches zero. Jerk is currently not supported.

The group state is "GroupStopping" when the MC\_GrpStop function block becomes active. While the axes group is in the GroupStopping state, no other function blocks can perform any motion on the same axes group. Once the path velocity reaches zero the **Done** output is TRUE. The **Execute** input must be set to FALSE before the group state can go to "GroupStandBy". Any coordinated moves in the buffer are flushed, and new coordinated moves can be queued up upon completion of the Stop command

## NOTE

This behavior is different than the MC\_GrpHalt command. For differences between MC\_GrpHalt and MC\_GrpStop, see "Differences between MC\_GrpHalt and MC\_GrpStop" on page 604.

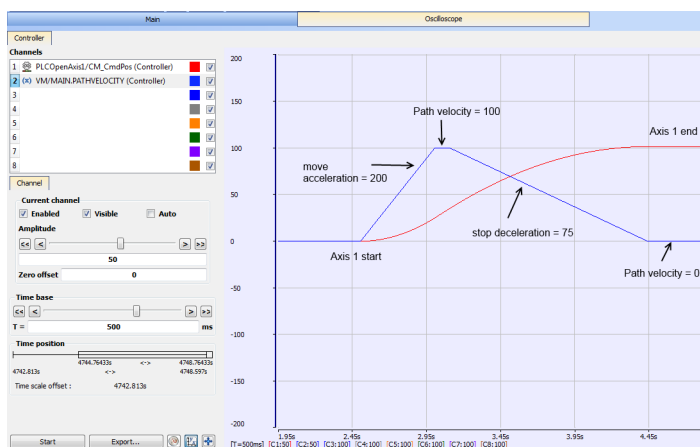


Figure 7-13: Oscilloscope Representation of linear coordinated move with a MC\_GrpStop

Exceptions:

- The deceleration rate from the MC\_GrpStop function block is only applied to the absolute and relative moves. Direct moves use the default deceleration value as defined by the AxisRef.
- MC\_GrpStop cannot occur if a group is not enabled.
- A MC\_GrpStop command *cannot* be aborted by any other commands (while MC\_GrpHalt can be).
- When there are two coordinated motion moves (*Active* and *Next*) in the queue, and the path velocity does not reach zero before the end of the *Active* move, the path velocity will continue to reach zero during the *Next* move.
- The deceleration rate will be increased if there is only one *Active* coordinated move and the path velocity cannot reach zero before the endpoint. This will prevent overshooting the endpoint.
- A MC\_GrpStop command does not prevent single axis motion from being performed.

### 12.1.2.7 Differences between MC\_GrpHalt and MC\_GrpStop

While [MC\\_GrpHalt](#) and [MC\\_GrpStop](#) both perform a controlled motion stop of all axes in an axes group, some differences exist between the operation of the function blocks.

- MC\_GrpStop can not be aborted by any other command. MC\_GrpHalt can be aborted by another MC\_GrpHalt command, a MC\_GrpStop command or a MC\_Move command such as MC\_MoveLinAbs, MC\_MoveCircRel, etc.
- While MC\_GrpStop is active, the group state (see [Group State Diagrams](#)) is 'GroupStopping'. While MC\_GrpHalt is active, the group state is 'GroupMoving'.
- When MC\_GrpHalt is complete, the group state goes to 'GroupStandBy'. When MC\_GrpStop is complete, the DONE output will be true. The EXECUTE input must be set to false before the group state can go to 'GroupStandBy'.

### 12.1.2.8 Handling Axis Errors

Coordinated Motion Error handling is configurable on a per axis group basis. When a PLCopen axis error occurs the [Default Behavior](#) is for all axes in the group stop. This means motion interpolation stops, active and next queues are cleared, and the group state "GroupErrorStop" is enabled. Additionally, the position loop on the drives in the group are opened and the drives are disabled.

The [Optional Behavior](#) when a PLCopen axis error occurs is for only the PLCopen axis that caused the error to have its position loop opened and that drive will be disabled. The application is then expected to control the motion for the remaining axes in the group. The optional feature may be used when stopping all axes is worse than just having one axis stop.

To configure the optional behavior, use the function block [MC\\_GrpWriteBoolPar](#), setting the parameter 'IGNORE\_AXIS\_ESTOP' (1000) to TRUE. To check if the optional behavior has been set, call the function block [MC\\_GrpReadBoolPar](#), reading the state of parameter 'IGNORE\_AXIS\_ESTOP' (1000). If the result is true, the optional behavior is enabled.

The types of PLCopen Axis Errors that affect coordinated motion include:

- Drive errors
- Drive communication errors
- User causes E-stop using [MC\\_EStop](#) on an axis in a group

#### 12.1.2.8.1 Default Behavior

In the default motion error handling case, a user can detect errors using any of the following group function blocks:

- [MC\\_GrpReadStatus](#) — Output GroupErrorStop is true if a group error has occurred.
- [MC\\_GrpReadError](#) — Error output will be true and Error ID output will be set to 12 if the error is due to E-stop.

- The originating Coordinated Motion command ([MC\\_MoveLinAbs](#), [MC\\_MoveCircAbs](#), etc.) will return Error ID 12

To determine which PLCopen axis has generated the error, call [MC\\_ReadAxisErr](#) for each axis in a group. The output ErrorID will be set to 12. In addition, [MC\\_ReadStatus](#) for each axis in the group can be called where the output ErrorStop will be true if in a E-stop condition

Once an error has been detected, the error must first be resolved. [MC\\_GrpReset](#) can be called to reset all PLCopen axis errors. This also resets the group status from GroupErrorStop to GroupStandby. The Done output of [MC\\_GrpReset](#) will be TRUE if all axis errors have been reset. This function block may take up to 3 seconds to reset some error conditions.

#### 12.1.2.8.2 Optional Behavior

##### TIP

This is configurable on a per group basis.

In the optional motion error handling case, the group status GroupErrorStop will not be set when an axis is in error. **The application is responsible for monitoring and handling error conditions.** The remaining axes in the group will continue moving.

Errors can be detected using the following group function block:

- [MC\\_GrpReadBoolPar](#) — Read the result of parameter ID 'AXIS\_ESTOP\_ACTIVE' (1001). If the result is TRUE, an axis error exists. Note that the group function blocks used for the default case of detecting errors ([MC\\_GrpReadStatus](#) and [MC\\_GrpReadError](#)) will not return an error.

To determine which PLCopen axis has generated the error, call [MC\\_ReadAxisErr](#) for each axis in a group. The output ErrorID will be set to 12. In addition, [MC\\_ReadStatus](#) for each axis in the group can be called where the output ErrorStop will be TRUE if in a E-stop condition.

Once an error has been detected, the error must first be resolved. [MC\\_ResetError](#) can then be called on each axis in error. This function only sends a request to the drive to clear any error. The error will not yet be reset when this function returns. [MC\\_ReadStatus](#) will still have to be called to verify that the drive error has been resolved.

#### 12.1.2.8.3 Recovery of the System State After an Axis Error

Recovery from axis errors is more complex with the addition of Coordinated Axes Groups (PLCopen Part 4) to KAS Runtime, This is due to additions in the operation of PLCopen Motion and Coordinated motion. This includes:

- Addition of the Coordinated motion Boolean variable that allows the Runtime to IGNORE\_AXIS\_ESTOP(ID number 1000).
- Addition of the Coordinated motion Boolean variable that allows the Application to read the state of the Runtime Axis errors: AXIS\_ESTOP\_ACTIVE (ID number 1001).
- Addition of [MC\\_GrpReadBoolPar](#)
- Addition of [MC\\_GrpWriteBoolPar](#)

When the application needs to take control of the default behavior where all axes in a group stop when any of them detect an Estop condition, and the Estop condition does not in itself stop all axes (Loss of EtherCAT communication), the application can set the Group Boolean parameter IGNORE\_AXIS\_ESTOP (ID := 1000) using [MC\\_GrpWriteBoolPar\(\)](#). When set to true, the runtime will try to keep axes in the same group as a faulting axis still able to be commanded. The intent is to not give up control of the commanded motion, but allow the application to substitute error handling motion that allows a group of axes to stop in a controlled manner. For example: command the remaining axes to a Home position, or a relative move away from the position of the faulted axis.

Once the IGNORE\_AXIS\_ESTOP parameter is set TRUE, the Group of axes will not enter the ERRORSTOP state when an axis encounters an ESTOP condition but remain in STANDBY or MOVING state. Instead, the AXIS\_ESTOP\_ACTIVE (ID := 1001) parameter can be monitored to trigger a response that will abort the current operation and take control of the axes that can still be

controlled. To determine which axis has faulted, the MC\_ReadStatus() FB can be used to monitor each axis and select the appropriate single axis or Coordinated motion commands to execute to take control of the motion and recover from the error.

When the error recovery is completed and the faulting axis restored to normal operation, in order to restart Coordinated Motion, an MC\_GrpSetPos() command must be issued to tell the Coordinated Motion Engine where the axes in the faulted Axes Group are then at, following restoration of the single axis fault(s). This command must be issued in order to reset the faulted axis.



Alternatively, if the MC\_GrpWriteBoolPar() function is issued to set the IGNORE\_AXIS\_ESTOP parameter to FALSE, the Group will enter the ERRORSTOP state if an Axis error has not been reset, and the default behavior and default usage of MC\_GrpResetError() function can be used to clear faults instead of the MC\_GrpSetPos() function.

## 12.2 Motion Techniques

This chapter explains advanced concepts and procedures related to **motion techniques** that are possible with the KAS IDE.

### 12.2.1 PLC Online Change

#### ! IMPORTANT

You have to save  and compile  your project before doing an online change.

This section provides a detailed description of the PLC Online Change functionality. See [Using PLC Online Change](#) for an overview of using this functionality.

#### 12.2.1.1 What is Online Change

Online Change enables you to update your PLC application on the fly, while it is running on the controller. You do not need to stop the controller, download the new code and start again. You only need to modify, recompile and download the new code as shown in the figure below; and then ask the controller to switch the execution to the new application.

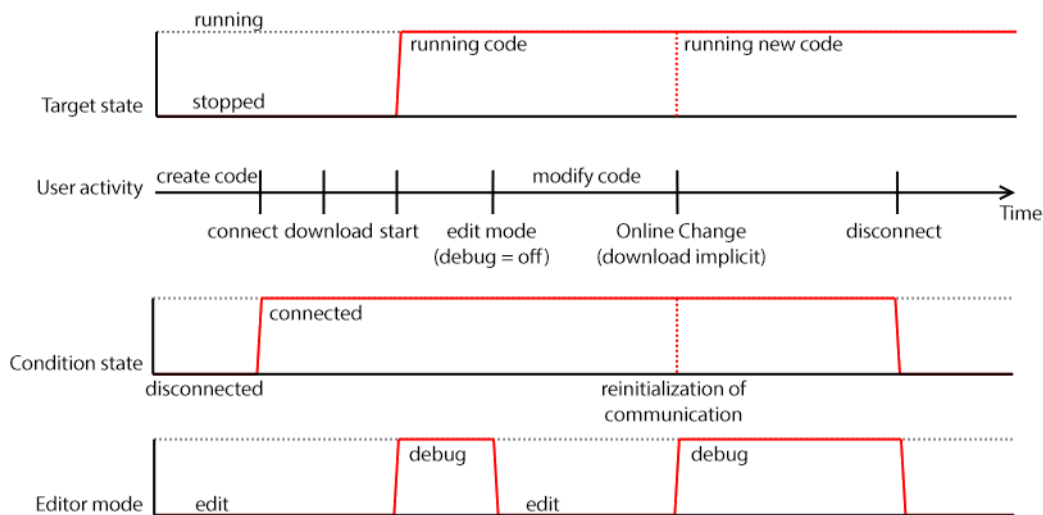


Figure 7-14: Online Change - Process Diagram

#### 👉 TIP

This capability applies only to PLC code. This is not supported in the PipeNetwork editor, the PLCOpen axis, or any other part of the system.

#### NOTE

Depending on the PLC code size, the time to perform the Online Change operation can take more than one [cycle](#). In that case, you can miss one PLC cycle before the changeover becomes effective.

This duration is also displayed in the [Log](#) window with an **INFO** level message as follows: **Online Change done in X µs**. For more details, click [here](#). The INFO measurement corresponds to the duration for the code hotswap. The download and loading of new code in memory is not taken into account in this measurement because they occur when the previous code is still running.

This feature is used in the following situations:

- **Development phase:** you can modify the application and apply these modifications incrementally without stopping the controller

- **Update in production:** you can update the running motion application (for instance with a bug fix release) without stopping the whole production chain

When Online Change is enabled, you can perform the following kinds of changes on the fly:

- Rename a program
- Change the code of a program
- Change the condition of an SFC transition or the actions of an SFC step
- Create, rename or delete global and local variables
- Create, rename or delete global and local function block instances
- Rename [Retain Variables](#)

The following **are not allowed**:

- Create or delete a program
- Add or remove Custom Libraries
- Add new PLC axes, change or delete PLC axes
- Add new cam profiles, change or delete existing cam profiles
- Add new Pipe Network blocks, change or delete existing Pipe Network blocks
- Add or remove variables in a Structure
- Create a new Structure or a new UDFB
- Change the local parameters and variables of a UDFB
- Change the type or dimension (or string length) of a variable or function block instance
- Create or delete Retain variables (their position in the runtime cannot be re-allocated)
- Change SFC charts: you cannot add or remove steps in the First Level of an SFC chart (but you can modify existing steps)
- Change the set of Input/Output or any modification that leads to an update in the [EtherCAT Motion Bus configuration](#)
- Pulse (P or N) contacts and coils (edge detection)

#### TIP

Using Pulse contacts in FFLD does not give any error, but the behavior of the contact during the switch is not always safe (for more details, as well as workaround, [Pulse Limitations with Online Change](#)).

- The WAIT and WAIT\_TIME instructions must not be used

#### IMPORTANT

The Online Change and Revert functions will fail while executing a WAIT.

- Loops in FBD with no declared variable linked. In this case, you need to explicitly insert a variable in the loop.

#### IMPORTANT

When Online Change is active and [custom libraries](#) are being used, some errors can occur during the compilation. This happens if you open your project on another PC, or under a different user account in Windows. To fix this limitation:

1. Deactivate the Online Change
2. Save and then reopen the project
3. Turn the Online Change back on if desired

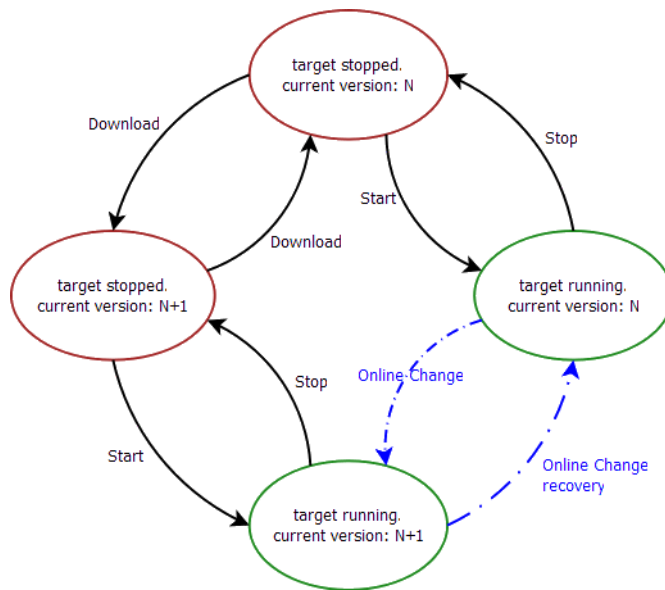
#### NOTE

Your new application can contain more variables than the previous one. A memory with sufficient pre-allocated space is defined for the eventual new variables. If you exceed this limit, a warning message is displayed.

For limitation about breakpoint with Online Change, [see "About Online Change" on page 221](#).

*About the states and transitions*





**Figure 7-15:** Online Change - States and Transitions

### 12.2.1.2 How to Activate Online Change

To allow Online Change, you need to open the PLC options and set the relevant parameters.

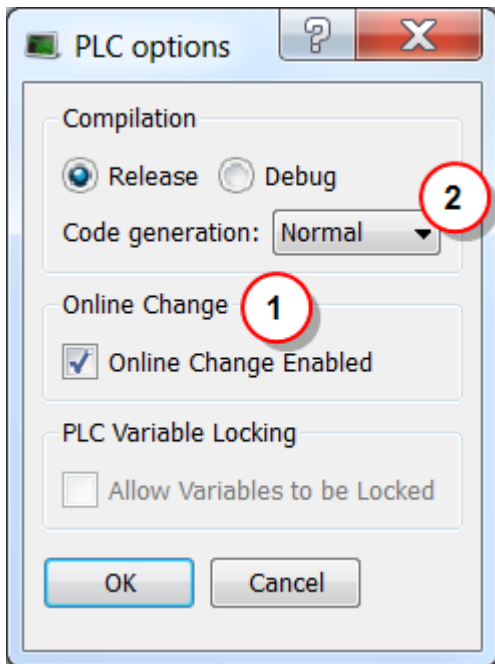


Figure 7-16: PLC Options - Online Change Enable

#### Set the parameters

This box allows you to enable or disable the **Online Change** feature (see call out **1**).

#### ⚠ IMPORTANT

If you deactivate the Online Change, the next PLC application generated is no more compatible for an online change, even if you re-activated the online change before the compilation.

As a result, you can only apply an Online Change to a running application under the two following conditions:

- The Online Change was already activated
- You have never deactivated the Online Change between the compilation of the running application and the compilation of the new application


Note: Check the [Controller Log](#) window for any errors that occur.

You also need to ensure that you have selected **Normal** code **2** as Online Change is not possible with **Optimized** code. Note that when **Optimized** code is selected, then Online Change is always deactivated.


Then you can compile your application, which now allows future changes on the fly.

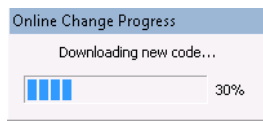
#### Switch to Edit mode

When you start the application, the Debug mode is automatically activated: you can see the values changing in the editors and the Dictionary, showing what is happening on the controller. In this mode the editor is read-only, so you are not able to modify the code.

To edit your code, go out of the Debug mode and enter the Edit mode by clicking the  button in the [Main Toolbar](#).

#### Perform the Online Change

When your new code has compiled correctly, you can perform the Online Change. To do so, click the  button. When you click this button, the KAS IDE opens a window showing the execution of current actions (download, activation of new code).



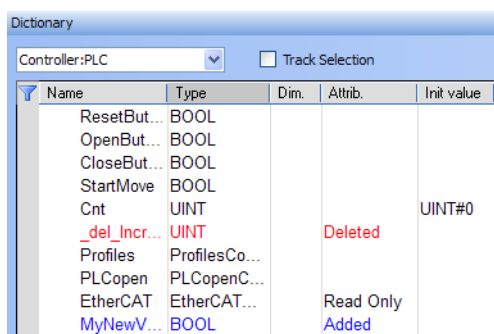
**Figure 7-17:** Online Change - Updating Controller Version

Once the Online Change is applied, the result is displayed in the window and you can click OK to acknowledge the operation and do a Warm start.

### Dictionary behavior

When the Online Change is enabled, the dictionary shows:

- new variables in blue
- deleted variables in red



**Figure 7-18:** Online Change - Dictionary

#### TIP

The deleted variables can be for new variables.

### 12.2.1.3 What is the Revert button

The **Revert** button is for security purposes. It allows you, after an Online Change, to revert your change quickly and go back to the previous application. That means switching the execution of the controller to the **P-code** that was running before the last Online Change (note that the source code in the KAS IDE is not replaced). The WAIT and WAIT\_TIME instructions can not be used with Revert.

After the Revert, the KAS IDE automatically goes back to Edit mode and is disconnected from the controller. The user must manually reconnect to the controller.

#### **NOTE**

You can go back to the previous version only when the Online Change feature is activated and while the controller is not stopped.

#### IMPORTANT

After a revert operation, the Online Change feature is **deactivated**.

The Revert button is active when you are connected and the controller is running.

Revert is not possible:

- if you did not perform an Online Change
- if the controller has been restarted since the previous Online Change

- after another Revert
- during a WAIT

#### 12.2.1.4 Difference between Local and Controller versions

When you restore a project with the Revert feature after an Online Change, KAS provides a tool to show the differences between two versions of the project. This tool can help you in checking all modifications before the next Online Change. It is also a useful tool when you want to compare your code with the last version after a Revert.

For more details, refer to [Compare PLC Programs](#).

#### 12.2.1.5 Pulse Limitations with Online Change

At the first cycle, the pulse evaluation is ignored, and the memory is updated. This memory enables the pulse evaluation from the second cycle.

When we apply the Online Change between  $t_0$  and  $t_1$ , the cases where this method is not correct are the two following:

- When we want to detect a **falling** edge:

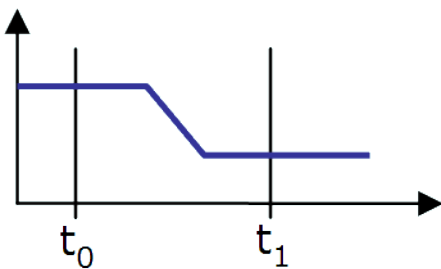


Figure 7-19: Pulse Limitations with Falling Edge

- When we want to detect a **rising** edge:

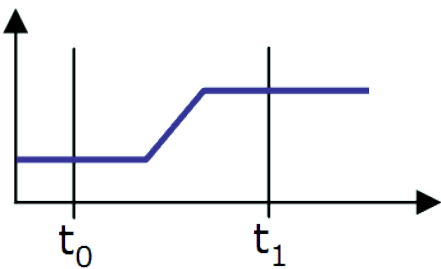


Figure 7-20: Pulse Limitations with Rising Edge

#### **TIP**

If you want to avoid this limitation, you must use declared instances of R\_TRIG and F\_TRIG function blocks.


#### **NOTE**

This limitation is temporary and is going to be fixed in a future release.

## 12.2.2 Using PLC Online Change

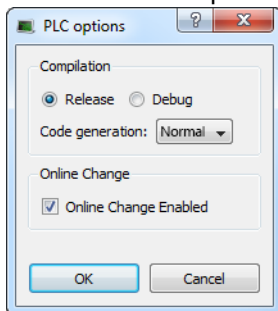
This section provides an overview of how to use Online Change. See [PLC Online Change](#) for descriptions of the functionality.





### 12.2.2.1 Set up an application

1. Create a new PLC application.
2. Connect to a controller and scan for EtherCAT devices.
3. Add logic and function blocks to the application.
4. Compile the project ,



### 12.2.2.2 Enable Online Change

1. Select the PLC Options button from the tool bar.



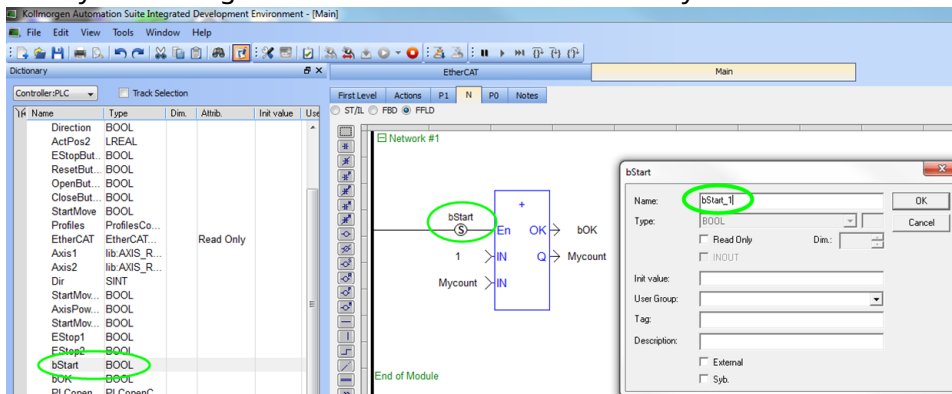
2. Enable Online Change.
3. Compile the project , connect  and download  the application to the device.
4. Start executing the application. 



### 12.2.2.3 Using Online Change

1. Enable the Toggle Edit/Debug mode button in the tool bar.  → . Note that application variables will not be updated, even though the application is running.

Local: UnnamedProject:3 Controller: UnnamedProject:3 Drives inactive **Running** **Connected**

2. Modify a local or global variable name in the Dictionary.




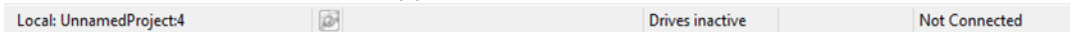
3. Compile  and click the PLC Online Change download button  from the tool bar to download the changes to the controller. Note that the status bar has changed from "Running" to "Paused" during the download, and back to "Running" once the download has finished. The applications variables should be updating.


**NOTE**

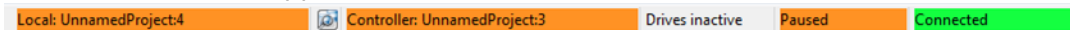
If Online Change is disabled while the application is running, even temporarily, downloading will fail.

**12.2.2.4 Revert Online Change**

1. Click the PLC Online Change Revert button  from the tool bar.
2. A message should be displayed stating that revert was successful.
3. Note the reverted state of the application in the IDE.



4. Connect to the device. 
5. Note the state of the application and the device.



6. From this state, Online Change can only be used again after ...:
  1. Stopping the application
  2. Making changes
  3. Recompile
  4. Download the application to the controller

**12.2.3 What Are Fast Inputs?**

Fast inputs allow a high-speed application to get position information about the occurrence of an external event at a higher resolution than the cycle time. Thanks to the precise timing of external events, an application can improve its control algorithm, resulting in higher operating performance. Fast (or high-speed) inputs are digital inputs of a drive that are configured to latch the time at which they are triggered.

A position or time capture can be triggered either by the positive (rising) edge or by the negative (falling) edge of the digital input. The [AKD Drive Configuration](#) has two capture engines which can be freely linked to any input. These high speed inputs can be used in application which, when triggered, caused a drive position to be captured and reported back to the controller.

**NOTE**

Only digital inputs 1 and 2 can be used as Fast Inputs.

**TIP**

When using S300 or S700, Fast input has to be enabled by setting the drive keywords IN1MODE to 26 and IN2MODE to 26. This can be achieved using DriveGUI configuration tool.

See the following sections for fast input information specific to the motion engines.

**12.2.3.1 Fast Inputs with Pipe Network Motion**

**TIP**

For the Axis and Trigger Pipe Network blocks, refer to [MLAxisCfgFastIn](#) which arms a Fast Input and returns the latched timestamp when the Fast Input event occurs.

Registration Operation Related FB (Trigger FB)	Homing operation related FB (Axis FB)
<a href="#">MLTrigSetEdge</a>	<a href="#">MLAxisRstFastIn</a>
<a href="#">MLTrigIsTriggered</a>	<a href="#">MLAxisIsTriggered</a>
<a href="#">MLTrigReadPos</a>	<a href="#">MLAxisCfgFastIn</a>

When the input is triggered, the timestamp is latched. With EtherCAT, the timestamp sent to the KAS IDE via the [MLAxisTimeStamp](#) or [MLTrigReadTime](#) function blocks is based on the distributed

clock that manages the reference clock (for more details on this concept, see ["Distributed Clock \(Synchronization\)" on page 439](#)). The KAS IDE converts this timestamp into a relative offset inside the cycle.

It is also possible to configure a Fast Input to latch the motor position instead of latching the time (see ["AKD Drive Configuration" on page 460](#)). However, when working with KAS, time latching is more useful, because the positions of all the drives in the application can then be interpolated by means of the [Trigger](#) block with the [MLTrigReadPos](#) function block.

### 12.2.3.2 Fast Inputs with PLCopen Motion

For PLCopen, refer to [MC\\_TouchProbe](#), which arms a Fast Input and returns the latched position when the Fast Input event occurs.

Other PLCopen function blocks which arm a Fast Input and return the latched position to the function block when the event occurs are shown below.

Registration operation	Homing operation
<a href="#">MC_MachRegist</a>	<a href="#">MC_Reference</a>
<a href="#">MC_MarkRegist</a>	<a href="#">MCFB_StepAbsSwitchFastInput</a>
	<a href="#">MCFB_StepLimitSwitchFastInput</a>

Table 7-2: PLCopen function blocks which arm Fast Inputs and return the latched position.

### 12.2.4 Torque Feed-forward

The torque feed-forward tells the controller what forces is required to move the axis in an arbitrary trajectory.

Here are the major features of torque feed-forward:

- Torque feed-forward results in virtually instantaneous response of the system.
- Feedback control loops (using PID loop or similar) take a finite amount of time before reacting.
- Torque feed-forward relies on an imperfect model of the system. This means that the feed-forwards need help from the feedback control loop in order to get accurate motion.
- Torque feed-forward can make the bulk of the move very quickly, while the feedback control loops correct the small errors that remain. As a result, a faster settling time can be achieved than if torque feed-forward was not used.
- There is a common misconception that torque feed-forward is similar to control loops and result in instability. Torque feed-forward is open loop, so it cannot suffer from closed loop instability.
- Torque feed-forward is typically less sensitive to being misadjusted than closed loop parameters.
- Feedback control systems can be excited into instability by grossly misadjusted torque feed-forward. However, the amount of misadjustment in the torque feed-forward necessary to cause such instability is very rare.

## 12.2.5 PLCopen Homing

### 12.2.5.1 PLCopen Homing Description

The homing features provided in PLCopen create tools for homing of PLCopen axes. Homing may be performed utilizing the MC\_Reference function block, utilizing Custom Homing Library UDFB's or by writing your own homing cycles.

- Utilizing MC\_Reference  
The application specifies a position for an axis to be assigned to a reference position, then invokes the MC\_Reference function block to generate motion to move the axis to the reference location. The AKD capture engine (previously set up by the application via SDO commands) captures the position of the reference location. Based on the desired reference position and the captured actual position, the coordinate system is shifted to correlate the desired reference position to this location.
- Writing your own homing cycles  
UDFBs can be written to provide specific "canned" homing cycles based on feedback type, and desired homing sequences such as homing off of limit switches, encoder markers, homing to "zero" or null positions etc. by proper configuration of the AKD capture engine, the MC\_Reference and MC\_SetpositionMC\_SetPos function blocks.
- Utilizing Custom Libraries  
A library already contains a set of homing UDFB. Contact the [Support](#) for more information. To add the library to your project, refer to [Create and Use Custom Libraries](#)

### 12.2.5.2 PLCopen Homing Methods

The following common homing methods (among others) can be performed in PLCopen. This section details the setting of the ADK parameters and the PLCopen function blocks to accomplish these methods.

PLCopen does not limit you to these methods, as the capture engine is very configurable.

#### 12.2.5.2.1 Home using Current Position

Homing using the current position is simply accomplished using the MC\_SetPositionMC\_SetPos function block. Using this function block, the current position can be set to any value.

#### 12.2.5.2.2 Find Input

Homing using a drive input is accomplished by configuring the [AKD capture engine](#), and then using the MC\_Reference function block. The following capture engine parameters need to be configured, along with the following input parameters in the MC\_Reference.

- Capture Event has to be set to ignore preconditions (0)
- Capture edge – capture edge is programmed in the MC\_Referece block
- Capture Trigger must be set to the desired drive input (0-6)
- Capture mode must be set to capture position (0)
- Capture preselect is not used
- Capture Precondition edge is not used
- MC\_Reference inputs:  
Trigger\_Ref.InputID must be set to 0 or 1 to select which AKD capture engine to use  
Trigger\_Ref.Direction must be set to Rising (1) or Falling (2) to select Capture Edge  
Trigger\_Ref.Trigid is not required.  
Position input must be programmed to the desired position at the switch.  
Option input must be programmed to 0 for "use latched position".



### 12.2.5.2.3 Find Input then find Zero Angle

Homing using a drive input along with the zero angle is similar to "Find Input" except the position is defined at the zero angle of the feedback device, rather than the switch location. It is typically used for resolver feedback.

- Capture Event must be set to ignore preconditions (0)
- Capture edge – capture edge is programmed in the MC\_Referece block
- Capture Trigger must be set to the desired drive input (0-6)
- Capture preselect is not used
- Capture Precondition edge is not used
- Capture mode must be set to capture position (0)
- MC\_Reference inputs:  
 Trigger\_Ref.InputID must be set to 0 or 1 to select which AKD capture engine to use.  
 Trigger\_Ref.Direction must be set to Rising (1) or Falling (2) to select switch capture edge.  
 Trigger\_Ref.Trigid is not required.  
 Position input must be programmed to the desired position at the null closest to the switch.  
 Option input must be programmed to identify the number of poles the resolver has.

### 12.2.5.2.4 Find Input then find Index

Homing using a drive input along with the index is similar to "Find Input" except the position is defined at the index pulse of the feedback device, rather than the switch location. It is typically used for incremental encoder feedback. To accomplish this, a precondition is used in the capture engine. Specifically, the input is the precondition, and the index is the event. The reference method looks for the switch first, and then the index pulse.

- Capture Event must be set to the desired switch operation. Typically set to 1 to require the edge of the switch. Set to 2 or 3 if the state of the switch is required.
- Capture preselect must be set to the desired drive input (0-7)
- Capture edge – capture edge of index pulse is programmed in the MC\_Referece block
- Capture Trigger must be set to the desired index input (10 = primary index,11 = tertiary index)
- Capture mode must be set to capture position (0)
- MC\_Reference inputs:  
 Trigger\_Ref.InputID must be set to 0 or 1 to select which AKD capture engine to use  
 Trigger\_Ref.Direction must be set to Rising (1) or Falling (2) to select Capture Edge  
 Trigger\_Ref.Trigid is not required.  
 Position input must be programmed to the desired position at the index pulse.  
 Option input must be programmed to 0 for "use latched position".

### 12.2.5.2.5 Find Index

Homing using a drive index pulse is accomplished by configuring the AKD capture engine, and then using the MC\_Reference function block. The following capture engine parameters need to be configured, along with the following input parameters in the MC\_Reference.

- Capture Event must be set to ignore preconditions (0)
- Capture edge – capture edge is programmed in the MC\_Referece block
- Capture Trigger must be set to the desired index input (10 = primary index,11 = tertiary index)
- Capture mode must be set to capture position (0)
- Capture preselect is not used
- Capture Precondition edge is not used
- MC\_Reference inputs:  
 Trigger\_Ref.InputID must be set to 0 or 1 to select which AKD capture engine to use  
 Trigger\_Ref.Direction must be set to Rising (1) or Falling (2) to select Capture Edge  
 Trigger\_Ref.Trigid is not required.  
 Position input must be programmed to the desired position at the index pulse.  
 Option input must be programmed to 0 for use latched position.



### 12.2.5.3 AKD Capture Engine Configuration

The AKD capture engine provides a broad range of capabilities for configuration of the capture event(s). Furthermore, it is capable of configuring preconditions to allow the application programmer to specify sequential events or conditions that must be met before the capture event can be triggered. The capture Engine in the AKD is configured with SDO #0x3460 (subindexes 1 to 10). The AKD supports two capture engines (0 and 1); the application programmer must configure the desired engine.

Sub Index #	Function
1	Trigger for capture engine 0
2	Trigger for capture engine 1
3	Mode for capture engine 0
4	Mode for capture engine 1
5	Capture Event for capture engine 0
6	Capture Event for capture engine 1
7	Precondition edge for capture engine 0
8	Precondition edge for capture engine 1
9	Preselect for capture engine 0
10	Preselect for capture engine 1

The following section details the configuration parameters for the ADK capture engines.

- Capture event (SDO object #0x3460 subindex engine 0 = 5/engine 1 = 6 )
  - 0 = ignore preconditions
  - 1 = trigger edge after the precondition edge
  - 2 = trigger edge while precondition = 1
  - 3 = trigger edge while precondition = 0
- Capture edge – capture edge is programmed in the MC\_Reference function block.
- Capture Trigger (SDO object #0x3460 subindex 1/2 )
  - 0 = general input 1
  - 1 = general input 2
  - ...
  - 6 = general input 7
  - 7 = rs485 input 1
  - 8 = rs485 input 2
  - 9 = rs485 input 3
  - 10 = primary index
  - 11 = tertiary index

For more details, refer to CAP0.PRESELECT, CAP1.PRESELECT section.

- Capture precondition edge (SDO object #0x3460, subindex 7/8 )
  - 0 = reserved
  - 1 = precondition with rising edge
  - 2 = precondition with falling edge
  - 3 = precondition with rising and falling edges
- Capture preselect (SDO object #0x3460 subindex 9/10 )
  - 0 = general input 1
  - 1 = general input 2
  - ...
  - 6 = general input 7

- 7 = rs485 input 1
- 8 = rs485 input 2
- 9 = rs485 input 3
- 10 = primary index
- 11 = tertiary index
  
- Capture mode (SDO object #0x3460 subindex 3/4 )
  - 0 = capture position
  - 1 = capture internal time
  - 2 = capture EtherCAT distributed time (DCT)
  - 3 = capture zero angle position

## 12.2.6 Pipe Network Homing

UDFBs can be written to provide specific "canned" homing cycles based on feedback type. Contact the [Support](#) for more information.

### 12.2.6.1 Fast Homing Example with the Pipe Network Motion Engine Axis Pipe Block

This use case explains how to use the motion library functions of the axes when you want to detect the positive edge of the first Fast Input in the drive, and read its associated timestamp.

[MLAxisIsTriggered](#) can be used to accurately perform a homing procedure in a user-defined procedure.

A Setup sequence example.

1. [MLAxisCfgFastIn](#)(PipeNetwork.AXIS1, MLFI\_FIRST, MLFI\_RISING\_EDGE)
  - Configure Fast Input 0 of AXIS1 to be triggered on the positive edge.

#### NOTE

Since tMLFI\_FIRST and MLFI\_SECOND correspond to the physical Fast Inputs 1 and 2. Therefore the drive must be configured in order to link fast input 1 with engine 0, and fast input 2 with engine 1.

- The first argument indicates the Axis pipe block in the Pipe Network that represents the drive to be configured
  - The second argument identifies which of the two Fast Inputs of the drive is configured (can be 0 or 1)
  - The third argument can indicate detection of positive edge when set to 1 and detection of negative edge when set to 2  
Note that if set to 0, Fast Input is disabled
2. [MLAxisIsTriggered](#)(PipeNetwork.AXIS1, 0, 1)
    - This function returns true if Fast Input 0 of AXIS1 has been triggered on the positive edge.
    - The meaning of the arguments is the same as in [MLAxisCfgFastIn](#)
  3. [MLAxisTimeStamp](#)(PipeNetwork.AXIS1, 0, 1)
    - This function returns the time in microseconds when the Fast Input was triggered on the positive edge  
This time is relative to the start of the drive cycle time and its value is explained in the section [How To Interpret a Timestamp](#).
    - The meaning of the arguments is the same as in [MLAxisCfgFastIn](#)
  4. [MLAxisRstFastIn](#)(PipeNetwork.AXIS1, MLFI\_FIRST)
    - This function resets the Fast Input 0 of AXIS1. The reset keeps the configuration of the Fast Input, but it rearms it so it can be triggered again
    - The meaning of the first two arguments is the same as in [MLAxisCfgFastIn](#)
  5. Follow-up Motion

The following is typical code used in a homing procedure.

```
CASE StepCounter OF
0:
  MLAxisRstFastIn (PipeNetwork.Feeder,MLFI_FIRST);
  MLAxisMoveVel (PipeNetwork.Feeder,250.0); //Jog Feeder Axis to search for
  sensor input
StepCounter := 1;
```

```

1:

IF MLAxisIsTriggered(PipeNetwork.Feeder,MLFI_FIRST,MLFI_RISING_EDGE)
  THEN MLAxisAbs(PipeNetwork.Feeder,MLAxisCmdPos(PipeNetwork.Feeder));
  //Stop motion when sensor is reached
  StepCounter := 2;
END_IF;

2:

IF MLAxisGenIsRdy(PipeNetwork.Feeder)
  THEN MLAxisWritePos(PipeNetwork.Feeder,0); //Set Feeder Axis position
  to zero
  StepCounter := 3;
END_IF;

```

Fast Homing based on the drive's high speed capture mechanism is also supported by the following Kollmorgen KUDFBs in the IDE Function Block Library: [MLFB\\_HomeFindHomeFastInput](#), [MLFB\\_HomeFindHomeFastInputModulo](#), [MLFB\\_HomeFindLimitFastInput](#), and [MLFB\\_HomeFindLimitFastInputModulo](#).

The Axis Position can also be derived from the Time Stamp by with the following code:

```

TimeStamp := any_to_lreal(MLAxisTimeStamp( iAxisID, any_to_dint
(ibFastInputNumber), 1+any_to_dint(ibHomeSwitchMode) ))/1000000;
CalculatedTriggerPosition := (CurrentVelocity-
PreviousVelocity)/2/iCycleTime*TimeStamp*TimeStamp+PreviousVelocity*TimeS
tamp+PreviousPosition;

```

## 12.2.7 PLCopen Registration

Registration is a technique used to maintain the positional accuracy in repetitive processes. It uses a Fast Input switch, typically a photo eye, to measure product position and adjust the axis (or axes) to compensate for variations. There are two basic forms of registration: single-axis registration and master/slave registration.

### 12.2.7.1 Deciding Which Function Blocks to Use for Registration

There are two good methods by which to set up Registration, each using a different set of function blocks. Method 1 is easy to set up, thus we recommend trying it first.

	Method 1: Using MC_TouchProbe	Method 2: Using MC_MarkRegist or MC_MachRegist
<b>Key Function Blocks</b>	<a href="#">MC_TouchProbe</a> , <a href="#">Trigger_Ref</a> structure, and <a href="#">MC_MoveSuperimp</a>	<a href="#">MC_MarkRegist</a> or <a href="#">MC_MachRegist</a> , <a href="#">Trigger_Ref</a> structure, <a href="#">MC_MoveSuperimp</a> , and <a href="#">MC_ReadParam</a>
<b>Details</b>	<p>This method incorporates:</p> <ul style="list-style-type: none"> <li>MC_TouchProbe to capture the drive position using the high speed capture engine in the drive.</li> <li>The Trigger_Ref structure is used to set up the drive's capture engine.</li> <li>MC_TouchProbe outputs a signal when a capture has occurred and also the position captured.</li> </ul> <p>Additional code is created to make motion corrections (typically with MC_TouchProbe) based on the captured position value compared to a reference position or distance.</p>	<p>This method incorporates:</p> <ul style="list-style-type: none"> <li>MC_MarkRegist or MC_MachRegist, which in addition to capturing a position from a drive automatically calculates the position correction and can either make the offset move (linear acceleration / deceleration) or outputs the calculated position correction to be done by another function block (typically with MC_TouchProbe).</li> <li>The Trigger_Ref structure is used to set up the drive's capture engine.</li> </ul>
<b>Advantages</b>	<p>Basic implementation.</p> <ul style="list-style-type: none"> <li>Fast to develop due to having fewer input values to determine .</li> <li>Simple function block functionality leads to less debug time.</li> </ul>	<p>Incorporates additional functionality including:</p> <ul style="list-style-type: none"> <li>Automatically calculates the number of good and bad registration marks.</li> <li>The option to only make corrective moves based on a good registration mark.</li> <li>Automatically make the corrective motion (linear acceleration / deceleration only).</li> <li>Position scaling differences are internally calculated if the axis in which the position is captured and the axis in which the motion compensation is made are different.</li> </ul>
<b>Example</b>	<a href="https://www.kollmorgen.com/en-us/developer-network/registration-plcopen-example-project/">https://www.kollmorgen.com/en-us/developer-network/registration-plcopen-example-project/</a>	<a href="https://www.kollmorgen.com/en-us/developer-network/registration-plcopen-example-project/">https://www.kollmorgen.com/en-us/developer-network/registration-plcopen-example-project/</a>

### 12.2.7.2 Single-Axis Registration

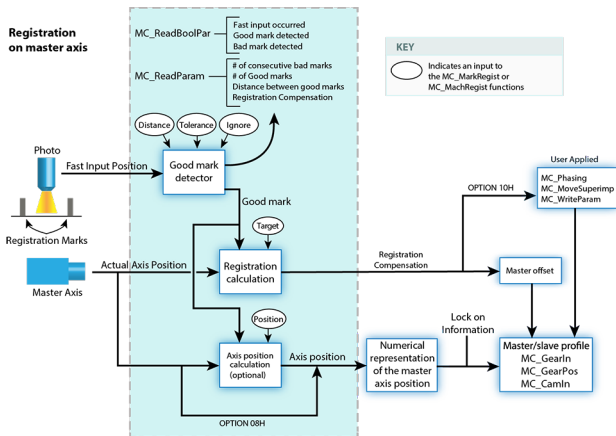
Single-axis registration is performed on an axis running a discrete move such as MC\_MoveAbsolute or MC\_MoveRelative. When the Fast Input latches the position of the product, the axis position is reset, typically to zero. This resets the axis's coordinates for each product to accommodate for variations in the distance between products and keep the process synchronized to the product over many repetitions.

### 12.2.7.3 Master/Slave Registration

Master/slave registration is performed on an axis running a master/slave move such as MC\_GearIn or MC\_CamIn. It can be performed by tracking the position of the master axis (Master Registration) or tracking the position of the slave axis (Slave Registration) or both. This type of registration adjusts the positional relationship between the master and slave axes to accommodate for variations in the distance between products and keep the process synchronized to the product over many repetitions.

#### 12.2.7.3.1 Master Registration

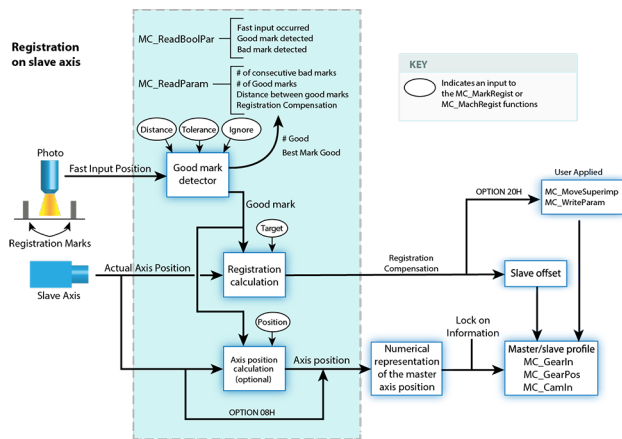
Master registration is performed by having the Fast Input switch trigger on a registration mark controlled by the master axis. When the Fast Input latches the position of the master axis at this mark, the distance between this position and the position of the previous mark is compared to an expected distance. This difference is added to the slave axis's master offset to adjust the position of the slave axis with respect to the position of the master.



#### 12.2.7.3.2 Slave Registration

Slave registration is performed by having the Fast Input switch trigger on a registration mark controlled by the slave axis. When the Fast Input latches the position of the slave axis at this mark, the distance between this position and the position of the previous mark is compared to an expected distance. This difference is added to the slave axis's slave offset to adjust the position of the slave axis with respect to the position of the master.





The figure [Registration](#) below shows an example of a printing application using registration. The axis controlling the web is the master and the axis controlling the print head is the slave. When the photo eye detects a registration mark on the web, the master position is latched. The amount of registration compensation required is calculated by comparing the actual distance between the marks to the expected distance. That difference is added to the slave axis's master offset. This adjusts the positional relationship between the web and the print head so that each print on the web is placed accurately.

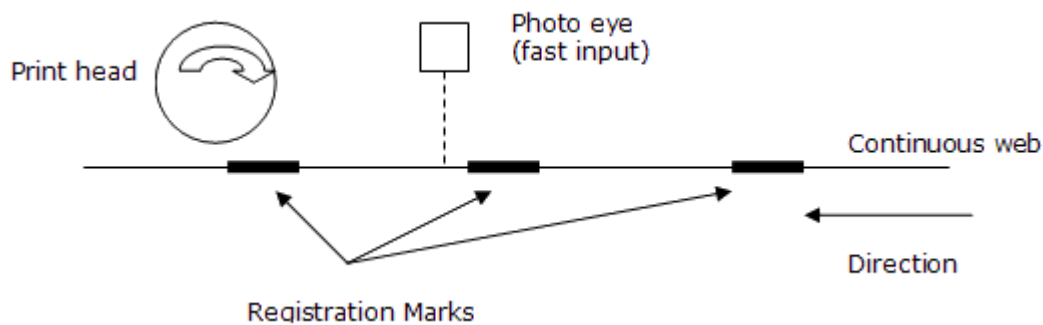


Figure 7-21: Registration

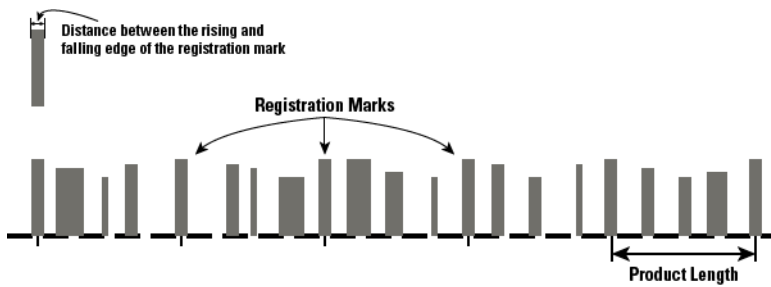
#### 12.2.7.4 Registration Application Guide

In many closed-loop servo systems it is often necessary to maintain synchronization and accurate positioning repeatedly throughout a process. This can be difficult when the product or process itself is inconsistent. Using registration helps you to overcome this difficulty.

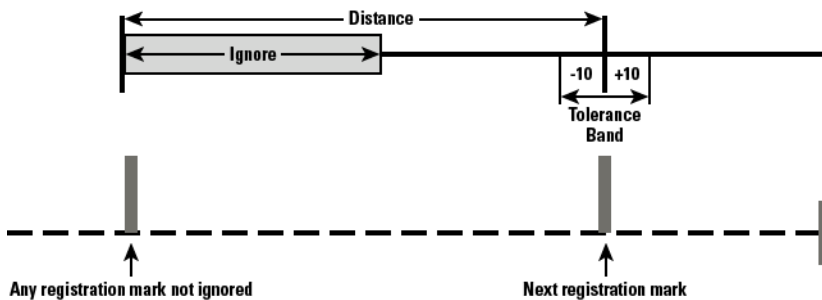
Many factors can contribute to inconsistency. Some examples are:

- Working with non-rigid material which may stretch or shrink during processing.
- Working with the mechanics of a system where the revolution of a feedback device may give you, for example, 5975 counts on one revolution and 5974 on the next.
- Unevenly spaced products on a web or belt.
- Materials which are sensitive to temperature, humidity, pressure, etc.

To overcome the various product and process inconsistencies registration capabilities may be required and can be applied on any servo or digitizing axis, and with any move type. Typically in these applications, sensors are used to detect the position of the product. With non-rigid materials, which may stretch or shrink or are unevenly spaced, a photo eye can detect registration marks on the material. With rigid products or processes a proximity sensor can detect leading or trailing product edges for material spacing.



With registration, the registration sensor is wired to the fast input on the servo drive providing the means for the registration trigger that will capture the axis position at the instance of the registration event. When the event occurs, the system will calculate a registration correction to compensate for the inconsistency and then can apply the correction to the registration axis and/or change the numerical representation of the registration axis position.



This is important in applications such as printing, packaging, and converting where the process must be precisely coordinated and any non-rigid material cannot be depended upon to retain dimensional relationships. These applications usually involve master/slave moves. Not only does registration provide correction but the fast input signals can also be used as repeatable references to which the master and all subsequent slaves continually synchronize. Two main types of registration are [Mark to Mark Registration](#) and [Mark to Machine Registration](#).

#### 12.2.7.4.1 Mark to Mark Registration

Mark to Mark Registration is implemented with the function block MC\_MarkRegist and is based on the desired distance between two registration marks. This distance is used to qualify a “good” mark and then calculate a registration correction. Examples of the different mark to mark registration variations include:

- Clear Lane Registration
- Print Registration
- Product Registration
- Rotary Registration

*Clear Lane Registration* is the most common type of registration used in industry. A dedicated lane on the material is reserved solely for registration purposes. Only the registration marks in the lane will trigger the sensor and fast input on the servo drive. The distance between one registration mark and the next mark is the basis for registration correction calculations.

*Print Registration* is less common than Clear Lane Registration. Print registration involves picking out a distinct distance between print features. Here too, this distance is the basis for registration correction calculations. Print registration is used when it is not practical to have a clear lane.

*Product Registration* uses cycle position where the relationship of the product position in the process cycle is important. Product registration can occur synchronously or asynchronously. In synchronous product registration, products typically flow continuously through the process. Registration will measure and calculate small adjustments required to move each product to the correct position. The distance between products is the basis for registration correction calculations. Asynchronous applications may require the axis to sit and wait for the product to pass the sensor

and, when it does, initiate a move and also apply registration correction based on the position of the product.

*Rotary Registration* is used to maintain a fixed axis position reference point on a rotary axis that has non-integer feedback in one revolution. If the axis position is not adjusted, the actual axis position would walk away from the desired position every revolution because of the non-integer number of feedback units per revolution.

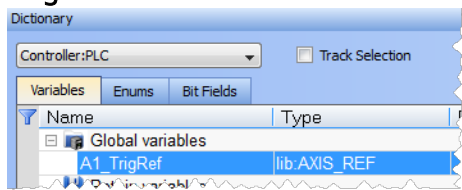
#### 12.2.7.4.2 Mark to Machine Registration

Mark to Machine Registration is implemented with the function block MC\_MachRegist and is based on a desired target position on a machine axis. Like mark to mark registration, mark to machine registration uses the desired distance between two registration marks to qualify a “good” mark (using Clear Lane, Print, and Product registration). But instead of using the mark to mark length, it uses an axis position as its basis for correction.

#### 12.2.7.5 Fast Homing Example with the PLCopen Motion Engine

This example shows how to use the axes' motion library functions to detect the positive edge of the first Fast Input in the drive, and read its position. [MC\\_TouchProbe](#) and the Trig\_Ref library function are used to accurately perform a homing procedure.

1. Set up an instance of the TrigRef function in the Dictionary. For this example, the name **A1\_TrigRef** is used.



2. Define the TrigRef values.

```
A1_TrigRef.InputID := 0; // configure Drive Capture Engine Used
A1_TrigRef.TrigID := 0; // configure which Drive to capture an
input
A1_TrigRef.Direction := 1; // configure Signal transition Direction
of capture
A1_TrigRef.TrigMode := 1; // configure Time or Position Capture
```

3. Activate the TouchProbe function

```
Inst_MC_TouchProbe( True, AxisID, A1_TrigRef, FALSE, 0, 0 );
```

#### NOTE

The position capture can be time-based ( `A1_TrigRef.TrigMode := 0` ) or position-based ( `A1_TrigRef.TrigMode := 1` ).

#### TIP

Fast Homing based on the drive's high speed capture mechanism is also supported by the following Kollmorgen KUDFBs in the IDE Function Block Library [MCFB\\_StepAbsSwitchFastInput](#) and [MCFB\\_StepLimitSwitchFastInput](#).

## 12.2.8 Pipe Network Registration and Fast Homing

### 12.2.8.1 Fast Homing Example with the Pipe Network Motion Engine Axis Pipe Block

This use case explains how to use the motion library functions of the axes when you want to detect the positive edge of the first Fast Input in the drive, and read its associated timestamp.

[MLAxisIsTriggered](#) can be used to accurately perform a homing procedure in a user-defined procedure.

A Setup sequence example.

1. [MLAxisCfgFastIn](#)(PipeNetwork.AXIS1, MLFI\_FIRST, MLFI\_RISING\_EDGE)

- Configure Fast Input 0 of AXIS1 to be triggered on the positive edge.

**NOTE**

Since tMLFI\_FIRST and MLFI\_SECOND correspond to the physical Fast Inputs 1 and 2. Therefore the drive must be configured in order to link fast input 1 with engine 0, and fast input 2 with engine 1.

- The first argument indicates the Axis pipe block in the Pipe Network that represents the drive to be configured
  - The second argument identifies which of the two Fast Inputs of the drive is configured (can be 0 or 1)
  - The third argument can indicate detection of positive edge when set to 1 and detection of negative edge when set to 2  
Note that if set to 0, Fast Input is disabled
2. [MLAxisIsTriggered](#)(PipeNetwork.AXIS1, 0, 1)
- This function returns true if Fast Input 0 of AXIS1 has been triggered on the positive edge.
  - The meaning of the arguments is the same as in [MLAxisCfgFastIn](#)
3. [MLAxisTimeStamp](#)(PipeNetwork.AXIS1, 0, 1)
- This function returns the time in microseconds when the Fast Input was triggered on the positive edge  
This time is relative to the start of the drive cycle time and its value is explained in the section [How To Interpret a Timestamp](#).
  - The meaning of the arguments is the same as in [MLAxisCfgFastIn](#)
4. [MLAxisRstFastIn](#)(PipeNetwork.AXIS1, MLFI\_FIRST)
- This function resets the Fast Input 0 of AXIS1. The reset keeps the configuration of the Fast Input, but it rearms it so it can be triggered again
  - The meaning of the first two arguments is the same as in [MLAxisCfgFastIn](#)
5. Follow-up Motion

The following is typical code used in a homing procedure.

```
CASE StepCounter OF
0:
  MLAxisRstFastIn (PipeNetwork.Feeder,MLFI_FIRST);

  MLAxisMoveVel (PipeNetwork.Feeder,250.0); //Jog Feeder Axis to search for
  sensor input

  StepCounter := 1;

1:
  IF MLAxisIsTriggered (PipeNetwork.Feeder,MLFI_FIRST,MLFI_RISING_EDGE)
```

```

THEN MLAxisAbs (PipeNetwork.Feeder,MLAxisCmdPos (PipeNetwork.Feeder));
//Stop motion when sensor is reached
StepCounter := 2;
END_IF;

2:

IF MLAxisGenIsRdy (PipeNetwork.Feeder)
THEN MLAxisWritePos (PipeNetwork.Feeder,0); //Set Feeder Axis position
to zero
StepCounter := 3;
END_IF;

```

Fast Homing based on the drive's high speed capture mechanism is also supported by the following Kollmorgen KUDFBs in the IDE Function Block Library: [MLFB\\_HomeFindHomeFastInput](#), [MLFB\\_HomeFindHomeFastInputModulo](#), [MLFB\\_HomeFindLimitFastInput](#), and [MLFB\\_HomeFindLimitFastInputModulo](#).

The Axis Position can also be derived from the Time Stamp by with the following code:

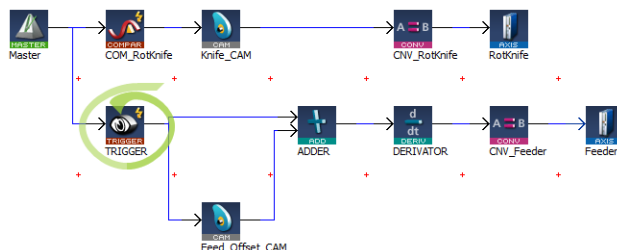
```

TimeStamp := any_to_lreal(MLAxisTimeStamp( iAxisID, any_to_dint
(ibFastInputNumber), 1+any_to_dint(ibHomeSwitchMode) ))/1000000;
CalculatedTriggerPosition := (CurrentVelocity-
PreviousVelocity)/2/iCycleTime*TimeStamp*TimeStamp+PreviousVelocity*Time
Stamp+PreviousPosition;

```

### 12.2.8.2 Registration Position Capture Example with Pipe Network Trigger Block

A Pipe Network Trigger block can be used for registration applications to capture a pipe position based on a timestamp which is set when a drive's fast input changes state. A Trigger block is placed in the Pipe Network precisely where the pipe position needs to be captured.



**Figure 7-22:** Example of using a Pipe Network Trigger block for position capture.

The Pipe Network motion engine captures the pipe position by using the timestamp received from the drive and interpolating two consecutive pipe position updates. The timestamp will have a delay getting into the pipe network from the drive (typically 1-2 cycles) due to sensor delay and the communication latency through the EtherCAT network (typically 1-2 cycles).

```

Corrected timestamp: = Fast input timestamp - DelayCompensation

```

To account for this delay, [MLTrigWriteDelay](#) provides delay compensation to the pipe position returned in [MLTrigReadPos](#) based on the expected delay.

The Trigger block is first configured by clicking on the block in a pipe network and setting the following parameters.

Function	Description
<b>INPUT_AXIS</b>	Defines the axis whose Fast Input is used. This name is the same given to the corresponding axis block in the Pipe Network
<b>INPUT_ID</b>	Indicates which one of the two available Fast Inputs in that particular axis is used. The value can be MLFI_FIRST or MLFI_SECOND for the trigger block to be triggered on the arrival of the first or the second input respectively.  Specify one of the following <b>constants</b> : MLFI_FIRST or MLFI_SECOND for the trigger block to be triggered on the arrival of the first or the second input respectively.
<b>TRIGGER_MODE</b>	Indicates if the trigger block responds to the rising edge or the falling edge of the Fast Input Specify one of the following <b>constants</b> : MLFI_RISING_EDGE or MLFI_FALLING_EDGE

**Figure 7-23:** Configuration of the Trigger block

After configuring the Trigger block, the order of calls to its motion library functions is as follows:

1. [MLTrigWriteDelay](#)(DINT,TriggerID,LREAL,delay)
  - This function sets the delay compensation, typically starting with a value of 2 usec + (1 or 2) EtherCAT cycles.
2. [MLTrigSetEdge](#)(PipeNetwork.TRIGGER1,MLFI\_FIRST,MLFI\_RISING\_EDGE)
  - This function reconfigures the edge of a trigger lock.
  - This function only needs to be called if the desired edge is different than the edge specified in configuration of the trigger block or if the edge is different than the previous capture.
3. [MLAxisCfgFastIn](#)(PipeNetwork.AXIS1, MLFI\_FIRST, MLFI\_RISING\_EDGE)
  - This function call is necessary at least one time, even if the Trigger pipe block is configured properly
4. [MLTrigIsTriggered](#)(PipeNetwork.TRIGGER1)
  - This function returns TRUE if the Fast Input associated to the Trigger pipe block given as argument has been triggered
5. [MLTrigReadPos](#)(PipeNetwork.TRIGGER1)
  - This function returns the position of the Pipe Network at the time that the Fast Input associated with the Trigger pipe block was issued

#### NOTE

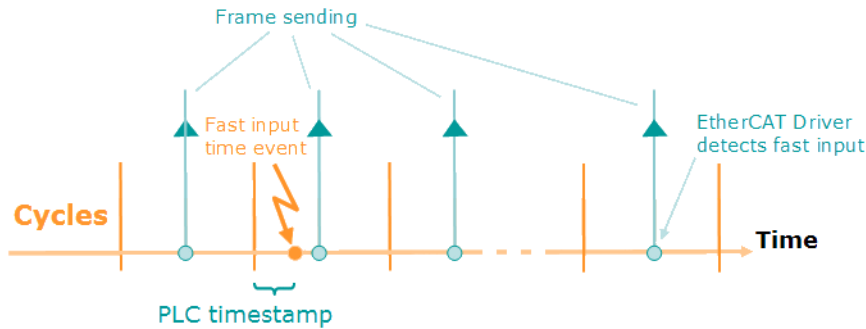
You have to correct the position by taking into account the delay due to the number of cycles needed to read the timestamp of the Fast Input. You can use [MLTrigWriteDelay](#) to address this issue.

6. [MLTrigReadTime](#)(PipeNetwork.TRIGGER1)
  - This function returns the time associated with the Fast Input as explained in [How To Interpret a Timestamp](#).  
Note that this function is of lesser importance compared to the previous one.
7. [MLTrigClearFlag](#)(PipeNetwork.TRIGGER1)
  - This function rearms the Trigger pipe block
8. [MLAxisRstFastIn](#)(PipeNetwork.AXIS1, MLFI\_FIRST)
  - This function rearms the Axis pipe block

After the Trigger block pipe position is determined and calculations are made in the Registration program, a corrective move is typically done using a [Cam](#) Pipe Network block for precision motion control.

### 12.2.9 How To Interpret a Timestamp

The timestamp is based on the EtherCAT system time. For this value to make sense, distributed clock must be activated in the drive (see [Distributed Clock tab](#)) and in the EtherCAT master.



**Figure 7-24:** PLC Timestamp Related to Fast Input Event

The timestamp returned is relative to the beginning of the cycle in which the Fast Input is triggered. It is called PLC timestamp since it is the value that you can use in PLC programs.

### 12.2.10 Tuning Controller Performance for EtherCAT Communication Latency

The response time for reading axis demand position data varies between different drive models. The response times also vary in reporting back TxPDO data such as actual position and time-based capture values. These different response times can lead to different lag in applications which use camming, gearing, and registration features. The KAS IDE has several configuration parameters which allow you to tune your system to work best.

For example, Drive A has a 1 cycle latency for TxPDO data, and Drive B has 2 cycles latency. The `MC_AXIS_PARAM_VELCOMP_FACTOR` parameter can be used to specify 1 (for Drive A), and 2 (for Drive B) cycles.

The features and corresponding parameters are listed below.

#### **TIP**

If the drives in an existing system are being changed, such as during an upgrade from AKD to AKD2G drives, then the parameters will need to be retuned to maintain optimal performance.

Feature	Parameters	Read Me For More Information
PLCopen Camming	Axis parameters <code>MC_AXIS_PARAM_VELCOMP_FACTOR</code> , <code>MC_AXIS_PARAM_VELCOMP_FILTER</code>	<a href="#">Axis Parameters</a> <a href="#">Velocity Compensation</a>
PLCopen Gearing		
PLCopen TouchProbe	Axis parameter <code>MC_AXIS_PARAM_SENSOR_DELAY</code>	<a href="#">MC_TouchProbe and Time-Based Capture (KDN Article)</a> <a href="#">PLCopen Registration</a>
PLCopen Registration		
PipeNetwork External Feedback	Use a Phaser block to compensate for the delay	<a href="#">Motion Library - Phaser</a>
PipeNetwork Trigger Block	Trigger Delay	<a href="#">MLTrigReadDelay</a> <a href="#">MLTrigWriteDelay</a>

**NOTE**

The AKD drive supports an object to configure the EtherCAT input data for immediate or 1 cycle delay (default for backwards compatibility). For more details see \_\_\_\_.

**12.2.11 Error Management**

When a non-fatal error occurs and motion must be stopped quickly, the following procedure can be taken:

For each axis:

Step	Example Application Code
Send Stop Command for each axis	<code>MLAxisStop(PipeNetwork.AXI_A1_Axis, TRUE, DEF_A1_StopDec);</code>
Stop the Axis Motion Generator	<code>MLAxisMoveVel(PipeNetwork.AXI_A1_Axis, 0.0);</code>
Wait for Axis to be stopped	<code>AxisStatus := MLAxisStatus(PipeNetwork.AXI_A1_Axis);</code> <code>IF AxisStatus.11 THEN</code> <code>MLAxisStop(PipeNetwork.AXI_A1_Axis,FALSE,DEF_A1_StopDec);</code>
Turn power off(disable) all the axes	<code>MLAxisPower(PipeNetwork.AXI_A1_Axis,FALSE);</code>
Disconnect Pipe Network from the axis	<code>MLCNVDisconnect(PipeNetwork.CNV_A1);</code>

For the machine:

Step	Example Application Code
Stop Command at the master block level	<code>MLMstRunMLMstRun(PipeNetwork.MASTER, 0.0);</code>
Wait for Master command to be stopped	<code>IF A1_AckState = DEF_StateErrorStop</code> <code>AND A2_Ackstate = DEF_StateErrorStop</code> <code>AND MLBlkIsReady(PipeNetwork.MASTER)</code> <code>THEN</code> <code>PrintF('*** ErrorStop M1=%i ***', M1_StatusWord,0,0,0);</code> <code>M1_AckState := DEF_StateErrorStop;</code>

**NOTE**

This code is only applicable to PipeNetwork applications. It will not work in a PLCopen application.

This procedure for error management is based on the **Project Structure Guidelines** as described in [Application Software Structure - Implementation](#) (→ p. 695)

For information on **restarting the motion**, refer to [Restarting Motion with PipeNetwork](#) (→ p. 632)

**12.2.12 Restarting Motion with PipeNetwork**

An advantage of the Pipe Network is the ability to minimize machine downtime and reduce material waste when a non-fatal error occurs. After stopping the motion with `MLAxisStop` command, it can be restarting by using the `MLAxisReAlign` function block.

**⚠ IMPORTANT**

`MLAxisReAlign` must be called after the `MLAxisStop` command, otherwise all motion commands are ignored



For each axis:

Step	Example Application Code
Check Axis Status	<pre>AxisStatus := MlAxisStatus(PipeNetwork.AXI_A1_Axis); IF AxisStatus.6 THEN   StepCounter := 1; END_IF;</pre>
Turn axis back on (re-enable)	<pre>IF MlAxisPower(PipeNetwork.AXI_A1_Axis, PowerUp) THEN   StepCounter := 2; END_IF;</pre>
Calculate position difference between the Reference and Actual Positions	<pre>DeltaPos := (MlAxisCmdPos(PipeNetwork.AXI_A1_Axis) - MlAxisReadActPos(PipeNetwork.AXI_A1_Axis));</pre>
Determine how far to move	<pre>IF DeltaPos &gt; LREAL#0.5*DEF_A1_PosPeriod THEN   DeltaPos := DeltaPos - DEF_A1_PosPeriod; ELSE   IF DeltaPos &lt; LREAL#-0.5*DEF_A1_PosPeriod THEN     DeltaPos := DeltaPos + DEF_A1_PosPeriod;   END_IF; END_IF; MlAxisReAlign(PipeNetwork.AXI_A1_Axis, 1000.0, 1000.0, 100.0, DeltaPos); StepCounter := 3;</pre>
Wait for move to be completed	<pre>IF MlAxisReAlgnRdy(PipeNetwork.AXI_A1_Axis) THEN   StepCounter := 4; END_IF;</pre>

For the machine:

Step	Example Application Code
Execute multi-axis move	<pre>MLMstRun MLMstRun(PipeNetwork.MASTER, 500);</pre>

### 12.2.13 Superimposed Axes with PLCopen

This feature allows the application program to superimpose the moves of multiple axes ("Superimposed Axes") on top of the move of another axis ("Receiving Axis"). This is performed internally by adding the command deltas of the Superimposed Axes to the command delta of the Receiving Axis. Up to four different Superimposed Axes can be superimposed upon a Receiving Axis.

### 12.2.14 Overtravel Conditions

The PLCopen and PipeNetwork motion engines can respond to overtravel conditions from the AKD and AKD2G drive family. As a safety feature, AKD and AKD2G drives ignore motion commands once an overtravel is detected in the direction of a hardware limit switch. The PLCopen or PipeNetwork motion engine applies a stop action to axes when these overtravel conditions occur.

#### NOTE

Generation 1 AKD drives require setting a fieldbus parameter to enable this behavior. See [Configuring Generation 1 AKD Drives](#) below.

#### 12.2.14.1 PLCopen Behavior

A PLCopen axis issues an emergency stop of the axis when it experiences an overtravel condition. The axis remains enabled under servo control once the emergency stop action completes. This action puts the axis into the "ErrorStop" state (see [PLCopen State Machine](#)).

To recover from this state:

1. Call [MC\\_ResetError](#) on the axis.
2. Command motion in the direction opposite to what was previously commanded.

If you are uncertain about which direction the axis was traveling in, check the states of the I/O points configured for the positive and negative limit switches on the drive.

#### 12.2.14.2 Coordinated Motion Behavior

An axis that is part of an axes group issues an emergency stop on the axes group when it experiences an overtravel condition. All axes remain enabled under servo control once the emergency stop action completes. This action puts the axes group into an error state.

To recover from this state:

1. Call [MC\\_GrpReset](#) on the axes group
2. Check the states of the I/O points configured for the positive and negative limit switches on the drives associated with the axes to figure out which axis and which limit switch cause the error.
3. Command motion in the direction opposite to what was previously commanded.

#### 12.2.14.3 PipeNetwork Behavior

A PipeNetwork axis issues a stop action on the axis when it experiences an overtravel condition. This is similar to calling [MLAxisStop](#). Bit 11 of [MLAxisStatus](#) is set and the axis remains enabled under servo control once the emergency stop action completes. The axis will no longer be accepting pipe positions. The pipe that the axis is connected to will continue functioning as before and any master will continue generating new positions. The axis which experienced overtravel simply no longer responds to new pipe positions.

To recover from this state, call [MLAxisReAlign](#) to realign the axis with the pipe. See the page [Restarting Motion with PipeNetwork](#) to understand how to accomplish this.

#### NOTE

If a hardware limit flag is being triggered at the program startup by a PipeNetwork axis, [MLAxisReAlign](#) may need to be called immediately before motion occurs. In this case, the distance to travel should be at least as long as the maximum distance the hardware limit flag can remain active.

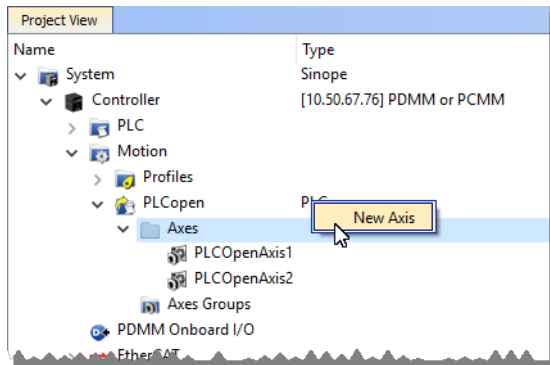
#### 12.2.14.4 Configuring Generation 1 AKD Drives

When using generation 1 AKD drives, bit 14 of the parameter FBUS.PARAM05 (see [FBUS.PARAM1 TO FBUS.PARAM10](#)) must be set high to allow KAS IDE to respond to the hardware limit switches, If FBUS.PARAM05 originally had a value of 0, typing `FBUS . PARAM05 16384` would turn on bit 14. Remember to save your drive parameters to flash memory once bit 14 is set, otherwise the behavior will turn off once the drive is power cycled.

## 12.2.15 How To Create a Digitizing Axis

Digitizing axes are supported by PLCopen-based projects. Following is the general process. The specifics will vary depending upon your EtherCAT network.

1. From the Project View, right-click on the *Controller > Motion > PLCopen* item and select **New Axis**.



2. Give the axis a name, note that spaces are not supported. See [Common Axis Parameters](#) for details.
3. Digitizing axes have *Bus* data which needs to be defined. See [Digitizing Axis Parameters](#) for details.
4. Lastly, define the [Axis Data Parameters](#).

The following list provides more detail about each method or circumstance under which a digitizing axis may be created.

- [How To Create a Digitizing Axis When EtherCAT Devices Are Defined](#)
- [How To Create a Digitizing Axis When EtherCAT Devices Are Not Connected](#)
- [How To Create a Digitizing Axis When Using an Imported ENI File](#)
- [How To Create a Digitizing Axis From Code](#)

Also see [Working With A Digitizing Axis in PLCopen](#) for more in-depth information about digitizing axes.

*What happens when the device providing feedback to the digitizing axis is removed?*

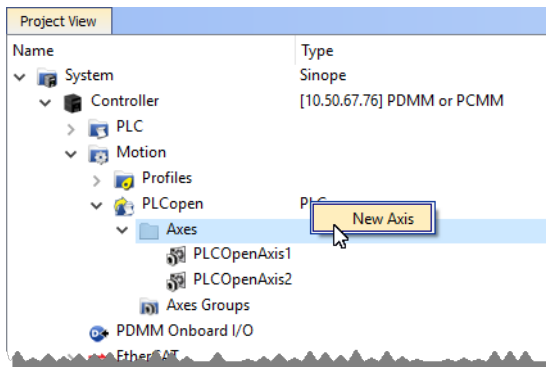
The feedback position will become zero if the device providing the Feedback Position for the Digitizing axis is removed using the [ECATDeviceAction](#) function block. The feedback position data coming from the device will automatically be connected to the digitizing axis when the device is reconnected to the EtherCAT network.

### 12.2.15.1 How To Create a Digitizing Axis When EtherCAT Devices Are Defined

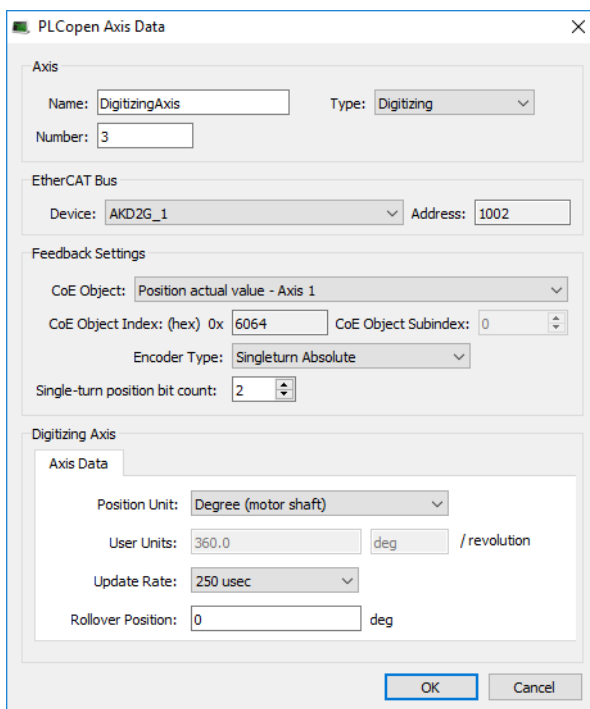
This process assumes that you have already scanned the EtherCAT network and assigned devices to axes.



1. From the Project Tree, right-click on the *Controller > Motion > PLCopen* item and select **New Axis**.



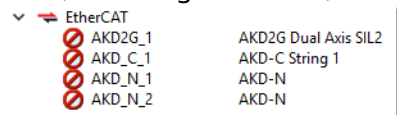
2. Give the axis a name and click **OK**. Note that spaces are not supported and select "**Digitizing**" as the **Type**. See [Common Axis Parameters](#) for details.
3. Digitizing axes have *Bus* and *Feedback Settings* data which needs to be defined. Double-click on the newly defined axis. Using an AKD or AKD2G will automatically set certain parameters, and hide that data. For example the Encoder Resolution for AKD drives is automatically set to 20-bit. See [Digitizing Axis Parameters](#) for details.



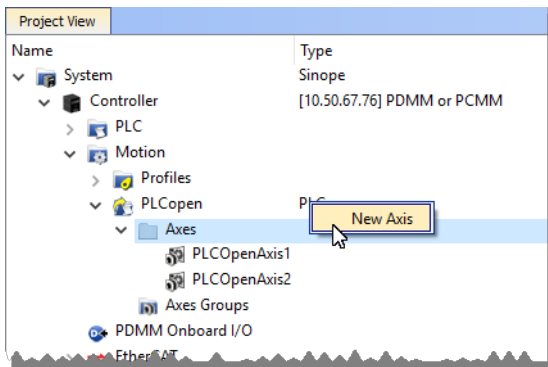
4. Lastly, define the [Axis Data Parameters](#).

### 12.2.15.2 How To Create a Digitizing Axis When EtherCAT Devices Are Not Connected

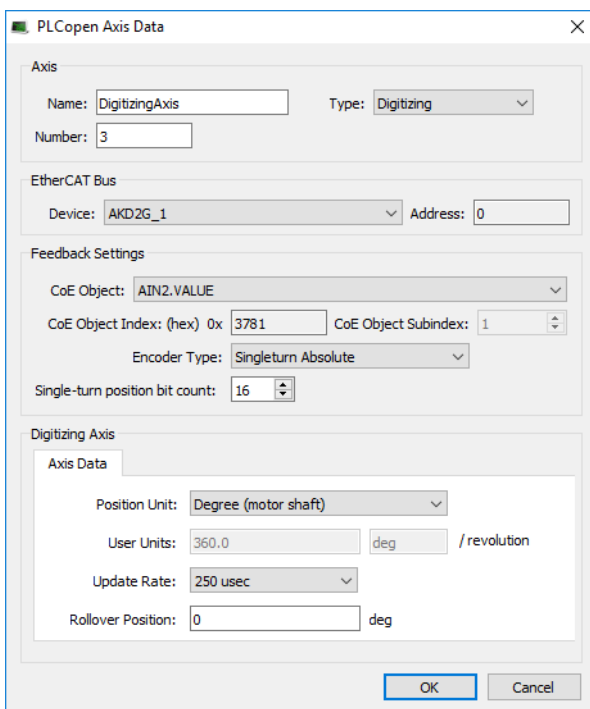
This process assumes that you are setting up a project and are not connected to the EtherCAT network. In this example we have added several devices to the EtherCAT branch of the Project View, including an AKD2G, an AKD-C, and two AKD-Ns.



1. From the Project Tree, right-click on the *Controller > Motion > PLCopen* item and select **New Axis**.



2. Give the axis a name and click **OK**. Note that spaces are not supported and select **"Digitizing"** as the **Type**. See [Common Axis Parameters](#) for details.
3. Digitizing axes have *Bus* and *Feedback Settings* data which needs to be defined. Double-click on the newly defined axis and select the **Device** which will provide the feedback information and the feedback parameters, including the CoE information that will be used. See [Digitizing Axis Parameters](#) for details.

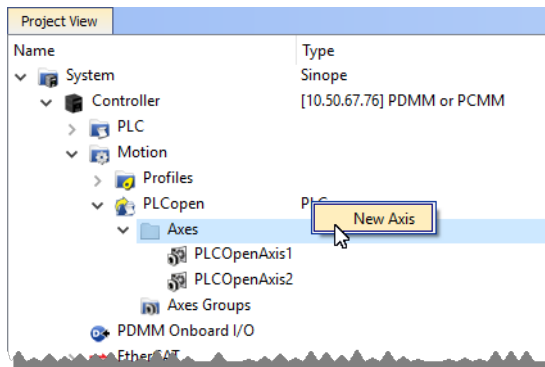


4. Lastly, define the [Axis Data Parameters](#).

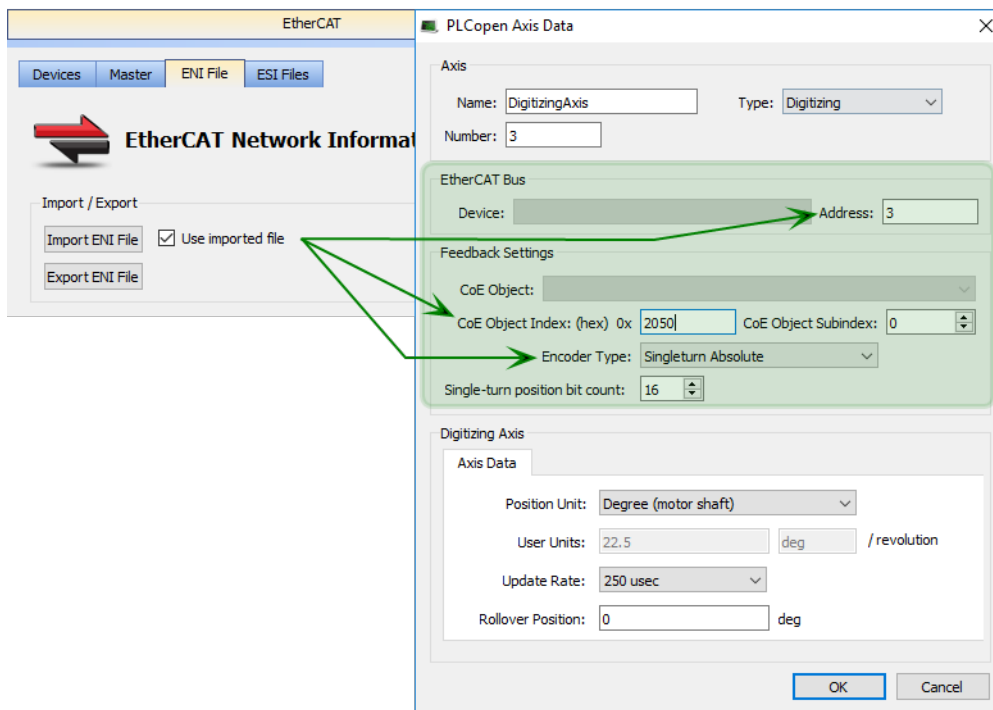
### 12.2.15.3 How To Create a Digitizing Axis When Using an Imported ENI File

This process assumes that you are setting up a project when using an imported ENI file. See [ENI File Tab](#) for information about using ENI files.

1. From the Project Tree, right-click on the *Controller > Motion > PLCopen* item and select **New Axis**.



2. Give the axis a name and click **OK**. Note that spaces are not supported and select **"Digitizing"** as the **Type**. See [Common Axis Parameters](#) for details.
3. Digitizing axes have *Bus* and *Feedback Settings* data which needs to be defined. The ENI file will provide some information about the digitizing axis but you will need to provide the **Address** of the device to use as well as the **CoE Object Index**, **Subindex**, and the **Encoder Resolution**. See [Digitizing Axis Parameters](#) for details.



4. Lastly, define the [Axis Data Parameters](#).

#### 12.2.15.4 How To Create a Digitizing Axis From Code

Following is a method for creating digitizing axes without using the GUI.

1. The [MC\\_CreatePLCAxis](#) function block is used to create the axis.
2. The [MC\\_InitAxisFeedback](#) function is used to initialize the feedback.

```
(* ----- Create PLCopen axis 'Digit1' *)
MC_CreatePLCAxis( 'Digit1', 'EtherCATDriver', 1001, 3, 1, 1, 1, 1,
LREAL#0, 4 );
MC_InitAxisFeedback( 3, 24804, 2, 32 );
```

### 12.2.16 Working With A Digitizing Axis in PLCopen

This topic provides the basics of setting up a digitizing axis and describes how to accomplish some common tasks. See [Common Axis Parameters](#) for reference material.

- A Digitizing Axis requires a feedback position. The feedback position can come from a secondary feedback of a drive or an EtherCAT device. Typically the AKD or S300/S700 drives provide the feedback position via the CoE object **0x2050**.

Input (Tx) PDOs

Select Input (Tx) PDOs

0x1B23 Inputs

Index	Subindex	Object Name	Size [bit]	PLC Variable
0x2050	0	Position actual val...	32	(Global)/ActPosA1_Feedback2
0x20...	0	Latch status word	16	
0x20...	0	Latch position	32	
0x3470	4	AIN.VALUE	16	Managed in I/O tab
0x6041	0	Status word	16	
0x6063	0	Position actual val...	32	
0x606C	0	Velocity actual val...	32	
0x6077	0	Torque actual value	16	
0x60F4	0	Following error	32	
0x60...	0	Digital inputs	32	Managed in I/O tab

- The Digitizing Axis is created by setting the Type to "Digitizing", selecting the EtherCAT device which will provide the feedback, and setting the correct CoE object.

PLCopen Axis Data

Axis

Name: DigitizingAxis      Type: Digitizing

Number: 3

Bus

Device: AKD\_1      Address: 1001

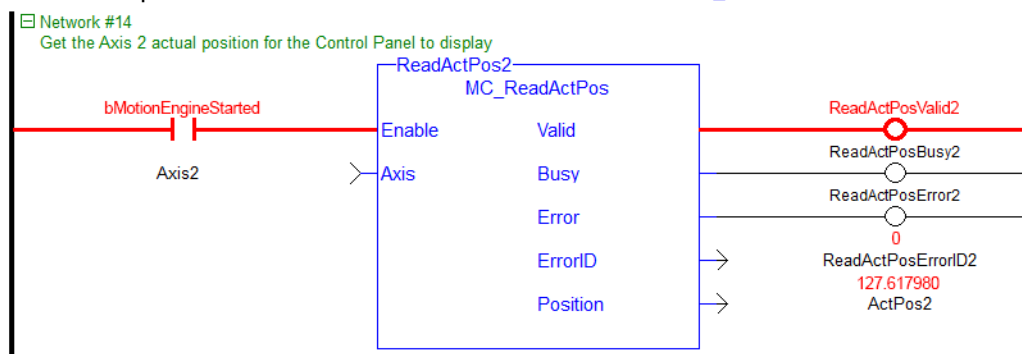
Feedback Settings

CoE Object: Position actual value 2

CoE Object Index: (hex) 0x 2050      CoE Object Subindex: 0

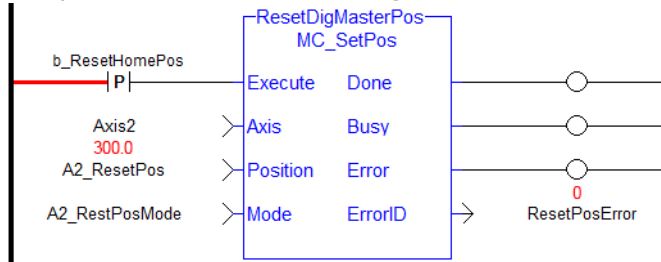
**NOTE**  
 First generation AKD drives will use certain pre-set values.

- The actual position in user units can be read with the [MC\\_ReadActPos](#) function block.





- The position can be offset using the [MC\\_SetPos](#) function block.



- For AKD drives, the polarity of the feedback signal can be changed using drive parameter [FB2.DIR](#).

### Parameters

This page lists the current values of all parameters on the device.

Full Name	Value	Units	Parameter	Read/Write
<b>Feedback 1</b>				
<b>Feedback 2</b>				
FB2 counting direction (impact...	0		FB2.DIR	read-write
Position Feedback Resolution	1,000		FB2.ENGRES	read-write
Feedback 2 Mode	0 - Input - A/B Sig...		FB2.MODE	read-write
FB2 MOTORPHASE	0 deg		FB2.MOTORPHA...	read-write
FB2.MOTORPOLES	6		FB2.MOTORPOL...	read-write
Feedback 2 scaled position	0.000 Counts		FB2.P	read-only
Feedback 2 counting direction	0		FB2.PDIR	read-write

**NOTE**

If an Endat 2.2 device is being used, set [FB3.DIR](#) and change [DRV.HANDWHEELSRC](#) to 3 in order to tie the PDO index 2050 parameter that is read from FB2 to FB3.

What happens when the device providing feedback to the digitizing axis is removed?

The feedback position will become zero if the device providing the Feedback Position for the Digitizing axis is removed using the [ECATDeviceAction](#) function block. The feedback position data coming from the device will automatically be connected to the digitizing axis when the device is reconnected to the EtherCAT network.

### 12.2.16.1 Digitizing Axis - Encoders Without a Z-Channel

If the encoder does not have a Z-channel there may be some unrelated drive faults associated with the secondary feedback signal. To disable this, change the setting for [FAULT130.ACTION](#) (or fault 131, 132, etc.) in the AKD parameters to 1.

### Parameters

This page lists the current values of all parameters on the device.

Full Name	Value	Units	Parameter	Read/Write
<b>Fault Actions</b>				
Gets/Sets the Resulting Action for Fa...	1 - Ignore (fault will not be reported)		FAULT130.ACTION	read-write
Gets/Sets the Resulting Action for Fa...	0 - Disable Amplifier		FAULT131.ACTION	read-write
Gets/Sets the Resulting Action for Fa...	0 - Disable Amplifier		FAULT132.ACTION	read-write
Gets/Sets the Resulting Action for Fa...	0 - Disable Amplifier		FAULT134.ACTION	read-write
Gets/Sets the Resulting Action for Fa...	0 - Disable Amplifier		FAULT139.ACTION	read-write
Gets/Sets the Resulting Action for Fa...	0 - Disable Amplifier		FAULT451.ACTION	read-write
Gets/Sets the Resulting Action for Fa...	4 - Generate Warning (motion will con...		FAULT570.ACTION	read-write
Gets/Sets the Resulting Action for Fa...	0 - Disable Amplifier		FAULT702.ACTION	read-write

## 12.2.17 Practical Applications - Camming

The following topics provide procedures you may find useful when programming camming.

### 12.2.17.1 Positioning an Axis Before Starting Camming

The function block [MC\\_CamStartPos](#) is used to position a slave axis in its starting position for a [MC\\_CamIn](#) move with a slave absolute profile. To position the slave axis for the [MC\\_CamIn](#) move, the typical programming sequence is:

1. With the master axis at standstill, call [MC\\_CamStartPos](#) to determine the start position for the slave axis.
2. Call [MC\\_MoveAbsolute](#) to move the slave axis to its start position.
3. Call [MC\\_CamIn](#) with `StartMode = 0` (Start mode). The [MC\\_CamIn](#) inputs *MasterOffset*, *SlaveOffset*, *MasterScaling*, and *SlaveScaling* should have the same values as used in the call to [MC\\_CamStartPos](#).

See also: [Cam Profile Switching](#)

### 12.2.17.2 Resuming Camming After an E-Stop

The [MC\\_CamResumePos](#) function block is used to return a slave axis to its profile position after an event (such as an E-stop) caused the slave axis to go off path. To return a slave axis to its [MC\\_CamIn](#) profile position, the typical programming sequence is:

1. With the master axis at standstill, call [MC\\_CamResumePos](#) to determine the profile position for the slave axis.
2. Call [MC\\_MoveAbsolute](#) to move the slave axis to the position calculated by [MC\\_CamResumePos](#).
3. Call [MC\\_CamIn](#) with `StartMode = 1` (Resume mode).

### 12.2.17.3 Cam On The Fly

In addition to creating and modifying a Cam Profile using the IDE, an application programmer can also create or modify a new Cam Profile directly from their application. This allows new Cam Profiles to be defined while the application is still online, without stopping the machine to load a new application. An application programmer might use this feature to modify their application at runtime to adjust their Cam Profiles for varying product shapes and sizes.

The following steps will guide you through creating and building a new Cam Profile on the fly.

1. Before the motion engine is started, reserve memory for the profile that will be created on the fly. This is done using [MLProfileCreate](#).
2. Define the profile properties and profile data points. This can be done at any time but must be done prior to calling [MLProfileBuild](#).
3. Build the profile with [MLProfileBuild](#). This takes a number of cycles and the profile will not be ready until the "Done" flag is set to TRUE.
4. The newly created profile can now be used exactly like a profile built in the IDE; it can be used in any number of camming relationships using any of the PipeNetwork and PLCOpen Cam Profile functions/function blocks.

Once you have created a Cam Profile using [MLProfileBuild](#), if you need to modify that Cam Profile you must first release the existing Cam Profile. The following steps will release the Cam Profile.

1. Ensure the profile is not in use (Deactivate the existing profile). In Pipe Network perform a [MLCamSwitch](#) on an active Pipe to a different Profile or deactivate the pipe. In PLCOpen, perform an [MC\\_CamOut](#) on the profile or abort a profile move.
2. Call [MLProfileRelease](#) with the desired ProfileID as its input to release the profile so it can be modified.

3. When the "Done" flag is set, the ProfileID will be free. Modify the existing profiles data and then rebuild the profile [MLProfileBuild](#).

**NOTE**

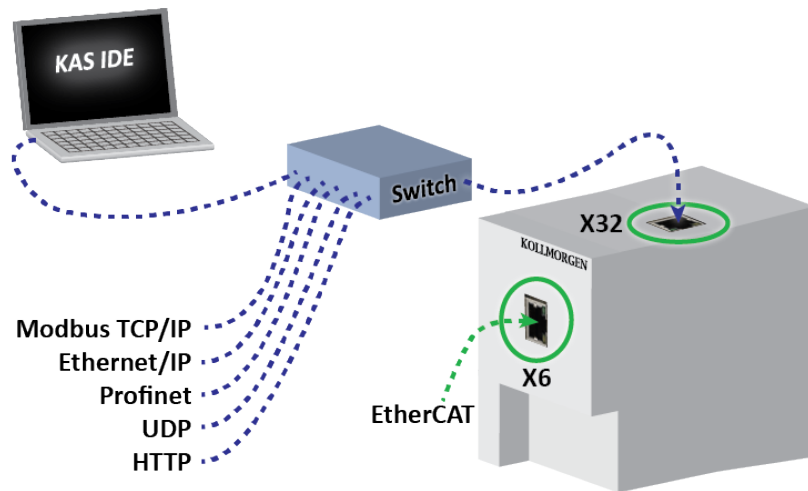
If a cam profile is Active, by having called [MC\\_CamTblSelect](#) and then [MC\\_CamIn](#) in PLCopen or [MLCamSwitch](#) in PipeNetwork, it cannot be released. In this situation use a second non-active cam to release, build (or rebuild), then activate.

An application is limited to 256 Cam Profile IDs. If additional profiles are desired, an existing profile can be released with [MLProfileRelease](#). This frees its ProfileID so it can be used in the construction of a new profile. Releasing a profile is simple, and is performed with the following steps.

1. Ensure that the profile is not in use (Deactivate the existing profile). See for more details.
2. Call [MLProfileRelease](#) with the desired ProfileID as its input to release the profile so it can be modified.
3. When the "Done" flag is set, the ProfileID will be free and the old profile can be used for a new profile, calling either [MLProfileInit](#) or [MLProfileBuild](#).

## 12.3 Motion Bus and Fieldbuses

Depending on the fieldbus used in your project (EtherCAT, Profinet, etc.), you have to make use of the configuration tools discussed below. The AKD PDMM or PCMM can be connected to multiple Ethernet devices simultaneously by connecting a switch to the X32 port on the top of the device.



### TIP

See [Performance Guidelines](#) for information on how the CPU load is affected when transmitting and receiving data between network devices and PLC variables

### 12.3.1 EtherCAT

- For configuration, [Configure EtherCAT Motion Bus](#)
- For I/O mapping, [Map Input and Output to Variables](#)
- For error management, [EtherCAT Error Messages](#)

See the [Beckhoff Web site](http://www.beckhoff.se/english.asp?download/elconfg.htm) for EtherCAT XML Device Description (<http://www.beckhoff.se/english.asp?download/elconfg.htm>).

### 12.3.2 Ethernet/IP

The KAS Runtime includes a fully integrated Ethernet/IP Adapter driver and Scanner driver for exchanging data with Ethernet/IP tag-based devices such as PLCs. Ethernet/IP Adapter (server), Scanner (client), Tag Client, and FlexIO/Point IO configurations are supported. The mapping of PLC variables to Profinet is as simple as a drag and drop.

- Setting up [Ethernet/IP IO Scanner \(Client\)](#)
- Setting up [Ethernet/IP Adapter \(Server\)](#)
- Setting up [Ethernet/IP Tag Client](#)

### 12.3.3 Modbus & TCP/IP

- Setting up [Modbus Slave](#)

### 12.3.4 Profinet

This fieldbus is Profibus over ethernet. We define one controller and have 1 or more devices.

- For configuring the controller, see [Profinet IO RT Controller Configuration](#) .
- For configuring devices, see [Profinet IO RT Device Configuration](#).

### 12.3.5 Add Unsupported EtherCAT Device

#### NOTE

This procedure is for **advanced** users only

When your project contains EtherCAT devices that are not supported by KAS, you have to create the configuration with an external tool, and perform the following steps:

1. Get the AKD device description ESI file from the official AKD distribution
2. Ensure all the device description ESI files are available for the external tool
3. Use the external tool and do all the configuration, including the following points:
  - Set the Cycle Time
  - Turn on the distributed clocks option for all slave drives in order to share a global system time through EtherCAT
  - Assign PDO to each drive (inputs and outputs)
  - Set the mode of operation of the drives into **position** mode
  - Insert variable names and do the mapping (see details [below](#))
4. Use the external tool to export the ENI description file
5. In KAS, [Import](#) the ENI file describing all the EtherCAT devices included in your project

#### ! IMPORTANT

Importing an external ENI file overrides all EtherCAT project device information and configuration settings in the IDE. The following views and configurations are *not* applicable when using an imported ENI file:

- Project View: All devices located under the EtherCAT node
- EtherCAT Device View tabs:
  - General Properties
  - PDO Selection/Mapping
  - Distributed Clock
  - CoE Init-Commands
- Slice I/O Properties
- Mapping PLC Variables to Slice I/O or PDO objects

Information displayed in the views may not match the imported ENI file.

#### 12.3.5.1 How to modify the EtherCAT image in cyclic mode

In your application program, when integrating non-standard EtherCAT devices, use the following function blocks to update [EtherCAT frame](#):

- ECATWriteData(Function)
- ECATReadData(Function)

#### 12.3.5.2 How to configure EtherCAT device

You need to use the following Functions Blocks:

- ECATWriteSdo (Function Block)
- ECATReadSdo (Function Block)

#### 12.3.5.3 How to map PLC variables

When you use an XML network description file generated with an external configurator, you need to add special tags to the PDO names to ensure the PLC variables can be mapped to IO channels. The tags must comply with the following convention:

```
@Scope.VariableName+StartBit-Size
```

Field	Description
@	prefix with character @ the PLC variable names of each of the image attributes that must be mapped
Scope	<p>Scope can be:</p> <ul style="list-style-type: none"> <li>• (Global)</li> <li>• (Retain)</li> <li>• ProgramName</li> </ul> <p>Note that even for the case of nested child SFC programs, the variables still belong to a unique well defined subprogram</p> <p>Parentheses must surround the scope when it is Global or Retain.</p>
+StartBit	<i>(Optional)</i> Integer that defines the bit from which the data must be written or read from the PLC variable
-Size	<i>(Optional)</i> Integer that defines the number of consecutive bits in the image which must be copied to/from the PLC variable. When present, this setting has precedence over the <BitSize> tag of the XML file.

#### Examples:

```
(Global) .MachineState
(Global) .bLedStatus:0-1
(Global) .bLedStatus:1-1
(Global) .myINT:+4
main.variable:3+4-8
```

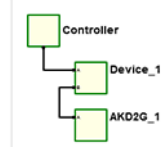
#### NOTE

This convention is applicable for simple variables. KAS does not yet support mapping for **Structs and Arrays**.

### 12.3.6 Set Up FSoE Master and an AKD2G with SafeMotion Monitor

This topic describes how to manually set up FSoE PDOs and copy links for a FSoE Master and an AKD2G with SafeMotion Monitor. This example uses a simple EtherCAT topology consisting of a PCMM (the **Controller**), a BBH SCU safety master (**Device\_1**), and an AKD2G with SMM Basic safety drive (**AKD2G\_1**).

EtherCAT Network Topology



#### **TIP**

If using a BBH SCU-1-EC and SafePLC2 software, then the FSoE PDOs can be automatically setup by the KAS IDE and SafePLC2. For more details see [Working with SafePLC2 Projects](#). For all other third party FSoE Masters, use the procedure described below.

#### 12.3.6.1 Create Safety Application

First, use the 3rd party Safety PLC software to create a safety application for the FSoE master. For example, the application will contain a FSoE master and an AKD2G with Axis1 and Axis2 as slave devices.

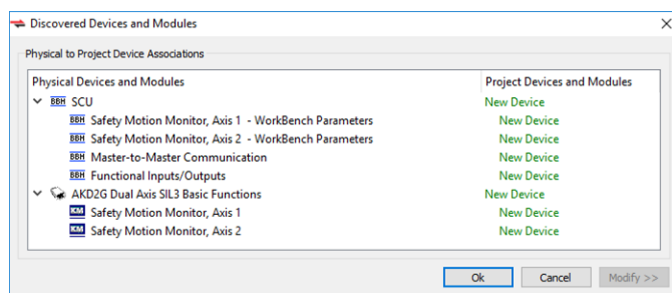
1. Set the AKD2G Connection IDs to match the the AKD2G SafeIDs.
2. Compile and download the safety application to the FSoE Master.
3. Run the safety application in FSoE Master.

The state will be "Run Intern" and the FSoE state will be "8".

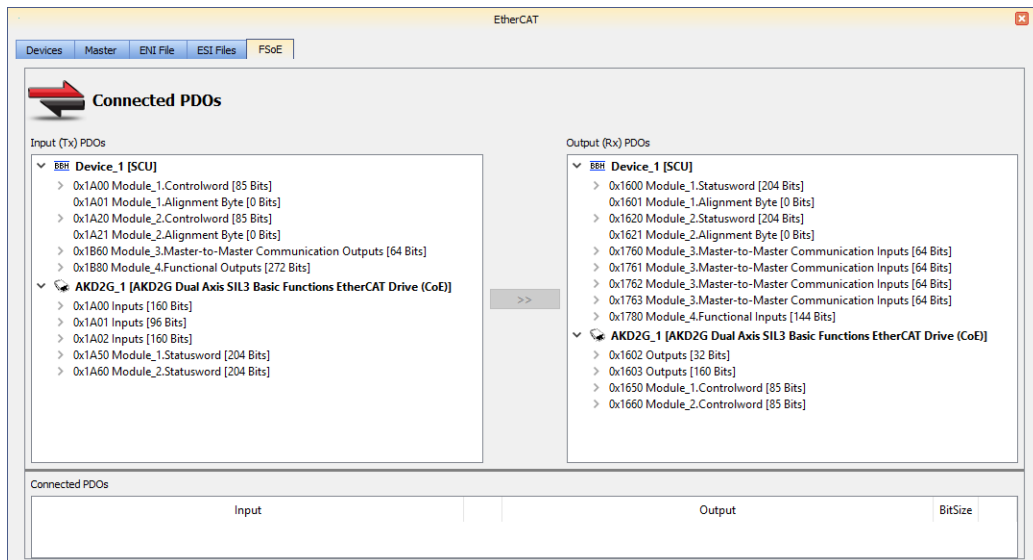
#### 12.3.6.2 KAS IDE Project Set Up

To finish the example, a project is created in the KAS IDE using the 2-axis Pipe Network template.

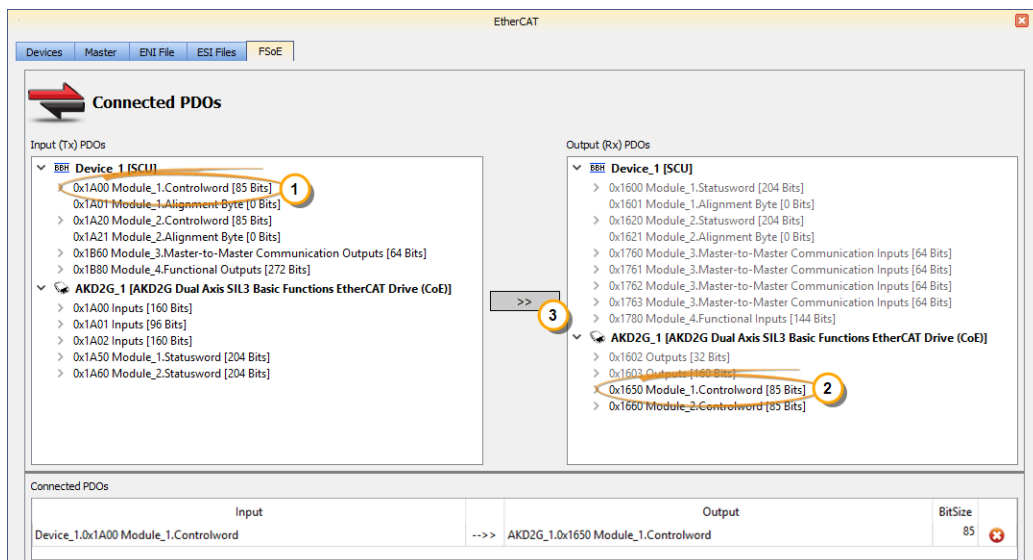
1. The IP address of the PCMM controller is set in the project.
2. Scan the EtherCAT network to discover the devices.



3. We are prompted to recompile the project due to changes in the EtherCAT topology. We respond "No" because we have not set up the PDO copy links.
4. To connect the PDOs we open the [FSoE / PDO Connections Tab](#) which is populated with Input and Output PDOs.



5. To connect PDOs we choose an input, an output, and click the **Connect** button. Connections are displayed at the bottom of the tab.



6. We make the following connections:

- Device\_1[SCU] PDO 0x1A00 to AKD2G\_1 PDO 0x1650
- Device\_1 [SCU] PDO 0x1A20 to AKD2G\_1 PDO 0x1660
- AKD2G\_1 PDO 0x1A50 to Device\_1 [SCU] PDO 0x1600
- AKD2G\_1 PDO 0x1A60 to Device\_1 [SCU] PDO 0x1620

7. Compile, download, and run the KAS application.

8. The BBH master has changed the state to "Run" and the FSoE state will be "4".

The Safety Master and the Safety Slave are now exchanging data over EtherCAT.

See Also:

- [AKD2G Safety Parametrization Using FSoE with SCU-1-EC and PCMM/AKD PDMM](#)
- [FSoEParamsInit](#)
- [Troubleshooting FSoE Safety Parameters](#)

### 12.3.7 AKD2G Safety Parametrization Using FSoE with SCU-1-EC and PCMM/AKD PDMM



The safety features in the AKD2G safety drive (Basic and Advanced) can be activated through either FSoE (Fail Safe over EtherCAT) or digital input or both. The source of activation is configured inside the safety parameters and it needs to match the system requirements.

This document describes how to configure and enable safety functions through BBH's SafePLC and KAS IDE applications.

### 12.3.7.1 Requirements

- Hardware**
- SCU-1-EC ([BBH FSoE Master](#))
  - PCMM or AKD PDMM
  - AKD2G safety drive
- Software**
- SafePLC2 ([BBH Safety Software](#))
  - KAS IDE - v3.05 (or newer)

### 12.3.7.2 How-To

Following are the general steps for safety parameterization via the "direct" method using the KAS IDE and BBH SafePLC2 software.

1. Create a PLCopen-based project in the KAS IDE.
2. Scan the EtherCAT network.
3. Configure the safety modules for the AKD2G devices.
  - Choose either the "SRA Parameters" or "SRA Parameters configure via WorkBench" module, based on your application need.
4. When using the "SRA Parameters" profile, add the [FSoEParamsInit](#) function block to your project.
 

This function block enables the controller to read the safety parameters from the FSoE Master (SCU-1-EC) and write them to the AKD2G.
5. Follow the steps listed in [Working with SafePLC2 Projects](#) if SafePLC2 is installed on the same computer or [Working with SafePLC2 Projects](#) if it is installed on a different computer than the KAS IDE.
6. Compile and download the SafePLC application to the BBH safety master (SCU-1-EC) and run the application.
7. Compile the KAS IDE project.
8. Run the compiled KAS application.
9. Execute the [FSoEParamsInit](#) function block when the application is ready and EtherCAT is in OP state.

#### See Also:

- [Working with SafePLC2 Projects](#)
- [Set Up FSoE Master and an AKD2G with SafeMotion Monitor](#)
- [FSoEParamsInit](#)
- [Troubleshooting FSoE Safety Parameters](#)

## 12.3.8 Fieldbus Editor

### 12.3.8.1 PCMM/AKD PDMM Ethernet/IP Data Communications

The following applies to both the Scanner and Adapter

- Up to 4 different connections:
  - One Adapter Connection with one O -> T Assembly and one T -> O Assembly
  - Up to 3 Scanner Connections each with one O -> T Assembly and one T -> O Assembly
  - Total: Up to 8 Assemblies (4 O -> T and 4 T -> O) total
- Connections can be:
  - All to one product (with unique IP Addresses)
  - Or up to 4 products (with 4 unique IP Addresses)
- Each Assembly can transfer up to 500 bytes
- PxMM max communications: 8 x 500 = 4000 bytes
- Assemblies and Connections can be mixed in any suitable manner. Examples:
  - Set up the PCMM/AKD PDMM to talk to 4 different Adapters with each one having a O -> T Assembly and T-A Assembly
  - Set up the PCMM/AKD PDMM:
    - as a Scanner (Client) to connect to 3 different Ethernet/IP Servers (Adapters) each with one O -> T Assembly and one T -> O Assembly
    - and also set up as a Server (Adapter) to connect to a Scanner (Client) with one O -> T Assembly and one T -> O Assembly

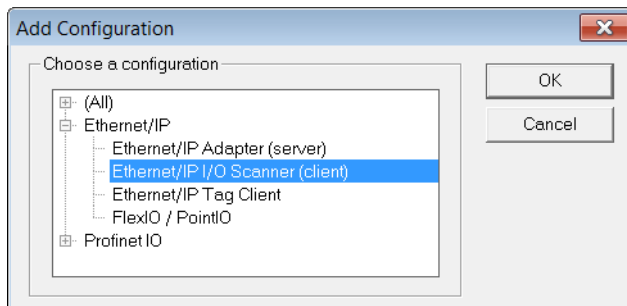
### 12.3.8.2 Ethernet/IP IO Scanner (Client)

The KAS Runtime includes a fully integrated Ethernet/IP client driver for exchanging CIP I/O assemblies as an Ethernet/IP scanner in your applications.

#### 12.3.8.2.1 Data Exchange - Configuration

A dedicated configuration tool is integrated in the KAS IDE.

1. Double-click the **Fieldbus** node in the project explorer to open it
2. Click the Insert Configuration icon  to add the Fieldbus configuration
3. Then select the **Ethernet/IP IO Scanner** in the configuration selector




The configuration is represented as a tree:

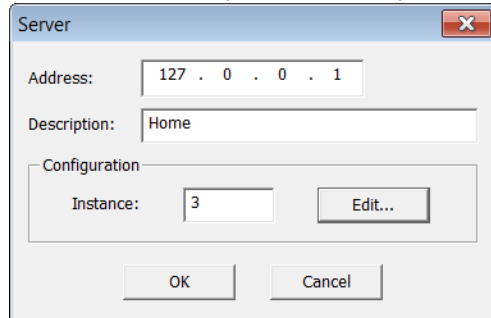
- Ethernet/IP IO Scanner
  - Server (an Ethernet/IP adapter device) (\*)
    - IO Assembly (Originator to Target)
      - Exchanged Variable (\*)
    - IO Assembly (Target to Originator)
      - Exchanged Variable (\*)

(\*) The items with this mark can appear several times in the configuration.

In addition to the IO configuration, explicit messaging can be performed using the blocks [eipReadAttr](#) and [eipWriteAttr](#).

### 12.3.8.2.2 Configuration of the Server

Click the Insert Master icon  to declare a server (slave adapter). Each server is identified by its IP address and an optional Description text.



Three blocks of data are exchanged with the server for each connection:

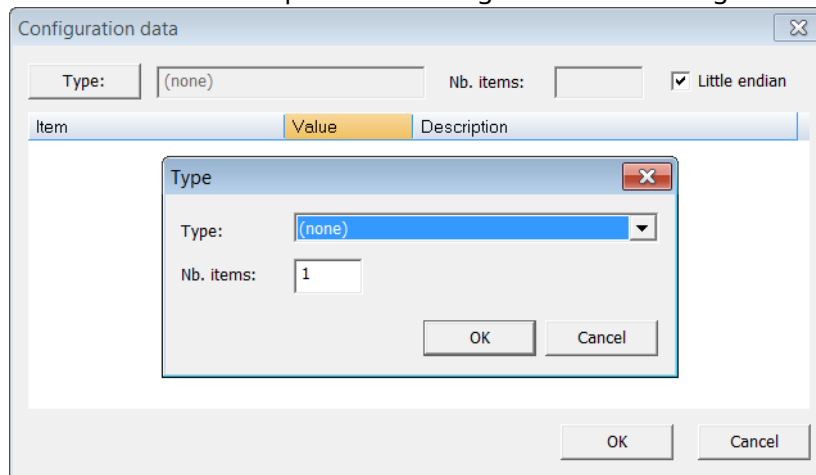
- Outputs (originator to target)
- Inputs (target to originator)
- Configuration: a static block sent to the server at connection time.

The instance number for the configuration may be specified in the properties of the server.

### 12.3.8.2.3 Editing the Configuration Data Block

While most devices do not expect any configuration it may happen that some data is required. For this, double click on the server in the tree and press the **Edit...** button in the dialog box.

The configuration is entered as a list of items. Use the **Type** button to specify the number of items and their data types. Structured data types defined in the KAS IDE are supported. Simply enter values in the grid. In case you exchange multiple byte integers (e.g. WORD) you must specify whether the device expects little or big endian formatting.



### 12.3.8.2.4 Configuring an Assembly

Click the Insert Slave icon  to declare a CIP I/O assembly. Each assembly is identified by:

Identifier	Meaning
<b>Type</b>	Direction of the I/O assembly. Can be one of: <ul style="list-style-type: none"> <li>• Originator to Target (outputs)</li> <li>• Target to Originator (inputs)</li> </ul>
<b>Instance</b>	Instance of the CIP assembly

Identifier	Meaning
<b>Config.instance</b>	The instance number of the configuration assembly. The default value is 100.
<b>Size</b>	Data size in bytes
<b>Connection type</b>	Type of the CIP connection. Can be Point To Point or MultiCast
<b>Priority</b>	CIP priority: Low, High, Scheduled or Urgent
<b>32 bit header</b>	Check this option if a 32 bit header is to be sent on notifications
<b>RPI(ms)</b>	Minimum period for notification of changes, in milliseconds (range is 10 - 10000 ms)
<b>Description</b>	Optional description text

### 12.3.8.2.5 Configuring Variables

IEC61131-3 variables may be mapped on the data of the assembly. For each variable you must specify:

Identifier	Meaning
<b>Symbol</b>	The name of the IEC61131-3 variable
<b>Offset</b>	Offset in bytes in the assembly data
<b>Bit</b>	Bit offset in the selected byte if format is "Bit"
<b>Format</b>	Format of the data in the assembly
<b>Mode</b>	Kind of data exchanged through the variable: <ul style="list-style-type: none"> <li>• <i>Data Exchange</i>: a type of input or output data in the assembly</li> <li>• <i>Server OK</i>: indicates the status of the IP connection to the server</li> <li>• <i>I/O connection OK</i>: indicates the status of the CIP I/O connection</li> <li>• <i>Last UCMM error</i>: CIP code fo the last error occurred during a connection</li> </ul>

#### NOTE

The data limit is: 500 bytes of data maximum O(originator)->T(target) and 500 bytes of data maximum T(target) -> O(originator). This is based on the Ethernet/IP specification.

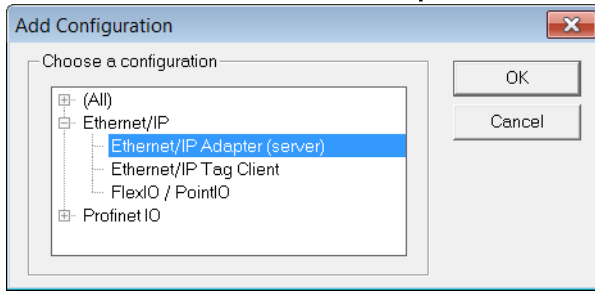
### 12.3.8.3 Ethernet/IP Adapter (Server)

The KAS Runtime includes fully integrated Ethernet/IP server driver for exchanging CIP I/O assemblies as an Ethernet/IP adapter in your applications.

#### Data Exchange - Configuration

A dedicated configuration tool is integrated in the KAS IDE.

1. Double-click the Fieldbus node in the project explorer to open it
2. Click the Insert Configuration icon  to add the Fieldbus configuration
3. Then select the **Ethernet/IP Adapter** in the configuration selector



The configuration is represented as a tree:


- Ethernet/IP IO Scanner
  - Served I/Os and objects
    - IO Assembly or Vendor Specific Object (\*)
      - Exchanged Variable (\*)


(\*) The items with this mark can appear several times in the configuration.

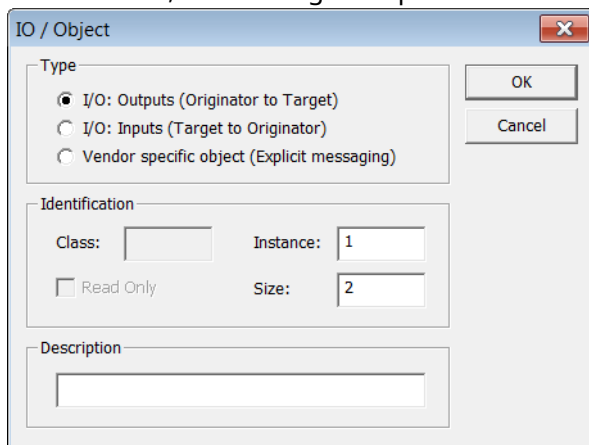
### Configuration

The following items can be configured at the root level.

Identifier	Meaning
Use LAN2	obsolete
IP Address	IP address of the Ethernet adapter used

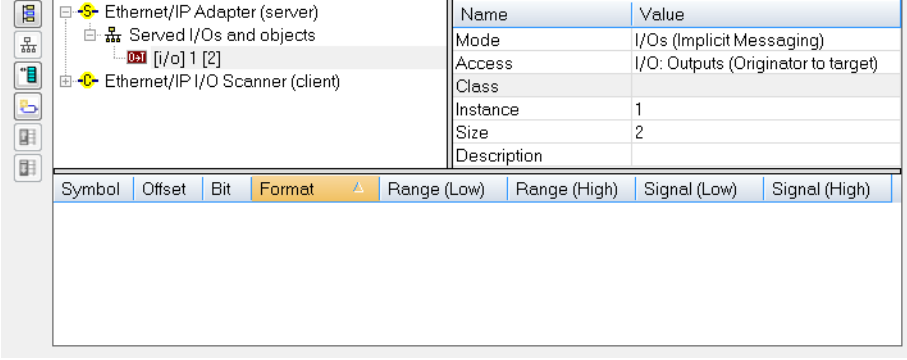
Click the Insert Master icon  to declare a server (adapter device). Each server is identified by its IP address and an optional description text.

Select the **Served I/Os and objects node**, then click the Insert Slave icon  to declare a CIP I/O assembly or a vendor specific object. Up to 4 input and 4 output assemblies are supported by the KAS Runtime, even though it is possible to create more in the KAS IDE.



Each assembly is identified by:

Identifier	Meaning
Mode	Kind of CIP object. Can be one of: <ul style="list-style-type: none"> <li>• I/O assembly</li> <li>• Vendor specific object</li> </ul>

Identifier	Meaning														
<b>Access</b>	In case of a vendor specific object, this property defines the access rights: <ul style="list-style-type: none"> <li>• Read/Write = free access</li> <li>• Read Only = the client (scanner) cannot write the object data</li> </ul>														
<b>Class</b>	CIP class in case of a vendor specific object. This field should be ignored in case of an I/O assembly. <div style="border: 1px solid gray; padding: 5px; margin-top: 10px;">  <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Name</th> <th>Value</th> </tr> </thead> <tbody> <tr> <td>Mode</td> <td>I/Os (Implicit Messaging)</td> </tr> <tr> <td>Access</td> <td>I/O: Outputs (Originator to target)</td> </tr> <tr> <td>Class</td> <td></td> </tr> <tr> <td>Instance</td> <td>1</td> </tr> <tr> <td>Size</td> <td>2</td> </tr> <tr> <td>Description</td> <td></td> </tr> </tbody> </table> </div>	Name	Value	Mode	I/Os (Implicit Messaging)	Access	I/O: Outputs (Originator to target)	Class		Instance	1	Size	2	Description	
Name	Value														
Mode	I/Os (Implicit Messaging)														
Access	I/O: Outputs (Originator to target)														
Class															
Instance	1														
Size	2														
Description															
<b>Instance</b>	Instance of the CIP assembly or object														
<b>Size</b>	Data size in bytes														
<b>Read Only</b>	Specify that a vendor-specific object is only readable by clients														
<b>Description</b>	Optional description text														

When defining a vendor specific objects, the following attributes are available for scanners:

1 (get only) = size of the object data

3 (get/set) = object data

Then you can map IEC61131-3 variables on the data of the assembly, for each variable you must specify:

Identifier	Meaning
<b>Symbol</b>	The name of the IEC61131-3 variable
<b>Offset</b>	Offset in bytes in the assembly data
<b>Bit</b>	Bit offset in the selected byte if format is "Bit"
<b>Format</b>	Format of the data in the assembly

**TIP**

You can drag a variable from the [Dictionary](#) directly to a slave item.

**NOTE**


The data limit is: 500 bytes of data maximum O(originator)->T(target) and 500 bytes of data maximum T(target) -> O(originator). This is based on the Ethernet/IP specification.

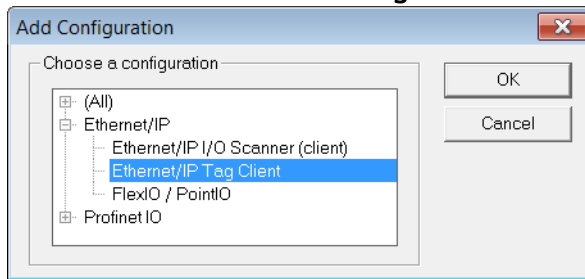
**12.3.8.4 Ethernet/IP Tag Client**

The KAS Runtime includes fully integrated Ethernet/IP client driver for exchanging tags with Ethernet/IP tag based devices such as PLCs.

**Data exchange - configuration**

A dedicated configuration tool is integrated in the KAS IDE.

1. Double-click the Fieldbus node in the project explorer to open it
2. Click the Insert Configuration icon  to add the Fieldbus configuration
3. Then select the **Ethernet/IP Tag Client** in the configuration selector




The configuration is represented as a tree:

- Ethernet/IP Tag Client
  - Server (an Ethernet/IP adapter device) (\*)
    - Tag (generally an array) (\*)
      - Exchanged variable (\*)


(\*) The items with this mark can appear several times in the configuration.

Driver and configurator are optimized for exchanging arrays (tags declared as arrays in the PLC). However it is also possible to exchange single tags.

### Configuration

Click the Insert Master icon  to declare an server (slave device). Each server is identified by its IP address and an optional description text.

Then you need to configure tags such as declared in the PLC:

- The easiest way is to right-click on the server in the tree and select the **Add ARRAY Tag** command in the contextual menu. Then you enter the properties of the tag request and the symbol of the corresponding array to be used in your IEC61131-3 application. Configuration of the tag and mapping of all array items is performed automatically.
- Alternatively you can click the Insert Slaver icon  to declare the tag and map some variables later on.

A tag request is identified by:

Identifier	Meaning
<b>Tag name</b>	The name of the tag such as declared in the PLC
<b>PLC Slot</b>	PLC slot number
<b>Mode</b>	Read or Write (note that the same tag can be configured twice for both reading and writing)
<b>Nb Elements</b>	Number of array items to read or write
<b>Offset</b>	O-based index of the first item to read or write in the array
<b>Tag data type</b>	Data type of the tag such as declared in the PLC. Available Types are: <ul style="list-style-type: none"> <li>▪ BOOL (single Boolean variable on 1 byte - 00=FALSE / FF=TRUE)</li> <li>▪ SINT (8 bit signed integer)</li> <li>▪ INT (16 bit signed integer)</li> <li>▪ DINT (32 bit signed integer)</li> <li>▪ DWORD (32 bit string)</li> </ul> <p>DWORD should be selected if the tag is declared in the PLC as an array of bits.</p>

Identifier	Meaning
Period(ms)	You can specify in this parameter a period for continuously sending the request. Enter "0" for a request sent "on demand"
Timeout	Request timeout in milliseconds

IEC61131-3 variables are mapped on the data of the tag, for each variable you must specify:

Identifier	Meaning
Symbol	The name of the IEC61131-3 variable
Offset	Offset in bytes in the assembly data
Bit	Bit offset in the selected byte if format is "Bit"
Format	Format of the data in the assembly
Mode	Kind of data exchanged through the variable: <ul style="list-style-type: none"> <li>• Data Exchange: a piece of input or output data in the assembly</li> <li>• Server OK: indicates the status of the IP connection to the server</li> <li>• Send Request Now: will be used as a command for activating the request</li> <li>• [transaction counter]: increased each time the request is sent</li> <li>• [general status]: CIP error code (0 = OK)</li> <li>• [extended status]: CIP extended error code (0 = OK)</li> </ul>

The tag will be read or written:

- periodically if a non zero period is specified in the tag configuration
- when a variable configured as "Send Request Now" becomes TRUE

In the case of a command variable, the variable is automatically reset to FALSE when the request is sent.

#### NOTE

The data limit is: 500 bytes of data maximum O(originator)->T(target) and 500 bytes of data maximum T(target) -> O(originator). This is based on the Ethernet/IP specification.

### 12.3.8.5 PDMM/PCMM EDS file for EtherNet/IP

The KAS installation contains an EtherNet/IP EDS file for the Kollmorgen controller. This file may be needed by other controllers, (PLCs, PCs etc.) to configure the EtherNet/IP communication with a AKD PDMM.

The `KAS_Controller_EIP.eds` file is located in the `\Astrolabe\Bin\EDS` directory, where the KAS software was installed. By default, the EDS file is located:

```
C:\Program Files (x86)\Kollmorgen\Kollmorgen Automation
Suite\Astrolabe\Bin\EDS\KAS_Controller_EIP.eds
```

Revision	Compatibility	Change
1.0	KAS v2.7 only	Initial revision
1.1	KAS v2.8 only	Updated vendor information
1.2	KAS v2.9 only	Support up to 4 input and output assemblies when KAS Runtime is configured as adapter (server), up from 1.

Table 7-3: Revision history for AKD PDMM EDS file for EtherNet/IP

#### 12.3.8.5.1 Using EDS Files



An EDS file may be required by a third party tool when the KAS controller is configured as an Ethernet/IP Adapter.

KAS adapter input/output images are configurable according to the user's need. A configuration matching the adapter must be defined on the scanner side.

#### NOTE

The data limit is: 500 bytes of data maximum O(originator)->T(target) and 500 bytes of data maximum T(target) -> O(originator). This is based on the Ethernet/IP specification.

The data block sizes in EDS files are predefined in bytes: 0, 1, 2, 4, 8, 16, 32, 64, 128, 256, 500. Use the closest existing configuration size that exceeds the needed amount of data. For example, if 40 bytes is sent O->T and 64 bytes is sent T->O, two 64 byte values must be selected.

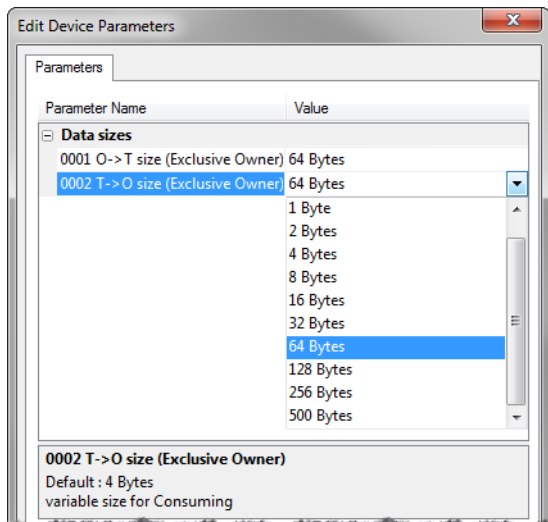


Figure 7-25: Example of setting byte sizes in a third-party Network Configurator.

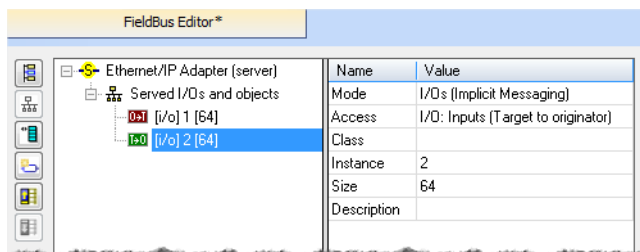



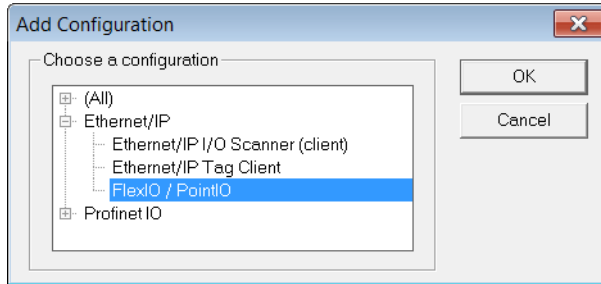
Figure 7-26: Example of setting byte sizes in the KAS IDE Fieldbus Editor


#### 12.3.8.6 Ethernet/IP FlexIO / PointIO

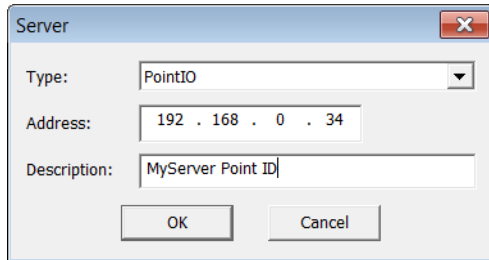
Before establishing the connection to the POINT IO, these modules require configuration. This is done through the WEB interface of the POINT IO bus coupler.


1. Double-click the Fieldbus node in the project explorer to open it
2. Click the Insert Configuration icon  to add the Fieldbus configuration

- Then select the **FlexIO/PointIO** driver in the configuration selector



- Click the Insert Master icon  and select PointIO



- Click the Insert Slave icon . Only modules in the list are supported. When inserting, the module variables can be declared automatically by checking Declare variables and set a prefix.

#### NOTE

Modules need to be inserted in the right order.

Configuration is ready and you can download the application to the KAS Runtime.

### 12.3.8.7 Modbus Slave

The KAS Runtime includes fully integrated slave functions for enabling Modbus communication on a serial link or Ethernet.

This communication is done in the background, asynchronously, at the cycle time (20-1000 milliseconds) specified in the Controller Properties (see ["Configure the Controller" on page 119](#)). Variables defined in the HMI to describe the interface (see [Map Variables to HMI](#)) are passed to the AKD PDMM/PCMM this way. This means there is no data coherency in the data exchange because the variables read by the Modbus do not come from the same PLC cycle. As this data has a rather low priority and is interpreted by human feedback, it should never be noticed by the user.

#### NOTE

Please note that Kollmorgen HMIs are limited to communicating no more frequently than every 100 milliseconds.

#### TIP

It is possible to have two Modbus master devices communicating with a controller at the same time, accessing the same variables. An example of this would be having two separate HMI panels running the same program on a single, large machine.

#### 12.3.8.7.1 Protocol specification

The protocol supported is Open Modbus on Ethernet. The following Modbus function codes are supported:

1	read coils
2	read bit inputs
3	read holding registers

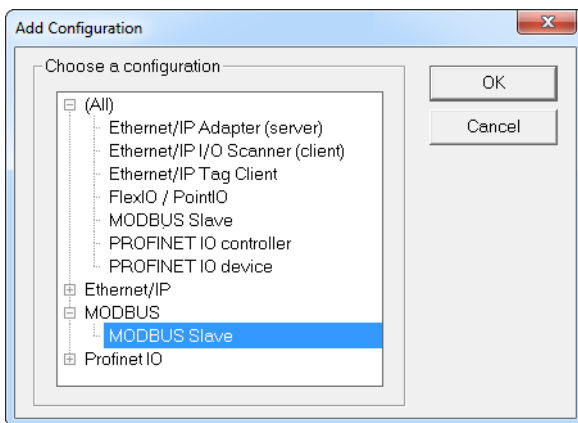
4	read input registers
5	write 1 coil
6	write 1 register
15	write $n$ coils
16	write $n$ registers

As a default, the first valid address for each kind of data is 1. If you use Modbus devices with other addressing conventions, you can change the base offset for each kind of data using the **Tools > Addresses** menu command.

### 12.3.8.7.2 Data exchange - configuration

A dedicated configuration tool is integrated in the KAS IDE. To run it,

1. Select the **Other Modbus devices** option in the Controller Properties dialog box. See [Configure the Controller](#) for more information.
2. Double-click the **Fieldbus** node in the project explorer to open it
3. Click the Insert Configuration icon to add the Fieldbus configuration
4. Then select the **Modbus Slave** in the configuration selector



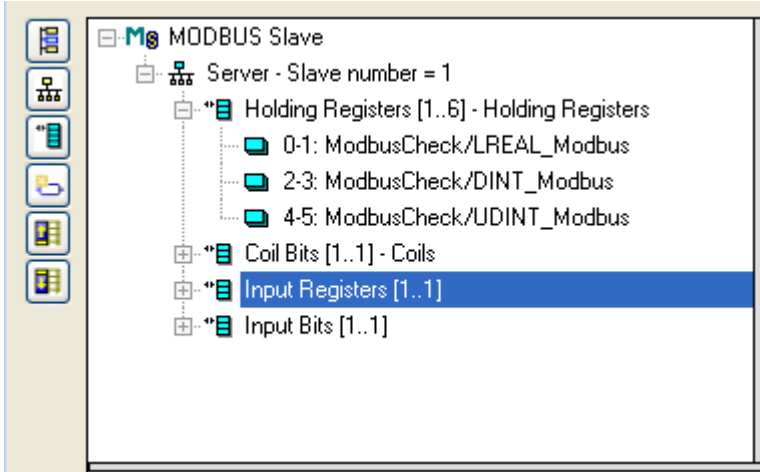
The Modbus Slave configuration is represented as a tree:

- Modbus Slave
  - Slave number (variables that can be accessed from external Modbus masters)
    - Input bits data block (read by masters)
      - Variable (\*)
    - Input words data block (read by masters)
      - Variable (\*)
    - Coil bits data block (forced by masters)
      - Variable (\*)
    - Holding bits data block (forced by masters)
      - Variable (\*)

(\*) The items with this mark can appear several times in the configuration.

### 12.3.8.7.3 Modbus Slave configuration

You need to configure the Modbus Slave in order to make variables visible from external Modbus masters such as SCADA systems. Below is a simple example of slave configuration:



Double click on the **Server** item to setup the Modbus slave number that will identify the runtime application. When the local server is selected, use the Insert Slave/Data Block menu command to insert Modbus data blocks. The following kinds of block are available:

- Input Bits: bits read by external masters (function 2).
- Coil Bits: bits forced by by external masters (function 5 or 15).
- Input Registers: words read by external masters (function 4).
- Holding Registers: words forced by external masters (function 6 or 16).

#### **TIP**

**Input Registers** require 15-25% less CPU time to exchange data than **Holding Registers** do. For optimal controller performance use Input Registers where possible.

Each data block is identified by a Modbus base address and a number of items (bits or words).

#### **NOTE**

Read and write requests sent by Modbus masters will be denied if the range specified in the request does not fit within a data block defined in the configuration. Requests overlapping two data blocks will be denied. For example, if you configure a block of 16 words starting at address 1 and another block of 16 words starting at address 17, a request for read or write of 32 words starting at address 1 will be denied and an "address error" exception will be reported.

When a server data block is selected, use the 'New symbol' command to map a variable to an item of the data block. Each variable is identified by a valid symbol of a variable in the open project and an offset in the data block according to Modbus addressing.

- For exchanging Boolean variables through Modbus words, a hexadecimal mask is available in order to define to which bit of a word a variable is attached. For example, enter the mask "0001" to map a Boolean variable to the less significant bit of a word.
- For exchanging 32 bit variables (DINT, REAL...), you can select to map the variable on two consecutive words.

You can sort the variables of each data block according to their offset using the 'Sort symbols' menu command at any time.

#### **12.3.8.7.4 Data types**

You can freely map a variable of any data type to a Modbus item. The Runtime automatically converts the value to the type of the variable.

- For exchanging Boolean variables through Modbus words, a hexadecimal mask is available in order to define to which bit of a word a variable is attached. For example, enter the mask "0001" to map a Boolean variable to the less significant bit of a word.
- For exchanging 32 bit variables (DINT, REAL...), you can select to map the variable on two consecutive words.

**NOTE**

- 64 bit variable (LINT and LREAL) cannot be extracted directly without lost of accuracy or data.

**12.3.8.8 Profinet IO RT Controller Configuration**

The KAS IDE contains a fully integrated configurator for Profinet IO RT Controller.

The Profinet maximum data size is 1440 bytes Input length and 1440 bytes Output length. To calculate the size, use the slot configuration view to count the number of bits per slot and convert to bytes. For example, 7 slots containing 16 outputs of 32 bits each, would be  $7 * 16 * 32 = 3584$  bits = 448 bytes.

**IMPORTANT**

Referring to the Profinet standard, the units of a Profinet network are named as IO Controllers (Masters) and IO Devices (Slaves).

**TIP**

To use some of the additional features of the Profinet controller fieldbus editor, such as "Browse Network for Slaves", you need to install the "CD Prot" driver. The installer can be found in [<Kollmorgen Automation Suite installation directory>\Bin\CDProtDriver](#).

Please note that your computer will need to be rebooted after installing the driver.

**NOTE**

Profinet is only supported on AKD PDMs and PCMMs.

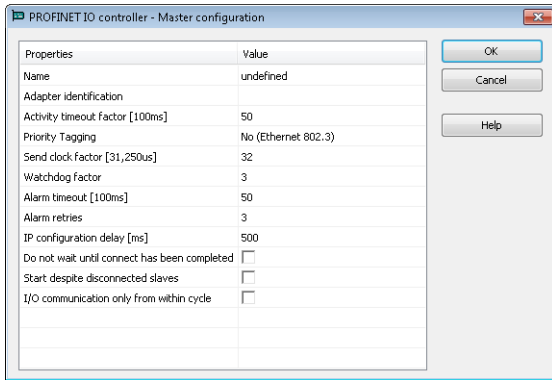
**NOTE**

Unlike an EtherCAT fieldbus which has a dedicated Ethernet interface, the Profinet fieldbus communicates through the same Ethernet interface as Modbus and other non-fieldbus communication. *There is no prioritization of Profinet over the other communication protocols, so frequent HTTP or IDE Oscilloscope communication can cause delay in the Profinet communications, and ultimately Profinet frame loss.*

**12.3.8.8.1 Configuration**

The I/Os of the Profinet network must be connected to the variables via a Profinet IO controller. Start the declaration of a Profinet controller with:

1. Insert > Insert Master/Port.  
The following window opens.



These parameter can be changed:

Parameter	Meaning
<b>Name</b>	<ul style="list-style-type: none"> <li>• A device name can consists of labels and must follow these conventions:</li> <li>• 1 or more labels, separated by [.]</li> <li>• Total length is 1 to 240</li> <li>• Label length is 1 to 63</li> <li>• Labels consist of [a...z0...9-]</li> <li>• Labels do not start with [-]</li> <li>• Labels do not end with [-]</li> <li>• The first label does not start with "port-xyz" or "port-xyz-abcde" with a,b,c,d,e, x, y, z = 0...9</li> <li>• Device names do not have the form n.n.n.n, n = 0...999</li> <li>• Labels do only start with 'xn-' if RFC 3490 is applied</li> </ul>
<b>Adapter identification</b>	MAC address of the Profinet IO controller or name of the network connection (Windows XP or younger and Windows CE 4.1 or younger). This must be set to <code>tsec0</code> .
<b>Activity timeout factor [100ms]</b>	Timeout for the connection establishment to the devices (maximum time between beginning of connection establishment and the first cyclic data exchange); Timeout factor based on 100ms.
<b>Priority Tagging</b>	Yes: cyclic data exchange without priority tag. No: data exchange with priority tag.
<b>Send clock factor [31,250µs]</b>	Send clock factor. Multiple of 31,250µs (32 = 1ms).
<b>Watchdog factor</b>	<p>Watchdog factor (default 3): The Watchdog factor defines how many frames may be missing until the device is set back. Watchdog triggers:</p> <ol style="list-style-type: none"> <li>frames may be lost; e.g. due to bad cabling.</li> <li>frames may arrive delayed due to blocking situations in the network; e.g. due to a non-separated network.</li> </ol> <p>The watchdog may be triggered as soon as HTTP file transfer occurs between the AKD PDMM or PCMM and the IDE when the default value is used. Increase this value if you encounter frequent watchdog triggers.</p>
<b>Alarm timeout [100ms]</b>	Alarm timeout (default 50).
<b>Alarm retries</b>	Number of alarm retries (default 3).

Parameter	Meaning
<b>IP configuration delay [ms]</b>	IP configuration delay (default 500): Defines the time to wait whether some devices are not ready after start.
<b>Do not wait until connect hat been competed</b>	On: the application starts immediately. Off: the application starts after all devices are connected.
<b>Start despite disconnected slaves</b>	On: the application starts despite configured but not found devices. Off: the application starts if all configured devices were found only.
<b>I/O communication only from within cycle</b>	On: Run I/O communication from within VM-cycle Off: Run I/O communication outside VM-cycle.

2. Mark the controller and click Insert > Insert Slave/Datablock to add a Profinet IO device.

Properties	Value
Name	<undefined>
Adapter MAC address	00:00:00:00:00:00
VLAN ID	0
IP address	0.0.0.0
Subnet mask	0.0.0.0
Default gateway	0.0.0.0
Object UUID	dea00000-6c97-11d1-8271-000000000000
Reduction ratio	1
Phase	1
Module IDs	
Module I/Os	
Module configuration	
Prefix	

These parameter can be changed:

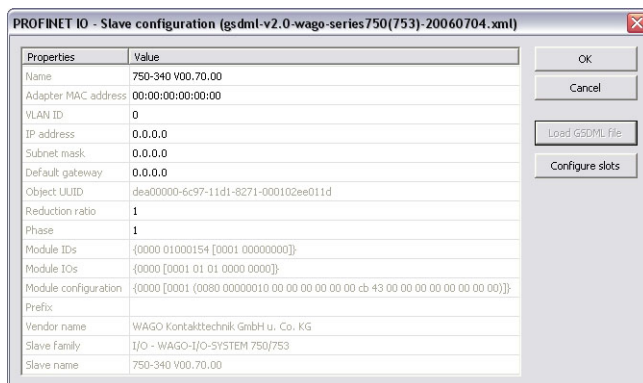
Parameter	Meaning
<b>Name</b>	<ul style="list-style-type: none"> <li>• A device name can consists of labels and must follow these conventions: <ul style="list-style-type: none"> <li>• 1 or more labels, separated by [.]</li> <li>• Total length is 1 to 240</li> <li>• Label length is 1 to 63</li> <li>• Labels consist of [a...z0...9-]</li> <li>• Labels do not start with [-]</li> <li>• Labels do not end with [-]</li> <li>• The first label does not start with "port-xyz" or "port-xyz-abcde" with a,b,c,d,e, x, y, z = 0...9</li> <li>• Device names do not have the form n.n.n.n, n = 0...999</li> <li>• Labels do only start with 'xn-' if RFC 3490 is applied</li> </ul> </li> </ul>
<b>Adapter MAC address</b>	MAC address of the Profinet IO device
<b>VLAN ID</b>	Virtual LAN ID
<b>IP address</b>	IP address of the device
<b>Subnet mask</b>	Subnet mask for the IP address of the device

Parameter	Meaning
<b>Default gateway</b>	Default gateway
<b>Object UUID</b>	UUID of the device
<b>Reduction ratio</b>	Reduction ratio (default 16): The Reduction ratio defines the frequency for data to be exchanged with the device. The transfer rate is calculated by $\langle \text{Send clock factor} \rangle * 31,250 \mu\text{s} * \langle \text{Reduction ratio} \rangle$ . The Send clock factor is a master parameter with a default value of 32 (all known devices work with this Send clock factor). With the default setting of 16 the data transfer rate is 48ms; for 2 it is 2ms...; Most devices support Reduction ratio settings of 1,2,4,8,16,32...
<b>Phase</b>	Phase
<b>Module IDs</b>	Module IDs of the device modules.
<b>Module IOs</b>	Module IOs of the device modules.
<b>Module configuration</b>	Module configuration.
<b>Prefix</b>	Prefix for the variables.

**NOTE**

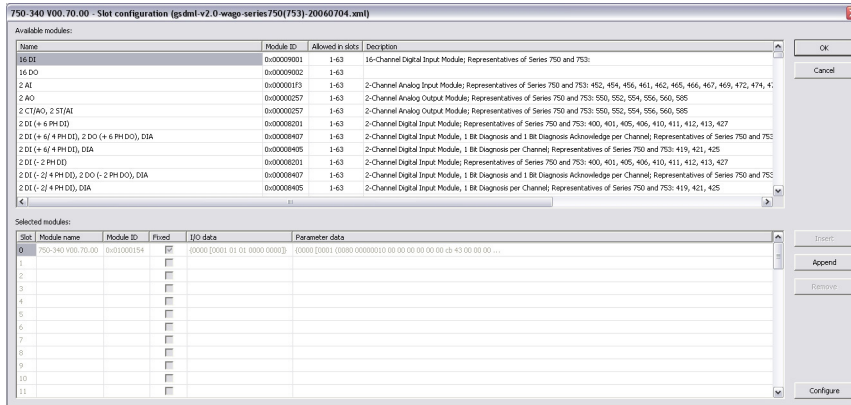
With the calculation of  $\langle \text{Watchdog factor} \rangle * \langle \text{Reduction ratio} \rangle * \langle \text{Send clock factor} \rangle * 31,250 \mu\text{s}$  you get the time that may expire between two frames until the device is set back. I.e. for the default settings  $3 * 32 * 31,250 \mu\text{s}$  the connection will be set back after 3ms of missing frames. For office networks this time is rather low. It is recommended to use a Reduction ratio of 32 and a Watchdog factor of 24 for such networks. I.e. the connection will be reset after  $24 * 32 * 32 * 31,250 \mu\text{s} = 768 \text{ ms}$ . If this data exchange rate is too low please separate the PROFINET IO network from the office network (e.g. by a router).

3. Click Load GSDML file to import the necessary GSDML file.





4. Do the slot configuration after the GSDML file import.



5. Select the modules in the upper list. With the buttons **Insert** and **Append** the modules are copied to the lower list.

You can not configure each module. Only modules with some sub modules respectively with a sub module with parameter data can be configured. Mark the according module in the lower list and click the **Configure** button.

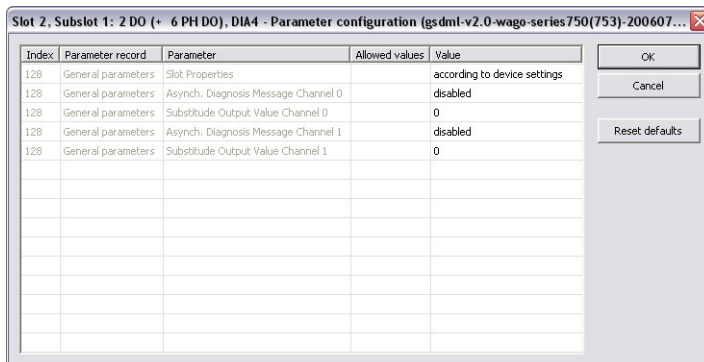
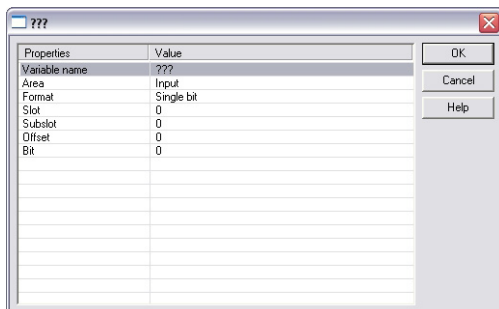


Figure 7-27: Example of configuring sub-modules.

6. Now you can connect the variables with the I/Os. Use **Insert > Insert/Set Variable** in order to append a variable to a device.



These parameter can be changed:

Parameter	Description
<b>Variable name</b>	Variable name following the IEC 61131-3 syntax.
<b>Area</b>	Output, Output IOCS, Output IOPS, Input, Input IOCS, Input IOPS, device status, or PNIO status.

Parameter	Description
Format	32 bit float, Signed 16 bit integer, Signed 32 bit integer, Signed 8 bit integer, Single bit, Unsigned 16 bit integer, Unsigned 32 bit integer, Unsigned 8 bit integer.
Slot	Slot Number
Subslot	Subslot Number
Offset	Offset
Bit	Bit

**NOTE**  
 The offset of a variable is relative to a sub module. Thus also depending from a slot and subslot. The offset of the first variable of a sub module is always 0.

- All settings can be changed in the grid too. The information show refers to the items below of the selected item in the configuration tree.

Variable name	Area	Format	Slot	Subslot	Offset	Bit
_0_Slot14_Subslot1_Output0	Output	Signed 16 bit integer	14	1	0	0
_0_Slot15_Subslot1_Output0	Output	Signed 16 bit integer	15	1	0	0
_0_Slot16_Subslot1_Output0	Output	Signed 16 bit integer	16	1	0	0
_0_Slot17_Subslot1_Output0	Output	Signed 16 bit integer	17	1	0	0
_0_Slot18_Subslot1_Output0	Output	Signed 16 bit integer	18	1	0	0
_0_Slot19_Subslot1_Output0	Output	Signed 16 bit integer	19	1	0	0
_0_Slot20_Subslot1_Output0	Output	Signed 16 bit integer	20	1	0	0
_0_Slot2_Subslot1_Output0	Output	Unsigned 8 bit integer	2	1	0	0
_0_Slot3_Subslot1_Input0	Input	Unsigned 8 bit integer	3	1	0	0
_0_Slot4_Subslot1_Input0	Input	Unsigned 8 bit integer	4	1	0	0
_0_Slot5_Subslot1_Input0	Input	Unsigned 8 bit integer	5	1	0	0
_0_Slot6_Subslot1_Input0	Input	Unsigned 8 bit integer	6	1	0	0
_0_Slot7_Subslot1_Input0	Input	Unsigned 8 bit integer	7	1	0	0
_0_Slot8_Subslot1_Input0	Input	Unsigned 8 bit integer	8	1	0	0
_0_Slot8_Subslot1_Output0	Output	Unsigned 8 bit integer	8	1	0	0
_0_Slot1_Subslot1_Input0	Input	Unsigned 16 bit integer	1	1	0	0
_0_Slot14_Subslot1_Output1	Output	Signed 16 bit integer	14	1	2	0
_0_Slot15_Subslot1_Output1	Output	Signed 16 bit integer	15	1	2	0
_0_Slot16_Subslot1_Output1	Output	Signed 16 bit integer	16	1	2	0
_0_Slot17_Subslot1_Output1	Output	Signed 16 bit integer	17	1	2	0

If the GUI Views is online with a target system the grid shows the real-time data of the variables.

The Profinet maximum data size is 1440 bytes Input length and 1440 bytes Output length. To calculate the size, use the slot configuration view to count the number of bits per slot and convert to bytes. For example, 7 slots containing 16 outputs of 32 bits each, would be  $7 * 16 * 32 = 3584$  bits = 448 bytes.

### 12.3.8.8.2 Data types

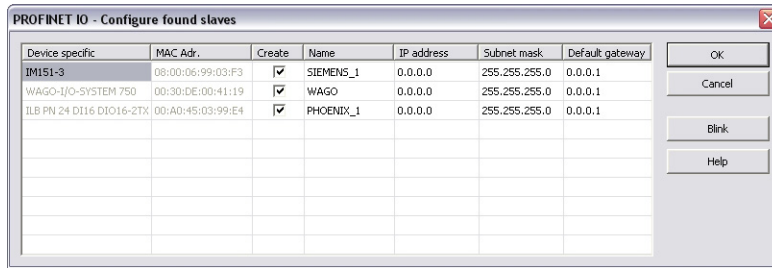
You can connect variables of any data type to the Profinet I/Os. The Runtime converts the values of the I/Os to the type of the variable. STRING variables are not supported.

### 12.3.8.8.3 Additional features

**TIP**  
 To use some of the additional features of the Profinet controller fieldbus editor, such as "Browse Network for Slaves", you need to install the "CD Prot" driver. The installer can be found in [<Kollmorgen Automation Suite installation directory>\Bin\CDProtDriver](#). Please note that your computer will need to be rebooted after installing the driver.

#### 12.3.8.8.3.1 Browse network for slaves

This command is available in the context menu of the network adapter. All connected Profinet IO devices are listed up.



Select the devices you want to add in the tree via the checkbox in the column `Create`.

The Network Link (or another) -LED of the marked device blinks for three seconds when pressing the `Blink` button.

Set the device names (column `Name`) here. A device name can consist of labels and must follow these conventions:

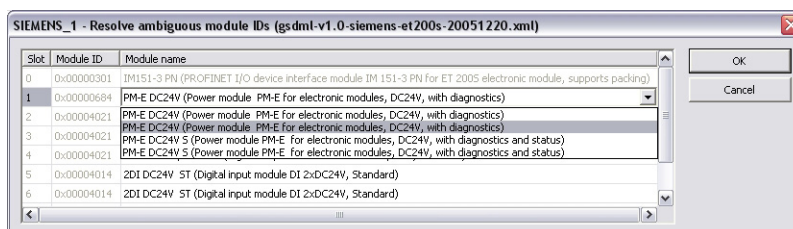
- 1 or more labels, separated by [.,].
- Total length is 1 to 240.
- Label length is 1 to 63.
- Labels consist of [a...z0...9-].
- Labels do not start with [-].
- Labels do not end with [-].
- The first label does not start with "port-xyz" or "port-xyz-abcde" with a,b,c,d,e, x, y, z = 0...9.
- Device names do not have the form n.n.n.n, n = 0...999.
- Labels do only start with 'xn-' if RFC 3490 is applied.

Set the IP address of the device here. By clicking into the grid the Default gateway will be set automatically.

Click `OK` after setting the IP-parameters (address, subnet mask, default gateway).

### 12.3.8.8.3.2 Configuration of devices

After adding the Profinet devices to the controller they need to be configured. Double click on the referring device. Load the GSDML file. If ambiguous module IDs are found they are shown in a window. Choose the right module in the combo-box.



#### NOTE

Automatic creation of variables can be done primal after this step.

### 12.3.8.8.3.3 Set slave station name

With this context menu command it is possible to rename the slave names.

### 12.3.8.8.3.4 Read module configuration

With this context menu command it is possible to read out the module configuration again.

### 12.3.8.8.3.5 Create variables

Find this command in the context menu of the device. Based on the defined device modules the referring variables are generated.

Within the same dialog it is possible to define the variables for

- Device diagnosis
- IOxS

#### 12.3.8.8.3.6 Device diagnosis

Based on the Profinet standard the referring variables can be generated:

- CycleCounter [UINT].
- Status [BOOL].
- DataValid [BOOL].
- ProviderState [BOOL].
- StationsProblemIndikator [BOOL].

#### 12.3.8.8.3.7 Create IOxS for slave modules

Based on the defined device modules the referring IOPS- and IOCS-variables are generated.

#### 12.3.8.8.4 How to Resolve Errors

##### 12.3.8.8.4.1 Device is not found

- Check if device is switched on
- Check the network connection
- Ensure the correct name was set on the device

##### 12.3.8.8.4.2 Error setting the IP configuration

- Ensure that the IP configuration is valid and appropriate for your network
- Ensure the VLAN ID is setup correctly in the device settings

##### 12.3.8.8.4.3 Timeout error

- Ensure that the IP configuration is valid and appropriate for your network
- Increase the IP configuration delay

##### 12.3.8.8.4.4 Other errors

- Ensure you use the correct device name
- Ensure you have chosen the correct GSDML file (also check if the GSDML file version is appropriate for the firmware version of the device)

##### 12.3.8.8.4.5 Connect response error

Depending on the PNIO status error check, if the settings listed in the table below are set appropriate for your device.

PNIO status	Check the specified setting
1C010003	Slave\ObjectUUID (correct GDML file)
DB81010A	Master\Activity timeout factor
DB81010B	Master\Controller name
DB81010C	Master\Controller name
DB810207	Master\RT-Class
DB81020A	Master\Send clock factor
DB81020B	Slave\Reduction ratio
DB81020C	Slave\Phase
DB81020F	Master\Watchdog factor
DB810210	Master\Watchdog factor
DB8103,*	Slave\Module IDs,IOs
DB810407	Master\Alarm timeout

PNIO status	Check the specified setting
DB810408	Master\Alarm retries

Table 7-4: PNIO status error codes on connect and the related settings in the configuration

#### 12.3.8.8.4.6 Module configuration is different

- Ensure you have configured the modules present on the device (If you have read the module configuration from the device, ensure you have selected the correct modules in the resolve ambiguous modules dialog displayed after selecting the GSDML file).
- Ensure you have chosen the correct GSDML file (also check if the GSDML file version is appropriate for the firmware version of the device).

#### 12.3.8.8.4.7 Writing parameterization error (with status 0xDF80\*)

- Ensure you have configured the modules present on the device (If you have read the module configuration from the device, ensure you have selected the correct modules in the resolve ambiguous modules dialog displayed after selecting the GSDML file).
- Ensure you have chosen the correct GSDML file (also check if the GSDML file version is appropriate for the firmware version of the device).

If your error could not be resolved using the hints above, update the firmware of your device and the GSDML file to the latest version and try again.

For further analyses record the PNIO communication using [Wireshark](http://www.wireshark.org/) (<http://www.wireshark.org/>).

- Start Wireshark and select Capture\Options from the menu.
- Choose the network interface used for the PNIO communication and enter the Capture filter below:

```
ether proto 0x8892 or (ether proto 0x8100 and ether[16:2]=0x8892) or
udp port 0x8894
```

- Then click start.
- Start the runtime with your application several times and then stop the Wireshark capturing by selecting Capture\Stop from the menu. You can save your recorded traffic by selecting File\Save from the menu.

#### 12.3.8.8.5 Coding of PNIO status

The PNIO status is a 32 bit unsigned value. It is composed of 4 byte values, which define the meaning of the status. For positive responses PNIO status has a value of zero.

Bit	Meaning
24-31	ErrorCode
16-23	ErrorDecode
8-15	ErrorCode1
0-7	ErrorCode2

Table 7-5: Coding of PNIO status for negative responses

ErrorCode	Meaning
DB	Error in connect response
DC	Error in release response
DD	Error in control response
DF	Error in write response

Table 7-6: Meaning of ErrorCode for negative responses

ErrorDecode	Meaning
80	Read/Write service
81	Connect, Control, Release service

Table 7-7: Meaning of ErrorDecode for negative responses

ErrorCode1	Meaning
A1	write error
A2	module failure
A3-A6	reserved
A7	busy
A8	version conflict
A9	feature not supported
AA-AF	device specific
B0	invalid index
B1	write length error
B2	invalid slot/subslot
B3	type conflict
B4	invalid area/API
B5	state conflict
B6	access denied
B7	invalid range
B8	invalid parameter
B9	invalid type
BA	backup
BB-BF	device specific
C0	read constrain conflict
C1	write constrain conflict
C2	resource busy
C3	resource unavailable
C4-C7	reserved
C8-CF	device specific

Table 7-8: Meaning of ErrorCode1 for ErrorDecode = 80

ErrorCode1	Meaning
01	Connect Parameter Error, Faulty ARBlockReq
02	Connect Parameter Error, Faulty IOCRBlockReq
03	Connect Parameter Error, Faulty ExpectedSubmoduleBlockReq
04	Connect Parameter Error, Faulty AlarmCRBlockReq
05	Connect Parameter Error, Faulty PrmServerBlockReq
06	Connect Parameter Error, Faulty MCRBlockReq
07	Connect Parameter Error, Faulty ARRPCBlockReq
08	Read Write Record Parameter, Error Faulty Record

ErrorCode1	Meaning
14	IODControl Parameter Error, Faulty ControlBlockConnect
15	IODControl Parameter Error, Faulty ControlBlockPlug
16	IOXControl Parameter Error , Faulty ControlBlock after a connection establishment
17	IOXControl Parameter Error, Faulty ControlBlock after a plug alarm
28	Release Parameter Error, Faulty ReleaseBlock
40	RMPM (Device state machines, device resources)

Table 7-9: Meaning of ErrorCode1 for ErrorDecode = 81

ErrorCode1	Meaning
00	ArgsLength invalid
01	Unknown Blocks
02	IOCR Missing
03	Wrong AlarmCRBlock count
04	Out of AR Resources
05	AR UUID unknown
06	State conflict
07	Out of Provider, Consumer, or Alarm Resources
08	Out of Memory
09-FF	Reserved

Table 7-10: Meaning of ErrorCode2 for ErrorCode1 = 40

For ErrorCode1 <> 40 ErrorCode2 refers to a field in the block specified by ErrorCode1.

ErrorCode2	Meaning
00	Block type
01	Block length
02	Block version (high byte)
03	Block version (low byte)

Table 7-11: Meaning of ErrorCode2 for ErrorCode1 &lt;&gt; 40

ErrorCode2	Meaning
04	AR Type
05	AR UUID
06	Session key
07	Initiator MAC address
08	Initiator Object UUID
09	AR Properties
0A	Activity timeout factor
0B	UDP RT port
0C	Station name length
0D	Station name

Table 7-12: Meaning of ErrorCode2 for ErrorCode1 = 01 (AR block request)

ErrorCode2	Meaning
04	IOCR Type
05	Reference
06	LT
07	IOCR properties
08	IO data length
09	Frame ID
0A	Send clock factor
0B	Reduction ratio
0C	Phase
0D	Sequence
0E	Frame send offset
0F	Watchdog factor
10	Data hold factor
11	Tag header
12	IOCR multicast MAC address
13	Number of APIs
14	API
15	Number of IO data objects
16	Slot
17	Subslot
18	IO data object offset
19	Number of IOCS
20	Slot
21	Subslot
22	IOCS offset

Table 7-13: Meaning of ErrorCode2 for ErrorCode1 = 02 (IOCR block request)

ErrorCode2	Meaning
04	Number of APIs
05	API
06	Slot
07	Module ident number
08	Module properties
09	Number of submodules
0A	Subslot
0B	Submodule ident number
0C	Submodule properties
0D	Data description
0E	Data length
0F	IOPS length
10	IOCS length

Table 7-14: Meaning of ErrorCode2 for ErrorCode1 = 03 (Expected submodule block request)



ErrorCode2	Meaning
04	Type
05	LT
06	AlarmCR Properties
07	RTA timeout factor
08	RTA retries
09	Local alarm reference
0A	Maximum alarm data length
0B	Alarm CRT Tag high
0C	Alarm CRT Tag low

Table 7-15: Meaning of ErrorCode2 for ErrorCode1 = 04 (AlarmCR block request)

ErrorCode2	Meaning
04	Sequence number
05	AR UUID
06	API
07	Slot number
08	Subslot number
09	Padding
0A	Index
0B	Data length
0C	Target AR UUID

Table 7-16: Meaning of ErrorCode2 for ErrorCode1 = 8 (Read/write record block request)

ErrorCode2	Meaning
05	Padding
06	Session key
07	Padding
08	Control block command
09	Control block properties

Table 7-17: Meaning of ErrorCode2 for ErrorCode1 = 16 (IOXControl block request)

### 12.3.8.9 Profinet IO RT Device Configuration

The KAS IDE contains a fully integrated configurator for Profinet IO RT Device.

The Profinet maximum data size is 1440 bytes Input length and 1440 bytes Output length. To calculate the size, use the slot configuration view to count the number of bits per slot and convert to bytes. For example, 7 slots containing 16 outputs of 32 bits each, would be  $7 * 16 * 32 = 3584$  bits = 448 bytes.

#### ⓘ IMPORTANT

Referring to the Profinet standard, the units of a Profinet network are named as IO Controllers (Masters) and IO Devices (Slaves).

#### NOTE

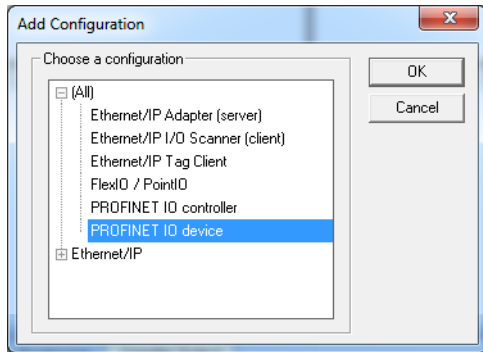
Profinet is only supported on AKD PDMMs and PCMMs.

### 12.3.8.9.1 ProfinetIO RT Device configuration

The Runtime manages a mapping table which contains the Profinet IO Inputs and Outputs. An appropriate configuration tool is integrated in the KAS.

To start the configuration:

1. Open the fieldbus configuration window.
2. Right click on the windows and choose **Insert > Insert Network**. The following window opens.



3. Select **PROFINET IO device** and click **OK**.

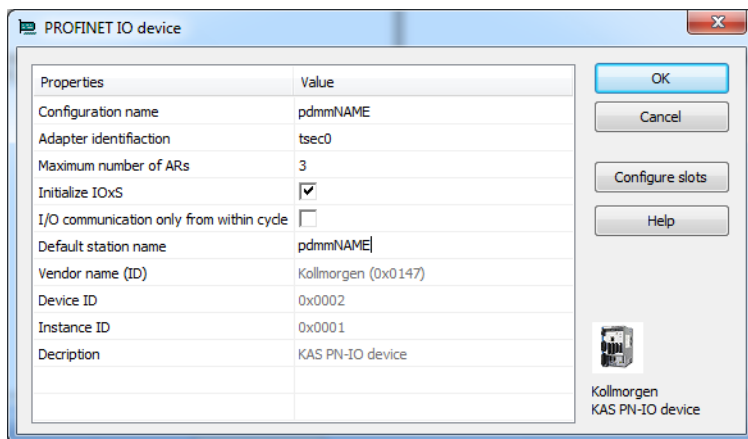
The configuration is represented as a tree:

- Profinet IO Configuration
  - Profinet IO device (\*)
    - Group (\*)
      - Variable (\*)

(\*) These items can appear several times in the configuration (depending on the bus topology).

The I/Os of the Profinet network must be connected to the variables via a Profinet IO device.

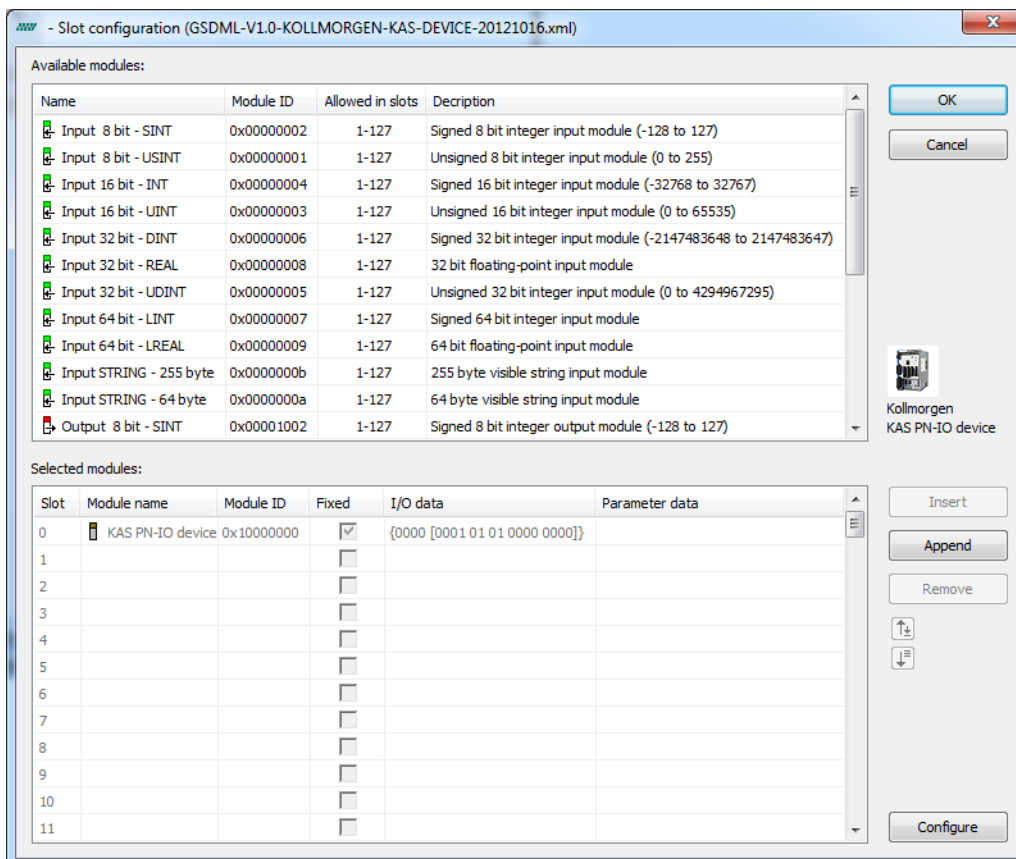
4. Start the declaration of a Profinet device by right clicking on the PROFINET IO device network and choosing **Insert Master**. The following window opens:



These parameters can be changed:

Parameter	Meaning
<b>Name</b>	<p>A device name can consists of labels and must follow these conventions:</p> <ul style="list-style-type: none"> <li>• 1 or more labels, separated by [.]</li> <li>• Total length is 1 to 240</li> <li>• Label length is 1 to 63</li> <li>• Labels consist of [a . . . z 0 . . . 9 -]</li> <li>• Labels do not start with [-]</li> <li>• Labels do not end with [-]</li> <li>• The first label does not start with "port-xyz" or "port-xyz-abcde" with a,b,c,d,e, x, y, z = 0...9</li> <li>• Device names do not have the form n.n.n.n.n, n = 0 . . . 999</li> <li>• Labels do only start with 'xn-' if RFC 3490 is applied</li> </ul>
<b>Adapter identification</b>	Must be "tsec0"
<b>Maximum # of ARs</b>	Maximum number of alarm retries (default 3).
<b>Initialize IOxS</b>	<p>On: Initialize IOxS with good status.                      Off: No initialization of IOxS.</p>
<b>IO communication only from within cycle</b>	<p>On: Run IO communication from within VM-cycle                      Off: Run IO communication outside VM-cycle.</p>
<b>Default station name</b>	Name of the station.

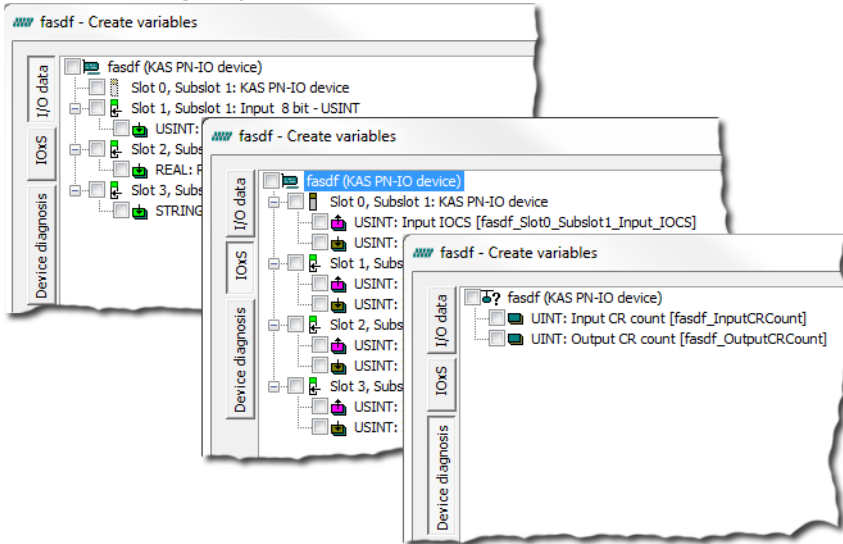
5. Click on Configure Slots.



6. Select the modules in the upper list. With the buttons **Insert** and **Append** the modules are copied to the lower list.

You can not configure each module. Only modules with some sub modules respectively with a sub module with parameter data can be configured. Mark the according module in the lower list and click the **Configure** button.

7. Click OK to close the Slot Configuration and click OK to close the device window.
8. Right click on the master and select Create Variables. This will automatically populate the variables and groups.



**! IMPORTANT**

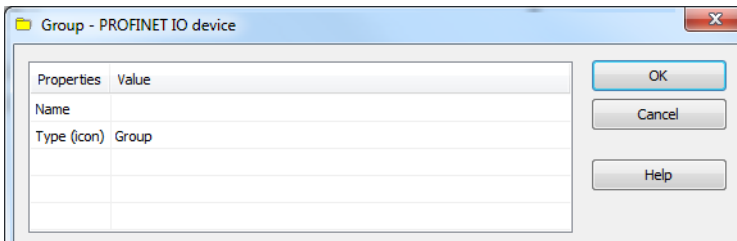
Every Profinet variable is expanded to a set of Boolean variables in PLC by default. A SINT slot, for example, will be mapped to eight PLC BOOL variables. Therefore, if you have many configured slots, many PLC variables will be produced. The AKD PDMM or PCMM will be slowed by a large amount of PLC variables.

To avoid this, you can right click on a slot in the *Create variables* dialog and select *Pack bits*. Doing so with a SINT slot, for example, will create one SINT variable in KAS instead of eight BOOL variables. This will help reduce the number of PLC variables and reduce the load on the AKD PDMM or PCMM.

**↖ TIP**

The *Pack bits* action may be applied to all slots by right clicking on the root node in the *Create variables* dialog.

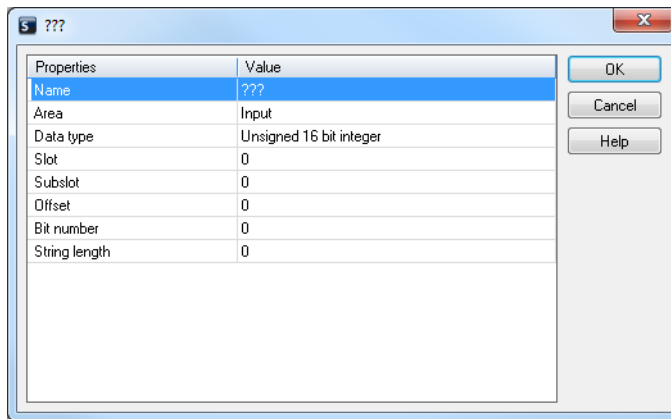
9. Mark the device and right click on it, then choose **Insert Slave** to add a group.



These parameters can be changed:

Parameter	Meaning
Name	Name of the group
Type (icon)	Icon used for the group

10. Now you can connect the variables with the I/Os. Right click on the group and choose **Insert Symbol** in order to append a variable to a device.



These parameters can be changed:

Parameter	Description
<b>Variable name</b>	Variable name following the IEC 61131-3 syntax.
<b>Area</b>	Output, Output IOCS, Output IOPS, Input, Input IOCS, Input IOPS, device status.
<b>Format</b>	32 bit float, Signed 16 bit integer, Signed 32 bit integer, Signed 8 bit integer, Single bit, Unsigned 16 bit integer, Unsigned 32 bit integer, Unsigned 8 bit integer.
<b>Slot</b>	Slot Number
<b>Subslot</b>	Subslot Number
<b>Offset</b>	Offset
<b>Bit</b>	Bit

The offset of a variable is relative to a sub module. Thus also depending from a slot and subslot. The offset of the first variable of a sub module is always 0.

All settings can be changed in the grid too. The information show refers to the items below of the selected item in the configuration tree.

Name	Area	Data type	Slot	Subslot	Offset	Bit number
Broadcom_Slot1_Subslot1_Input0_Bit0	Input	Single bit	1	1	0	0
Broadcom_Slot1_Subslot1_Input0_Bit1	Input	Single bit	1	1	0	1
Broadcom_Slot1_Subslot1_Input0_Bit2	Input	Single bit	1	1	0	2
Broadcom_Slot1_Subslot1_Input0_Bit3	Input	Single bit	1	1	0	3
Broadcom_Slot1_Subslot1_Input0_Bit4	Input	Single bit	1	1	0	4
Broadcom_Slot1_Subslot1_Input0_Bit5	Input	Single bit	1	1	0	5
Broadcom_Slot1_Subslot1_Input0_Bit6	Input	Single bit	1	1	0	6
Broadcom_Slot1_Subslot1_Input0_Bit7	Input	Single bit	1	1	0	7
Broadcom_Slot0_Subslot1_Input_IOCS	Input IOCS	Unsigned 8 bit int...	0	1	0	0
Broadcom_Slot0_Subslot1_Input_IOPS	Input IOPS	Unsigned 8 bit int...	0	1	0	0
Broadcom_Slot1_Subslot1_Input_IOCS	Input IOCS	Unsigned 8 bit int...	1	1	0	0
Broadcom_Slot1_Subslot1_Input_IOPS	Input IOPS	Unsigned 8 bit int...	1	1	0	0

If KAS is connected to a target system and the system is running, the grid shows the real-time data of the variables.

The Profinet maximum data size is 1440 bytes Input length and 1440 bytes Output length. To calculate the size, use the slot configuration view to count the number of bits per slot and convert to bytes. For example, 7 slots containing 16 outputs of 32 bits each, would be  $7 * 16 * 32 = 3584$  bits = 448 bytes.

### 12.3.8.9.2 Data types

You can connect variables of any data type to the Profinet I/Os. The Runtime converts the values of the I/Os to the type of the variable. **STRING** variables are not supported.

### 12.3.8.9.3 Additional features

#### 12.3.8.9.3.1 Create Variables

Find this command in the context menu of the device. Based on the defined device modules the referring variables are generated.

Within the same dialog it is possible to define the variables for

- Device diagnosis
- IOxS

#### 12.3.8.9.3.2 Device Diagnosis

This retrieves the device state information. Based on the Profinet standard the referring variables can be generated:

- InputCRCCount [UINT].
- OutputCRCCount [UINT].

#### 12.3.8.9.3.3 Create IOxS for Slave Modules

Based on the defined device modules the referring IOPS- and IOCS-variables are generated.

Input modules/modules without IO data:

- The IOPS state is managed by the device. This is the status sent by the module. If the data is invalid, the controller has to ignore it.
- The IOCS state is managed by the controller. The controller can indicate if it can't handle the data.

Output modules:

- The IOPS state is managed by the controller it reflects the status of the data sent by the controller. If the data is invalid the device has to ignore it
- The IOCS state is managed by the device. Le device can indicate to the controller that the data can't be handled.

### 12.3.8.9.4 How to resolve errors

#### 12.3.8.9.4.1 Device is not found

- Check if device is switched on
- Check the network connection
- Ensure the correct name was set on the device

#### 12.3.8.9.4.2 Error setting the IP configuration

- Ensure that the IP configuration is valid and appropriate for your network
- Ensure the VLAN ID is setup correctly in the device settings

#### 12.3.8.9.4.3 CL-RPC Lookup

### 12.3.9 Timeout error

- Ensure that the IP configuration is valid and appropriate for your network
- Increase the IP configuration delay

#### 12.3.9.0.0.1 Other errors

- Ensure you use the correct device name
- Ensure you have chosen the correct GSDML file (also check if the GSDML file version is appropriate for the firmware version of the device)

### 12.3.9.0.0.2 Connect response error

### 12.3.10 Timeout error

- Ensure that the IP configuration is valid and appropriate for your network
- Increase the IP configuration delay

#### 12.3.10.0.0.1 Connect response error

Depending on the PNIO status error check, if the settings listed in the table below are set appropriate for your device.

PNIO Status	Check the specified setting
1C010003	Slave\ObjectUUID (correct GDML file)
DB81010A	Master\Activity timeout factor
DB81010B	Master\Controller name
DB81010C	Master\Controller name
DB810207	Master\RT-Class
DB81020A	Master\Send clock factor
DB81020B	Slave\Reduction ratio
DB81020C	Slave\Phase
DB81020F	Master\Watchdog factor
DB810210	Master\Watchdog factor
DB8103,*	Slave\Module IDs,I/Os
DB810407	Master\Alarm timeout
DB810408	Master\Alarm retries

Table 7-18: PNIO status error codes on connect and the related settings in the configuration

#### 12.3.10.0.0.2 Module configuration is different

- Ensure you have configured the modules present on the device (If you have read the module configuration from the device, ensure you have selected the correct modules in the resolve ambiguous modules dialog displayed after selecting the GSDML file).
- Ensure you have chosen the correct GSDML file (also check if the GSDML file version is appropriate for the firmware version of the device).

#### 12.3.10.0.0.3 Writing parameterization error (with status 0XDF80\*)

- Ensure you have configured the modules present on the device (If you have read the module configuration from the device, ensure you have selected the correct modules in the resolve ambiguous modules dialog displayed after selecting the GSDML file).
- Ensure you have chosen the correct GSDML file (also check if the GSDML file version is appropriate for the firmware version of the device).

If your error could not be resolved using the hints above, update the firmware of your device and the GSDML file to the latest version and try again.

For further analyses record the PNIO communication using [Wireshark](http://www.wireshark.org/) (<http://www.wireshark.org/>).

- Start Wireshark and select Capture\Options from the menu.
- Choose the network interface used for the PNIO communication and enter the Capture filter below:

```
ether proto 0x8892 or (ether proto 0x8100 and ether[16:2]=0x8892) or
udp port 0x8894
```

- Then click start.
- Start the runtime with your application several times and then stop the Wireshark capturing by selecting Capture\Stop from the menu. You can save your recorded traffic by selecting File\Save from the menu.

### 12.3.10.0.1 Coding of PNIO status

The PNIO status is a 32 bit unsigned value. It is composed of 4 byte values, which define the meaning of the status. For positive responses PNIO status has a value of zero.

Bit	Meaning
24-31	ErrorCode
16-23	ErrorDecode
8-15	ErrorCode1
0-7	ErrorCode2

Table 7-19: Coding of PNIO status for negative responses

ErrorCode	Meaning
DB	Error in connect response
DC	Error in release response
DD	Error in control response
DF	Error in write response

Table 7-20: Meaning of ErrorCode for negative responses

ErrorDecode	Meaning
80	Read/Write service
81	Connect, Control, Release service

Table 7-21: Meaning of ErrorDecode for negative responses

ErrorCode1	Meaning
A1	write error
A2	module failure
A3-A6	reserved
A7	busy
A8	version conflict
A9	feature not supported
AA-AF	device specific
B0	invalid index
B1	write length error
B2	invalid slot/subslot
B3	type conflict
B4	invalid area/API
B5	state conflict
B6	access denied



ErrorCode1	Meaning
B7	invalid range
B8	invalid parameter
B9	invalid type
BA	backup
BB-BF	device specific
C0	read constrain conflict
C1	write constrain conflict
C2	resource busy
C3	resource unavailable
C4-C7	reserved
C8-CF	device specific

Table 7-22: Meaning of ErrorCode1 for ErrorDecode = 80

ErrorCode1	Meaning
01	Connect Parameter Error, Faulty ARBlockReq
02	Connect Parameter Error, Faulty IOCRBlockReq
03	Connect Parameter Error, Faulty ExpectedSubmoduleBlockReq
04	Connect Parameter Error, Faulty AlarmCRBlockReq
05	Connect Parameter Error, Faulty PrmServerBlockReq
06	Connect Parameter Error, Faulty MCRBlockReq
07	Connect Parameter Error, Faulty ARRPCBlockReq
08	Read Write Record Parameter, Error Faulty Record
14	IODControl Parameter Error, Faulty ControlBlockConnect
15	IODControl Parameter Error, Faulty ControlBlockPlug
16	IOXControl Parameter Error , Faulty ControlBlock after a connection establishment
17	IOXControl Parameter Error, Faulty ControlBlock after a plug alarm
28	Release Parameter Error, Faulty ReleaseBlock
40	RMPM (Device state machines, device resources)

Table 7-23: Meaning of ErrorCode1 for ErrorDecode = 81

ErrorCode1	Meaning
00	ArgsLength invalid
01	Unknown Blocks
02	IOCR Missing
03	Wrong AlarmCRBlock count
04	Out of AR Resources
05	AR UUID unknown
06	State conflict
07	Out of Provider, Consumer, or Alarm Resources
08	Out of Memory

ErrorCode1	Meaning
09-FF	Reserved

Table 7-24: Meaning of ErrorCode2 for ErrorCode1 = 40

For ErrorCode1 <> 40 ErrorCode2 refers to a field in the block specified by ErrorCode1.

ErrorCode2	Meaning
00	Block type
01	Block length
02	Block version (high byte)
03	Block version (low byte)

Table 7-25: Meaning of ErrorCode2 for ErrorCode1 &lt;&gt; 40

ErrorCode2	Meaning
04	AR Type
05	AR UUID
06	Session key
07	Initiator MAC address
08	Initiator Object UUID
09	AR Properties
0A	Activity timeout factor
0B	UDP RT port
0C	Station name length
0D	Station name

Table 7-26: Meaning of ErrorCode2 for ErrorCode1 = 01 (AR block request)

ErrorCode2	Meaning
04	IOCR Type
05	Reference
06	LT
07	IOCR properties
08	IO data length
09	Frame ID
0A	Send clock factor
0B	Reduction ratio
0C	Phase
0D	Sequence
0E	Frame send offset
0F	Watchdog factor
10	Data hold factor
11	Tag header
12	IOCR multicast MAC address
13	Number of APIs
14	API

ErrorCode2	Meaning
15	Number of IO data objects
16	Slot
17	Subslot
18	IO data object offset
19	Number of IOCS
20	Slot
21	Subslot
22	IOCS offset

Table 7-27: Meaning of ErrorCode2 for ErrorCode1 = 02 (IOCR block request)

ErrorCode2	Meaning
04	Number of APIs
05	API
06	Slot
07	Module ident number
08	Module properties
09	Number of submodules
0A	Subslot
0B	Submodule ident number
0C	Submodule properties
0D	Data description
0E	Data length
0F	IOPS length
10	IOCS length

Table 7-28: Meaning of ErrorCode2 for ErrorCode1 = 03 (Expected submodule block request)

ErrorCode2	Meaning
04	Type
05	LT
06	AlarmCR Properties
07	RTA timeout factor
08	RTA retries
09	Local alarm reference
0A	Maximum alarm data length
0B	Alarm CRT Tag high
0C	Alarm CRT Tag low

Table 7-29: Meaning of ErrorCode2 for ErrorCode1 = 04 (AlarmCR block request)

ErrorCode2	Meaning
04	Sequence number
05	AR UUID
06	API

ErrorCode2	Meaning
07	Slot number
08	Subslot number
09	Padding
0A	Index
0B	Data length
0C	Target AR UUID

Table 7-30: Meaning of ErrorCode2 for ErrorCode1 = 8 (Read/write record block request)

ErrorCode2	Meaning
05	Padding
06	Session key
07	Padding
08	Control block command
09	Control block properties

Table 7-31: Meaning of ErrorCode2 for ErrorCode1 = 16 (IOXControl block request)

### 12.3.11 Modular EtherCAT Concept

#### 12.3.11.1 Introduction

A modular approach to machine design has many benefits, including flexibility, scalability, and improved efficiency. Modular machines are split into sub-sections with each sub-section containing all the mechanical and electrical devices, motors, sensors, etc. as well as corresponding application code, to work together to provide a functional segment in the manufacturing workflow. The advantages to modular design are realized when the modules can be re-used. Examples include multiple module instances in the same machine and modules replicated across multiple machines.

Special consideration should be given to the physical layout, EtherCAT network topology, and the application code architecture to design a modular machine. Modular machines are well suited to distributed EtherCAT device topologies, such as the AKD-C/N drives, but can also be applied to other EtherCAT device(s).

#### Examples of Modular Machine Design Uses

- Machine with Multiple Variants:
  - a. Machine Variant = Base module + one, two, (or more) additional module(s)
  - b. Base Module - Each machine variant has the same base set of EtherCAT devices.
  - c. Additional Module(s) - Each module contains one (or more) EtherCAT devices.
- Module Maintenance:
  - a. One (or more) modules are removed during machine operation.
  - b. The machine continues to operate with the remaining base + module(s).
- Share Modules across Machines in a Factory Line:
  - a. Remove a module from machine A.
  - b. Continue to operate machine A with the remaining base + module(s).
  - c. Move the module to machine B.

#### 12.3.11.2 Modular Design Guidelines

The following sections provide some helpful guidelines for modular design:

*Single project file (.kas) with maximum EtherCAT topology*

Start with the full topology of EtherCAT devices planned for the maximum machine configuration. Perform an EtherCAT scan to discover the full set of EtherCAT devices and save the project file. Then, the PLC code in the KAS project can be adapted to handle the cases where modules containing EtherCAT devices are disconnected and removed for different machine configurations, maintenance, etc.

*Base module contains the first EtherCAT device with Distributed Clocks*

The first EtherCAT device with Distributed Clocks (DC) must be included in the base module. A Kollmorgen drive (AKD, AKD2G, etc.) would be a good choice for the first EtherCAT device. The first EtherCAT device with Distributed Clocks and all devices upstream from the DC device cannot be disconnected/removed.

*PLCopen/PipeNetwork Axis to drive map is the same for all machine variants*

Map the axes to the drives for the maximum EtherCAT topology. The axis to drive map will remain consistent if drives are disconnected/removed. Whether drives are disconnected and removed from the end or from the middle of an EtherCAT network, the axis to drive map relationship will remain the same.

*Axis is simulated for a disconnected drive*

The PLC application code will need to adapt to the disconnected/removed devices. Because disconnected drives will behave as simulated axes/drives, the application code can perform many normal axis operations without code adaptations. This capability provides flexibility for how the application code is designed to handle disconnected/removed drives.

*Safety devices should not be disconnected/removed*

FSoE Masters may not support modular EtherCAT designs. If a safety device is disconnected and removed, the FSoE master will not allow normal machine operation.

Also see [ECATDeviceAction](#) for information on how to add the ability to connect/disconnect an EtherCAT device to your program..

### 12.3.11.3 Backing up Modular EtherCAT devices

The following steps should be taken to backup EtherCAT devices when the modular EtherCAT feature (see [Modular EtherCAT Concept](#) for more information) is used to remove devices to create a machine variant.

1. Run the KAS application which sets up the correct EtherCAT topology for the machine variant.
2. Stop the KAS application.

#### **ⓘ IMPORTANT**

Do not power cycle or reboot the controller between steps 2 and 3.

3. Backup the EtherCAT devices (see [Backup Steps](#)).

### 12.3.11.4 Limitations

- The first EtherCAT device with Distributed Clocks cannot be disconnected/removed and must remain the same. Only devices downstream from the first DC device can be removed.
- The device topology order cannot be rearranged.
- The Modular EtherCAT features are not supported with imported ENI files.

*What happens when the device providing feedback to the digitizing axis is removed?*

The feedback position will become zero if the device providing the Feedback Position for the Digitizing axis is removed using the [ECATDeviceAction](#) function block. The feedback position data coming from the device will automatically be connected to the digitizing axis when the device is reconnected to the EtherCAT network.

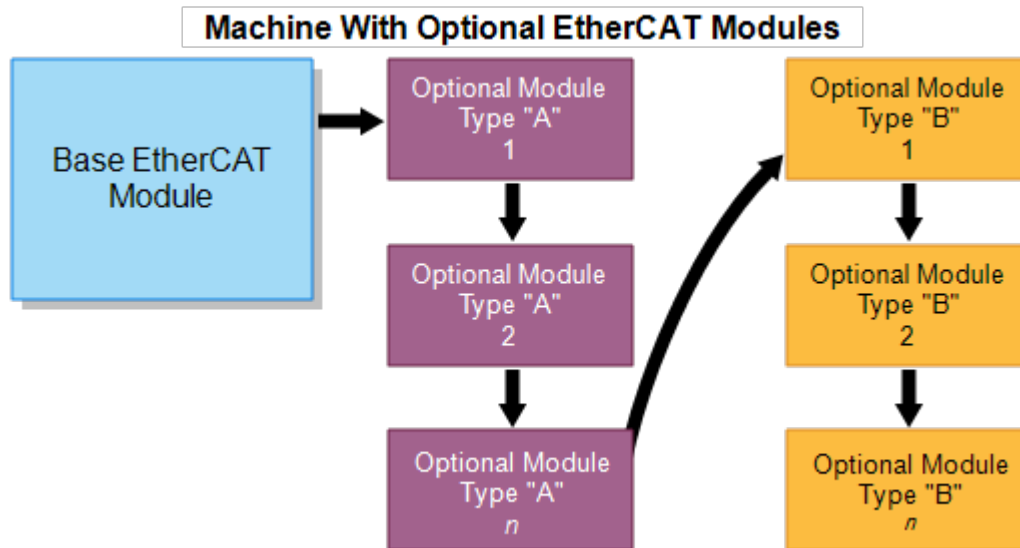
*The Restore function only supports exact matches in the topology of backedup EtherCAT devices.*

Currently the Restore function only supports conditions where the network topology exactly matches the topology of the EtherCAT devices backup.

There are a number of permutations and combinations which are possible when the Modular EtherCAT feature is used to create a machine with multiple variants. If a machine has a base EtherCAT module and a varying number of optional EtherCAT modules then it is recommended that multiple backups are created: a base EtherCAT module backup, and an optional EtherCAT module backup.

### Example

The system shown below has a base EtherCAT module and a varying number of optional type "A" modules and type "B" modules.



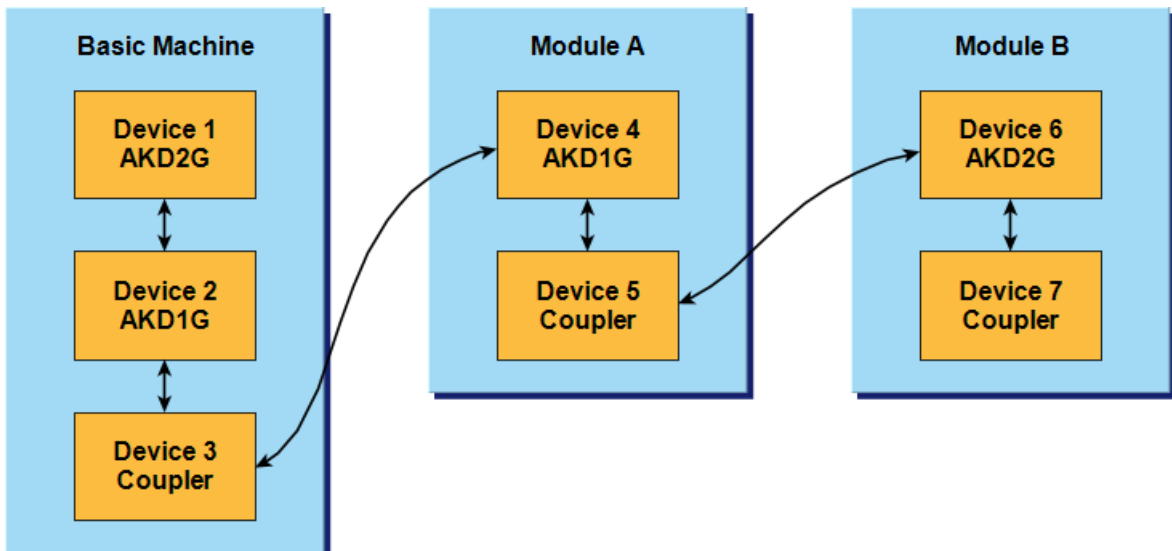
We recommend that EtherCAT Devices backups be made with the following configurations in order to meet different scenarios.

- Backup #1: Base EtherCAT Module + one of Optional Module Type A
- Backup #2: Base EtherCAT Module + one of Optional Module Type B

With the two EtherCAT Devices backups, you may be able to easily restore various combinations by removing the other Optional Modules, keeping only the module you need to restore.

#### 12.3.11.5 How to Create an Application with Modular EtherCAT

This section discusses how to create an application with modular EtherCAT for various scenarios. We will use the following configuration to describe the scenarios and code snippets. The configuration consists of a full machine with two different modules. The full EtherCAT network consists of seven devices.



### 12.3.11.5.1 Create a Machine Variant with the Basic Machine

1. As described in the [Modular Design Guidelines](#), start with a full topology of EtherCAT devices. In this example we have Device1 to Device7.
2. After initializing the motion engine use the [ECATDeviceAction](#) function block to disconnect Device 4, Device 5, Device 6, and Device 7 **before** calling the [MLMotionStart](#) function block.
3. After the devices are disconnected, call [MLMotionStart](#) and finish the code.

Following is a ST code snippet example.

```

//////MOTION INITIALIZATION//////////////////////////////////////
FirstCycle := TRUE;

On FirstCycle DO //Initialize the motion engine
    MLMotionInit( 1000);
    StatusMessage := 'Initializing Motion Engine';
END_DO;

MotionEngineStatus := MLMotionStatus(); //Check the current status of the
motion engine

ON MotionEngineStatus = MLSTATUS_INITIALISED DO //Once motion engine is
initialized, create CAM profiles and defined Axis, then start the motion
engine
    If not FirstInitComplete then
        Profiles( MLPR_CREATE_PROFILES );
        PLCopen( 0 );
        // Disconnect Device 4 through Device 7
        Inst_ECATDeviceAction(True, EtherCAT.Device_4 , DEVICE_ACTION_
DISCONNECT);
        if Inst_ECATDeviceAction.Error then
            //Handle the Error here
        End_if;
        // Repeat the steps done for Device 4 to disconnect Device
5, Device 6, Device 7
        // After disconnecting all the devices that are not part of
this machine variant, start the motion engine.
        MLMotionStart();
    
```

```

        FirstInitComplete := True;
    End_If;
END_DO;

```

### 12.3.11.5.2 Steps to Add a Module

Use the following procedure when a module needs to be added. In this example we will add Module A of our example project.

1. Stop the motion engine using [MLMotionStop](#).
2. Verify that motion engine is stopped.
3. Use the action "DEVICE\_ACTION\_CONNECT" in [ECATDeviceAction](#) to connect the EtherCAT devices (Device 4 and Device 5) corresponding to the module.
4. Start the motion engine using [MLMotionStart](#).

Following is a ST code snippet example.

```

// To Add a Module first stop the Motion Engine.
MLMotionStop();
MotionEngineStatus := MLMotionStatus();//Check the current status of
the motion engine

ON MotionEngineStatus = MLSTATUS_STOPPED DO

    // Connect Device 4 and Device 5 to add Module A
    Inst_ECATDeviceAction(True, EtherCAT.Device_4, DEVICE_ACTION_
CONNECT);
    If Inst_ECATDeviceAction.Error then
        //Handle the Error here
    End_if;
    Inst_ECATDeviceAction(True, EtherCAT.Device_5, DEVICE_ACTION_
CONNECT);
    If Inst_ECATDeviceAction.Error then
        //Handle the Error here
    End_if;

    // Start the Motion Engine
    MLMotionStart();

END_DO;

```

### 12.3.11.5.3 Steps to Remove a Module

Use the following procedure when a module needs to be removed. In this example we will add Module A of our example project.

1. Stop the motion engine using [MLMotionStop](#).
2. b. Verify that motion engine is stopped.
3. Use the action "DEVICE\_ACTION\_DISCONNECT" in [ECATDeviceAction](#) to disconnect the EtherCAT devices (Device 4 and Device 5) corresponding to the module.
4. Start the motion engine using [MLMotionStart](#).
5. See the code snippet below for \_\_\_?

Following is a ST code snippet example.

```

// To remove a Module first stop the Motion Engine.
MLMotionStop();

```



```

    MotionEngineStatus := MLMotionStatus();//Check the current status of
the motion engine

ON MotionEngineStatus = MLSTATUS_STOPPED DO

    // Disconnect Device 4 and Device 5 to add Module A
    Inst_ECATCHDeviceAction(True, EtherCAT.Device_4, DEVICE_ACTION_
DISCONNECT);
    If Inst_ECATCHDeviceAction.Error then
        //Handle the Error here
    End_if;

    Inst_ECATCHDeviceAction(True, EtherCAT.Device_5, DEVICE_ACTION_
DISCONNECT);
    If Inst_ECATCHDeviceAction.Error then
        //Handle the Error here
    End_if;

    // Start the Motion Engine
    MLMotionStart();

END_DO;

```

### 12.3.12 Performance Guidelines

Additional controller CPU usage is required when fieldbuses are used to transmit and receive data between network devices and PLC variables. The impact to PLC execution time and the CPU load can be estimated as follows.

$$\text{PLC Time} = (\text{Mapped PLC Variable Count} * \text{Exchange Time per Variable}) + \text{PLC Overhead}$$

AKD PDMM or PCMM model	Exchange Time per Variable
800 MHz	0.4 microseconds
1.2 GHz	0.2 microseconds

Table 7-32: Average Exchange Time per Variable by AKD PDMM / PCMM model used to calculate performance.

#### NOTE

The exact *Exchange Time per Variable* is dependent on the data type. BOOL variables take less time, while REAL variables require more time.

#### 12.3.12.1 EtherNet/IP

$$\text{CPU Usage \%} = (\text{PLC Time} / \text{ECAT Cycle Time}) + (\text{Assembly Count} * \text{Load per Assembly})$$

AKD PDMM or PCMM model	PLC Overhead	Load per Assembly
800 MHz	20 microseconds	0.0075 (0.75%)
1.2 GHz	10 microseconds	0.00375 (0.375%)

Table 7-33: Values by AKD PDMM and PCMM models used to calculate performance with EtherNet/IP.

The exact percent *Load per Assembly* is dependent on the data size per assembly and the *Requested Packet Interval (RPI)*. The *Load per Assembly* values listed above represent the maximum percentage, with 496 bytes per Assembly and *RPI* = 10 milliseconds.

#### 12.3.12.1.1 EtherNet/IP Performance Example

This example uses an 800 MHz AKD PDMM with 4x Assemblies, 372 mapped PLC variables, and a 1000 microsecond EtherCAT Cycle Time.

```
PLC Time = (372 * 0.4) + 20 = 169 microseconds  
CPU Usage % = (169 / 1000) + (4 * .0075) = 0.20 (20%)
```

Thus, exchanging data with 372 PLC variables, over EtherNet/IP, will increase the PLC execution time by 169 microseconds and reduce the CPU Idle Time by 20%.

## 12.4 Project Structure Guidelines

### 12.4.1 Introduction

By implementing a predefined structure for new projects, KAS tries to achieve the following goals:

- Efficiency in developing new applications
- High flexibility to keep only functionalities that are needed and to create the new ones that are required
- Safe applications due to an already tested and approved structure that optimize the resources usage (memory and processor load)
- Reliable framework that supports error, state, data and communication management
- Easier to exchange applications
- Less time needed to understand, maintain and teach an application (from a troubleshooting and support standpoint)
- Less documentation work is required since the main behavior of the Application is already documented (only the specific functionalities need some additional work)

### 12.4.2 External Files

Some items that belongs to your application (displayed in the [Project Explorer](#) ) are not embedded into the project file. For the domains listed below, KAS IDE also uses some resources that are stored in external files.



Domain	Description	File
<b>HMI</b> 	Using Kollmorgen HMI, simply tag the variables in the PLC environment to create an export file that describes the data to be exchanged between the PLC and the HMI. Import this Modbus mapping file into the HMI programming environment and use the variables as if they are local variables	<a href="#">KVB Project File</a>
<b>PLC</b>	The PLC programming environment gives you the possibility to create reusable components (UDFB), and template applications which can be customized to suit any given application	<a href="#">Create Custom Libraries</a> <a href="#">Read Common Constants</a>
<b>Motion</b>	The CAM editor lets you create complex CAM profiles online using a “graphical” interface. It is also possible to import existing CAM profile points into the CAM editor to allow you to reuse your existing machine building experience seamlessly	<a href="#">Import Cam Profile</a> <a href="#">How Do I Export the Oscilloscope Data?</a>
<b>Fieldbus</b>	Kollmorgen Automation Suite tightly integrates the EtherCAT motion bus (standard Ethernet-based cabling) to define all the network description	<a href="#">Import or Export EtherCAT ENI File</a>
<b>Drive</b> 	The AKD drive is fully embedded in the Kollmorgen Automation Suite but not all interwoven at one time. This makes future customization easier to get all the firmware features	<a href="#">Download AKD Firmware</a>

Table 7-34: - File location

**TIP**

The hyperlinks bring you to the relevant topic that contains more details.

## 12.4.3 Application Software Structure - Definitions

### 12.4.3.1 Modules to build up the Structure

#### 12.4.3.1.1 Structure Overview

You normally write the PLC program. Whereas Kollmorgen application team members create in most cases the motion control part.

The global software structure is built up with different modules placed on two different levels as showed on the figure below:

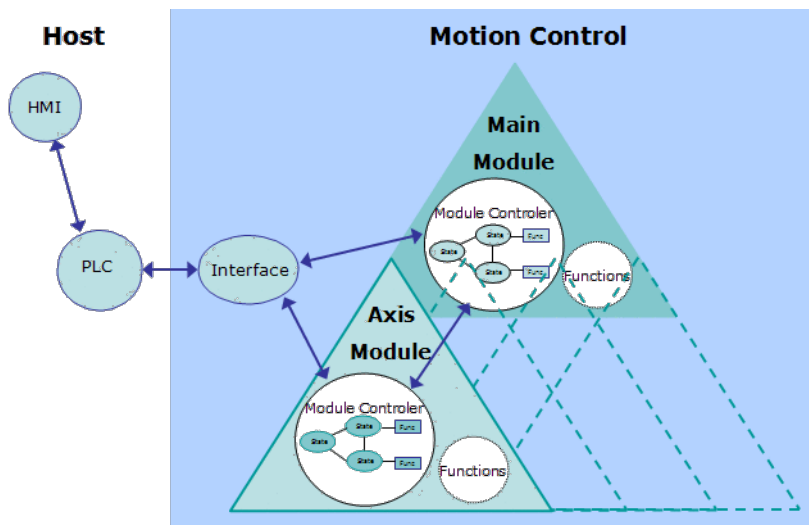


Figure 7-28: Software Structure Overview

#### Module Definition:

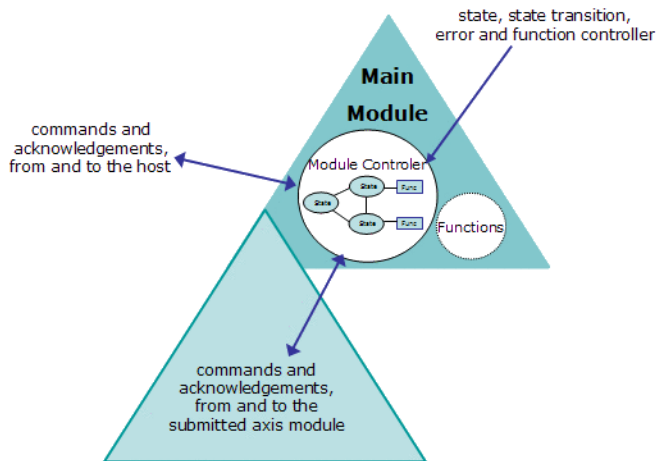
- A module is one unit of the software structure (triangle)
- It is controlled by one module from the next higher level and can in turn control several modules in the next lower level
- It never communicates with modules of the same level
- It can generally run independently from any other modules at the same or higher level

To have the structure running as a real application, it needs to be controlled by a PLC. As the PLC is not part of the application structure, only the main and axis modules are described here.

#### 12.4.3.1.2 Main Module description

The main module controls the functional work that globally affect the application (e.g. multi axes functions). It receives commands from the PLC and sends back acknowledgements. The main module does not directly act on the physical axes, but controls the axis modules that are linked to them.

Communication between main and axis modules is done via internally defined data channels.



**Figure 7-29: Main Module Description**

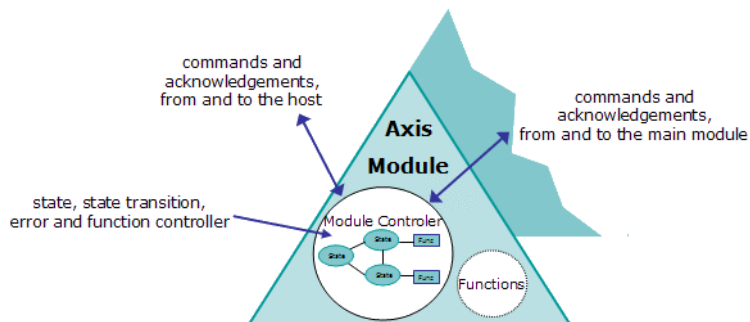
As shown on the figure, the main module consists of two parts:

- the **module controller** part is responsible for state, state transition, error and functions handling. It receives state transition and function call commands from the host, performs all needed actions and sends back some acknowledgements. In case of an error it reacts by itself and sends a message to the PLC. If requested, it activates state transitions and functions in the axis modules, by sending commands to them and waiting for acknowledgement. The main module controller also manages the error status of the submitted modules and performs the needed actions.
- the **functional part** consists of all functionalities needed for the current application. These functions can be state dependant (e.g. multi axes functions) or state independent (e.g. increase a speed value).

#### 12.4.3.1.3 Axis Module description

The axis module controls the functional work that affect the application one or more physical axes (e.g. single-axis functions). It receives commands from the PLC and sends back acknowledgements.

The axis module also communicates with its main module via the internally defined data channel.



**Figure 7-30: Axis Module Description**

As shown on the figure, the axis module consists of the same two parts as the main module:

- the **module controller** part is responsible for state, state transition, error and functions handling. If the axis module is not connected to its main module, it receives state transition and function call commands from the host, performs all needed actions and sends back some acknowledgements. If connected, state transition commands are received from its

main module and not from the host. In case of an error it only reacts by itself, if it is not connected to the main module.

- the **functional part** consists of all functionalities needed for the current physical axis. These functions can be state dependant (e.g. single axes functions) or state independent (e.g. increase a speed value).

### 12.4.3.2 State and Function Definitions

A state machine and some functions of general interest are implemented in the software structure. They are provided as examples of how to use the structure but can be adjusted to fulfil specific application usage (see also [How to add a new state](#) and [How to add a new function](#)).

#### 12.4.3.2.1 State transition Diagram

The following state machine has been defined.

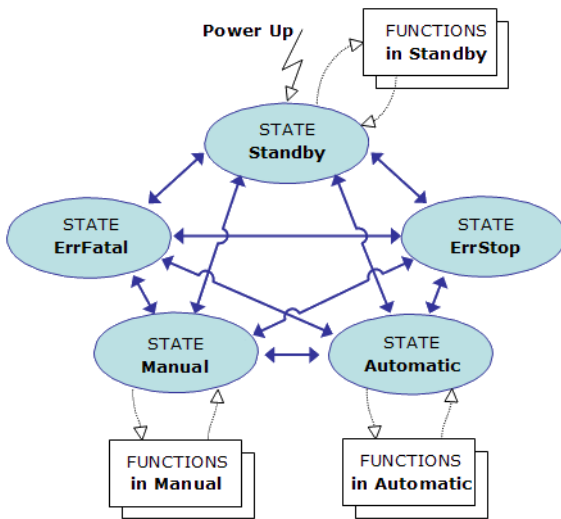

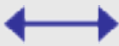
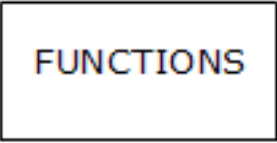


Figure 7-31: State Machine

#### Legend

	<p>Represents the current state of one or more modules (e.g. Standby state means that the module is waiting for commands)</p> <p>In addition to the static states, there are two other possible dynamic states:</p> <ul style="list-style-type: none"> <li>• <b>Busy</b>: when performing a state transition (exit old and enter new state)</li> <li>• <b>Running</b>: when working through a function</li> </ul>
<p>State Transition</p> 	<p>It is the transition from one state to another when performing the corresponding action</p>
	<p>Any needed machine or single-axis functionality (e.g. single-axis jog), which can be called out of one or several states</p>

All modules have the same states and state transitions. The state of a module is only influenced by other modules, if they are connected with each other.

### 12.4.3.2.2 State, state transitions and functions descriptions

The structure is built in such a way that state transitions are possible from the active state to any other existing states (except state ErrStop). After leaving state ErrStop (corresponding to a non-fatal error, which causes a stop and power off) the structure automatically recovers the state which was active before entering ErrStop. That means that all characteristics of the previous state are kept.

#### NOTE

Because functionalities are always specific to the application, none are included in the structure itself.

## 12.4.4 Application Software Structure - Implementation

This chapter describes how the software structure described before is implemented. Insofar as all modules are implemented and behave in the same way, only the main module is described in detail here.

### 12.4.4.1 SFC children building up the software

The following files contain all the data to build up the application. They are all required to ensure a successful compilation.

#### Parent SFC

Main	System start up and SFC children call
------	---------------------------------------

#### Main module SFC children

M1_StateController	state and function controller of the main module
M1_ErrorHandling	error handling of the main module
M1_IndependentFunctions	state independent functions of the main module
M1_Interface	interface to PLC

#### Axis module SFC children

Ai_StateController	state and function controller of the axis module
Ai_ErrorHandling	error handling of the axis module
Ai_IndependentFunctions	state independent functions of the axis module

With  $i = 1 \dots n$

### 12.4.4.2 Variables for the Interface

#### 12.4.4.2.1 List of variables

- M1\_CmdState
- bM1\_CallStandbyFunction1
- bM1\_CallStandbyFunction2
- bM1\_CallManualFunction1
- bM1\_CallManualFunction2
- bM1\_CallAutomaticFunction1
- bM1\_CallAutomaticFunction2
- bAi\_CallStandbyFunction1

- bAi\_CallStandbyFunction2
- bAi\_CallManualFunction1
- bAi\_CallManualFunction2
- bAi\_CallAutomaticFunction1
- bAi\_CallAutomaticFunction2
- bErrorReset

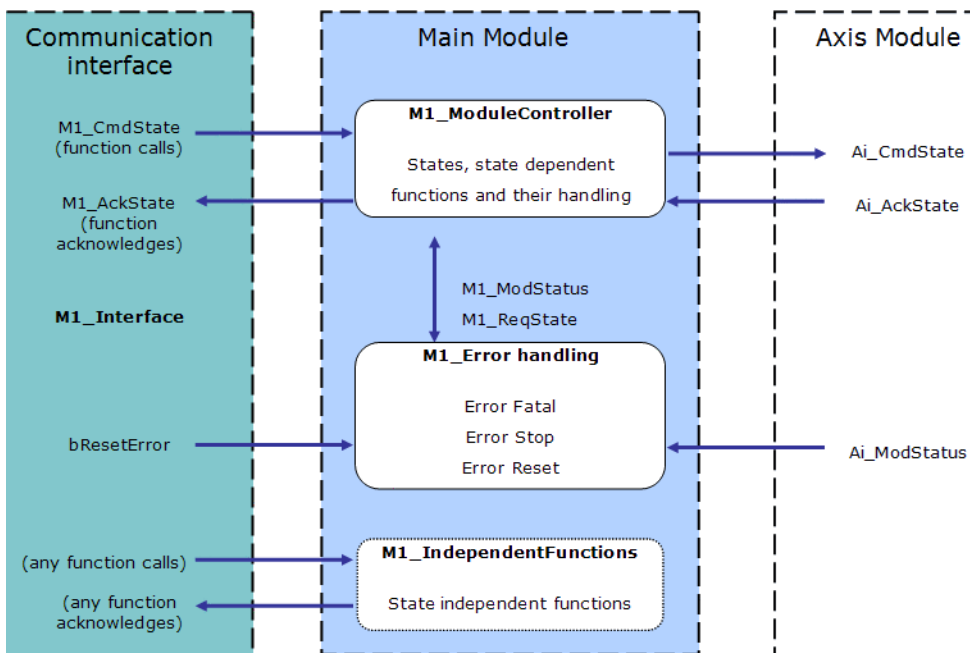
**12.4.4.2.2 List of output variables**

- M1\_AckState
- M1\_StatusWord
- bM1\_Running
- Ai\_StatusWord
- bAi\_Running

**12.4.4.3 Main module implementation description**

In the main module, all necessary state, state transition, error and function handling facilities are implemented for this level.

**Context diagram for the main module**



The following objects (variables, tasks...) are defined in the structure of the main module.

**12.4.4.3.1 M1\_CmdState**

**12.4.4.3.1.1 Description**

This internal word variable contains the actual state command value. It is automatically set to state 'Standby' during power up.



```

//*****
//**          State Defines          **
//*****
#define DEF_StateUndefined          0
#define DEF_StateStandby           1
#define DEF_StateManual             2
#define DEF_StateAutomatic          3
#define DEF_StateBusy               4
#define DEF_StateErrorStop          5
#define DEF_StateErrorFatal         6

```

#### 12.4.4.3.1.2 Usage

These state commands are usually set in the communication interface (see software listing of ACT\_M1\_Translate and ACT\_M1\_SimaticSimu) and must not be set directly from the host system. If additional or different state commands are needed, then the definitions described above can be modified accordingly.

#### 12.4.4.3.2 M1\_AckState

##### 12.4.4.3.2.1 Description

This internal word variable contains the actual state acknowledge value, as a result from the **M1\_CmdState** state command performed with success. Possible values are the same as for the state commands (see above).

##### 12.4.4.3.2.2 Usage

Out of this value the corresponding acknowledgements for the PLC can be created in the communication interface.

#### 12.4.4.3.3 M1\_ReqState

##### 12.4.4.3.3.1 Description

This internal word variable contains the internally active state. It is used for internal purpose only, to keep the actual state value, e.g. while performing a function. Possible values are the same as for the state commands (see above).

##### 12.4.4.3.3.2 Usage

Used by system, do not use it for application purpose.

##### 12.4.4.3.3.3 Description

This internal word variable contains the actual module status and error information. It is automatically set to the default value during power up. The meaning of the predefined **Module Error Bits** are as follows:

Bit	Bit	Bit	Bit	Bit	Bit	Bit	Bit	Bit	Bit	Bit	Bit	Bit	Bit	Bit	Bit
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Bits	Description
0	error stop reported by drive (drive error)
1	error fatal reported by Drive (lag error)
2	not used (motor temperature too high)
3	not used (external stop)
4	not used (negative limit switch reached)
5	not used (positive limit switch reach)
6	not used (not used)
7	not used (not used)
8	not used (state HW enable)
9	not used (state AS enable)
10	not used (axis is powered on)
11	not used (axis is homed)
12	not used (axis is running)
13	not used (pipe is connected)
14	error stop (error stop)
15	error fatal (error fatal)

#### 12.4.4.3.3.4 Usage

While the error bits are usually set only by the error handling (`M1_ErrorHandling`), the mode bits can be modified where ever needed in the application program (except in the interface). Several bits can be set at the same time. Several masks have been defined to test or modify the whole word. For each module, there is one mask to define the bits causing a fatal error (e.g. `MSK_M1_StatusErrorFatal`) and one for the stop error (e.g. `MSK_M1_StatusErrorStop`). To add errors and modes, the bits not already assigned by default can be used (i.e. bits 16 to 31).

#### 12.4.4.3.4 bErrorReset

##### 12.4.4.3.4.1 Description

This internal flag variable is used as the error reset command for the main and axis modules. It is reset during power up.

##### 12.4.4.3.4.2 Usage

Set and reset this flag to activate a reset of the module errors (`M1_StatusWord`, `Ai_StatusWord`).

#### 12.4.4.3.5 M1\_ErrorHandling

##### 12.4.4.3.5.1 Description

This program is responsible for the main module error handling. If an error occurs (in the main module or a submitted axis module), the corresponding bit in the module status (`M1_StatusWord`) is set. This causes the error reaction bits (`MSK_Mi_StatusErrorStop`, `MSK_Mi_StatusErrorFatal`) to be set in the module status word.

##### 12.4.4.3.5.2 Usage

Any additional error which needs to be treated has to be included in this program. Do not forget to modify the corresponding masks (`MSK_M1_StatusErrorFatal`, `MSK_M1_StatusErrorStop`) to cause the correct reaction on errors.

#### 12.4.4.3.6 M1\_ModuleController

##### 12.4.4.3.6.1 Description

This program is the heart of the whole controller and contains:

- a state manager sequence
- all state sequences
- and state dependent function sequences of the main module

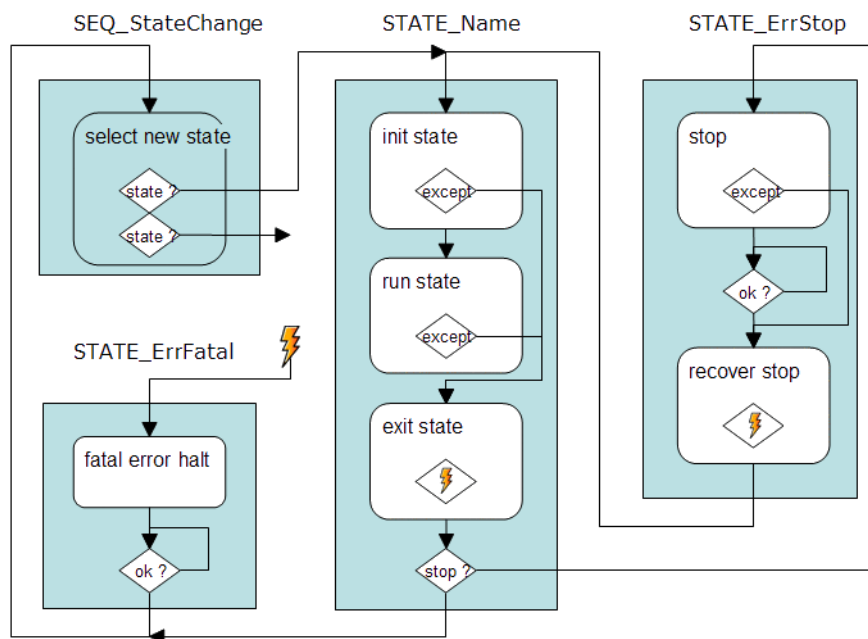
#### 12.4.4.3.6.2 Usage

Some rules have to be followed, when using and changing states and functions (see also [How to add a new state](#) and [How to add a new function](#)).

#### 12.4.4.4 States and Errors

##### 12.4.4.4.1 How States and Errors are treated

The figure below shows how states and errors are treated.



**StateChange** (state manager)

Activates the new state required by M1\_ReqState

**StateName** (state macro)

init state	- Initializes exceptions on new state M1_CmdState <> <sup>1</sup> M1_ReqState and on errors set in M1_StatusWord
	- Goes to exit state when an exception occurs
	- Performs all actions to properly enter this state (init variables, pipes, ...)
	- Sends commands to the submitted axis modules by setting Ai_CmdState to StateName and waits for their acknowledgement in Ai_AckState
	- Acknowledges end of initialization by setting M1_AckState to M1_ReqState
run state	- Waits for any function calls, activate function if called

<sup>1</sup><> means Not Equal

exit state	<ul style="list-style-type: none"> <li>- Performs all actions to properly leave this state</li> <li>- Acknowledges running by setting M1_AckState to 'busy'</li> <li>- If error stop occurs, activates STATE_ErrStop, otherwise sets new requested state M1_ReqState to M1_CmdState and activates StateChange</li> </ul>
------------	--

#### 12.4.4.4.2 How to add a new state

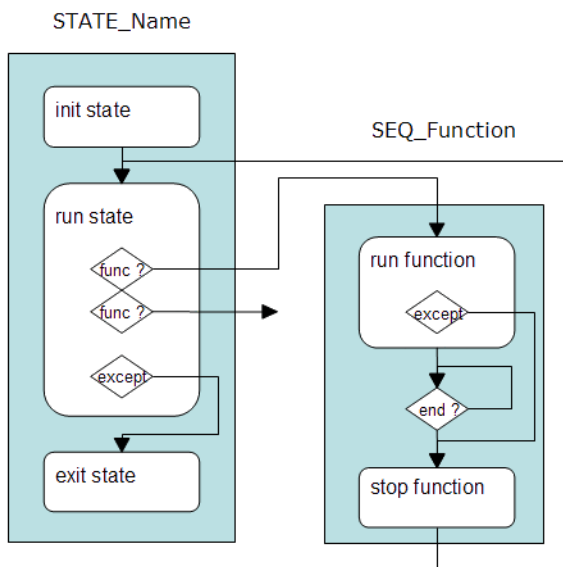
To add a new state, do as follows:

1. copy a similar existing state sequence
2. replace the old state name by the new one (e.g. 'Standby' by 'MyState')
3. modify both init and exit sections of the new state to perform the relevant actions
4. insert the needed function calls into the states run part
5. add the state call command line into the state change sequence
6. add the state definition values to the general declaration

#### 12.4.4.5 Functions linked to states

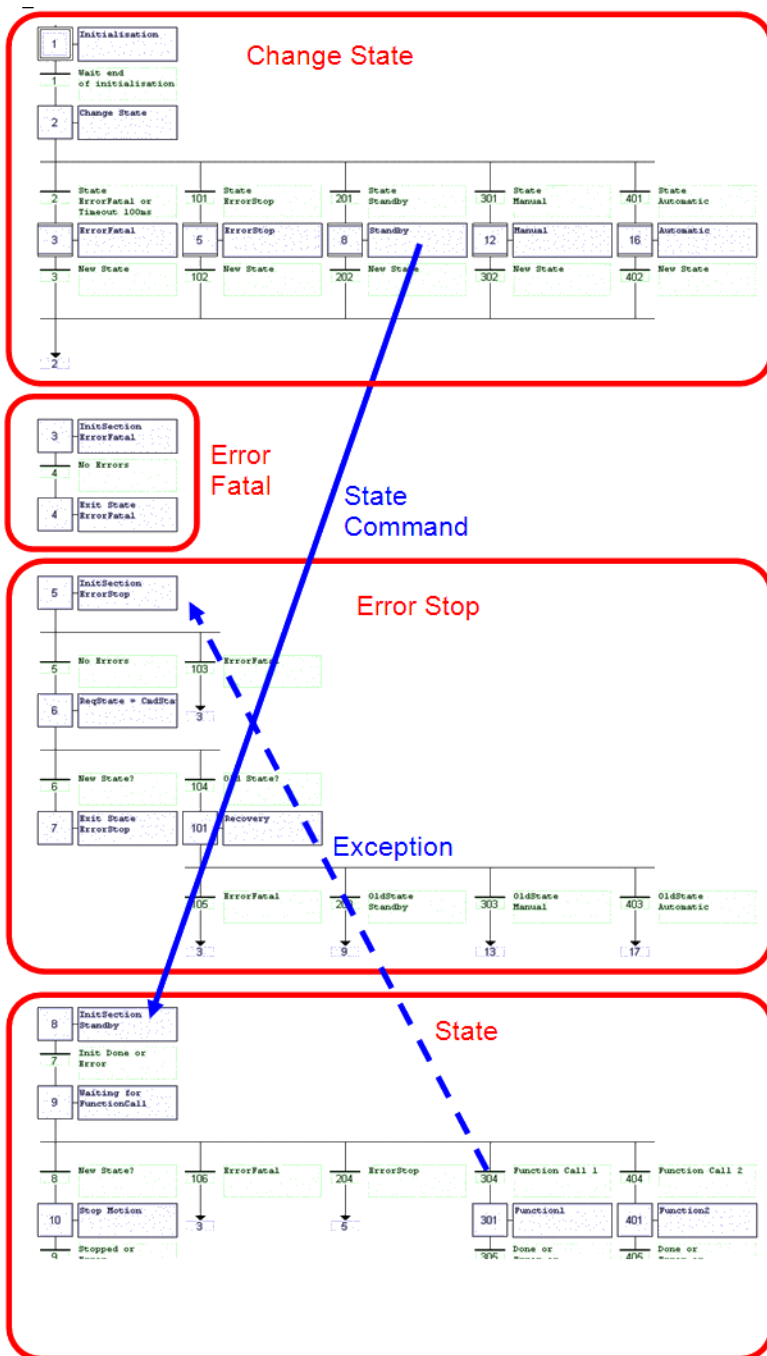
##### 12.4.4.5.1 How Functions are treated

The figure below shows how functions (that are state dependent) are treated.



#### Function (function step)

run function	<ul style="list-style-type: none"> <li>- Initializes exceptions on new state M1_CmdState &lt;&gt; M1_ReqState and on errors set in M1_StatusWord</li> <li>- Goes to exit function when an exception occurs</li> <li>- Acknowledges running</li> <li>- Performs all actions needed for the function until the function call command is reset</li> </ul>
stop function	<ul style="list-style-type: none"> <li>- Performs all actions to properly leave this function</li> <li>- Acknowledges end of exit, by setting M1_AckState to M1_ReqState</li> <li>- Returns to last state</li> </ul>



### 12.4.4.5.2 How to add a new function

To add a new function, do as follows:

1. copy a similar existing function sequence
2. replace the old function name by the new one (e.g. 'Running' by 'MyFunction')
3. modify the exit section of the new function to perform the relevant actions
4. insert the needed function code into the run part
5. add the function call command line to the state sequence where the function is used

## 12.5 Project Templates

KAS provides start up templates to help you get started (see how to use the project setup wizard [here](#)). There are separate templates for the Pipe Network motion engine and for the PLCopen motion engine:

Template Type	Template name	Description
PipeNetwork	1 Axis FFLD	Simple initialization, relative position move
	1 Axis ST	Simple initialization, relative position move
	2 Axes FFLD	Position and velocity moves, gearing, camming, and drive status
	2 Axes ST	Position and velocity moves, gearing, camming, and drive status
	2 Axes SFC	Position and velocity moves, gearing, camming, and drive status with optimized performance.
PLCopen	1 Axis FFLD	Simple initialization, relative position move
	1 Axis ST	Simple initialization, relative position move
	2 Axes FFLD	Position and velocity moves, gearing, camming, and drive status
	2 Axes ST	Position and velocity moves, gearing, camming, and drive status
	2 Axes SFC	Position and velocity moves, gearing, camming, and drive status with optimized performance.
Coordinated Motion	2 Axes - Linear / Circular	Raster Scan Motion Path, 2 PLCopen axes
	3 Axes - Linear / Circular	Raster Scan Motion Path, 2 PLCopen axes and 1 PipeNetwork axis
	3 Axes - Linear (3D)	Diamond/Square Motion Path, 3 PLCopen axes
KAS Runtime	Library	Allows you to create a custom library ( <a href="#">Create and Use Custom Libraries</a> )
Download Template from KDN	KDN	Upon selecting Finish, the dialog closes and a web browser opens a page on KDN which contains custom-built template files, which may have enhanced, or specific functionality. Download and save a template file, then open it from within the KAS IDE.

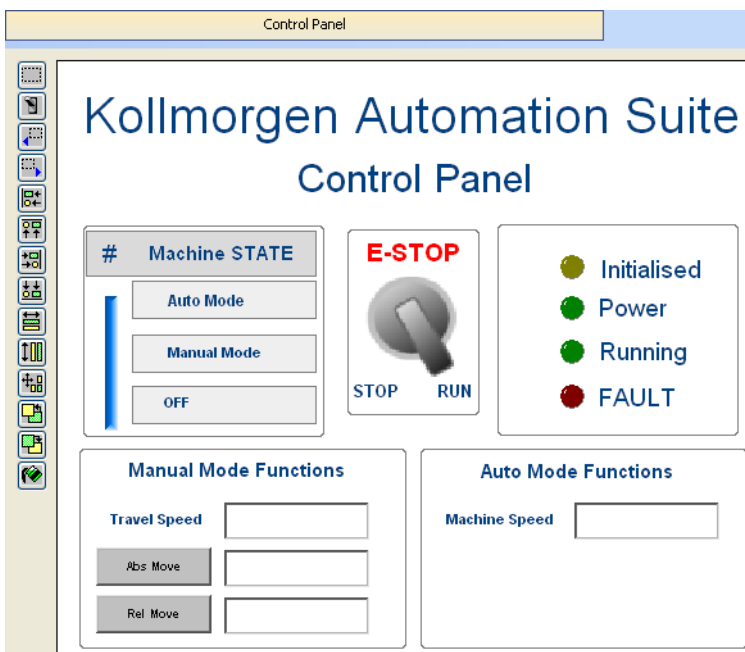
These templates come complete with software to:

- Create two or three axes of servo motion
- Enable the drives
- Perform simple motion

The templates contain variables which support these operations.

Name	Type	Dim.	Attrib.	Init value
TravelSp...	LREAL			LREAL...
MasterA...	LREAL			LREAL...
MasterDe...	LREAL			LREAL...
Machine...	LREAL			LREAL...
Axis1Sta...	DINT			
Axis2Sta...	DINT			
Machine...	DINT			0
bMaster...	BOOL			FALSE
bMaster...	BOOL			FALSE
bEStop	BOOL			FALSE
bLedStat...	BOOL	[0..3]		
Profiles	ProfilesCo...			
EtherCAT	EtherCAT...		Read Only	
PipeNet...	PNCCode			
Retain variables				
MachineLogic				
lastTrave...	LREAL			LREAL...
lastMach...	LREAL			LREAL...

Additionally, the templates contain a [Control Panel](#) for ease of running motion.



#### NOTE

You may need to modify the templates to run on your specific hardware.

### 12.5.1 Pipe Network 1-Axis Template with FFLD only

The 1-axes Pipe Network template has a FFLD program (called **Main**) that initializes, starts and runs the motion. Also included are controls to enable/disable the drive, clear faults, and display fault and axis position information in the Control Panel.

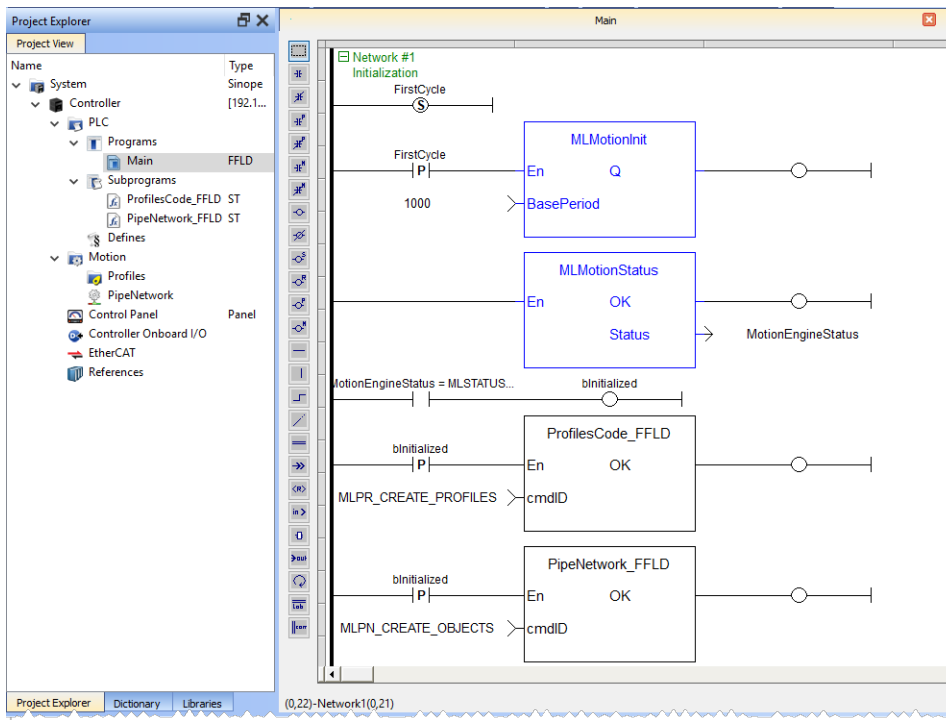
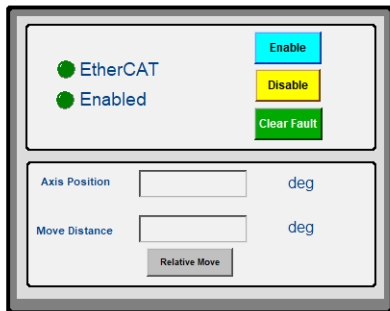


Figure 7-32: PN Template with FFLD - Main

## Kollmorgen Automation Suite

### Control Panel



### 12.5.2 Pipe Network 1-Axis Template with ST only

The 1-axis Pipe Network template has a ST program (called **Main**) that initializes, starts and runs the motion. Also included are controls to enable/disable the drive, clear faults, and display fault and axis position information in the Control Panel.

```

Project Explorer
Project View
Name      Type
System    Sinope [192.168...
  Controller
    PLC
      Programs
        Main      ST
      Subprograms
        ProfilesCode_FFLD ST
        PipeNetwork_FFLD ST
      Defines
    Motion
      Profiles
      PipeNetwork
      Controller Onboard I/O
      EtherCAT
      References
      Control Panel Panel

FirstCycle := TRUE;

ON FirstCycle DC //Initialize the motion engine
  Printf('Initialisation', 0, 0, 0, 0);
  MLMotionInit( 1000);
END_DC;

MotionEngineStatus := MLMotionStatus();//Check the current status <

ON MotionEngineStatus = MLSTATUS_INITIALISED DC//Once motion engine
  Printf('Creation of motion objects', 0, 0, 0, 0);
  Profiles(MLPR_CREATE_PROFILES);// Create all CAM profiles
  PipeNetwork(MLPN_CREATE_OBJECTS);// Creates all motion objects
  MLMotionStart();
END_DC;

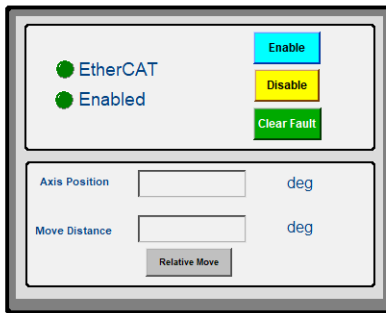
ON MotionEngineStatus = MLSTATUS_RUNNING DO
  bMotionEngineStarted := TRUE;
  // Activates the pipes
  PipeNetwork(MLPN_ACTIVATE);

  // Connect the axes to the pipes
  PipeNetwork(MLPN_CONNECT);
END_DC;
    
```

Figure 7-33: PN Template with ST - Main



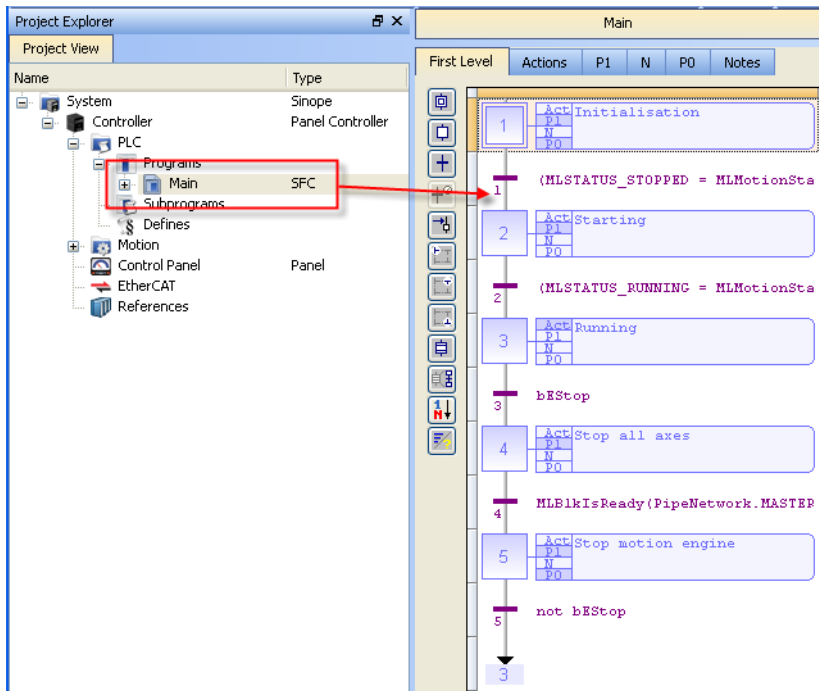
## Kollmorgen Automation Suite Control Panel



### 12.5.3 Pipe Network 2-Axes Template with SFC, ST, FFLD, and FBD

#### 12.5.3.1 PLC Programs

The 2-axes Pipe Network template has an SFC program (called **Main**) that initializes and starts the motion.



**Figure 7-34:** PN Template - Main

The Pipe Network Template contains an SFC child program called Machine Logic for running motion.

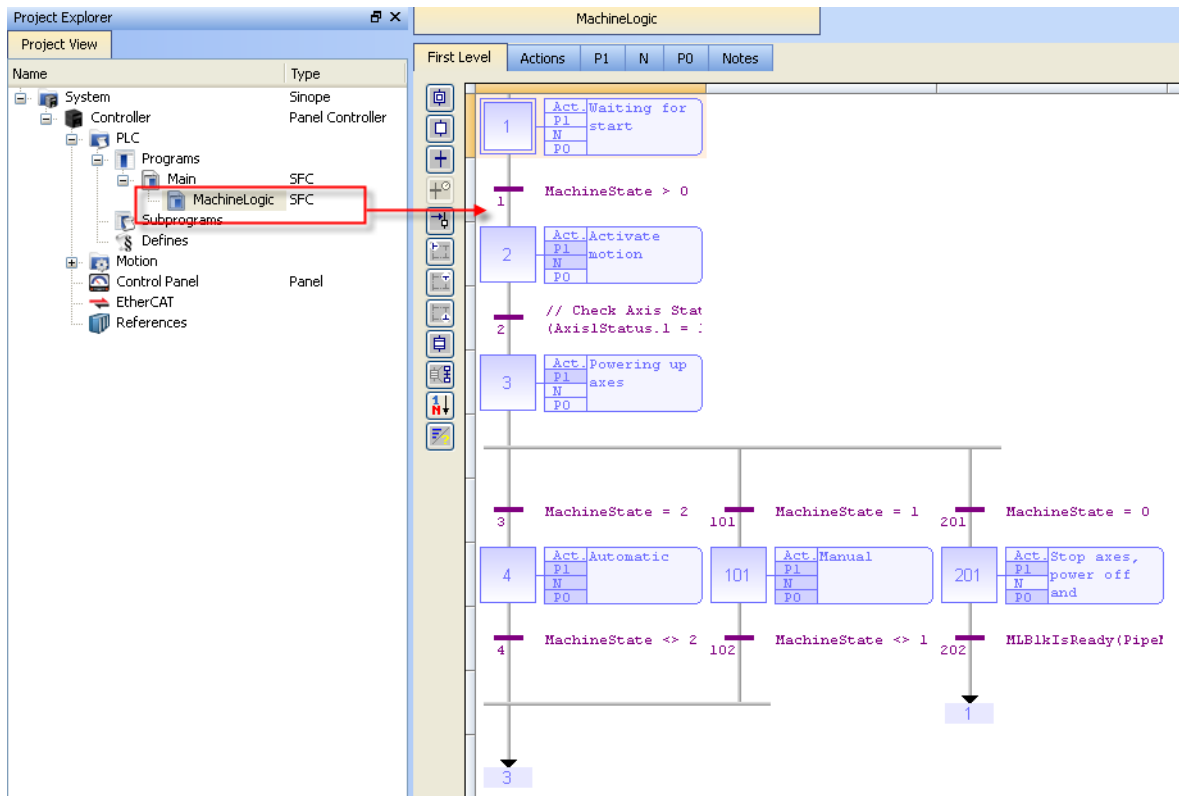
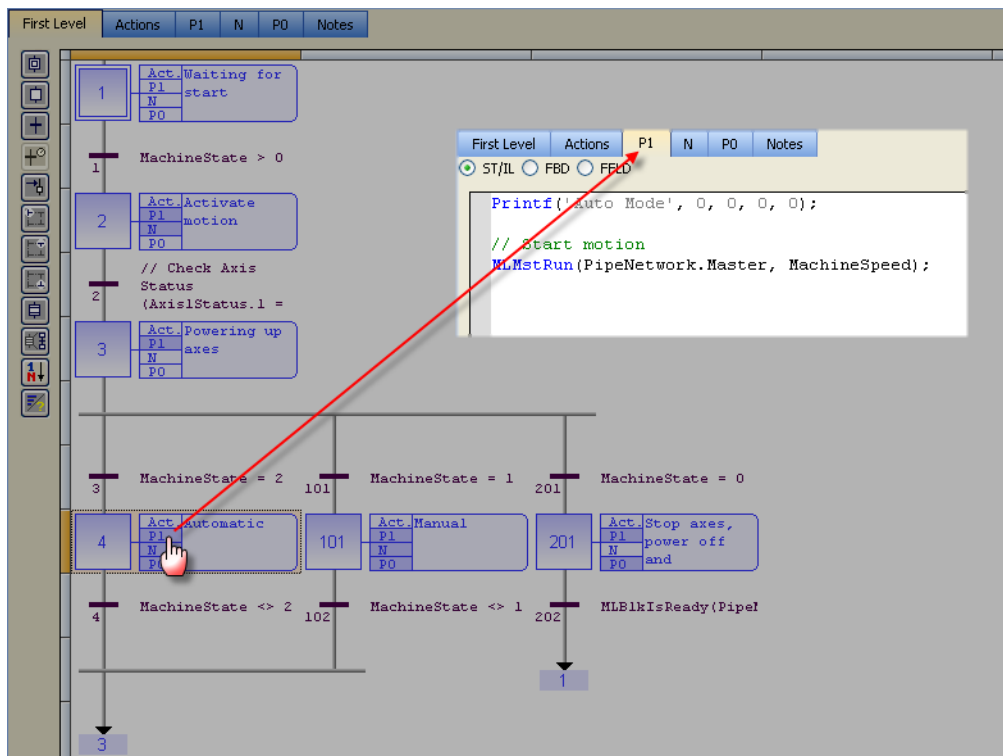
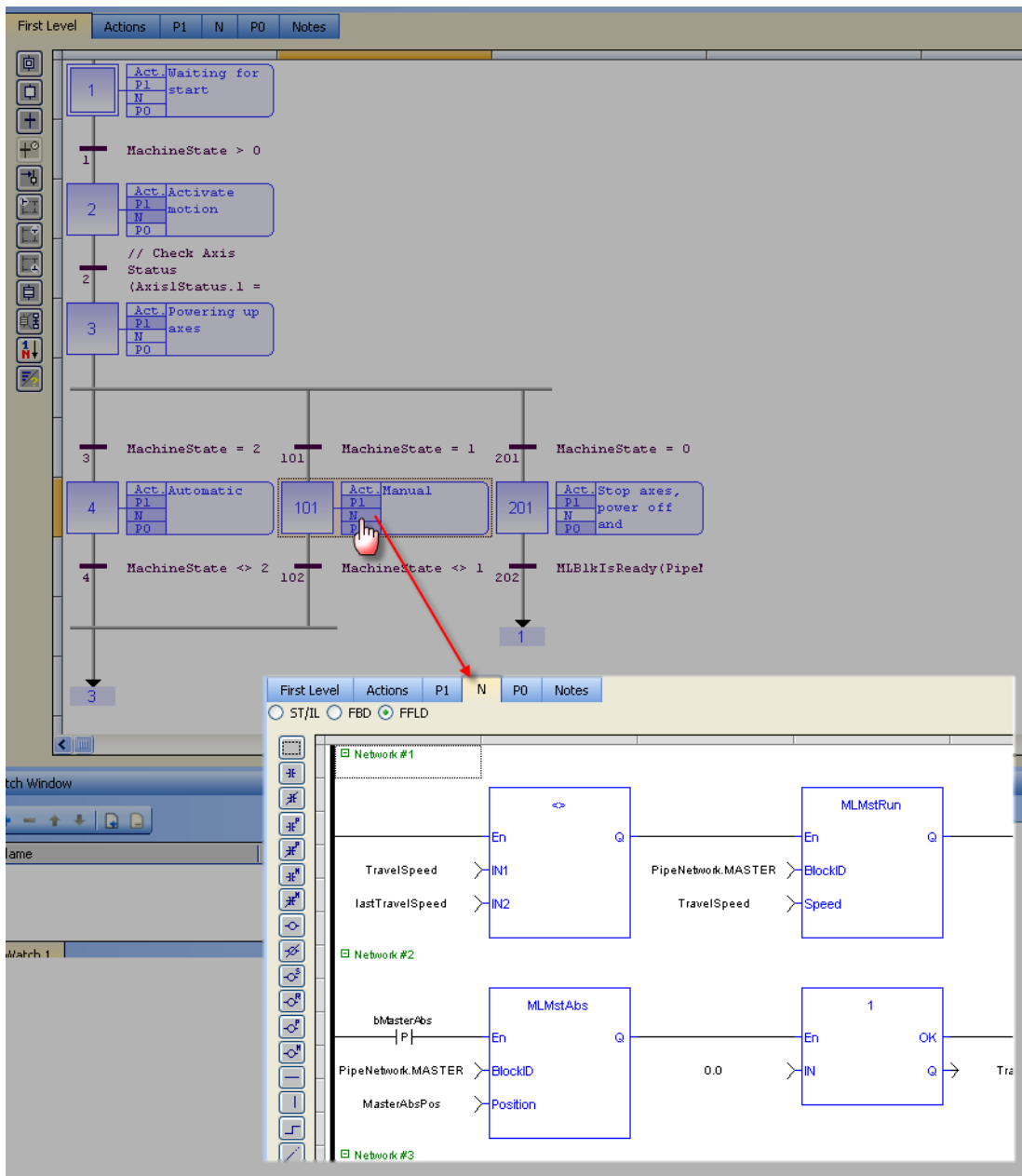


Figure 7-35: PN Template - MachineLogic

ST programs can be found in the P1 and P0 actions for many steps



FFLD programs can be found in the N action for steps 4 and 101



### 12.5.4 Motion

The template has a motion profile defined with the graphical Pipe Network editor.

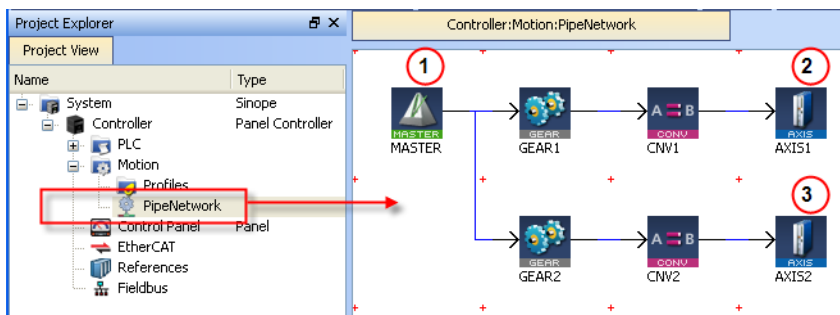


Figure 7-36: PN Template - Motion

The motion profile contains four different pipe blocks:

- The **Master** (see call out **1**) is the generator that allows a synchronization between the two pipes (**2** and **3**).
- The **Gear** modifies (with ratio and offset) the flow of values issued from the Master.
- The **Converter** controls the position of the axis.
- The **Axis** gives access to the physical remote drive

## 12.5.5 Control Panel

For more details, see [Design the Control Panel with the Internal Control Panel Editor](#)

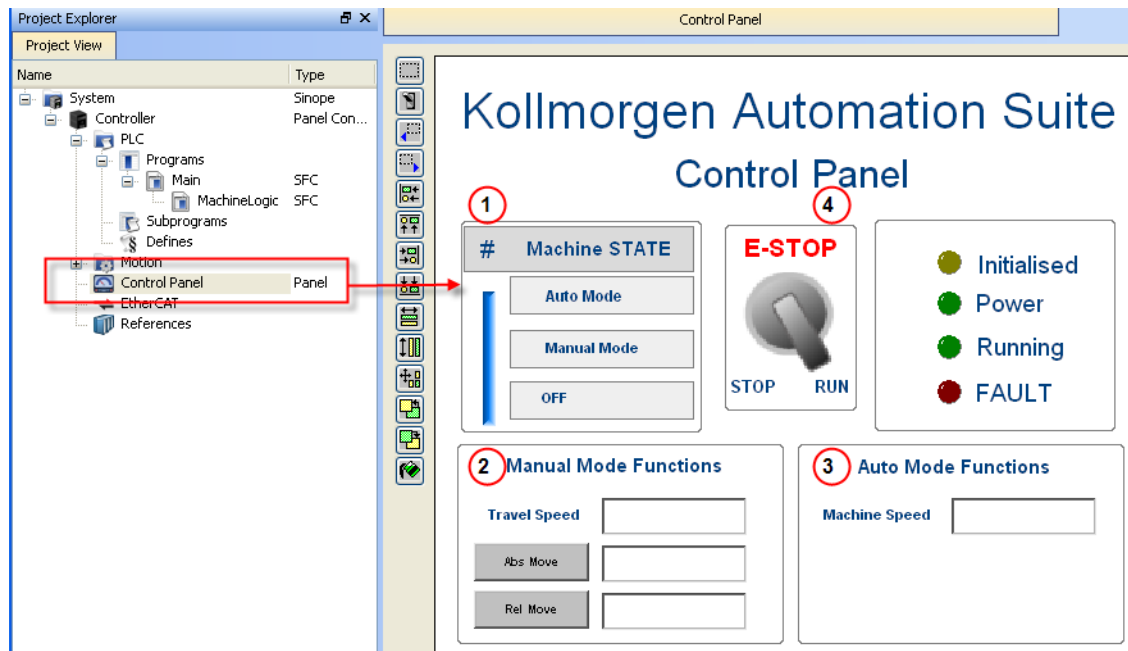


Figure 7-37: PN Template - Control Panel

Call out#	Description
<b>1</b>	Allows to choose how to run the axes between automatic and manual modes
<b>2</b>	In manual mode, you can set the speed. You can also set an absolute and relative move. When you click those commands, the two axes move to the specified position and the speed is reset to 0
<b>3</b>	In automatic mode, you can set the speed
<b>4</b>	When you click the emergency button, the machine state becomes OFF (see call out <b>1</b> ) and the two axes stop running

Table 7-35: PN Template - Control Panel

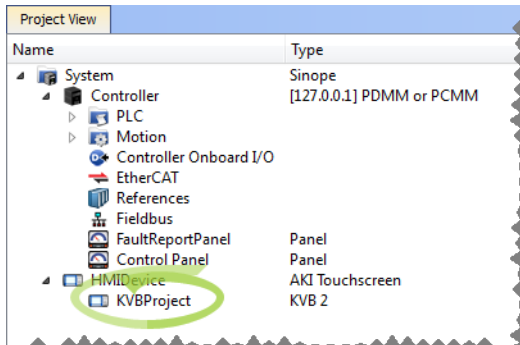
Based on the template, the project can be run:

- using the [KAS Simulator](#)
- with actual drives and motors (in this case, you first have to set up the axes in the EtherCAT part. For more details, see ["Configuring EtherCAT" on page 321](#))

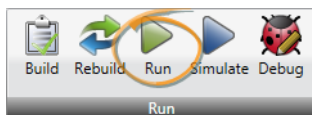
### 12.5.5.1 HMI

Also included in the project template is an HMI screen for the AKI terminals created in the KVB (Kollmorgen Visualization Builder) environment.

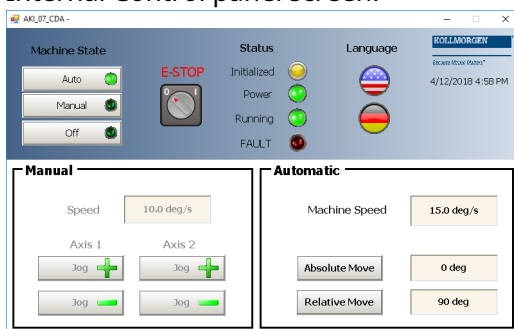
The KVB software is a separate installation from the KAS IDE. KVB can be opened from within KAS IDE by double-clicking on the **KVBProject** item in the project tree.



The project can be run in the KVB programming environment after the KAS IDE program has started running. Click on the Run button in the Project ribbon to start the program.



A screen, localized in English and German, is included which duplicate the controls on the IDE Internal Control panel screen.



## 12.5.6 Pipe Network 2-Axes Template with ST only

### 12.5.6.1 PLC Programs

The 2-axes Pipe Network template has a ST program (called **Main**) that initializes, starts and runs the motion.

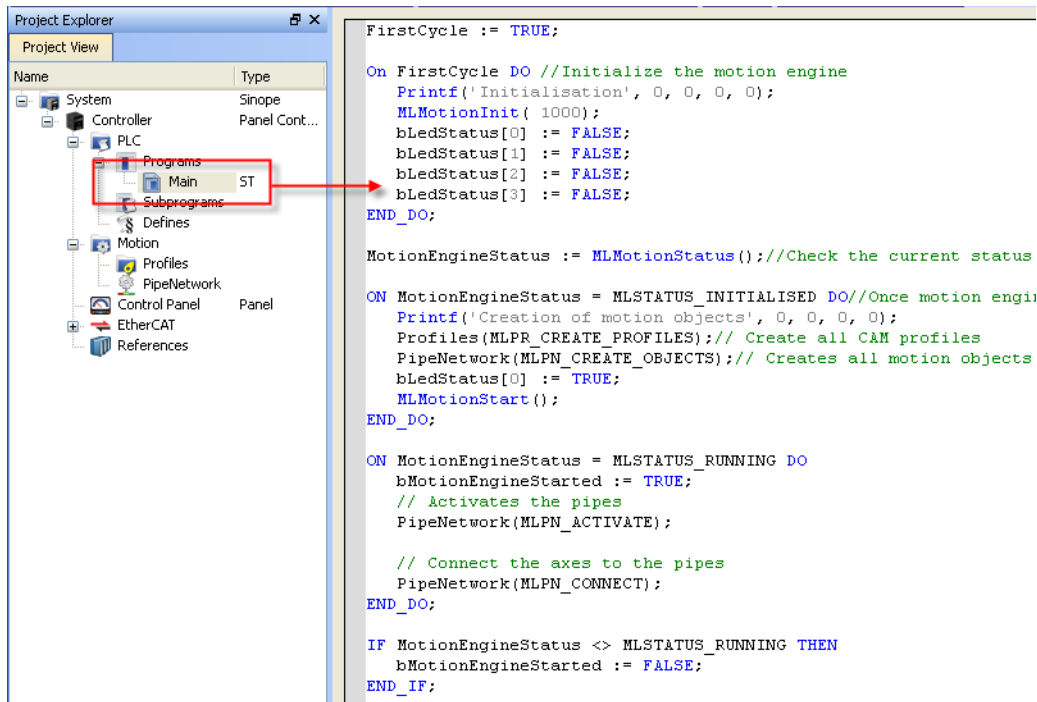


Figure 7-38: PN Template with ST - Main

## 12.5.7 Motion

The template has a motion profile defined with the graphical Pipe Network editor.

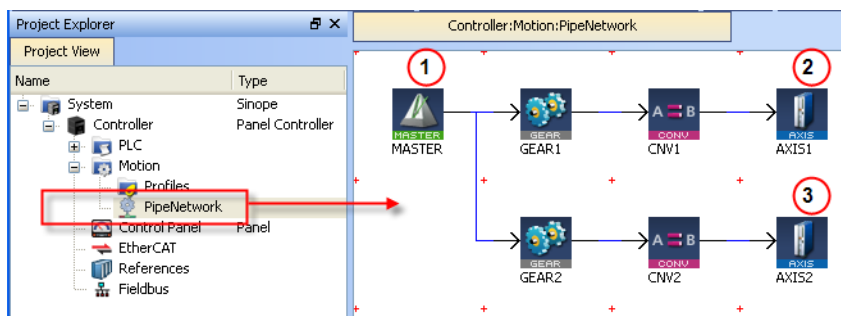


Figure 7-39: PN Template - Motion

The motion profile contains four different pipe blocks:

- The **Master** (see call out ①) is the generator that allows a synchronization between the two pipes (② and ③).
- The **Gear** modifies (with ratio and offset) the flow of values issued from the Master.
- The **Converter** controls the position of the axis.
- The **Axis** gives access to the physical remote drive

### 12.5.8 Control Panel

For more details, see [Design the Control Panel with the Internal Control Panel Editor](#)

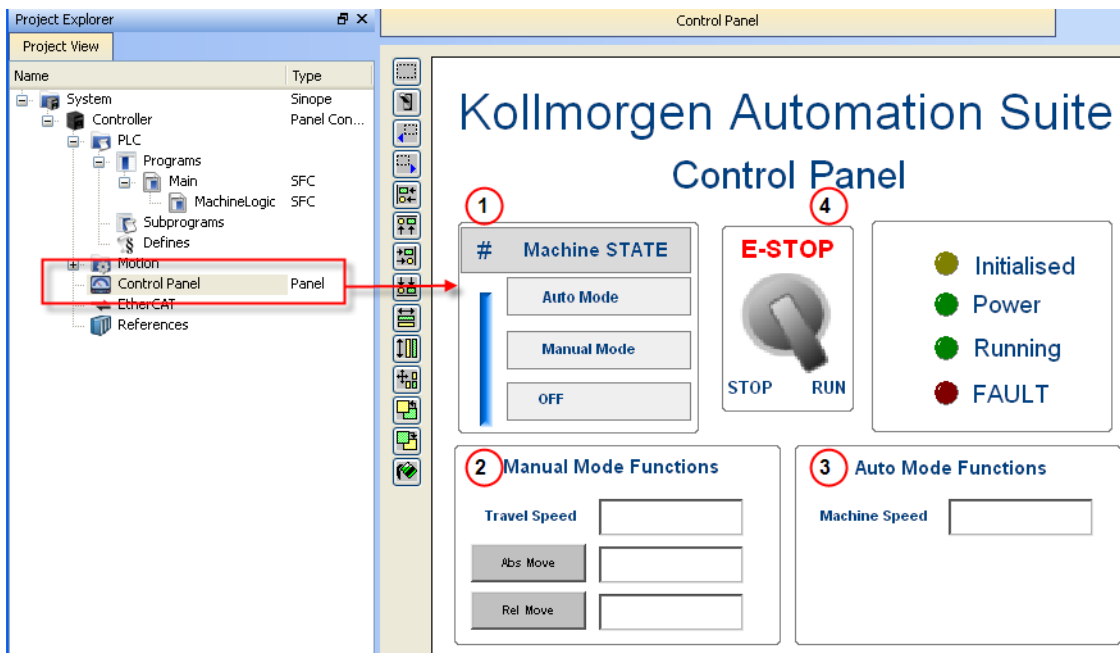


Figure 7-40: PN Template - Control Panel

Call out#	Description
1	Allows to choose how to run the axes between automatic and manual modes
2	In manual mode, you can set the speed. You can also set an absolute and relative move. When you click those commands, the two axes move to the specified position and the speed is reset to 0
3	In automatic mode, you can set the speed
4	When you click the emergency button, the machine state becomes OFF (see call out 1) and the two axes stop running

Table 7-36: PN Template - Control Panel

Based on the template, the project can be run:

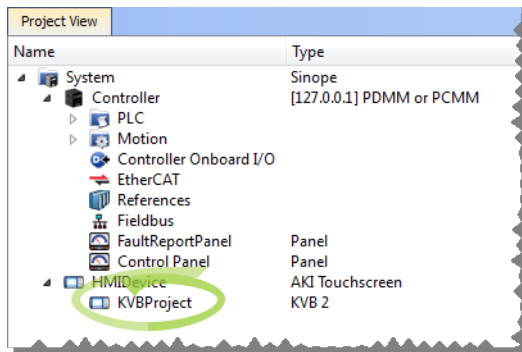
- using the [KAS Simulator](#)
- with actual drives and motors (in this case, you first have to set up the axes in the EtherCAT part. For more details, see ["Configuring EtherCAT" on page 321](#))

#### 12.5.8.1 HMI

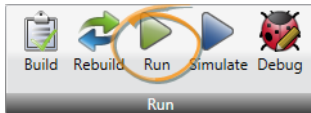
Also included in the project template is an HMI screen for the AKI terminals created in the KVB (Kollmorgen Visualization Builder) environment.

The KVB software is a separate installation from the KAS IDE. KVB can be opened from within KAS IDE by double-clicking on the **KVBProject** item in the project tree.

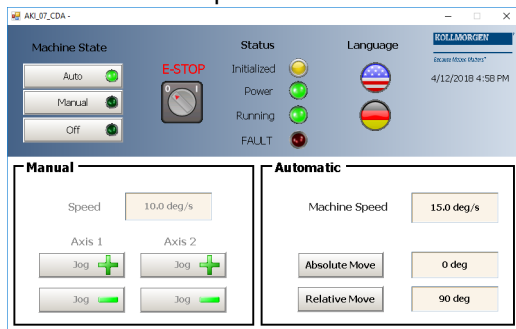




The project can be run in the KVB programming environment after the KAS IDE program has started running. Click on the Run button in the Project ribbon to start the program.



A screen, localized in English and German, is included which duplicate the controls on the IDE Internal Control panel screen.



## 12.5.9 Pipe Network 2-Axes Template with FFLD only

### 12.5.9.1 PLC Programs

The 2-axes Pipe Network template has a FFLD program (called **Main**) that initializes, starts and runs the motion.

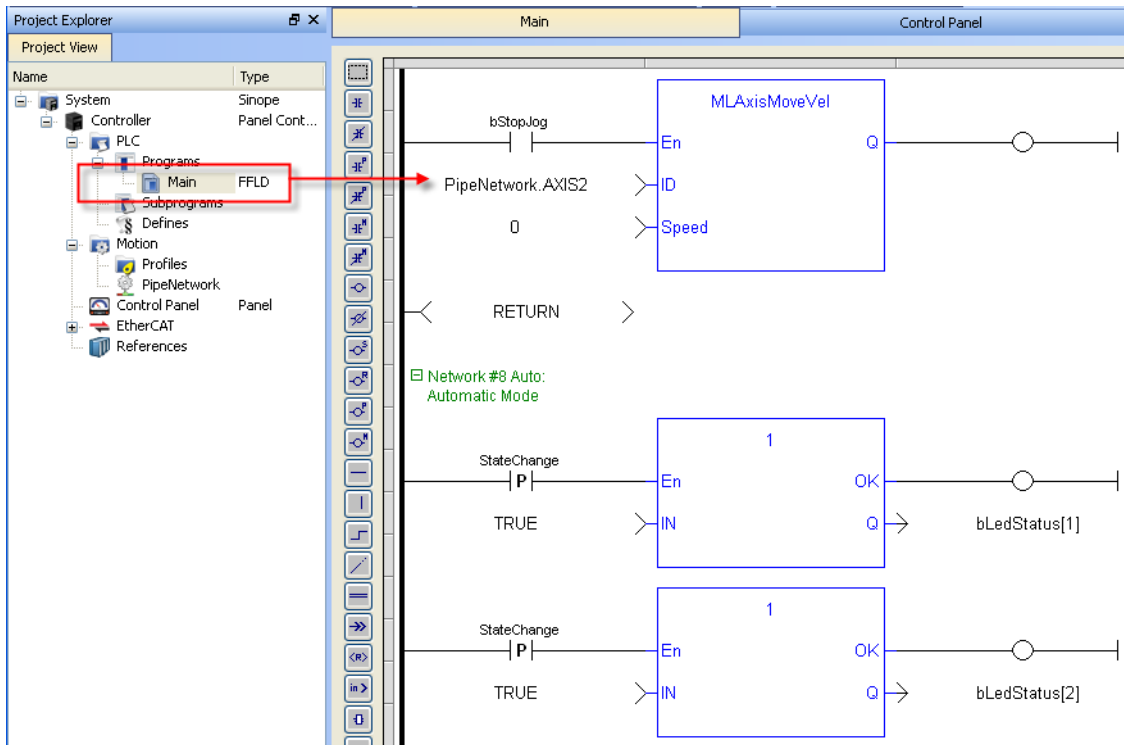


Figure 7-41: PN Template with FFLD - Main

## 12.5.10 Motion

The template has a motion profile defined with the graphical Pipe Network editor.

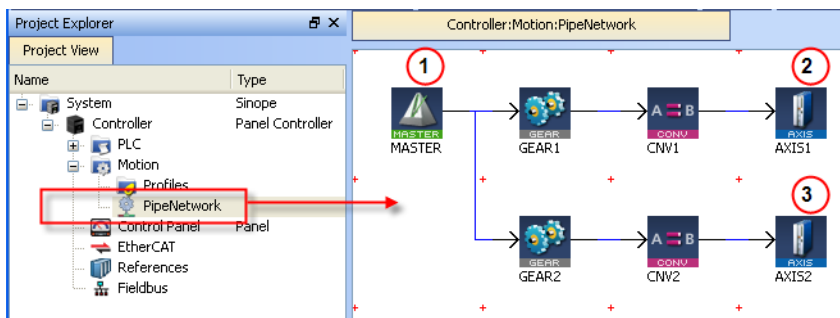


Figure 7-42: PN Template - Motion

The motion profile contains four different pipe blocks:

- The **Master** (see call out 1) is the generator that allows a synchronization between the two pipes (2 and 3).
- The **Gear** modifies (with ratio and offset) the flow of values issued from the Master.
- The **Converter** controls the position of the axis.
- The **Axis** gives access to the physical remote drive

### 12.5.11 Control Panel

For more details, see [Design the Control Panel with the Internal Control Panel Editor](#)

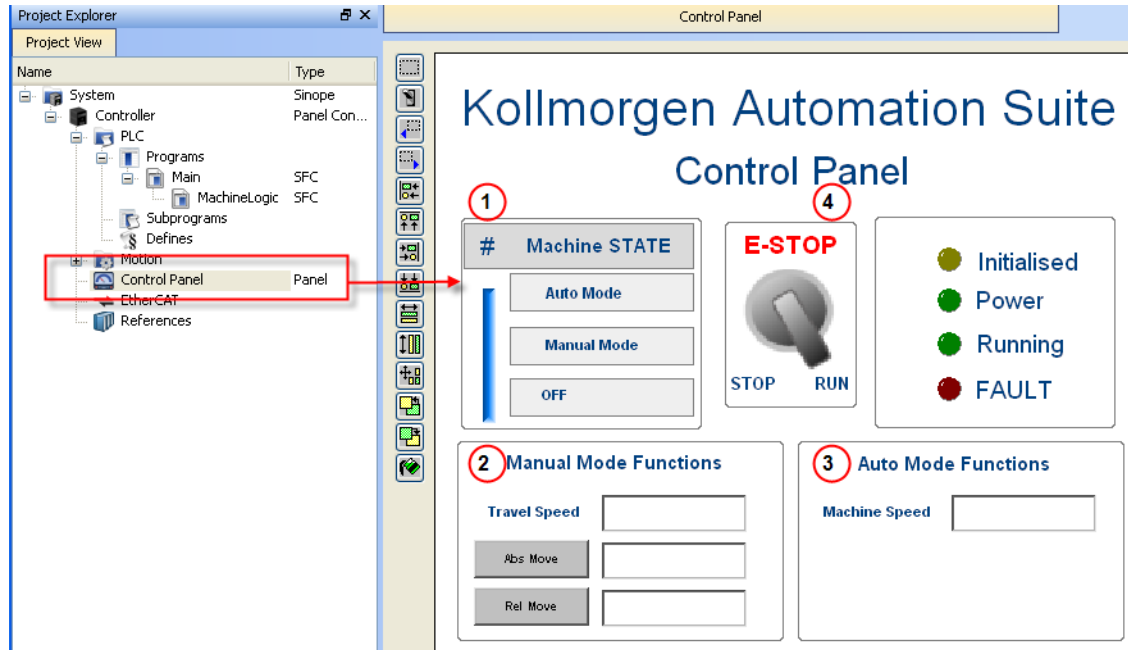


Figure 7-43: PN Template - Control Panel

Call out#	Description
1	Allows to choose how to run the axes between automatic and manual modes
2	In manual mode, you can set the speed. You can also set an absolute and relative move. When you click those commands, the two axes move to the specified position and the speed is reset to 0
3	In automatic mode, you can set the speed
4	When you click the emergency button, the machine state becomes OFF (see call out 1) and the two axes stop running

Table 7-37: PN Template - Control Panel

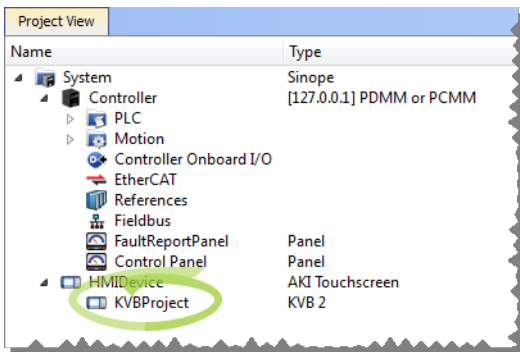
Based on the template, the project can be run:

- using the [KAS Simulator](#)
- with actual drives and motors (in this case, you first have to set up the axes in the EtherCAT part. For more details, see ["Configuring EtherCAT" on page 321](#))

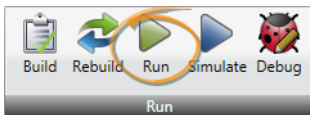
#### 12.5.11.1 HMI

Also included in the project template is an HMI screen for the AKI terminals created in the KVB (Kollmorgen Visualization Builder) environment.

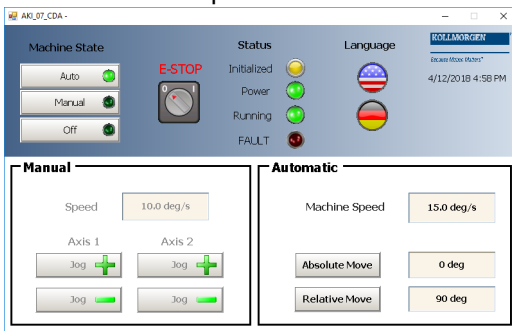
The KVB software is a separate installation from the KAS IDE. KVB can be opened from within KAS IDE by double-clicking on the **KVBProject** item in the project tree.



The project can be run in the KVB programming environment after the KAS IDE program has started running. Click on the Run button in the Project ribbon to start the program.



A screen, localized in English and German, is included which duplicate the controls on the IDE Internal Control panel screen.



### 12.5.12 PLCopen 1-Axis Template with FFLD

The 1-axis PLCopen template has a FFLD program (called **Main**) that initializes and starts the motion. Also included are controls to enable/disable the drive, clear faults, and display fault and axis position information in the Control Panel. A second program, FaultMonitor, reads the drive fault and position information. The programs are commented to ease their use.

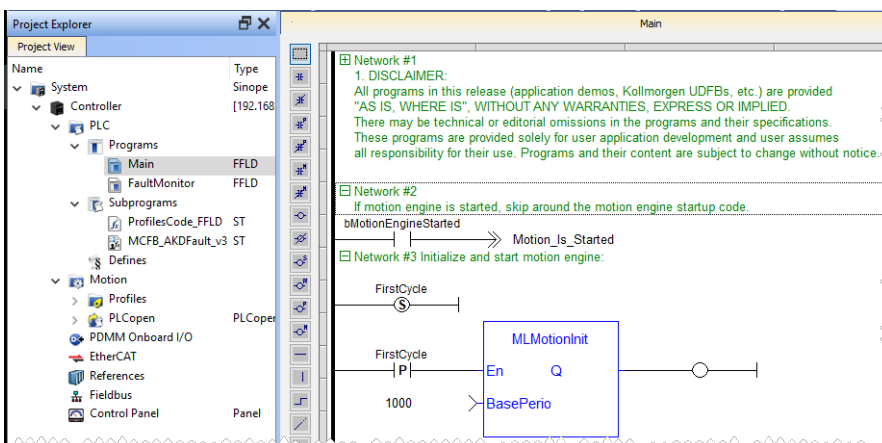
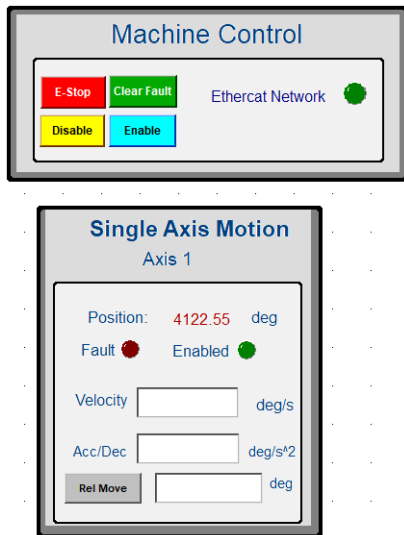


Figure 7-44: PLCopen Template with FFLD - Main



### 12.5.13 PLCopen 1-Axis Template with ST

This 1-axis PLCopen template has a ST program (called **Main**) that initializes, starts and runs the motion. Also included are controls to enable/disable the drives, clear faults, and display fault and axis position information in the Control Panel. A subprogram reads the drive fault and position information. The programs are commented to ease their use.

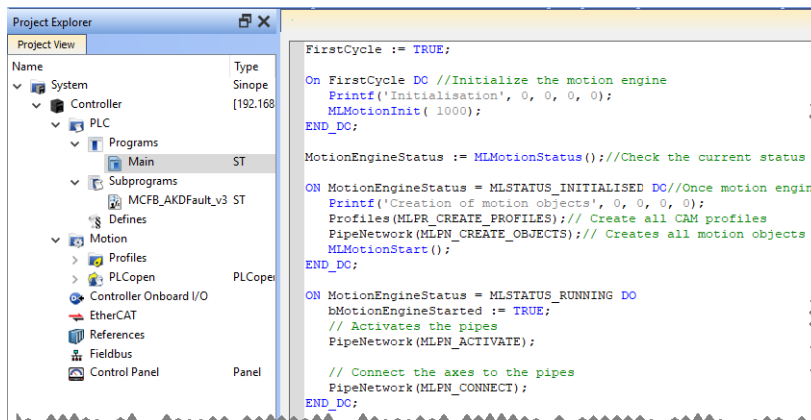
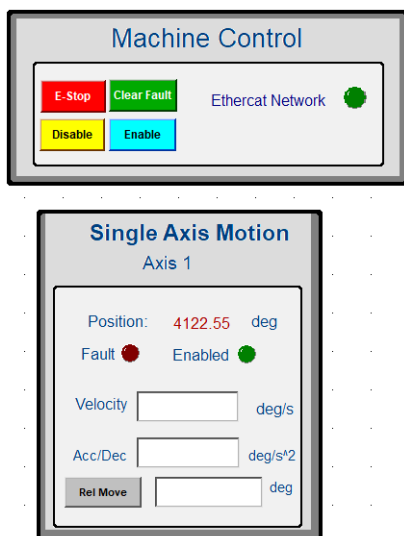


Figure 7-45: PLCopen Template with ST - Main



### 12.5.14 PLCopen 2-Axes Template with SFC and FFLD

This template project contains a two-axis template for getting started with a KAS application. It contains key project elements that will save development time. Additional axes can be added to the template applications that have three, four, five, or more axes. The templates are designed to easily add code to the programs and Control Panels that are in the template or to new ones. A project can contain many programs, and each program can be written in ST, SFC, FFLD, FBD, or IL programming languages. This template program can be run on the KAS Simulator or with a AKD PDMM or PCMM and AKD drives.

This project contains two axes where Axis 2 is slaved to Axis 1 at a 2:1 ratio.

#### 12.5.14.1 PLC Programs

The 2-axes PLCopen template has an SFC program (called **Main**) that initializes and starts the motion. The motion includes single axis jogging, incremental and absolute moves, and gearing motion between axes. Also included are controls to enable/disable the drives, clear faults, and display fault and axis position information in the Control Panel.

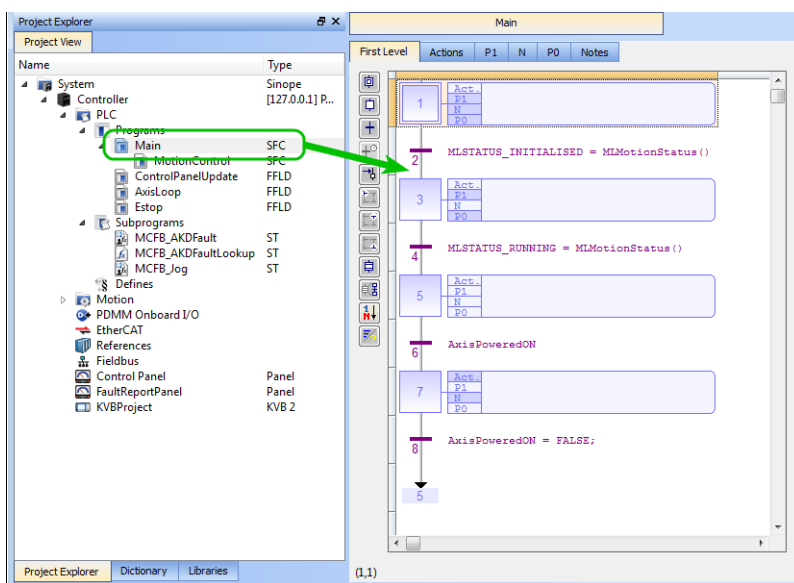


Figure 7-46: PLCopen - Template Main

Step 5 of the Main program in the PLCopen template contains the FFLD code for running the motion. As defined below with the MoveVelocity function block, the motion profile is based on a [trapezoidal](#) acceleration/deceleration.

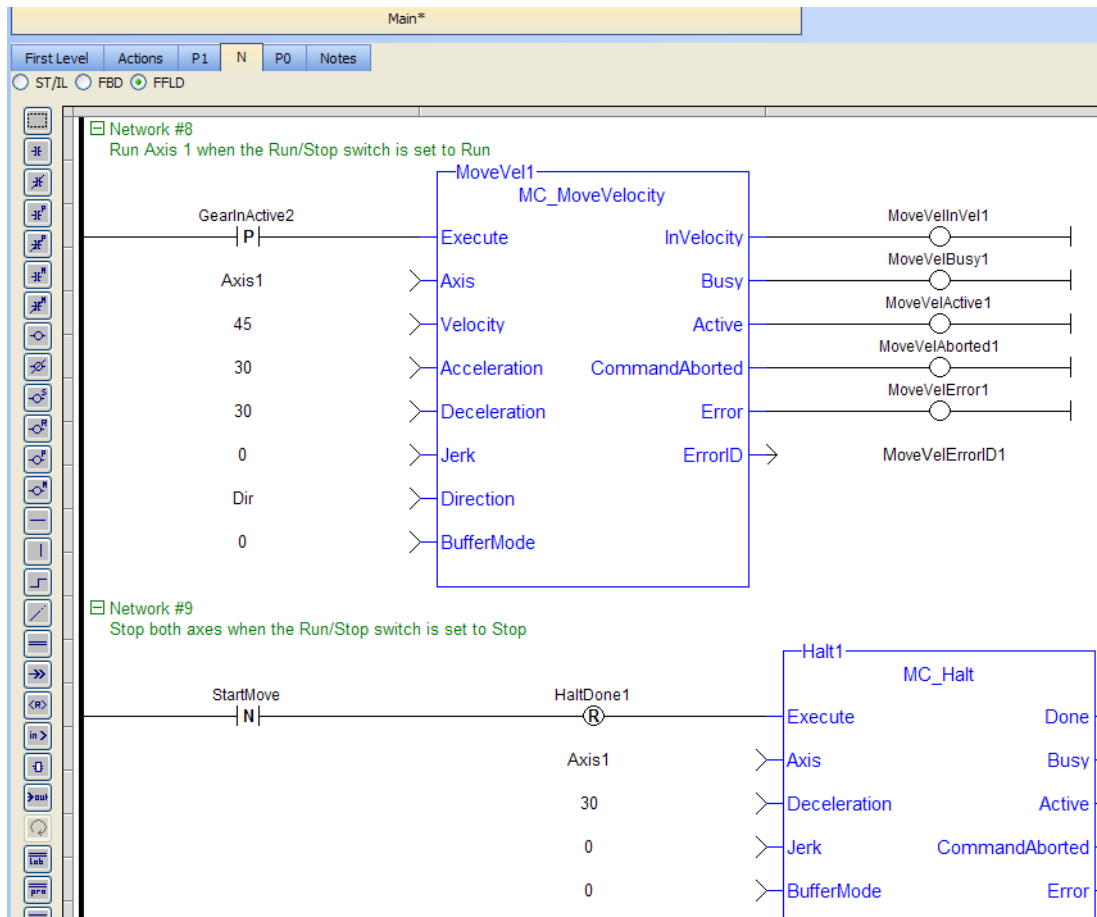


Figure 7-47: PLCopen Template - Step 5 of the Main

### 12.5.14.2 Motion

The template contains two PLCopen Servo axes where User Units, Update Rate, Rollover Position, and Axis Limits are defined as follows:

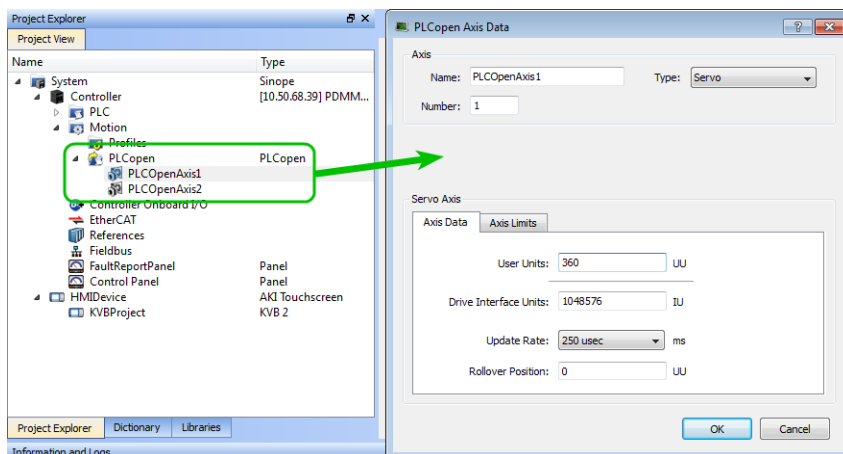


Figure 7-48: PLCopen Template - Motion

For more details on PLCopen axis parameters, see "Axis Data Parameters" on page 157

### 12.5.14.3 Control Panel

The template contains a Control Panel which works inside of the IDE when running a project, and serves as a useful tool to run and debug programs. The Control Panel consists of two screens, named **Control Panel** and **Fault Report Panel**.

**Control Panel** consists of three sections.

- **Multi-Axis Commands** which controls motion and shows gearing
- **Single Axis Motion Commands** which shows motion at the single axis level and resetting the axis position
- **Axis Status** which shows basic axis information

For more details, see [Design the Control Panel with the Internal Control Panel Editor](#)

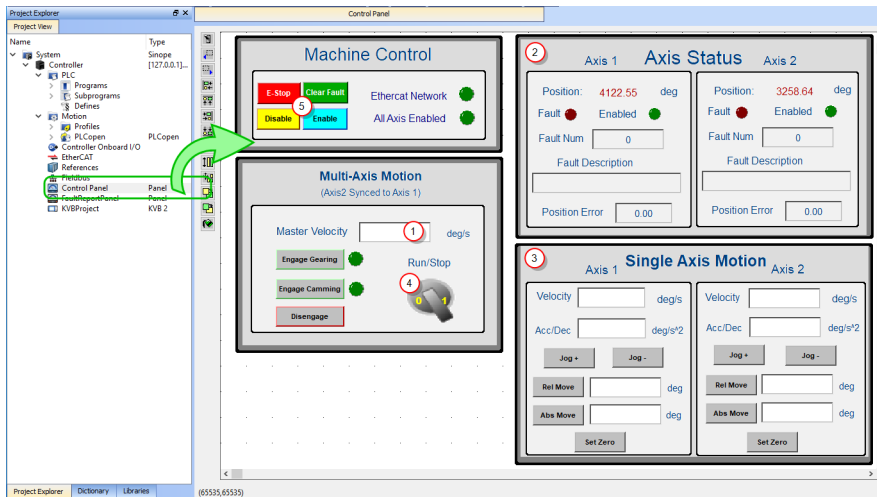


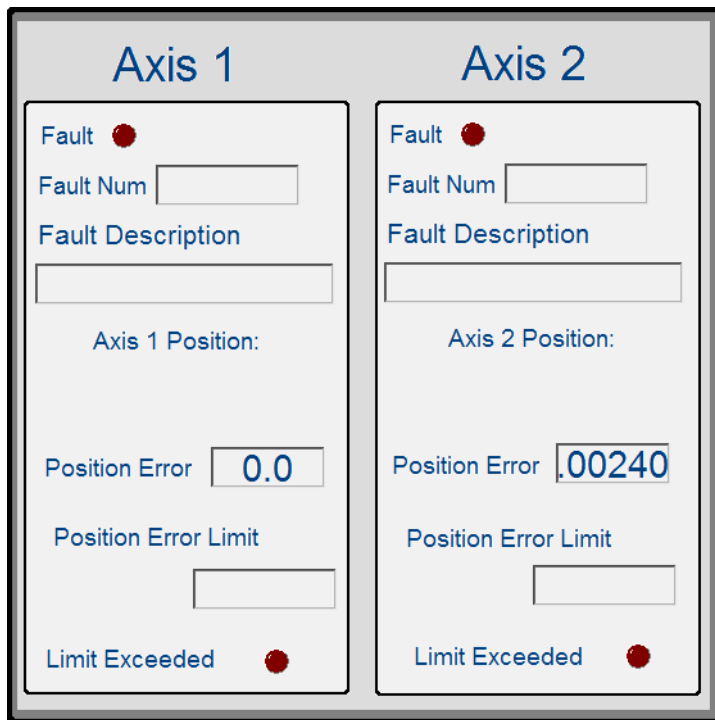
Figure 7-49: PLCopen Template - Control Panel

Call out#	Description
1	Allows you to set the speed
2	Displays the actual position for each axis
3	Control the axis motion
4	Start or stop the motion on the condition that the axes are enable (the green light must be switched on)
5	Allows for enabling or disabling the axes. After an emergency stop, you need to select the Reset and Enable commands before running the axes

Table 7-38: PLCopen Template - Control Panel elements

Fault Report Panel provides more detailed information about when a drive fault occurs.





**Figure 7-50:** PLCopen Template - Fault Report Panel

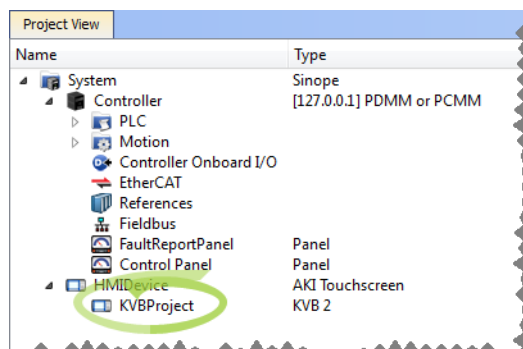
Based on the template, the project can be run:

- using the [KAS Simulator](#)
- with actual drives and motors (in this case, you first have to set up the axes in the EtherCAT part. For more details, see "[Configuring EtherCAT](#)" on page 321)

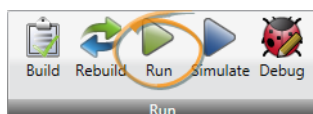
#### 12.5.14.4 HMI

Also included in the project template are HMI screens for the AKI terminals created in the KVB (Kollmorgen Visualization Builder) environment.

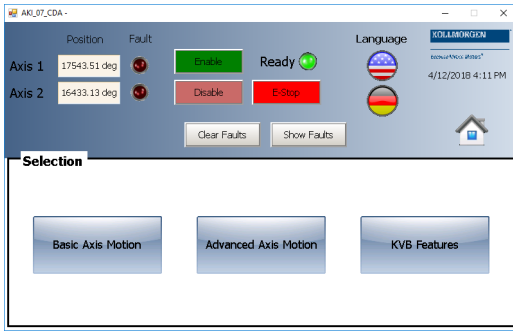
The KVB software is a separate installation from the KAS IDE. KVB can be opened from within KAS IDE by double-clicking on the **KVBProject** item in the project tree.



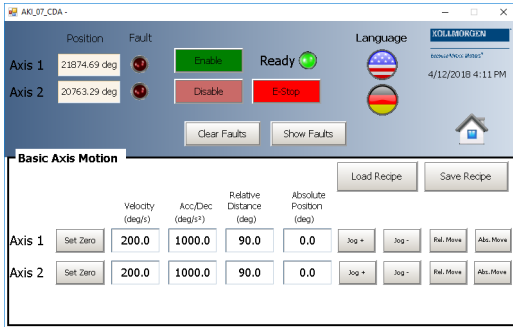
The project can be run in the KVB programming environment after the KAS IDE program has started running. Click on the Run button in the Project ribbon to start the program.



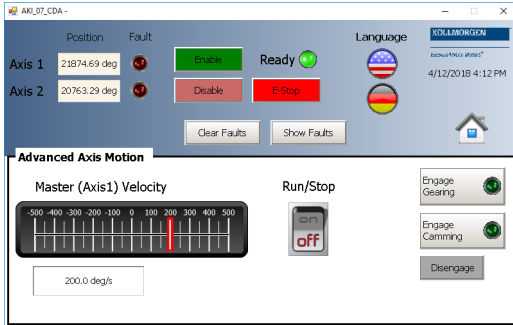
Four screens, localized in English and German, are included that duplicate the controls on the IDE Internal Control panel screens.



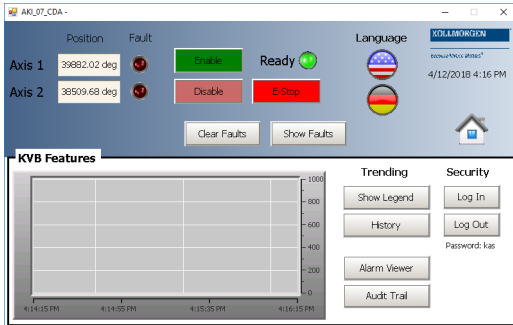
Home Screen



Basic Motion



Advanced Motion



KVB Features

### 12.5.15 PLCopen 2-Axes Template with ST

This template project contains a two-axis template for getting started with a KAS application. It contains key project elements that will save development time. Additional axes can be added to the template applications that have three, four, five, or more axes. The templates are designed to easily add code to the programs and Control Panels that are in the template or to new ones. A project can contain many programs, and each program can be written in ST, SFC, FFLD, FBD, or IL programming languages. This template program can be run on the KAS Simulator or with a AKD PDMM or PCMM and AKD drives.

#### 12.5.15.1 PLC Programs

This 2-axis PLCopen template has a ST program (called **Main**) that initializes, starts and runs the motion. The motion includes single axis jogging, incremental and absolute moves, and gearing motion between axes. Also included are controls to enable/disable the drives, clear faults, and display fault and axis position information in the Control Panel. A second program, FaultMonitor, reads the drive fault and position information.

The programs are commented to ease their use.

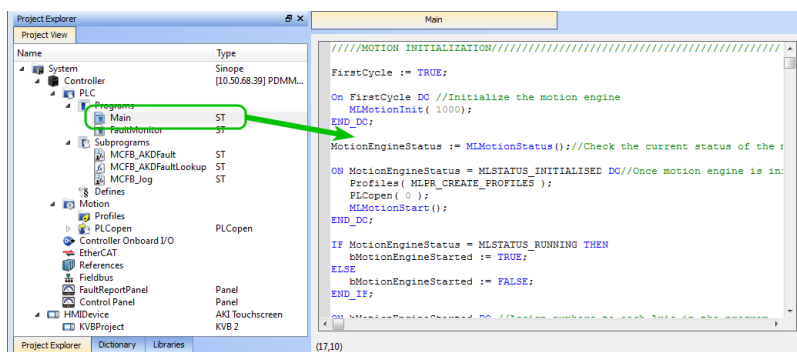


Figure 7-51: PLCopen Template with ST - Main

Also included are three subprograms (**MCFB\_AKDFault**, **MCFB\_AKDFaultLookup**, and **MCFB\_Jog**) that are Kollmorgen UDFBs and can be found in the Project Library section of the IDE.

#### 12.5.15.2 Motion

The template contains two PLCopen Servo axes where User Units, Update Rate, Rollover Position, and Axis Limits are defined as follows:

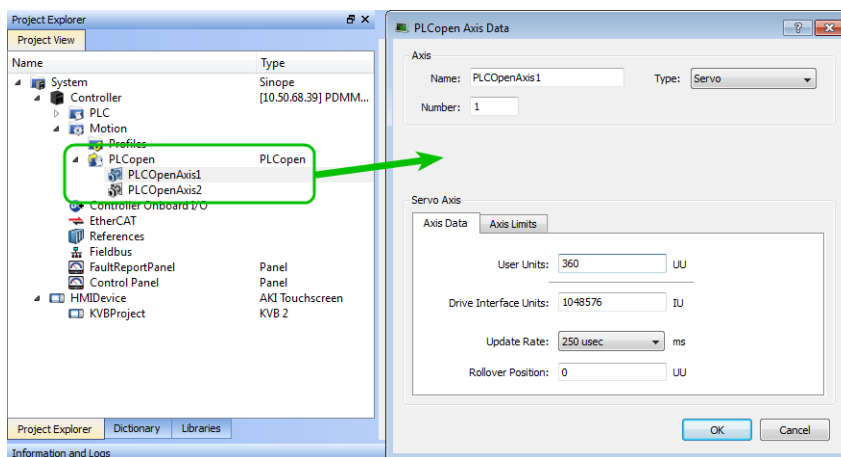


Figure 7-52: PLCopen Template - Motion

For more details on PLCopen axis parameters, see ["Axis Data Parameters"](#) on page 157

### 12.5.15.3 Control Panel

The template contains a Control Panel which works inside of the IDE when running a project, and serves as a useful tool to run and debug programs. The Control Panel consists of two screens, named **Control Panel** and **Fault Report Panel**.

**Control Panel** consists of three sections.

- **Multi-Axis Commands** which controls motion and shows gearing
- **Single Axis Motion Commands** which shows motion at the single axis level and resetting the axis position
- **Axis Status** which shows basic axis information

For more details, see [Design the Control Panel with the Internal Control Panel Editor](#)

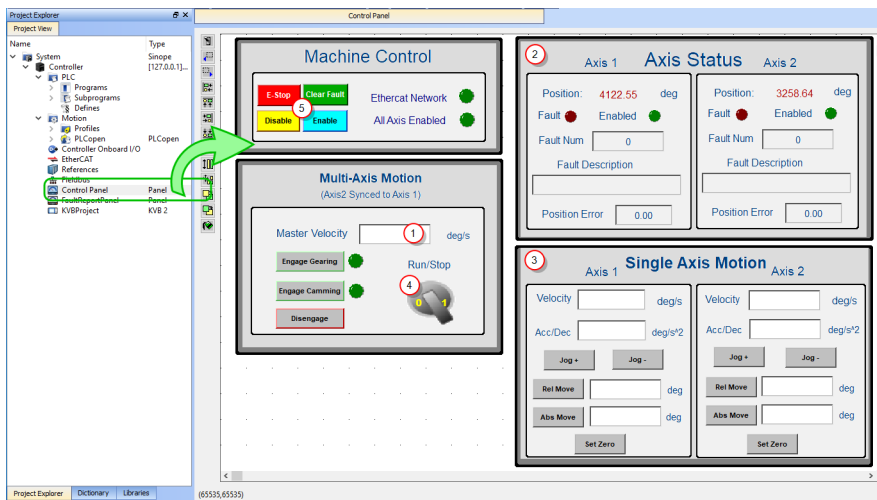


Figure 7-53: PLCopen Template - Control Panel

Call out#	Description
1	Allows you to set the speed
2	Displays the actual position for each axis
3	Control the axis motion
4	Start or stop the motion on the condition that the axes are enable (the green light must be switched on)
5	Allows for enabling or disabling the axes. After an emergency stop, you need to select the Reset and Enable commands before running the axes

Table 7-39: PLCopen Template - Control Panel elements

Fault Report Panel provides more detailed information about when a drive fault occurs.

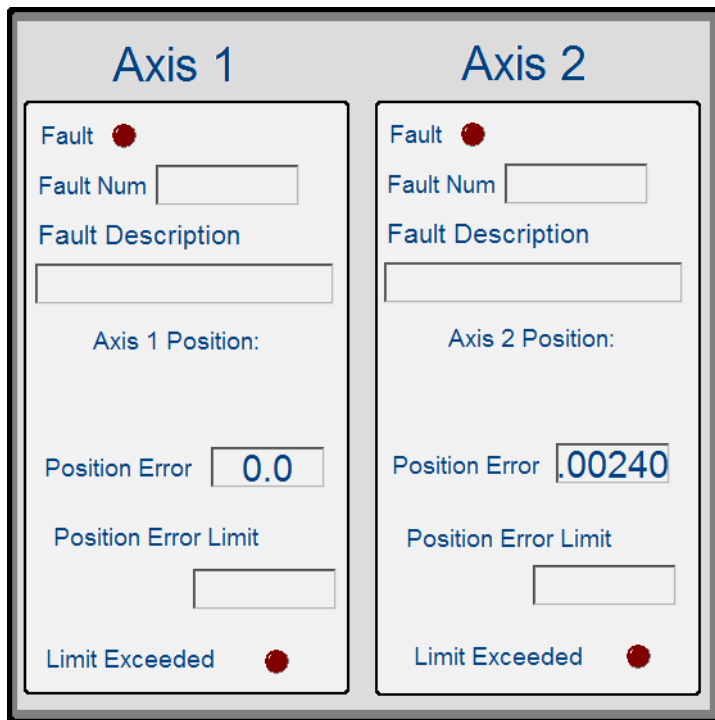
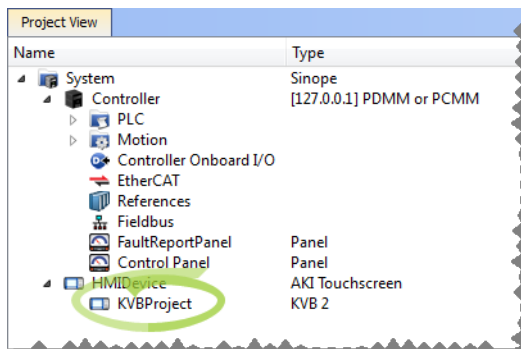


Figure 7-54: PLCopen Template - Fault Report Panel

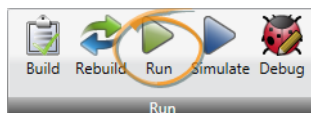
#### 12.5.15.4 HMI

Also included in the project template are HMI screens for the AKI terminals created in the KVB (Kollmorgen Visualization Builder) environment.

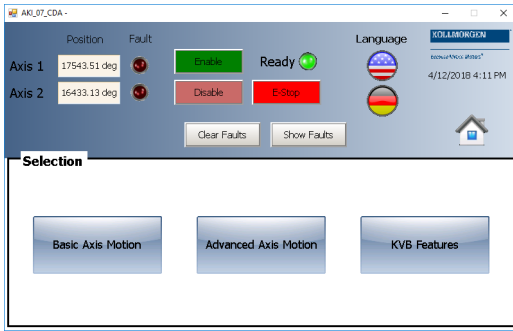
The KVB software is a separate installation from the KAS IDE. KVB can be opened from within KAS IDE by double-clicking on the **KVBProject** item in the project tree.



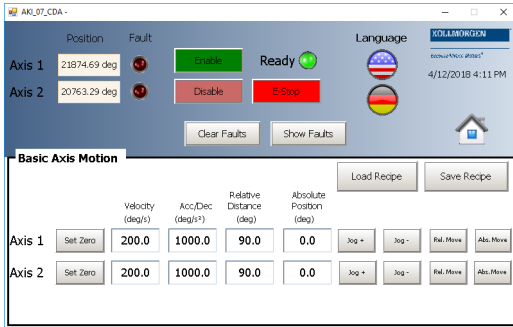
The project can be run in the KVB programming environment after the KAS IDE program has started running. Click on the Run button in the Project ribbon to start the program.



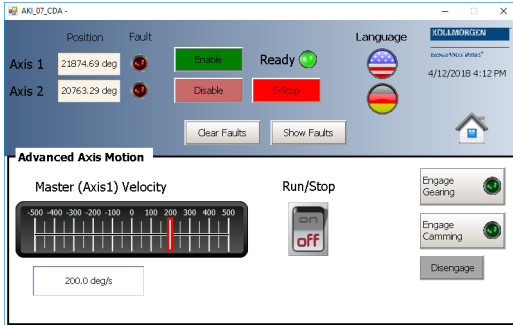
Four screens, localized in English and German, are included that duplicate the controls on the IDE Internal Control panel screens.



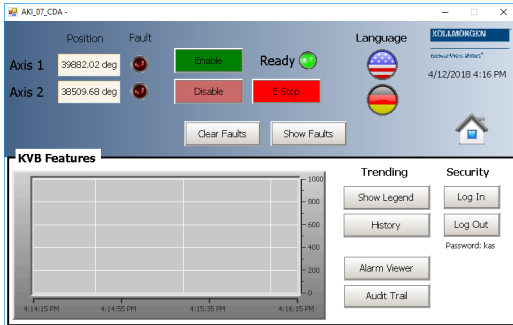
Home Screen



Basic Motion



Advanced Motion



KVB Features

### 12.5.16 PLCopen 2-Axes Template with FFLD

This template project contains a two-axis template for getting started with a KAS application. It contains key project elements that will save development time. Additional axes can be added to the template applications that have three, four, five, or more axes. The templates are designed to easily add code to the programs and Control Panels that are in the template or to new ones. A project can contain many programs, and each program can be written in ST, SFC, FFLD, FBD, or IL programming languages. This template program can be run on the KAS Simulator or with a AKD PDMM or PCMM and AKD drives.

This project contains two axes where Axis 2 is slaved to Axis 1 at a 2:1 ratio.

#### 12.5.16.1 PLC Programs

The 2-axes PLCopen template has a FFLD program (called **Main**) that initializes and starts the motion. The motion includes single axis jogging, incremental and absolute moves, and gearing motion between axes. Also included are controls to enable/disable the drives, clear faults, and display fault and axis position information in the Control Panel.

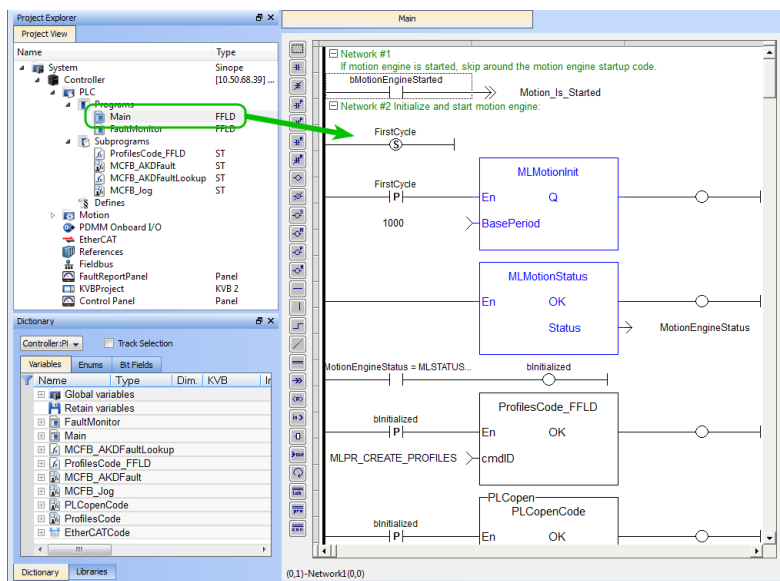
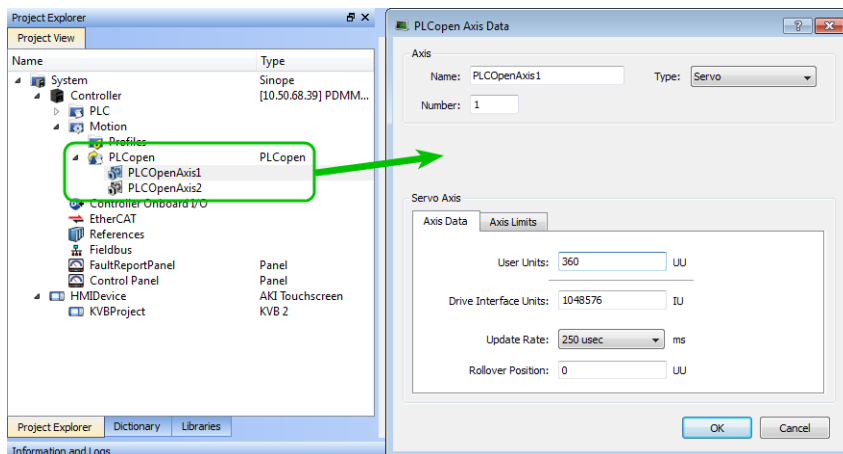


Figure 7-55: PLCopen Template with FFLD - Main

#### 12.5.16.2 Motion

The template contains two PLCopen Servo axes where User Units, Update Rate, Rollover Position, and Axis Limits are defined as follows:



**Figure 7-56: PLCopen Template - Motion**

For more details on PLCopen axis parameters, see ["Axis Data Parameters" on page 157](#)

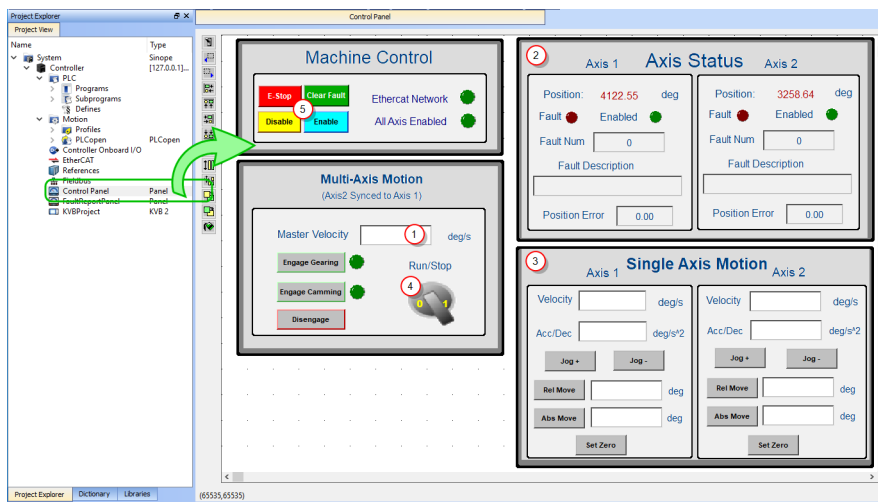
### 12.5.16.3 Control Panel

The template contains a Control Panel which works inside of the IDE when running a project, and serves as a useful tool to run and debug programs. The Control Panel consists of two screens, named **Control Panel** and **Fault Report Panel**.

**Control Panel** consists of three sections.

- **Multi-Axis Commands** which controls motion and shows gearing
- **Single Axis Motion Commands** which shows motion at the single axis level and resetting the axis position
- **Axis Status** which shows basic axis information

For more details, see [Design the Control Panel with the Internal Control Panel Editor](#)



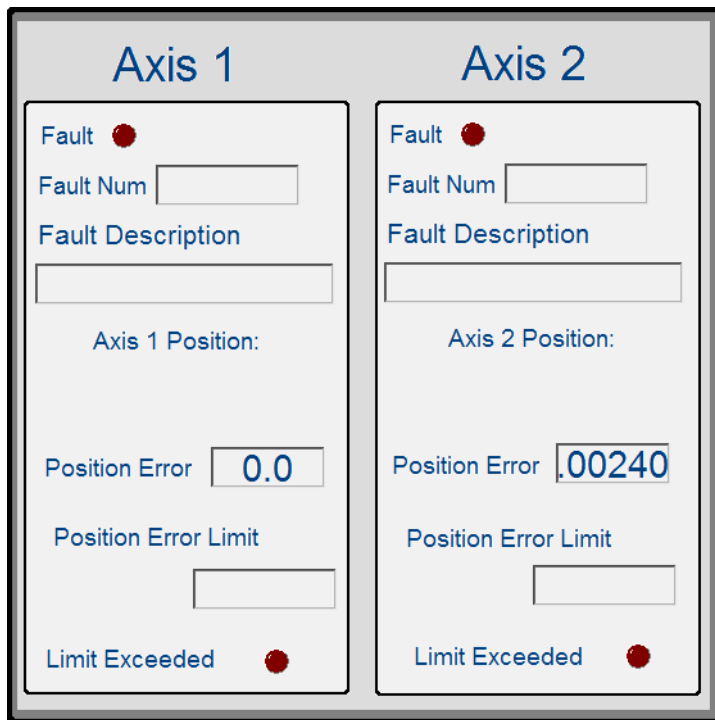
**Figure 7-57: PLCopen Template - Control Panel**

Call out#	Description
1	Allows you to set the speed
2	Displays the actual position for each axis
3	Control the axis motion
4	Start or stop the motion on the condition that the axes are enable (the green light must be switched on)
5	Allows for enabling or disabling the axes. After an emergency stop, you need to select the Reset and Enable commands before running the axes

Table 7-40: PLCopen Template - Control Panel elements

Fault Report Panel provides more detailed information about when a drive fault occurs.





**Figure 7-58:** PLCopen Template - Fault Report Panel

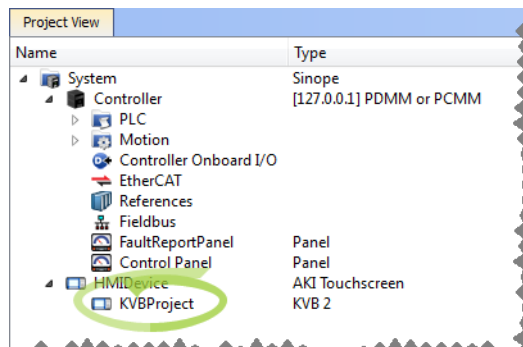
Based on the template, the project can be run:

- using the [KAS Simulator](#)
- with actual drives and motors (in this case, you first have to set up the axes in the EtherCAT part. For more details, see "[Configuring EtherCAT](#)" on page 321)

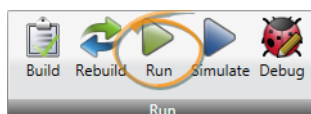
#### 12.5.16.4 HMI

Also included in the project template are HMI screens for the AKI terminals created in the KVB (Kollmorgen Visualization Builder) environment.

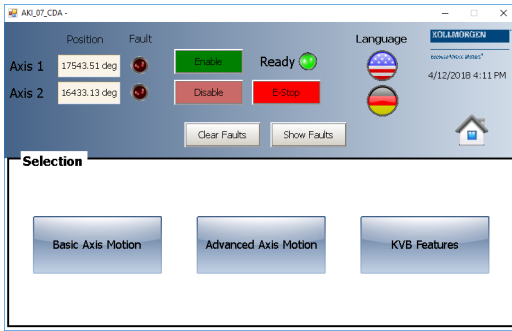
The KVB software is a separate installation from the KAS IDE. KVB can be opened from within KAS IDE by double-clicking on the **KVBProject** item in the project tree.



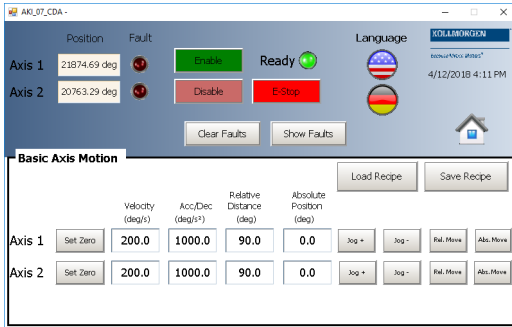
The project can be run in the KVB programming environment after the KAS IDE program has started running. Click on the Run button in the Project ribbon to start the program.



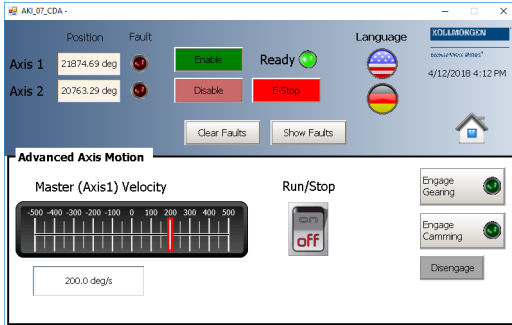
Four screens, localized in English and German, are included that duplicate the controls on the IDE Internal Control panel screens.



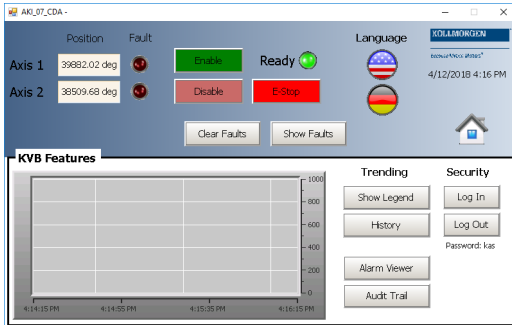
Home Screen



Basic Motion



Advanced Motion



KVB Features

### 12.5.17 Coordinated Motion 2-Axis Template

This project controls two axes in coordinated motion (PLCOpenAxis1 and PLCOpenAxis2).

#### 12.5.17.1 Programs

The program is Sequential Function Chart (SFC) containing both Structured Text (ST) and Free Form Ladder Diagram (FFLD) code.

The first five steps of the SFC create and initialize the axes and the coordinated motion axes group. Step six of the SFC interfaces with the Control Panel and performs a back-and-forth coordinated motion pattern with the two axes. The program utilizes coordinated motion direct moves, linear moves, circular moves, transitions and blending.

#### 12.5.17.2 Motion

To run the coordinated motion pattern, perform the following steps:

1. Download and start the application.
2. Press the "Enable" button to enable the axes and the axes group.
3. Press the "Home" button to move the axes to their zero position. (optional)
4. After the "Ready" light is illuminated, turn the "Cycle Start" switch to "1" and the axes will begin moving in programmed pattern.

#### 12.5.17.3 Control Panel

#	Description	GUI
1	Allows you to change the coordinated motion feedrate from 0% to 200%	
2	Displays the axis positions	
3	Indicates that the three axes are enabled and the axes group is ready and enabled for coordinated motion	
4	Begins the coordinated motion pattern	
5	Causes an Emergency Stop for all axes. The axes will be disabled.	
6	Resets the Emergency Stop and enables the axes	
7	Disables the two axes	
8	Enables the two axes and the axes group	
9	Moves all axes to their zero position	
10	Causes a reference by setting the axes' position to zero at their current location.	

## 12.5.18 Coordinated Motion 3-Axis Template

This project controls two axes in coordinated motion (PLCOpenAxis1 and PLCOpenAxis2), and a third independent axis (VERTICAL\_AXIS). This template demonstrates how to use coordinated motion PLCopen axes and a Pipe Network axis.

### 12.5.18.1 PLC Programs

The Coordinated Motion 3-Axis template has a Sequential Function Chart program (SFC) containing both Structured Text (ST) and Free Form Ladder Diagram (FFLD) code.

The first five steps of the SFC program create and initialize the axes and the coordinated motion axes group plus the Pipe Network axis.

Step 6 of the SFC interfaces with the Control Panel and performs a back-and-forth coordinated motion pattern with the two axes. The program utilizes coordinated motion direct moves, linear moves, circular moves, transitions and blending. It also performs basic moves for the third (Pipe Network) axis, to move down/up before and after the coordinated motion pattern.

### 12.5.18.2 Motion

This template uses both motion engines (Pipe Network and PLCopen) simultaneously.

#### **! IMPORTANT**

Coordinated motion can only be performed with PLCopen axes, Pipe Network axes do not support this feature. As this template demonstrates, PLCopen axes that perform coordinated motion can be mixed with independent Pipe Network axes.

There is no axis synchronization at the Motion Engine level between a PLCopen axis and a Pipe Network axis. Any synchronization between the axes must be performed inside the PLC application.

To run the complete motion pattern:

1. Download and start the application.
2. Press the "Enable" button to enable the axes and the axes group.
3. Press the "Home" button to move the axes to their zero position. (optional)
4. After the "Ready" light is on, turn the "Cycle Start" switch to "1" and the axes will begin moving in the programmed pattern.

## 12.5.18.3 Control Panel

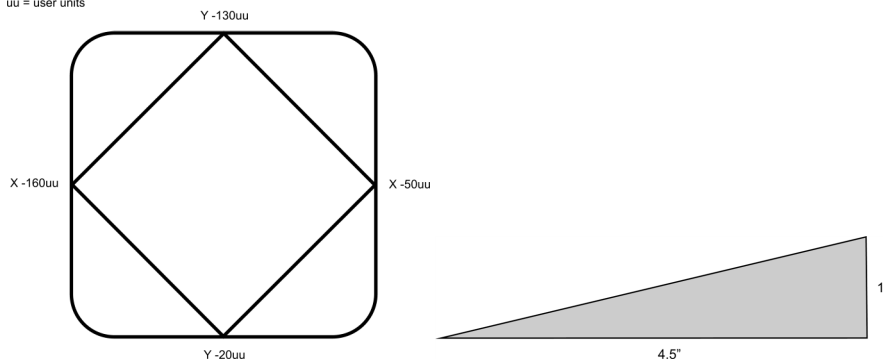
#	Description	GUI
1	Allows you to change the coordinated motion feedrate from 0% to 200%	
2	Displays the axis positions	
3	Indicator lights. Ready indicates that the three axes are enabled and the axes group is ready and enabled for coordinated motion. Error Stopped indicates that the axes group has been stopped and disabled by the E-Stop button.	
4	Begins the coordinated motion pattern	
5	Causes an Emergency Stop for all axes. The axes will be disabled.	
6	Resets the Emergency Stop and clears axis errors.	
7	Disables the three axes	
8	Enables the three axes and the axes group	
9	Moves all axes to their zero position	
10	Causes a reference by setting the axes' position to zero at their current location.	

### 12.5.19 Coordinated Motion 3-Axis (3D) Template

This project template controls three axes in coordinated motion (PLCOpenAxis1, PLCOpenAxis2 and PLCOpenAxis3) and demonstrates how to use 3D coordinated motion, transitions, blending and a homing cycle with PLCOpen axes. The path follows a square and diamond pattern on a plane which is rotated ~12.5 degrees about the Y axis with the center of rotation located at Z=30 X=-160.

The pattern and the platform for the plane are displayed below.

uu = user units



#### NOTE

This program was designed to run on the 3-Axis Demonstration machine shown in the video. It uses the machine's physical end-limit switches for the homing sequence. To run this program on different hardware, click on the **Sim Mode** button on the Control Panel. This will cause the program to execute in simulator mode, which will bypass the homing sequence. Also, the axis scaling parameters (entered in the PLCOpen Axis Data dialogs) may need to be changed to accommodate the different hardware.

#### 12.5.19.1 PLC Programs

The Coordinated Motion 3-Axis template has a Sequential Function Chart (SFC) program containing both Structured Text (ST) and Free Form Ladder Diagram (FFLD).

- The first five steps create and initialize the axes and the coordinated motion axes group.
- Step 6 specifies the coordinates of the square and diamond pattern. These coordinates are then rotated about the Y-axis.
- Step 7 monitors the Control Panel and performs two main functions. The first function is to reference the axes to establish a home position. The second function is to perform the 3-axis coordinated motion moves of the square diamond pattern.

This program provides examples of coordinated motion linear moves, transitions, blending, and a homing cycle.

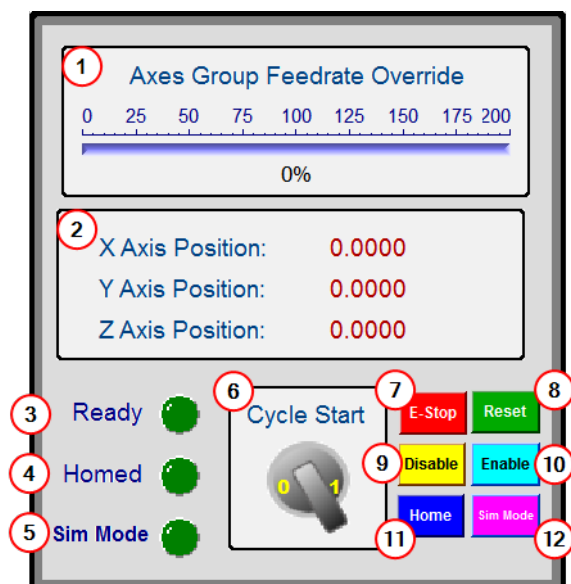
#### 12.5.19.2 Motion

To run the complete motion pattern:

1. Download and start the application.
2. Press the "Enable" button to enable the axes and the axes group.
3. After the "Ready" light is on:
  - If working with the 3-Axis Demonstration machine, press the "Home" button to reference the axes and move them to their home position.
  - If working with other hardware, press the "Sim Mode" button.
4. After the "Ready" and "Homed" lights are on, turn the "Cycle Start" switch to "1" and the axes will begin moving in the programmed pattern.

## 12.5.19.3 Control Panel


#	Description
1	Allows you to change the coordinated motion feedrate from 0% to 200%
2	Displays the axis positions
3	Ready Indicator light - "Ready" indicates that the three axes are enabled, and the axes group is ready and enabled for coordinated motion.
4	Homed Indicator light: "Homed" indicates that each axis has been referenced, and the axis group has moved to its zero position.
5	Simulation Mode indicator light: "Sim Mode" indicates that the homing functions have been bypassed and the program is set up as if the homing function has been performed.
6	Begins the coordinated motion pattern. "Ready" and "Homed" lights must both be on to execute motion.
7	Causes an Emergency Stop for all axes. The axes will be disabled. "Ready" and "Homed" lights will be turned off.
8	Resets the Emergency Stop and clears axis errors.
9	Disables the axes group. The axes have to be at standstill to disable the group
10	Enables the three axes and the axes group. This will turn on the "Ready" light.
11	Starts a homing function. Each axis will be referenced and the axis group will move to its zero position. Homing must be completed before executing a cycle start. When homing is complete, the "Homed" light will be turned on.
12	"Sim Mode" bypasses the homing function. The current position of each axis will be set to zero position. The homing function will not be performed. The "Sim Mode" light and "Homed" light will be turned on. This mode can be used when running on a simulator or when the hardware reference switches are not available.



## 13 PDMM and PCMM Errors and Alarms

When an [Error](#) or [Alarm](#) occurs, always check the controller log messages. The log messages will provide more details about the failure and the history of events leading up to the failure. From the log messages, you can determine the specifics about the cause of the failure to correct the underlying problem.

### 13.1 PCMM and PDMM Errors

Code	Description	Cause	Remedy	Clear <sup>‡</sup>
E01	Critical  temperature exceeded. The controller operation is stopped after 20 seconds, CPU will be put to sleep.	CPU temperature exceeded safe operating temperature limit.	Power-off. Check airflow and operating environment are within hardware specifications. Allow unit to cool before power-on.	HW
E02	Out of memory. KAS runtime is stopping.	Memory leak, memory corrupted, or hardware memory failure.	Power-off/on. If problem is recurrent, check release notes for firmware updates or return hardware for repair.	HW
E03	Fan failure.	CPU cooling fan was not able to operate properly.	Check temperature and monitor for High temp alarm (see A01). Return hardware for fan replacement.	HW
E10	Firmware is corrupted.	Flash memory corrupted during firmware download or flash hardware failure.	Re-download firmware or boot into recovery mode, download firmware, and power-off/on. If problem persists, return hardware for repair.	SW
E11	Flash is corrupted, no filesystem is available.	At startup the filesystem could not be mounted on the flash.	Reset to factory defaults. If problem persists, return hardware for repair.	SW
E12	Not enough flash memory available.	Flash memory is full, unable to write to flash.	Clean-up the flash memory by removing log files, application programs, recipes, or other data files.	SW
E13	Out of NVRAM space for <a href="#">retained variables</a> .	NVRAM is full.	Change application to reduce the amount of retained variables.	SW
E14	Reset to Factory Defaults failed.	Flash memory could not be formatted during a Reset to Factory Defaults procedure.	Try reset to factory defaults again from power-on. If problem persists, return hardware for repair.	SW



Code	Description	Cause	Remedy	Clear <sup>‡</sup>
E15	Cannot read/write files from/to a SD card	SD card is not plugged in or the file system is corrupt and cannot be mounted. PLC function failures will not cause this error.	Insert a valid SD card or reformat the SD card using Settings > SD Card > Format button.	SW
E16	Not enough space available on the SD card	SD card is full, unable to write to the SD card. PLC function failures.	Clean-up the SD card space by deleting files or re-format the card using Settings > SD Card > Format button.	SW
E17	Cannot connect to a shared directory	Improper shared directory configuration on remote computer or PCMM/PDMM. Improper permissions, access control, or security policy for computer's share configuration. Computer is not accessible via Ethernet network.	Check if the computer's shared directory is accessible. Check and Apply the KAS Application->Shared Directory configuration or power-off/on the controller.	SW
E20	Runtime plug-in, process, thread or application failed to start.	KAS runtime or application code failed to auto-start at boot. Caused by either a high controller temperature at power up, or a system error.	<ol style="list-style-type: none"> <li>1. Check the web server for other alarms or errors, such as temperature alerts, and resolve them.</li> <li>2. Power cycle the device</li> <li>3. If E20 is still being reported, reset to factory defaults.</li> <li>4. If the problem is recurrent, check release notes for firmware updates, or download firmware.</li> </ol>	HW
E21	Runtime process, thread, or driver failed to respond during operation.	KAS runtime code failed during normal operation.	Power-off/on. If problem is recurrent, check release notes for firmware updates.	HW
E22	Fatal error in PLC program, application stopped.	Virtual machine failed to execute an instruction.	Check the PLC application code for programming errors that could cause a fatal execution error, e.g. divide by zero, array out of bound, etc.. Check that the IDE and controller Runtime versions are compatible. Re-compile the application, download, and re-start.	SW
E23	CPU is overloaded. See <a href="#">CPU Overload (E23)</a> .	Either the motion engine did not complete or the PLC program did not complete within the timeout period due to excessive CPU load.	Stop the application or power-off/on. Reduce the sample rate, simplify the application, or reduce the application cycles and restart the application.	SW

Code	Description	Cause	Remedy	Clear <sup>‡</sup>
E24	PLC application cannot be started	<p>PLC application cannot be started, due to an existing condition. Possible reasons:</p> <ol style="list-style-type: none"> <li>1. Maintenance operation is in progress.</li> <li>2. Controller is in <a href="#">Online Configuration Mode</a>.</li> <li>3. AKD Restore failed.</li> <li>4. The IDE version of the compiled PLC code and controller runtime version do not match.</li> <li>5. Previous download failed.</li> </ol>	<p>Check the following:</p> <ol style="list-style-type: none"> <li>1. Controller web-server home page for any maintenance operation in-progress. Wait for the operation to finish.</li> <li>2. Connect to the controller with the IDE and disable <a href="#">Online Configuration Mode</a>.</li> <li>3. EtherCAT network topology by using the Scan network button in the web-server's Restore tab. Correct the physical topology and re-execute an AKD restore.</li> <li>4. IDE version (only major.minor.micro) should match with runtime version. To correct, install the correct version of IDE or Runtime.</li> <li>5. Connect IDE and download application.</li> </ol>	SW
E30	EtherCAT communication failure during operational mode.	Network communication error. This is commonly caused by a loose or bad EtherCAT connection occurring after the network successfully starts up. See IDE Controller log for more information	<ul style="list-style-type: none"> <li>• Read AKD parameter ECAT.DIAG for more information on where in the EtherCAT network the issue may be occurring. See <a href="#">Debugging Intermittent EtherCAT Communication Issues</a> on KDN for more information.</li> <li>• Check the EtherCAT network wiring and devices state. Re-start the application.</li> </ul>	SW
E31	EtherCAT communication failure during preop mode.	EtherCAT network operation failed due to a network communication error.	Check the EtherCAT network wiring and devices state. Re-start the application.	SW
E32	EtherCAT communication failure during bootstrap mode.	EtherCAT network operation failed due to a network communication error.	Check the EtherCAT network wiring and devices state. Re-start the application.	SW
E33	EtherCAT failed to initialize into operational mode.	EtherCAT network initialization failed. This is commonly caused by the Ethercat configuration in the project not matching the actual hardware. See IDE Controller log for more information.	Check the EtherCAT network wiring and devices state. Re-start the application.	SW
E34	EtherCAT failed to initialize into preop mode.	EtherCAT network initialization failed due to a network communication error.	Check the EtherCAT network wiring and devices state. Re-start the application.	SW

Code	Description	Cause	Remedy	Clear <sup>‡</sup>
E35	EtherCAT failed to initialize into bootstrap mode.	EtherCAT network initialization failed due to a network communication error.	Check the EtherCAT network wiring and devices state. Re-start the application.	SW
E36	EtherCAT failed to discover the expected devices.	EtherCAT network discovery failed due to a mismatch between the discovered and expected devices.	Check the EtherCAT devices and wiring order. Correct the device order wiring or re-scan the network, re-compile, and download the updated application. Re-start the application.	SW
E37	EtherCAT failed to return to init state.	EtherCAT network initialization failed due to a network communication error.	Check the EtherCAT network wiring and devices state. Re-start the application.	SW
E50	Backup to SD card failed	An unrecoverable error occurred during the backup operation.	Repeat the backup to SD card operation. If it fails again, replace the SD card.	SW
E51	Restore from SD card failed	An unrecoverable error occurred during the restore operation.	<b>Do not reboot the Controller!</b> Repeat the restore operation. If it fails again, reset the Controller to factory defaults. If the problem persists, return hardware for repair.	SW
E52	SD Backup files are missing or corrupt	The restore operation failed due to missing, incomplete, or corrupt files on the SD card.	Perform a backup operation before the restore or use and SD card with valid backup files.	SW
E53	SD Backup files are not compatible	Backup files on the SD card are not compatible.	Use a different SD card with Backup file(s) from a compatible controller.	SW

<sup>‡</sup> Items labeled "SW" can be cleared from the web server. Items labeled "HW" require a reboot to be cleared.

## 13.2 PCMM and PDMM Alarms

Code	Description	Cause	Remedy	Clear <sup>‡</sup>
A01	High temperature exceeded	CPU temperature near the safe operating temperature limit (120°C).	Check airflow and operating environment are within hardware specifications.	SW
A02	Low on memory.	Memory leak or corruption.	Power-off/on. If problem is recurrent, check release notes for firmware updates or return hardware for repair.	SW
A04	Low input voltage	+24 volt input power is +19 volts or less.	Check power supply voltage and connection to the controller.	SW
A12	Flash memory is low on free space.	Flash memory is almost full.	Clean-up the flash memory by removing log files, application programs, recipes, or other data files. Reset to factory defaults.	SW

Code	Description	Cause	Remedy	Clear‡
A21	Recoverable process or thread failed to respond during operation.	KAS non-runtime code failed during normal operation and was automatically restarted.	If problem is recurrent, power-off/on. Check release notes for firmware updates.	SW
A23	CPU is heavily loaded	CPU usage is too high for 5 (or more) seconds.	Reduce the sample rate, simplify the application, or reduce the application cycles.	SW
A30	EtherCAT missed a send frame during operation mode.	EtherCAT master was unable to send a frame for one or more cycles.	Reduce the controller CPU load, so it has enough Real-Time margin to send EtherCAT frames every cycle.	SW
A31	EtherCAT received a frame with a wrong Working Counter value.	<ul style="list-style-type: none"> <li>The slave EtherCAT device does not exist or cannot be physically reached due to connector/cable failure.</li> <li>Slave hardware failure</li> <li>Slave is still busy with a previous command and is not ready.</li> </ul>	<ul style="list-style-type: none"> <li>Check the EtherCAT cables and the connectors.</li> <li>Check the EtherCAT devices for any device errors.</li> </ul>	
A38	EtherCAT missed a receive frame during operation mode.	EtherCAT master did not receive, or received too late, a frame for one or more cycles.	<ul style="list-style-type: none"> <li>Read AKD parameter ECAT.DIAG for more information on where in the EtherCAT network the issue may be occurring. See <a href="#">Debugging Intermittent EtherCAT Communication Issues</a> on KDN for more information.</li> <li>Check the EtherCAT network wiring and devices, or decrease the EtherCAT cycle rate.</li> </ul>	SW
A40	Local digital IO missed a cyclic update	Local digital IO was not updated during a cycle or the updates are no longer synchronous.	Reduce the sample rate, simplify the application, or reduce the application cycles.	SW
A53	The Controller was replaced with a higher performance model.	The 1.2GHz model was restored using backup files from an 800MHz model.	Either replace the controller with an 800MHz model or use the functionally compatible, higher performance 1.2GHz model.	SW

‡ Items labeled "SW" can be cleared from the web server. Items labeled "HW" require a reboot to be cleared.

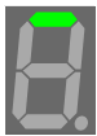
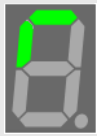
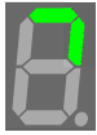
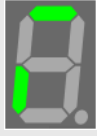
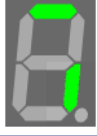
### 13.3 CPU Overload (E23)

If the Motion Engine or PLC program execution (VM) do not complete a full cycle within their respective timeout periods, an E23 error will be flashed on the 7-segment display.

Process	Timeout
Motion Engine	200 milliseconds
PLC Program (VM)	10 seconds

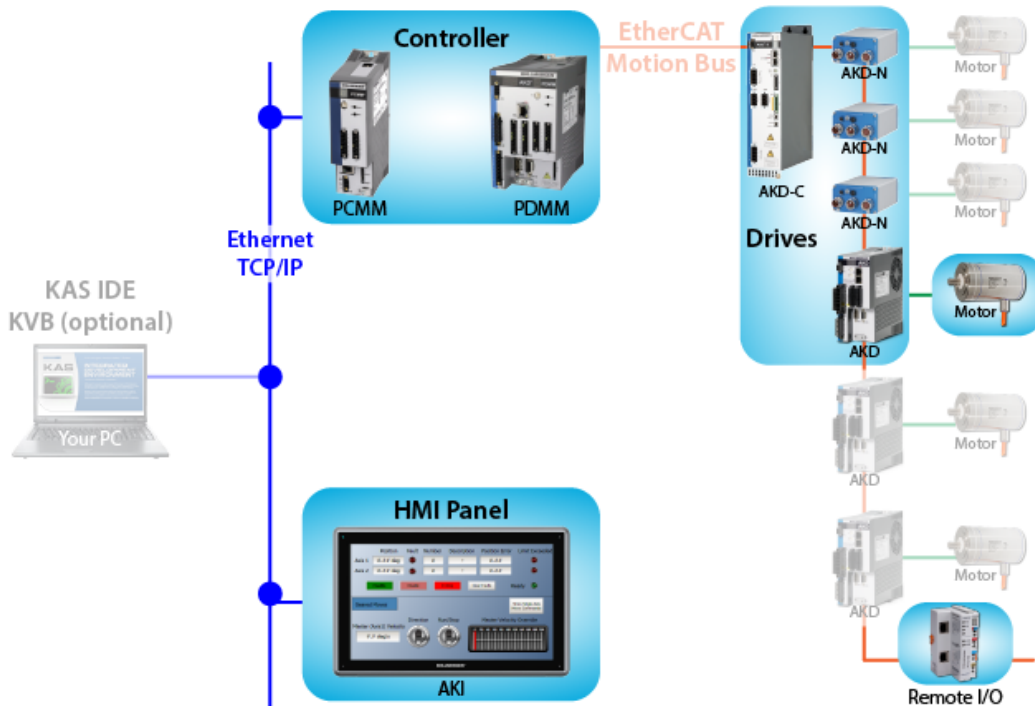
The Real-Time operation for EtherCAT and the Motion Engine have the highest priority in the controller. The PLC Program (VM) has the second highest priority in the controller. These processes will continue to execute, even if their timeout values are exceeded.

If the CPU overload is severe, there may not be enough CPU time to execute the background operations. The background operations include the 7-Segment display update, monitoring push-buttons, web-server, Modbus, and communications with the KAS IDE. The 7-Segment will indicate a CPU overload or frozen software task by displaying one of the following patterns:

Display	Meaning
	CPU overload is extreme.
	Motion thread is no longer able to execute.
	EtherCAT Rx thread is no longer able to execute.
	PLC thread is no longer able to execute.
	Non-realtime threads are no longer able to execute.

To recover from an E23, stop the application from the IDE or web-browser (KAS Application view). If the CPU overload is severe, the controller may not have enough CPU time to respond to the IDE or web-browser. In this case, you will need to power-off/on the controller. If the PCMM or AKD PDMM is configured for Auto-start, press and hold the B3 menu button at boot-time to prevent the application from automatically re-starting. Then, you will be able to connect to the Controller with the IDE.

## 14 KAS Component Manuals



### 14.1 HMI

HMI Part Number	Description	KVB	Manual
AKI2G-CDA-MOD-05T-000	Graphical Display 7" TFT LCD, Touchscreen	v2.20	
AKI2G-CDA-MOD-07T-000	Graphical Display 5" TFT LCD, Touchscreen	v2.20	
AKI2G-CDB-MOD-07T-000	Graphical Display 7" TFT LCD, Touchscreen	v2.20	
AKI2G-CDB-MOD-12T-000	Graphical Display 12" TFT LCD, Touchscreen	v2.20	
AKI-CDC-MOD-12T-000	Graphical Display 12.1" TFT LCD, Touchscreen	v2.0	
AKI-CDC-MOD-15T-000	Graphical Display 15.4" TFT LCD, Touchscreen	v2.0	
AKI-CDC-MOD-21T-000	Graphical Display 21.5" TFT LCD, Touchscreen	v2.0	

Table 8-1: List of KAS HMI

#### NOTE

Refer to our [Web site](#) for up-to-date information and material that are available in the **Automation Component Solutions** section. Note that you first need to log in before accessing KAS Literature.

Scan this QR code to access our web site with your mobile device.



### 14.1.1 HMI Accessories

Various accessories are available to compliment the HMI, including:

- Programming Cable, RS232 to HMI Terminal RS232
- Key cover for 5.7" Graphical Display
- Key cover for 10.4" Graphical Display
- Touch cover for 3.5" Touchscreen Graphical Display
- Touch cover for 5.7" Touchscreen Graphical Display
- Touch cover for 10.4" Touchscreen Graphical Display
- Touch cover for 15.1" Touchscreen Graphical Display
- 512 MB Compact Flash Industrial Grade
- 1 GB Compact Flash Industrial Grade

## 14.2 Controllers

Description	Model Number	Main Characteristics
PCMM, 800 MHz single core	<a href="#">AKC-PCM-MC-080-00N-00-000</a>	Standard Mutli-axis Controller
PCMM, 1.2 GHz single core	<a href="#">AKC-PCM-M1-120-00N-00-000</a>	High Performance Multi-Axis Controller
PCMM, 1.2 GHz dual core	<a href="#">AKC-PCM-M2-120-00N-00-000</a>	High Performance Multi-Axis Controller
AKD PDMM, 800 MHz	<a href="#">AKD-M0xxxx-MCEC-0000</a>	Standard Drive Resident Controller
AKD PDMM, 1.2 GHz	<a href="#">AKD-M0xxxx-M1EC-0000</a>	High Performance Drive Resident Controller

Table 8-2: List of KAS Controllers

### 14.2.1 PCMM Specifications

The PCMM programmable motion controller delivers a small yet powerful and cost-effective hardware platform ideally suited for modular or stand-alone machines that want the maximum in flexibility and performance.

<b>Processor</b>	800MHz (MC model), 1.2GHz (M1 model), or dual-core 1.2GHz (M2 model)
<b>Internal Memory</b>	64 MB Flash memory for program storage
<b>External Memory</b>	Removable SD Card slot (SD Card not included)
<b>Input Power</b>	24 Vdc @ 1.25 A
<b>Operating Temperature</b>	0 °C - 55 °C
<b>High Temperature Limit</b>	120°C
<b>Critical Temperature Limit</b>	127°C
<b>Sealing</b>	IP20
<b>Local I/O</b>	6 digital inputs, 2 digital outputs
<b>Motion Network</b>	EtherCAT, max 4kHz update rate
<b>Networking Protocols</b>	100BaseT connection supporting UDP, HTTP, Modbus TCP, Ethernet/IP, Profinet



<b>Dimensions</b>	174 mm (H) x 50 mm (W) x 111.5 mm (D)
<b>Certifications</b>	CE, UL

### 14.2.2 AKD PDMM Specifications

The AKD PDMM combines an AKD servo drive with a powerful, embedded motion controller in a single, compact package that can control up to 64 axes on your machine. It provides integrated control for multiple high-performance axes, complete I/O and HMI interfaces.

Model	Voltage	Continuous Current (Arms)	Peak Current (Arms)	Dimensions
AKD-M00306-xxEC-0000	120/240 VAC 1- and 3-Phase	3	9	168x89x156
AKD-M00606-xxEC-0000	120/240 VAC 1- and 3-Phase	6	18	168x89x156
AKD-M01206-xxEC-0000	120/240 VAC 1- and 3-Phase	12	30	192x107x187
AKD-M02406-xxEC-0000	120/240 VAC 1- and 3-Phase	24	48	248x96x228
AKD-M00307-xxEC-0000	240/400/480 VAC 3-Phase	3	9	256x99x185
AKD-M00607-xxEC-0000	240/400/480 VAC 3-Phase	6	18	256x99x185
AKD-M01207-xxEC-0000	240/400/480 VAC 3-Phase	12	30	256x99x185
AKD-M02407-xxEC-0000	240/400/480 VAC 3-Phase	24	48	306x99x228
AKD-M04807-xxEC-0000	240/400/480 VAC 3-Phase	48	96	385x185x225

<b>Processor</b>	800MHz (MCEC model) or 1.2GHz (M1EC model)
<b>Internal Memory</b>	64 MB Flash memory for program storage
<b>External Memory</b>	Removable SD card slot (not included)
<b>Input Power</b>	24 Vdc @ 1.25 A
<b>Operating Temperature</b>	0 °C - 55 °C
<b>High Temperature Limit</b>	120°C
<b>Critical Temperature Limit</b>	127°C
<b>Sealing</b>	IP20
<b>Local I/O</b>	6 digital inputs, 2 digital outputs
<b>Motion Network</b>	EtherCAT, max 4kHz update rate
<b>Networking Protocols</b>	100BaseT connection supporting UDP, HTTP, Modbus TCP, Ethernet/IP, Profinet
<b>Dimensions</b>	174 mm (H) x 50 mm (W) x 111.5 mm (D)
<b>Certifications</b>	CE, UL

### 14.2.3 NVRAM

KAS uses the NVRAM (non-volatile memory) to save [Retain Variables](#).

Hardware Type	NVRAM Size Allocation
Simulator	128 Kbytes
AKD PDMM or PCMM	32 Kbytes

Table 8-3: NVRAM Size Depending on Hardware

**! IMPORTANT**

Part of the NVRAM allocation is reserved to store some internal data (144 bytes). As a consequence, not all the complete physical NVRAM is available for the retain variables.

If the size is big enough, KAS updates the non-volatile memory to store the retain variables values. This operation is performed in the background every 20 seconds. The frequency increases to every 2 seconds when the application is running, and when you shutdown the application.

*Life expectancy of the NVRAM*

- **Data Retention:** At maximum operating temperature, the data written to the memory will be retained for 10 years.
- **Endurance:** The NVRAM memory has an endurance limit of 10<sup>14</sup> read/write accesses, (or more than 6,000,000 years of being accessed every 2 seconds).

**NOTE**

Using the retain variables is highly cycle time consuming. As a consequence, Kollmorgen strongly recommends to carefully monitor the system load with the [TraceTimes](#) command.

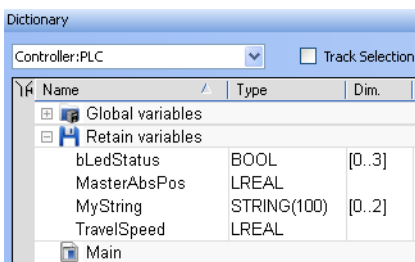
**14.2.3.1 How can I check the NVRAM space is enough to store my retain variables?**

To calculate the NVRAM space, you have to add the size of each retained variable according to:

- its data type as described [here](#)
- the numbers of elements in case you declare the variable as an [array](#)

Do not forget to add the 144 bytes as stated in the Warning above.




In the following example, the total size is: **3684 bits** (which is less than 0.5 Kbytes)



Variable	Size / element	Element no.	Total Size / variable
bLedStatus	1 bit	4	4
MasterAbsPos	64 bits	1	64
MyString	800 bits (100 bytes)	3	2400
TravelSpeed	64 bits	1	64
<i>Internal data</i>	<i>1152 bits (144 bytes)</i>	<i>na</i>	<i>1152</i>

### 14.3 BBH FSoE Master

The following BBH Safety equipment is required to program and operate the SCU-1-EC FSoE Safety master with the AKD2G, [AKT2G-SDI-004-000](#), [AKT2G-SDO-004-000](#), PCMM/AKD PDMM, and/or KAS IDE software.

Part Number	Description	Image
SCU-1-EC	FSoE Master - modular basic module	
SMX91	Programming adapter (cable)	
SafePLC2	Programming software, License including Hardlock	

These items are available for order from Kollmorgen.

## 14.4 Kollmorgen Safety Solutions

### KSM-compact Safety Module

- KSM 11 • For 1 or 2 safe axes
- KSM12 • Up to 2 expansion modules
- KSM 31 • Base unit with 16 safe I/O
- KSM 21R • Expandable up to 60 safe I/O
- KSM 5x • 1 safe relay output, expandable
- 2 pulse outputs, 2 signal outputs
- Expandable up to 6 pulse and 6 signal outputs
- Up to 800 function blocks
- Space-saving, compact design

### KSM-modular Safety PLC

- KSM 100-1 • Up to 12 safe axes
- KSM 100-2 • Up to 8 expansion modules
- KSM 100-4 • Base unit with 56 safe I/O
- KSM 121/-2 • Expandable up to 200 safe I/O
- KSM 122/-2 • 1 safe relay output, expandable
- KSM 122A • 2 pulse outputs, up to 10 signal outputs
- KSM 131 • Expandable up to 14 pulse and 22 signal outputs
- KSM 131R • Up to 3000 function blocks
- For applications with large number of interfaces

### Available Safety Functions (PLe & SIL3)

- Safe Stop Function: STO, SS1, SS2, SOS
- Safe Velocity Function: SLS, SSM, SSR, SMS
- Safe Position Function: SLP, SCA, SLI
- Safe Direction Function: SDI
- Safe Brake Function: SBC

Learn more about the safety functions on the [Kollmorgen Developer Network](#).

### Connectivity

- EtherCAT
- CANopen
- Profinet
- PROFIsafe and EtherCAT FSoE slave

## 14.5 Remote Input/Output Terminals

KAS remote I/Os provide a complete spectrum of bus couplers, digital and analog inputs, digital and analog outputs, stepper, counter, and thermocouple modules.







### Related Documents











Please find in the table below the list of each I/O component available.

#### AKT2G (EtherCAT) Terminals

I/O Terminal Part Number	I/O Terminal Description
<a href="#">AKT2G-AC-FAN-001</a>	Fan cartridge for EtherCAT and Bus Terminals
<a href="#">AKT2G-AN-240-000</a>	2-channel input terminal PT100 (RTD) for resistance sensors, 16 bit, 2-, 3-wire system
<a href="#">AKT2G-AN-400-000</a>	4-channel thermocouple input terminal, preset to type K, with wire breakage detection, 16 bit
<a href="#">AKT2G-AN-430-000</a>	4-channel analog input, parameterisable, -10/0...+10 V, -20/0/+4...+20 mA, 16 bit
<a href="#">AKT2G-AT-410-000</a>	4-channel analog output terminal 0...10 V, 12 bit, 1-wire system
<a href="#">AKT2G-AT-425-000</a>	4-channel analog output terminal -10 V...+10 V, 12 bit, 4 x 2-wire system
<a href="#">AKT2G-BRC-000-000</a>	Brake Chopper Terminal
<a href="#">AKT2G-DN-002-000</a>	Up/down counter 24 V DC, 100 kHz, 32 bit counter depth
<a href="#">AKT2G-DN-008-000</a>	8-channel digital input terminal 24 V DC, filter 3.0 ms, 1-wire system
<a href="#">AKT2G-DNH-008-000</a>	8-channel digital input terminal 24 V DC, filter 10 µs, 1-wire system
<a href="#">AKT2G-DT-008-000</a>	8-channel digital output terminal 24 V DC, 0.5 A, 1-wire system
<a href="#">AKT2G-ECT-000-000</a>	EtherCAT Coupler for E-bus terminals
<a href="#">AKT2G-EM-000-000</a>	Bus end cover for E-bus terminals, cover for power and E-bus contacts, grey
<a href="#">AKT2G-ENC-180-000</a>	1-channel incremental encoder interface, 32 bit
<a href="#">AKT2G-ENC-190-000</a>	Incremental encoder interface with differential input, 16/32 bit
<a href="#">AKT2G-PSF-024-000</a>	Power supply terminal with fuse, 24 V DC
<a href="#">AKT2G-SDI-004-000</a>	4-channel digital input terminal, Safety, 24 V DC
<a href="#">AKT2G-SDO-004-000</a>	4-channel digital output terminal, Safety, 24 V DC, 0.5 A
<a href="#">AKT2G-SM-L15-000</a>	Stepper motor terminal, 24 V DC, 1.5 A, vector control
<a href="#">AKT2G-SM-L50-000</a>	Stepper motor terminal, 50 V DC, 5 A, vector control

#### AKT (K-Bus) Terminals

I/O Terminal Part Number	Description	Link	Replaced By
<a href="#">AKT-AN-200-000</a>	2 Channel Thermocouple Input Module		<a href="#">AKT2G-AN-240-000</a>
<a href="#">AKT-AN-400-000</a>	4 Channel Thermocouple Input Module		<a href="#">AKT2G-AN-400-000</a>
<a href="#">AKT-AN-410-000</a>	4 Channel Analog Input Module, 0-10 VDC		<a href="#">AKT2G-AN-430-000</a>
<a href="#">AKT-AN-420-000</a>	4 Channel Analog Input Module, 0-20 mA		<a href="#">AKT2G-AN-430-000</a>
<a href="#">AKT-AN-810-000</a>	8 Channel Analog Input Module, 0-10 VDC		<a href="#">AKT2G-AN-430-000</a>
<a href="#">AKT-AN-820-000</a>	8 Channel Analog Input Module, 0-20 mA		<a href="#">AKT2G-AN-430-000</a>

I/O Terminal Part Number	Description	Link	Replaced By
AKT-AT-220-000	2 Channel Analog Output Module, 0-20 mA		—
AKT-AT-410-000	4 Channel Analog Output Module, 0-10 VDC		—
AKT-AT-420-000	4 Channel Analog Output Module, 0-20 mA		—
AKT-AT-810-000	8 Channel Analog Output Module, 0-10 VDC		<a href="#">AKT2G-AT-410-000</a>
AKT-AT-820-000	8 Channel Analog Output Module, 0-20 mA		—
AKT-DN-004-000	4 Channel Digital Input Module, 24 VDC 3ms		<a href="#">AKT2G-DN-008-000</a>
AKT-DN-008-000	8 Channel Digital Input Module, 24 VDC 3ms		<a href="#">AKT2G-DN-008-000</a>
AKT-DNH-004-000	4 Channel Digital Input Module, 24 VDC 0.2ms		<a href="#">AKT2G-DNH-008-000</a>
AKT-DNH-008-000	8 Channel Digital Input Module, 24 VDC 0.2ms		<a href="#">AKT2G-DNH-008-000</a>
AKT-DT-004-000	4 Channel Digital Output Module, 24 VDC 0.5A		<a href="#">AKT2G-DT-008-000</a>
AKT-DT-008-000	8 Channel Digital Output Module, 24 VDC 0.5A		<a href="#">AKT2G-DT-008-000</a>
AKT-DT-2RT-000	2 Channel Relay Output Module, 230 V AC 2.0A Rel.2NO Pot.-Free		—
AKT-ECT-000-000	EtherCAT Bus Coupler		<a href="#">AKT2G-ECT-000-000</a>
AKT-EM-000-000	Standard-Bus End Terminal		<a href="#">AKT2G-EM-000-000</a>
AKT-IM-000-000	Isolation / Separation Terminal		—
AKT-PS-024-000	Power Supply, 24 VDC		<a href="#">AKT2G-PSF-024-000</a>
AKT-PSF-024-000	Fused Power Supply with diagnostics, 24 VDC		<a href="#">AKT2G-PSF-024-000</a>
AKT-SM-L15-000	Stepper Motor Terminal, 24 VDC, 1.5 A		<a href="#">AKT2G-SM-L15-000</a>
AKT-SM-L50-000	Stepper Motor Terminal, 50 VDC, 5 A		<a href="#">AKT2G-SM-L50-000</a>

**ⓘ IMPORTANT**

24-volt power is not passed through AKT-AN-200-000 and AKT-AN-400-000 thermocouple modules. To get 24VDC power to devices that need it (such as an AKT\_AT-410-000 Output module) there are two possible solutions.

- Place the module requiring 24VDC before the thermocouple module.
- Add a power feed module (AKT-PS-024-000 or AKT-PSF-024-000) after the thermocouple module.

Table 8-4: List of KAS I/O Terminals

## 14.6 Drives

This section details the following drives:

AKD part number	Description
AKD-B00106	120/240 VAC 1.5A Drive
AKD-B00306	120/240 VAC 3A Drive
AKD-B00606	120/240 VAC 6A Drive
AKD-B01206	120/240 VAC 12A Drive
AKD-B02406	120/240 VAC 24A Drive
AKD-B04806	120/240 VAC 48A Drive
AKD-B00107	240/480 VAC 1.5A Drive
AKD-B00307	240/480 VAC 3A Drive
AKD-B00607	240/480 VAC 6A Drive
AKD-B01207	240/480 VAC 12A Drive
AKD-B02407	240/480 VAC 24A Drive
AKD-N00307	3A Distributed Servo Drive
AKD-N00607	6A Distributed Servo Drive

Table 8-5: List of AKD Drives

### Related Documents

For further information on drives, refer to the following manuals:

Drives Guide	Description
<a href="#">AKD PDMM User Manual</a>	Describes software installation, setup, and operation for the AKD PDMM drive. Includes basic topics and examples to help guide you in setting up and using the various features in the drive.
<a href="#">AKD PDMM Fault Card</a>	Describes AKD PDMM (including AKD) faults, warnings, error messages, and alarms. Provides cause and remedy instructions to help determine the specifics of the failure and to correct the underlying problem.
<a href="#">AKD Installation Manual</a>	Installation manual for AKD and AKD PDMM drives. Describes the AKD series of digital drives and includes mechanical, electrical, and software installation information needed to safely install AKD.
<a href="#">AKD2G Installation Manual</a>	Describes the AKD2G series of digital drives and includes mechanical, electrical, software and functional safety options.
<a href="#">AKD, AKD2G, S700 (in NA) Accessories Manual</a>	This manual describes the accessories for Kollmorgen digital drive systems and servo drive motors.
<a href="#">AKD EtherCAT Manual</a>	Describes the installation, setup, range of functions, and software protocol for the EtherCAT AKD product series,
<a href="#">AKD2G EtherCAT Manual</a>	AKD2G EtherCAT and CANopen Communications Manual describes the installation, setup, range of functions, and software protocol for the AKD2G product series.
<a href="#">S300 Reference Documentation</a>	Kollmorgen website that gives access to all S300 manuals.
<a href="#">S700 Reference Documentation</a>	Kollmorgen website that gives access to all S700 manuals.

Table 8-6: List of Drive Manuals

**NOTE**

The AKD manuals are located under:  
 C:\Program Files\Kollmorgen\AKD WorkBench 1.0.x.y\WebHelp  
 (x.y must be replaced with the version number)  
 (this location differs if you chose another location when installing AKD).

**14.6.1 About the AKD "NC" Drive Model**

The AKD "NC" drive is a drive variant introduced in 2021 with a substitute chip. It is identified with "NC" the part number:

**AKD—Pxxxxx—NCxx—xxxx**

For more details about the drive, please contact your local Kollmorgen representative.

**14.6.1.1 FAQ's regarding the "NC" drive**

*Which AKD firmware versions support the AKD "NC" drive?*

AKD firmware versions v01-20-00-003 and higher support the "NC" drive.

*Which versions of the KAS IDE are compatible with the AKD "NC" drive?*

KAS IDE versions 2.09 and higher are compatible. Kollmorgen recommends using v3.05 or higher because the drive features available in AKD firmware v01-20 are integrated with and supported by KAS IDE v3.05.

*Are there any backwards compatibility issues with the AKD "NC" drive?*

EtherCAT Devices Backup and Restore requires AKD firmware version v01-20-00-003 at a minimum. The firmware for the "NC" drive will not be available in your backup files if your existing backup contains older firmware. To update your backup files, please see [EtherCAT Devices Backup](#) for more details.

*Where can I find the KAS firmware package with "NC" support?*

The AKD firmware and KAS firmware packages are available in the downloads section of Kollmorgen website.

- [AKD backup/restore firmware package](#)
- [AKD firmware](#)

*Which AKD drive models can be swapped with the "NC" model?*

The "NB" models are functionally compatible and can be swapped, using the [EtherCAT Devices Backup and Restore](#) feature.

*What is the recommended usage for "NC" drives?*

Kollmorgen recommends using "NC" drives for new machines. We do not recommend using "NC" drives as replacement drives with existing machines, as it may require a drive firmware upgrade and an update to your existing EtherCAT devices backup files.



## 15 Troubleshooting

In addition to the topics contained in this section, the following topics may prove useful in troubleshooting.

- [Error Management](#) — this topic provides a programmatic procedure for when a non-fatal error occurs in the machine and motion needs to be halted quickly.
- [Practical Application: Using Trace Time To Measure CPU Load](#) — this topic provides a procedure for calculating CPU usage.

### 15.1 FAQs

#### 15.1.1 Basic

*Why does the Installer not Start when I insert the CD?*

Your Autorun feature may be deactivated. Open an Explorer window to see the autorun.exe file and use the Run command in the contextual menu to manually start the installer.

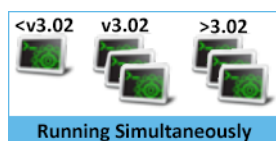
*Can I install more than one version of the KAS IDE?*

That depends on the version of KAS. You may have one version of KAS IDE v3.00 and earlier installed along with any number of later versions.

KAS version	Only 1 version	Multiple Installed Verions
< v3.00	✓	✗
v3.01 and later	✗	✓

*Can I run more than one version of the KAS IDE simultaneously?*

Yes, as long as one of the versions is v3.02 or later. With the release of v3.02 multiple instances of the KAS IDE are permitted. That includes running one older (<v3.02) instance and any number of newer instances. This includes multiple instances of the same version. The only limitation is available RAM.



#### **TIP**

The KAS IDE supports simultaneously running multiple instances. Double-clicking a shortcut or project file, or selecting the application from the Start menu will open another instance, Double-clicking on a project file more than once will open a copy of the project in another instance of the KAS IDE.

*How do I launch my project with a particular version of the KAS IDE?*

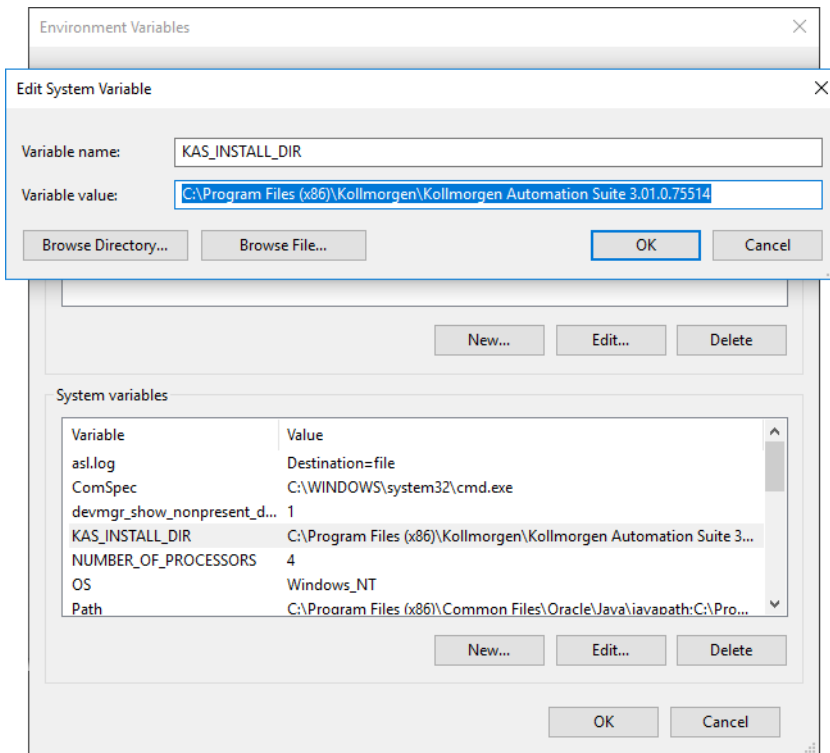
Right-clicking on a project file allows you to choose which of your installed versions of the KAS IDE to use for that instance.



How do I choose the default version of the KAS IDE to use?

The default version of the KAS IDE is that which is most recently installed. This can be changed to a preferred version by changing an Environment Variable.

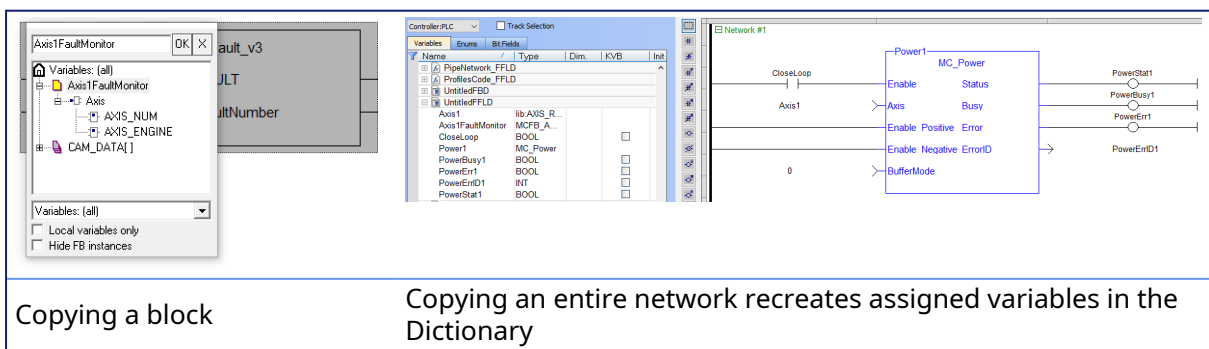
From the list of System variables, edit KAS\_INSTALL\_DIR. Enter the path to your preferred version.



Can I copy and paste or drag and drop between instances of the KAS IDE

Yes, the KAS IDE supports copying and pasting between simultaneously running instances. This is supported in both the same version and different version numbers. This includes drag and drop for supported objects, such as FBD blocks.

The KAS IDE will attempt to recreate what you have copied, but there are some limitations. For example, it may not automatically associate variables with a function. For example, if you copy and paste only a function block from an FFLD program, the assigned variables will not be copied and pasted, but if you select the entire network the KAS IDE will automatically create the variables for you.



Copying a block

Copying an entire network recreates assigned variables in the Dictionary

**NOTE**

Closing the KAS IDE automatically clears the Clipboard. Anything that was copied is cleared from memory. This means you can not copy something, close the KAS IDE, and then paste.

*Why does the KAS IDE not display all the items in the Project Explorer when I create a new project based on a template?*

A side effect with some remaining files that were not deleted properly can interfere with your new project. To fix this issue:

- Close your current project without saving
- Open Windows Explorer and go to **C:\Documents and Settings\user\Local Settings\Application Data\Kollmorgen\KAS\Project** , where "(user)" is the Windows' username you are currently logged in with
- Delete all the remaining files and folders
- You can now create your new project

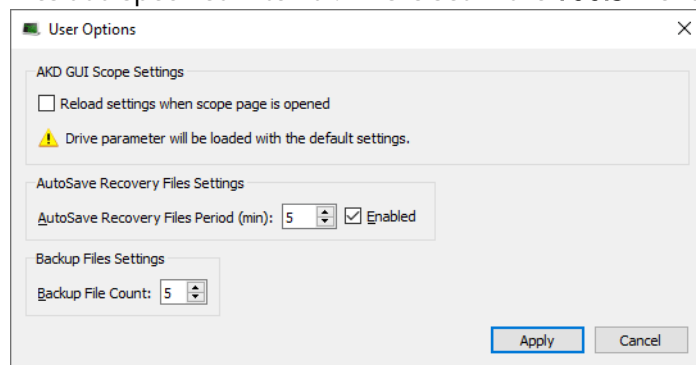
*Where can I get the latest User Manuals?*

Everything in our manuals may be found in the help. If you need a printable document or want stand-alone manuals, the documentation is available as [PDFs on KDN](https://www.kollmorgen.com/en-us/developer-network/kas-downloads/).  
<https://www.kollmorgen.com/en-us/developer-network/kas-downloads/>

[Learning Kollmorgen Automation Suite](#)

*Can I automatically save my files?*

The KAS IDE does have an AutoSave feature which automatically backs up your crash recovery files at a specified interval. This is set in the **Tools** menu.



*How does file recovery work?*

With the AutoSave function, project files are saved for recovery. More than one file may exist due to multiple versions of the KAS IDE being installed or used. When you restart the KAS IDE you will be prompted to recover a found file. When there is more than one file the recovery process will start with the latest date from the newest KAS IDE version. Assuming you choose to recover the file it is opened and you may resume work, but the older recovery files still exist.

The next time you launch the KAS IDE you will be prompted to recover the next oldest file. Choosing to not recover the file will delete it, and you will be prompted to open the next oldest file until none are left.

### 15.1.2 Setup

*How can I download new **Firmware** to my AKD Drive?*

*How are fieldbuses connected to the KAS Runtime?*

As depicted in figures found in [Different Implementations](#), the fieldbus serially links all the drives to the industrial PC.

**NOTE**

When the KAS IDE is used to deploy an automation system on a master drive (also known as programmable drive), the fieldbus serially links all the drives to the master drive.

*How does Kollmorgen Automation Suite communicate with a Host?*

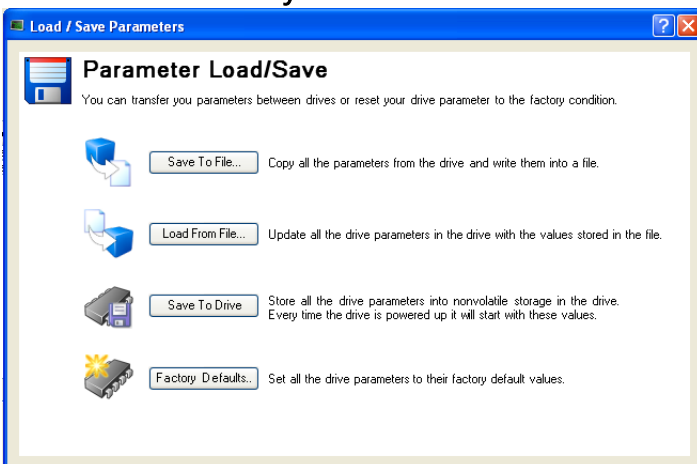
As described in [Communication and Fieldbus](#), KAS can communicate with outside world through Ethernet, CANopen, DeviceNet.

*Why does Online Configuration Mode not work after I reload the drive's factory default parameters?*

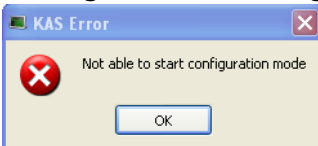
**Description**

This issue occurs when you perform the following

- Connect to the controller and download your application
- In the project explorer, open the EtherCAT properties
- Click the [Online Configuration Mode](#)
- In the project explorer, right-click on the AKD\_1 and select **Load/Save Parameter...**
- Then select the **Factory Defaults...** command to reset the drive to its default parameters



- Clicking the Online Configuration Mode leads to the following error

**Reason**

If you set the drive to its default parameters, then all the AKD parameters are restored and the unique ID (FBUS.PARAM03) used to identify the drive is lost.

**Solution**

You have to perform a new scan operation after setting the parameters to its default values

**TIP**

You can also clear the **Write a unique ID** option in the XML configuration tab (for more details, [ENI File Tab](#))

*How can I fix security issues?*

If you encounter any security issues during execution of Kollmorgen Automation Suite, refer to your IT department to set your proxy properly.

**Firewall**

You may have to define your firewall settings to allow accessing the IP addresses used by KAS (for instance, IP address of the target system, or localhost IP address for the KAS Runtime Simulator: 127.0.0.1).

### Port numbers

Port numbers have to be set properly in your firewall settings to avoid any trouble during communication, such as when [downloading](#) the application to the target, or plugging a probe to the softscope. Kollmorgen strongly recommends opening port numbers over 1024, as well as the range 502 to 520.

### 15.1.3 Computer / Controller

*How many axes can the KAS IDE manage in 1 ms?*

This number is mainly dependent upon the application and your PC's computing power. An average number would be 20 axes/ms

*How can I see the CPU load between the PLC and motion parts?*

This procedure allows you to determine if your controller is overloaded due to the PLC program or motion system load.

You can use the Softscope and the **Trace Times** button to display the following CPU loads:

- Cyclejitter (microseconds)
- Motion execution time (microseconds)
- PLC execution time (microseconds)
- Real Time Margin (microseconds)

To view the load, do as follows:

- Open the Softscope
- Plug four probes to any kind of data (see procedure [here](#))
- In the Control Panel, click the **TraceTimes** button

*I cannot log into the controller's webpage. What do I do?*

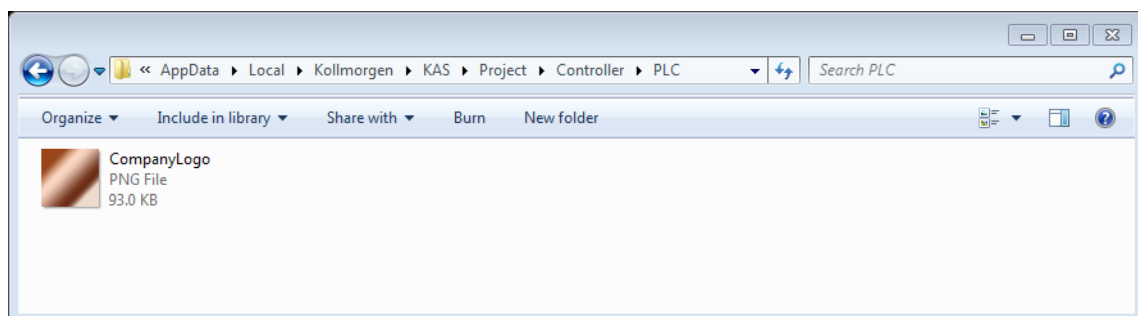
If you are unable to log into the controller's webpage using a valid user ID and password, the controller's file system may be corrupt. To recover the system you will need to use the push buttons on the controller (see [Booting from the Recovery Image](#)) to recover the system.

*How can I check the if there is enough NVRAM space to store my Retain Variables?*

For explanation, [How can I check the NVRAM space is enough to store my retain variables?](#)

*I have custom graphics in my project's control panel. Why don't they show up when the file is opened on other computers?*

Graphics need to be made portable with a project, otherwise the KAS IDE will attempt to point to the original location of the graphic files. To make graphics portable, copy them to `c:\Users\[user folder]\AppData\Local\Kollmorgen\Project\Controller\PLC`. After this the graphics are a part of the .KAS file.

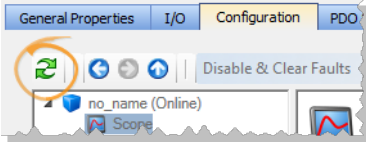


### 15.1.4 Performance

I am having trouble with the WorkBench navigation tree. What can I do?

Due to excessive ASCII SDO commands or very slow Ethernet communication, the WorkBench tree may not fully populate or navigation may be unresponsive. To recover from this situation:

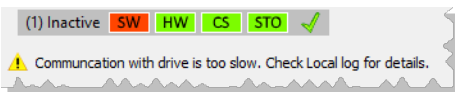
1. Press the **Refresh Tree** button in KAS IDE WorkBench view.



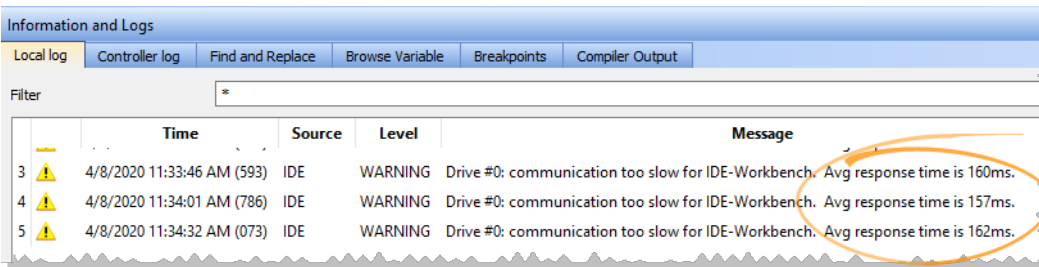
2. Disconnect and Reconnect the KAS IDE to restart KAS IDE-WorkBench communication.

What can I do if the KAS IDE's embedded WorkBench (drive Configuration tab) is slow or unresponsive?

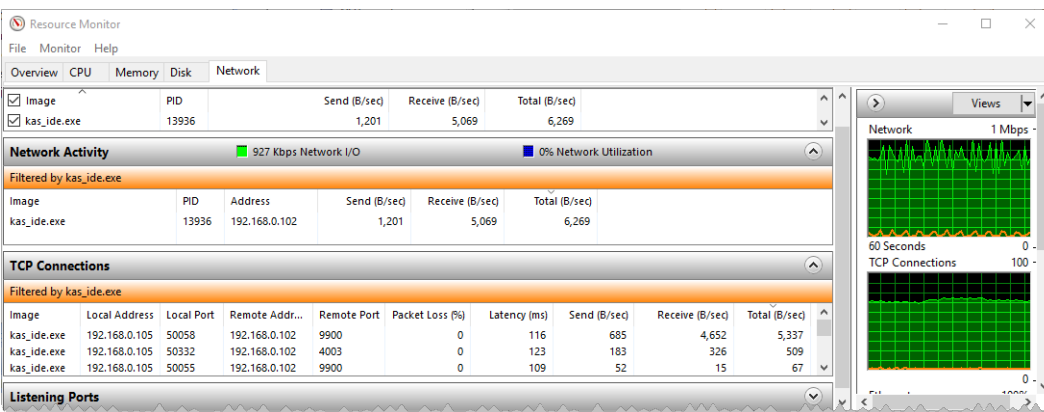
The status bar may display a warning if communication between the KAS IDE, controller, and drive is too slow.



To begin troubleshooting, configure the Local log for **WARNING** level (or lower) and check for "communication too slow" messages. The IDE-WorkBench navigation and views may become very slow or completely unresponsive if the average response time for drive communication is too long.



Communication delays between the KAS IDE and Controller can be caused by degraded Ethernet bandwidth or high latencies. You can use the Windows Resource Monitor (resmon.exe) and select **kas\_ide.exe** to monitor and diagnose high latency and/or poor bandwidth.



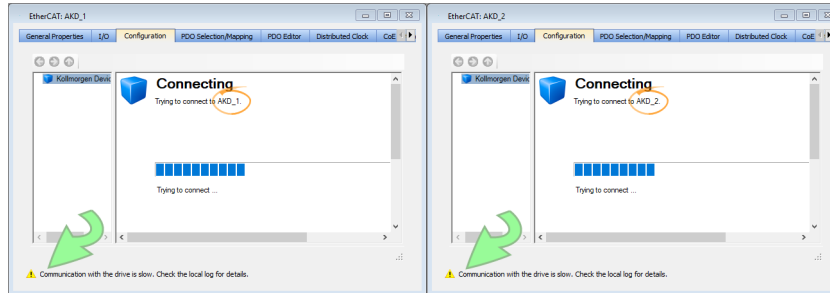
Communication delays between the Controller and Drive can occur while an application is running on the controller. Degraded drive communication can be caused by excessive ASCII SDO commands from the application code.

- Check your application code for calls to DriveParamRead/Write(...), DriveParamStrRead(...), or ECATRead/WriteSdo(...), and minimize the frequency of these calls.

- Use [Online Configuration Mode](#) with KAS IDE WorkBench to avoid ASCII SDO channel competition between your application and KAS IDE WorkBench

*I am having trouble connecting to drives with multiple WorkBench views/windows open. What can I do?*

There are times when trying to connect to multiple drives simultaneously with WorkBench is problematic, particularly on slower networks.



#### To resolve this situation:

- Close all of the open windows in the IDE workspace, leaving only one WorkBench window open.
- Connect the KAS IDE and run the project or use Online Configuration mode.

These actions reduce the network and KAS IDE loads. This allows the embedded WorkBench to establish communication and access the views.

#### To access multiple drives at the same time:

- Open one WorkBench window
- Wait for communication and initialization to complete successfully
- Open each additional WorkBench window one at a time, allowing enough time for each to successfully complete initialization.

### 15.1.5 EtherCAT

*Is EtherCAT limited to Master/Slave applications?*

No. As with every real-time Industrial Ethernet system, one device (the master) has to be in charge of the network management and organize the Medium Access Control. With EtherCAT, Slave-to-Slave communication is supported in two ways:

- topology dependent within one communication cycle ("upstream" device talks to "downstream" device)
- topology independent within two cycles.

Since EtherCAT is so much faster than competing systems, slave-to-slave communication using two cycles is faster, too.

*What is the maximum number of EtherCAT nodes per network?*

The maximum number of node addresses is 65,535. Typically, the number of the nodes on the network will be limited by the maximum EtherCAT frame size, which is 1500 bytes. Please see the Frame Size in the [EtherCAT Master Settings Tab](#) for more details.

*When can I expect my SDO command to be completed?*

If you need to rely on SDO communication to set the parameter of an EtherCAT device, you can do this with the ECATWriteSdo FB .

Being asynchronous and based on the EtherCAT mailbox, the SDO communication is not deterministic. So the EtherCAT master uses a polling mode to ensure the SDO command is

completed. Note that in operational mode, this polling is performed every 50 cycles <sup>1</sup>. As a consequence, you can expect the acknowledgement of your SDO command usually before less than 100 ms. So, a good practice is to set the update rate for SDO communication in your PLC application each 25 cycles.

[EtherCAT Motion Bus Concepts](#) for more details.

*When I compile I get a warning about the total number of PDO entries. How do I fix this?*

The AKD2G has a limit on the number of PDO objects which can be mapped, depending upon the sample rate.

Sample Rate	Cycle Time	Max. PDOs
4kHz	250µs	14 Rx, 14 Tx
2kHz	>500µs	30 RX, 30 Tx

This limitation can be hit during compilation with the following error:

```
EtherCAT: ERROR: In AKD2G_1, the total number of entries for all selected Tx PDOs is greater than the max allowed at a cycle time of 250 µs: 17 > 14
```

There are three possible solutions.

- Increase the Cycle Time – see [EtherCAT Master Settings Tab](#)
- Deselect unused PDOs – see [PDO Selection/Mapping Tab](#)
- Delete unused objects – see [Delete a CoE Object](#)

*Why is the PLC execution rate not the same as the EtherCAT rate with the KAS Simulator?*

When the application runs on the KAS Simulator, the PLC execution rate is approx. 10 milliseconds. KAS Simulator cannot execute the PLC programs faster because Windows is not able to handle timing less than 10ms.

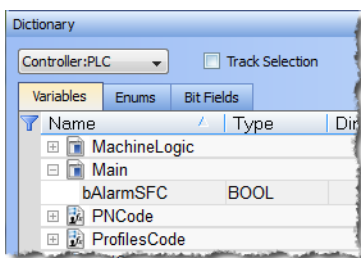
### 15.1.6 Programming

*How can I control the time execution for an SFC step?*

When you want to check the maximum time execution for an SFC step, you have to program this action based on the SFC alarm capability.

To show this status, you have to:

- In the Dictionary, declare a Boolean PLC variable linked to the related SFC program

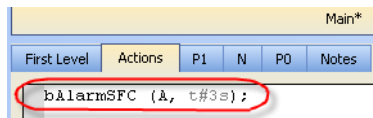


- Add the instruction in the Actions tab related to the SFC step, with first parameter set to **A** (for Alarm) as shown below

---

<sup>1</sup>To avoid overloading the controller, this rate is set according to the communication load, as well as the duration the AKD takes to process commands





**Figure 9-1:** SFC Step - Timeout Alarm

*How can I fix the Library Access issue?*

If you open a project containing a link on a library which is no longer available, a warning is displayed. To fix this issue, refer to [What happens when a library no longer exists?](#)

*What is the Fast Input?*

The Fast Input allows an application to get information about the occurrence of an external event at a higher resolution than the cycle time.

For more details, refer to [Fast Inputs with Pipe Network Motion](#)

*How do I implement feedback?*

There are two kinds of feedback:

#### 15.1.6.0.0.1 Primary feedback

With a S300 drive you can use a resolver for primary feedback.

#### 15.1.6.0.0.2 Secondary feedback

If a secondary feedback is required with your S300 drive, you can use a BiSS feedback device.

If you use the same setup with an S300 drive, the S300's EXTPOS parameter has to be set to **-11**. **IMPORTANT:** do not omit the negative sign!

To access secondary feedback, use a SAMPLER Pipe Network block. To configure the block use the [MLSmpConPNAxis](#), [MLSmpConPLCAxis](#), or [MLSmpConECAT](#) functions.

*How do I implement Torque Feed-forward?*

Current drives that support torque feed-forward are: [S300](#) and [AKD](#) drives.

To use torque feed-forward, you have to rely on a CONVERTER Pipe Network block. To configure the block use the [MLCNVConnectEx](#) function. The arguments must be:

- The Pipe Network block ID being configured
- The ID of the axis to which the torque feed-forward is applied
- The constant EC\_ADDITIVE\_TORQUE\_VALUE
- An ignored integer value (usually set to zero)

For more details, refer to the three following links:

- [Torque Feed-forward](#)
- [Guidelines for Choosing feed-forward Control in Industrial Applications](#)
- [Tuning with Feed-forwards](#)
- [Measurement-based Feed-forward Tuning](#)

*How is Torque Feed-forward Scaled?*

If I measure a number e.g. 500 as an input at the [Converter](#) block which is connected with the PDO object (Additive Torque Value 0x60B2), how many Amps are fed in the current loop at the AKD?

```
Current loop feed-forward value = Rated current x IL.KBUSFF x input at
converter block / 1000
```

For example, with an AKD where:

Rated current	3A
IL.KBUSFF	1.0
Additive Torque (PDO object)	500 units

Then

$$IL.FF = 3 \times 1.0 \times 500 / 1000$$

$$IL.FF = 1.5A$$

*What are the limitations with cams?*

There is no limitation with the cams, the number of cams, the number of cam points, etc.. The limitation is only given by the processing power of your PC.

*If a variable is associated with an I/O point value, would it get automatically updated?*

Yes, I/O points represent the state of real world values.

*How does the Pipe Network engine interact with a PLC program?*

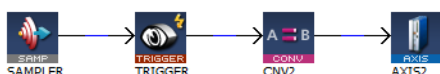
This item is explained [here](#)

*When I pressed Ctrl-V to paste Pipe Network blocks, nothing happened.*

The system needs to know where to place the copied blocks. Press Ctrl-V then use the mouse to point to where you want the copied blocks to be pasted.

*How can I capture the correct pipe position?*

There are network and mechanical delays in systems that will make capturing a correct pipe position difficult. To remove the delays, you should use a sampler block to capture the actual position of an axis. The following shows how this works.



The sampler block above is configured to capture the actual position of axis 1. The conversion and axis blocks are only needed to terminate the pipe. [MLTrigWriteDelay](#) may still be needed to compensate for sensor delay.

If you need to calculate the master pipe value corresponding to the captured axis position, you need to add the difference between the axis command position and the master pipe value.

*A hardware limit flag is being thrown by a PipeNetwork axis at the start of my program.*

If a hardware limit flag is being triggered at the program startup by a PipeNetwork axis, [MLAxisReAlign](#) may need to be called immediately before motion occurs. In this case, the distance to travel should be at least as long as the maximum distance the hardware limit flag can remain active.

## 15.2 Compiler Errors

The following errors may be seen when compiling a project.

"EtherCAT: ERROR: Failed to retrieve Vendor...", or  
 "EtherCAT: ERROR: Failed to retrieve EtherCAT device..."

These errors indicate either the ESI file for the EtherCAT device is missing or information is missing from the ESI file for a device and/or module.

Examples:

```

EtherCAT: ----- Generating EtherCAT Network Information (ENI)
file -----
EtherCAT: ERROR: Failed to retrieve Vendor for vendorID=0x21 in ESI
cache
Project compile failed

```

```

EtherCAT: ----- Generating EtherCAT Network Information (ENI)
file -----
EtherCAT: ERROR: Failed to retrieve EtherCAT device (vendorID=0x2,
productCode=0x44c2c52, revision=0x120000) in ESI cache
Project compile failed

```

To correct the error, identify which EtherCAT devices or modules are listed as “Unknown” in the Project View and import an appropriate ESI file. For more details, see: [Unknown – Missing ESI File](#).

*"EtherCAT: ERROR: In AKD\_1, PDO object index 0x6040, subIndex 0 is redundant", or  
 "Failed to export ENI file, check the logs for more details.", or  
 "EtherCAT: Warning: In AKD\_1, PDO object index 0x3470, subIndex 4 is redundant."*

These errors are indicative of redundant PDO entries, see ["Redundant PDO Entries" on page 348](#).

*Compile Optimized Code FAILED*

The following compiler output indicates a failure to compile the optimized ("C") code:

```

Controller:PLC:> ----- Compile Optimized Code -----
Controller:PLC:> Compiling "C" code.
Controller:PLC:> ERROR: Failed to compile PLC "C" code.
Controller:PLC:> Compilation log:
...
Controller:PLC:> End of Compilation log
Controller:PLC:> Compile Time Elapsed:1.37 seconds
Controller:PLC:> ----- Compile Optimized Code FAILED -----
---
```

Please send the following items to Kollmorgen technical support to help us resolve it with you.

1. the compilation log (copied from the compiler output)
2. the files in the following directory (the path can be copied and pasted into the Windows Explorer address bar):

%LOCALAPPDATA%\Kollmorgen\KAS\Project\Controller\PLC\CC\

Kollmorgen technical support contact information can be found in [Global Support Contacts](#), or at [Kollmorgen.com](#). Please call or e-mail the appropriate support for your region.

*Compiling optimized code fails due to user permissions*

A possible cause for optimized PLC code to fail compilation is if the user does not have permission to write to an output file or folder. The problem can be resolved by modifying the file permissions to allow writing to the file. Below are examples of error messages for specific files.

```

Controller:PLC:> ----- Compile Optimized Code -----
Controller:PLC:> ERROR: cannot remove old output file
"C:\Users\my.name\AppData\Local\Kollmorgen\KAS\Project\Controller\PLC\C
C\cclog.txt"
Controller:PLC: > To resolve the issue, set the permissions on the file

```

```

so that it is writeable.
Controller:PLC:> Compile Time Elapsed:0.02 seconds
Controller:PLC:> ----- Compile Optimized Code FAILED -----
---
```

**Figure 9-2:** The file clog.txt is not writeable.

```

Controller:----- Compile PLC -----
...
Controller:PLC:main
Controller:PLC:c:\users\my.name\appdata\local\kollmorgen\kas\project\co
ntroller\plc\cc\main.c:
Cant write output file
Controller:PLC:appli
Controller:PLC:Error(s) detected
Controller:----- PLC failed -----
```

**Figure 9-3:** The file main.c is not writeable.

### 15.3 CPU Load Reduction Techniques

There are several things you can change to reduce the CPU load:

- Reduce the cyclic update rate by increasing the [Cycle Time](#) in the EtherCAT view. This will have the biggest impact to improve the RealTimeMargin.
- Reduce the number of axes. This will decrease the MotionExecTime.
- Distribute the PLC program execution across multiple PLC cycles (see ["Define the PLC Cycle" on page 204](#)). This will decrease the PLCProgExecTime.

## 15.4 EtherCAT Diagnostics & Errors

### 15.4.1 EtherCAT Diagnostics

The EtherCAT nodes provide several types of diagnostic information through various registers. The KAS controller uses these diagnostic registers to detect error conditions in the EtherCAT network. This can include:

<b>Physical Link status</b>	Indicates if a hardware connection between two nodes is established.
<b>Link Lost Counter</b>	Indicates how many connection lost events have occurred between two nodes. Some EtherCAT devices will not detect the lost link event.
<b>Rx Error Counter</b>	This counter is incremented when the node detects any signal error detected by the hardware.
<b>Rx Invalid Frame Counter</b>	Invalid frame error is detected when the CRC (Cyclic Redundancy Check) does not match the received frame CRC.
<b>Forwarded Rx Error Counter</b>	Frame error detected by the previous node and forwarded to this node.
<b>EtherCAT Processing Unit Error Counter</b>	Counts errors in the EtherCAT frames such as frame length or a non-EtherCAT frame. Not all EtherCAT devices support this register.

- **Application Started:** When the application is started, the KAS controller resets the Link Lost Counter, Rx Error Counter, Rx Invalid Frame Counter, Forwarded Rx Error Counter, and EtherCAT Processing Unit Error Counter registers of all the nodes.
- **Application Running:** While the application is running, the KAS controller monitors every cyclic frame and detects any missing or invalid frames. If any frame is missed or invalid, it generates an [A38](#) alarm.
- **Application Stopped:** When the application is stopped, the controller will read the EtherCAT node diagnostic registers and process them. If any error condition is detected, an EtherCAT diagnostic log message is generated. The log messages can be used to identify the location of the communication failure.

The node register of **Link Lost Counter**, **Rx Error Counter**, **Rx Invalid Frame Counter**, **Forwarded Rx Error Counter**, and **EtherCAT Processing Unit Error Counter** will have non-zero values when a cable disconnection or an EtherCAT frame corruption occurs, The following example indicates that a communication failure occurred between Node 0 and Node 1.

```
EtherCAT | WARNING | EtherCAT diagnostics: AKD_1, port B (out) RX Error Counter is 255
```

```
EtherCAT | WARNING | EtherCAT diagnostics: AKD_2, port A (in) RX Error Counter is 255
```

### 15.4.2 EtherCAT Error Messages

This section covers the following error messages linked to the EtherCAT motion bus that are displayed in the [Information and Logs](#) window:

```
Working Counter failure limit exceeded due to failed node response to cyclic commands. Stopping EtherCAT network cyclic communication.
```

```
Link Error detected! Please, check controller connection.
```

Slave %s is not responding to acyclic frame. Please, check power supply or connection.

These messages can arise due to the following causes seen in the following table.

Error	Case Description	Results
<p><b>Wrong / Missing Device</b></p>	<p>The XML network configuration file contains the list of all EtherCAT devices present in the network.</p> <p>At the EtherCAT initialization phase, the master checks that:</p> <ul style="list-style-type: none"> <li>• Every physical device in the network corresponds to the configured devices (the master detects if the configuration does not match the physical devices)</li> <li>• The Standard I/O Couplers and I/O slices are correct by adding the proper commands in the network configuration file (this allows the detection of wrong or missing Standard I/O Coupler)</li> </ul>	<p>An Error log is generated with the relevant information.</p> <p>The EtherCAT startup is aborted, as well as the startup of the machine.</p>
<p><b>Link Loss / Device Fault</b></p>	<p>This kind of error can appear anytime in the EtherCAT communication, typically when a cable is disconnected or cut or whenever an EtherCAT device is damaged.</p> <p>The master has a mechanism that detects such situations.</p>	<p>An Error log is generated with the relevant information.</p> <p>The EtherCAT communication is aborted.</p> <p>If the network is cut, the drives on the side of the network disconnected from the master are moved into an error state (<b>F29</b>). They are automatically stopped and powered off.</p> <p>In addition, all still-reachable axes have to be stopped and powered off.</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p style="text-align: center;"><b>NOTE</b></p> <p>It can be necessary to put the axes in a safe position before powering it off (this action is application dependent).</p> </div>
<p><b>Frame Loss</b></p>	<p>For security, all frames sent must be received in a given timeout period (at least before the next cycle is started).</p> <p>The master detects this case by managing the appropriate timeout watchdogs.</p>	<p>An Error log is generated with the relevant information.</p> <p>The EtherCAT communication is aborted.</p>

Error	Case Description	Results
<b>Working Counter</b>	<p>When EtherCAT master (KAS controller) sends out an EtherCAT datagram it sets the Working Counter (WC) value to 0. The nodes receiving the frame increment the WC at the hardware level according to a precise rule. When the controller receives the frame it compares the WC with the expected value to detect error conditions.</p> <p>Possible Reasons for the Working Counter Error:</p> <ul style="list-style-type: none"> <li>• The slave EtherCAT device does not exist or cannot be physically reached due to connector/cable failure.</li> <li>• Slave hardware failure</li> <li>• Slave is still busy with a previous command and is not ready.</li> </ul>	<p>The controller generates an A31 alarm and logs a warning message when the WC values does not match the expected value. When the controller detects three such working counter failures within 1000 EtherCAT frames, it generates an E30 error and shuts down the EtherCAT communication.</p>

#### 15.4.2.1 Other Messages Linked to EtherCAT

For more information see ["Communication and Fieldbus" on page 45](#)

The following message is displayed if an error or inconsistency is discovered during the parsing of the XML file when the application is started:

```
Unable to open EtherCAT config file <file-name>
<file-name>:<line>:<column>: <parsing error>
```

#### 15.4.3 EtherCAT Communication Diagnosis Steps

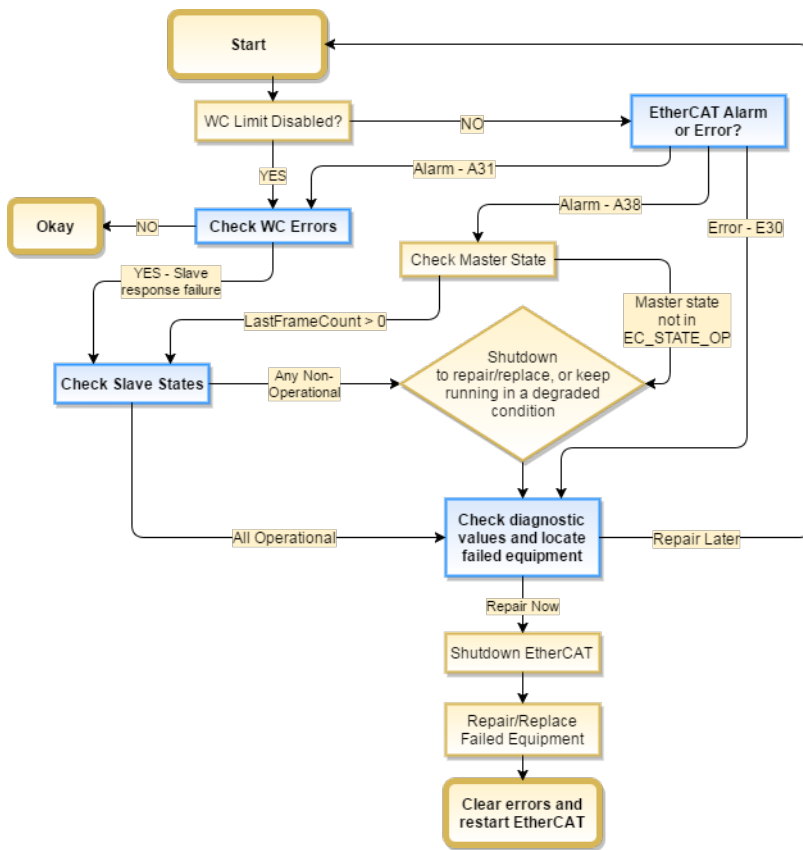
This topic covers the steps to diagnose EtherCAT communication errors, and provides examples for how an application program can help a technician detect the issue and repair it.

##### 15.4.3.1 Diagnosing EtherCAT Communication Errors

The flow chart below explains the steps involved in diagnosing the EtherCAT errors.

#### TIP

The blue fields in the flow chart are clickable. Each points to sample code that can be used in a program to help with diagnosis.



### 15.4.3.2 Code Examples for Diagnosing EtherCAT Communication Errors

Checking for Working Counter Errors:

This sample code checks for any Working Counter failure using the function `ECATWCStatus`. If there is one or more working counter errors then the return value will be greater than 0.

```

WC_ErrorCount := ECATWCStatus( 0 );
IF WC_ErrorCount > 0 THEN
  WC_error := TRUE; // There is a communication problem.
ELSE
  WC_error := FALSE; // No communication problem
END_IF;
  
```

Checking for existing EtherCAT Alarms and Errors:

The `GetCtrlErrors` function can be used to get the Errors and Alarms in the controller that are related to EtherCAT.

```

ControllerErrorStatus:= GetCtrlErrors(ActiveError, ActiveAlarm) ;
A31Active := ActiveAlarm[31]; // True if A31 is active.
A38Active := ActiveAlarm[38]; // True if A38 alarm is active
E30Active := ActiveError[30] // True if E30 is active.
// Process the alarms and Errors here.
  
```

Checking the Device States:

The device state can be read by the application using the function block `ECATDeviceStatus`.



```

Inst_ECATDeviceStatus(True, slaveAddress);
IF Inst_ECATDeviceStatus.Done THEN
  IF Inst_ECATDeviceStatus.LinkStatus = EC_LINK_NO_COMMUNICATION THEN
    device_not_reachable := TRUE; // Device is not reachable. Link to
the device is broken.
  END_IF
  IF Inst_ECATDeviceStatus.State <> EC_STATE_OP THEN
    device_not_in_OPMODE := TRUE; // Slave is not in Operational
mode.
  END_IF
END_IF;

```

#### Checking the Connections for Errors:

This sample code uses [ECATCommErrors](#) function block to identify the connections that have errors.

```

Inst_ECATCommErrors( TRUE, connections);
IF Inst_ECATCommErrors.Done THEN
  IF Inst_ECATCommErrors.ConnectionCount > 0 THEN
    // Process the connection errors here.
  END_IF;
END_IF

```

#### TIP

See the article "[Building KAS Applications with Built-in EtherCAT Diagnostics](#)" on KDN for a code sample and project file that uses EtherCAT diagnostics.

## 15.5 EtherCAT Coupler Error Handling And Diagnosis

This section provides information about the diagnostic LEDs for the EtherCAT Coupler (AKT-ECT-000).

### NOTE

This section is an excerpt of the [EtherCAT Coupler Technical Manual](#).

### 15.5.1 EtherCAT Diagnostic LEDs

After switching on, the ETHERCAT Bus Coupler immediately checks the connected configuration. Error-free start-up is indicated when the red I/O ERR LED goes out. If the I/O ERR LED blinks, an error in the area of the terminals is indicated. The error code can be determined from the frequency and number of blinks. See below for more information.

The ETHERCAT Bus Coupler has respectively a green and yellow LED at the RJ45 plug sockets, which indicate the state of the fieldbus (Figure 4.1). The RUN and ERROR LEDs (upper middle) indicate the state of the EtherCAT State Machine.

On the upper right hand side of the Bus Couplers are two more green LEDs that indicate the supply voltage. The left hand LED indicates the presence of the 24 V supply for the Bus Coupler. The right hand LED indicates the presence of the supply to the power contacts.

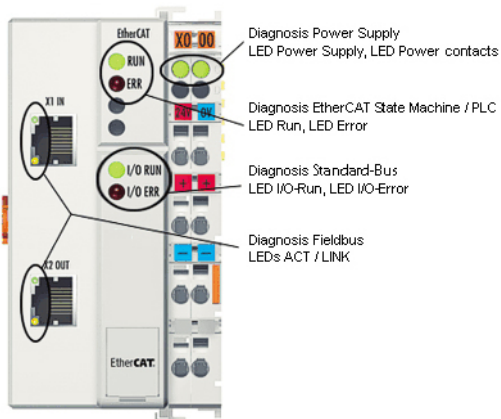


Figure 9-4: ETHERCAT Coupler Diagnostic LEDs

### 15.5.2 EtherCAT LED Power Supply Diagnosis

LED	Display	Description
Power Supply	Green Off	No operating voltage connected
	Green On	24 VDC operating voltage connected
Power Contacts	Green Off	No 24 VDC power connected to the power contacts
	Green On	24 VDC power connected to the power contacts

### 15.5.3 EtherCAT LED Off Power Supply Diagnosis

LEDs	Description
Left Green LED Off	Bus coupler has no power
Right Green LED Off	No 24 VDC power connected to the power contacts

### 15.5.4 LEDs for EtherCAT State Machine/PLC Diagnosis

LEDs	Display	Status	Description
Run	Green	Off	Init State of the EtherCAT State Machine: INIT = Initialization
		Blinking	Pre-operational State of the EtherCAT State Machine: PREOP = Pre-Operational
		Single Flash	Safe operational State of the EtherCAT State Machine: SAFEOP = Safe-Operational
		On	Operational State of the EtherCAT State Machine: OP = Operational
		Flashes	Bootstrap State of the EtherCAT State Machine: BOOT = Bootstrap (Update of the coupler firmware)
Error	Red	Off	— No errors
		Blinking	Err-Operational No Communication PLC error / Lost frames

### 15.5.5 LEDs for EtherCAT Connection Diagnosis

LEDs	Display	Status	Description
LINK (X1 IN)	Yellow	Off	No connection with the previous EtherCAT client
		On	Linked Previous EtherCAT-client connected
ACT (X1 IN)	Green	Blinking	Active Communication with the previous EtherCAT client
		Off	No connection with the previous EtherCAT client
		On	No communication with the previous EtherCAT client
LINK (X2 OUT)	Yellow	Off	Linked Next EtherCAT client connected
		On	Active Next EtherCAT client connected
ACT(X2 OUT)	Green	Blinking	Active Communication with the next EtherCAT client
		Off	No connection with the next EtherCAT client
		On	No communication with next previous EtherCAT client

### 15.5.6 LEDs for EtherCAT Data Diagnosis

LEDs	Display	Status	Description
I/O-Run	Green	Off	— EtherCAT Bus inactive
		On	— EtherCAT Bus active
LED Red; I/O Error	Error Code Argument	Description	Remedy
Persistent, continuous blinking		EMC problems	<ul style="list-style-type: none"> <li>• Check power supply for overvoltage or under voltage peaks</li> <li>• Implement EMC measures</li> <li>• If a EtherCAT Bus error is present, it can be localized by a restart of the coupler (by switching it off and then on again)</li> </ul>

LED Red; I/O Error	Error Code Argument	Description	Remedy
1 Pulse	0	EEPROM checksum error	Set manufacturer's setting with the configuration software
	1	Code buffer overflow	Insert fewer Bus Terminals. The programmed configuration has too many entries in the table Software update required for the Bus Coupler
	2	Unknown data type	Software update required for the Bus Coupler
2 Pulses	0	Programmed configuration has an incorrect table entry	Check programmed configuration for correctness
	n (n > 0)	Table comparison (Bus Terminal n)	Incorrect table entry
3 Pulses	0	EtherCAT Bus command error	<ul style="list-style-type: none"> <li>No Bus Terminal inserted</li> <li>One of the Bus Terminals is defective; halve the number of Bus Terminals attached and check whether the error is still present with the remaining Bus Terminals. Repeat until the defective Bus Terminal is located.</li> </ul>
4 Pulses	0	EtherCAT Bus data error, break behind the Bus Coupler	Check whether the n+1 Bus Terminal is correctly connected; replace if necessary.
	n	Break behind Bus Terminal n	Check whether the Bus End Terminal is connected.
5 Pulses	n	EtherCAT Bus error in register communication with Bus Terminal n	Exchange the nth Bus Terminal
14 Pulses	n	nth Bus Terminal has the wrong format	Start the Bus Coupler again, and if the error occurs again then exchange the Bus Terminal
15 Pulses	n	Number of Bus Terminals is no longer correct	Start the Bus Coupler again. If the error occurs again, restore the manufacturers setting using the configuration software
16 Pulses	n	Length of the EtherCAT Bus data is no longer correct	Start the Bus Coupler again. If the error occurs again, restore the manufacturers setting using the configuration software

## 15.6 Troubleshooting FSoE Safety Parameters

Safety parameters can be set by using the [FSoEParamsInit](#) function block. This operation requires proper configuration of the FSoE master, FSoE safety devices, and the EtherCAT master (the PCMM or AKD PDMM).

*Are there any errors?*

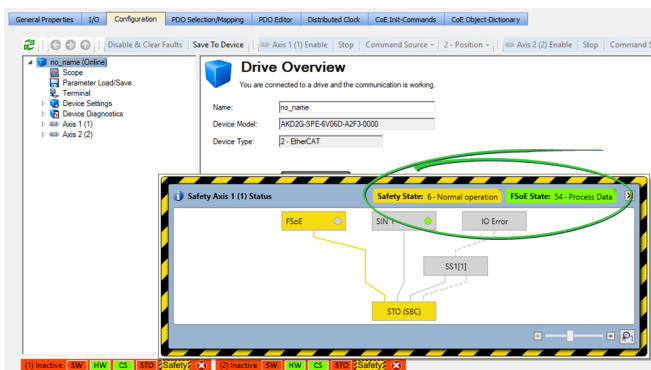
The first thing to do is to check the ErrorID output of [FSOEParamsInit](#). There are some common failure cases that can be identified from the ErrorID values and the Controller log messages.

<b>ECERR_DEVICE_NOTREADY</b>	<p>The Safety parameters can only be sent after the EtherCAT network state is in op-mode. To verify EtherCAT is ready:</p> <ul style="list-style-type: none"> <li>• Verify the motion engine is running. Use <a href="#">MLMotionStatus</a> to verify the status is MLSTATUS_RUNNING before calling FSOEParamsInit.</li> <li>• Check the EtherCAT state directly. Use <a href="#">ECATMasterStatus</a> to verify the state is EC_STATE_OP.</li> </ul>
<b>ECERR_DEVICE_FSOE_MASTER_INVALID</b>	<p>This error is generated if the argument specified for <a href="#">FSOEParamsInit</a>'s FSOEMasterAddress is not a FSOE Master.</p> <ul style="list-style-type: none"> <li>• Check to ensure that the EtherCAT address specified for the FSOE master is correct.</li> <li>• Check the supported FSOE Masters list.</li> </ul>
<b>ECERR_DEVICE_TIMEOUT</b>	<p>This can occur if the FSOE Master is not responding or is not configured properly.</p> <ul style="list-style-type: none"> <li>• Check the FSOE Master diagnostic LEDs, Error Codes/Messages, log messages, and any other diagnostic tools available from the FSOE Master vendor. For example, with the BBH SCU-1-EC FSOE master, use the SafePLC2 software.</li> </ul>

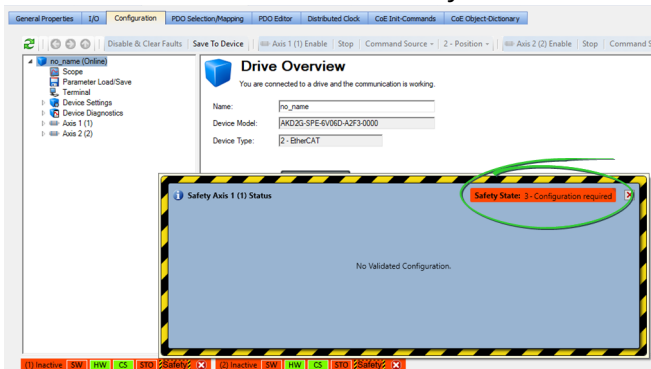
*FSOEParamsInit(...) is successful but the AKD2G Safety State does not reach 6 - Normal operation.*

If FSOEParamsInit(...) is successful but the AKD2G Safety State does not reach 6 - Normal operation then the CRC expected by the AKD2G SMM does not match the CRC sent by the FSOE master.

1. Use WorkBench or the KAS IDE's integrated WorkBench (Configuration tab for the AKD2G) to diagnose.
2. Click the "Safety" item in the status bar.
  - If FSOE is operating normally the Safety State will be 6 - Normal Operation and FSOE State will be 54 - Process Data.



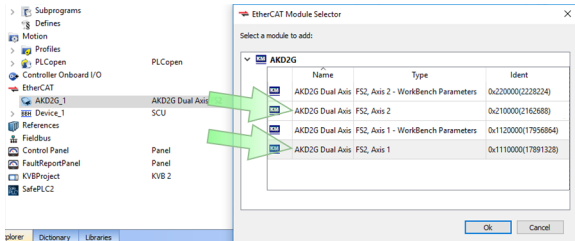
- If the CRC does not match the Safety State will be 3 - Configuration required.



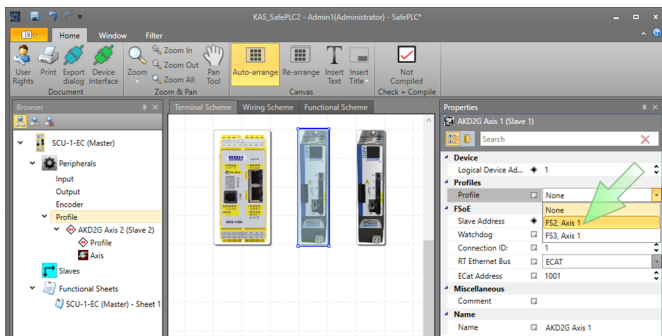
*AKD2G module(s) are incorrectly set to use "Workbench Parameters"*

The AKD2G has two options for configuring safety parameters which are determined by selecting the modules associated with the AKD2G in the KAS IDE Project Tree.

If the module(s) are set for "Workbench Parameters" then the FSoE Master cannot set the AKD2G's safety parameters. The "WorkBench Parameter" modules are set by default when the AKD2G is scanned and discovered. The modules must be changed to accept the parameter from the Safety Maser if you want to use [FSoEParamsInit](#). The picture below shows the appropriate modules for using [FSoEParamsInit](#) with AKD2G.



Additionally, the AKD2G profile in the SafePLC2 project should be set to "FSx Axis Y" in order to send the parameters to the AKD2G.



*FSoEParamsInit returns with Done = TRUE and Error = FALSE, but no parameters are initialized.*

This can occur in two situations:

- if the FSoE safety parameters have been previously configured correctly
- if the FSoE Master has not been configured properly.

[FSoEParamsInit](#) will only initialize safety parameters if the FSoE master determines there are parameters to set in the safety device(s). Please check the FSoE master configuration using the vendor's software. For example, with the BBH SCU-1-EC FSoE master, use the SafePLC2 software.

**See Also:**

- [Set Up FSoE Master and an AKD2G with SafeMotion Monitor](#)
- [AKD2G Safety Parametrization Using FSoE with SCU-1-EC and PCMM/AKD PDMM](#)
- [FSoEParamsInit](#)

## 15.7 Troubleshooting KAS IDE and SafePLC2

Following are several situations you may encounter when you are setting up a KAS project involving SafePLC2.

*The ESI file for a Safety device is reported as missing.*

EtherCAT devices which are missing ESI files will be identified in the KAS IDE's EtherCAT **Devices** tab.

#	Name	Vendor	Product	Model
1	AKD2G_1	Kollmorgen	AKD2G Dual Axis FS2 EtherCAT Drive (CoE)	AKD2G
2	Device_1	BBH Products GmbH	SCU	N/A
3	Coupler_1	Kollmorgen	AKT2G-ECT-000-000 EtherCAT Gateway	N/A
4	DI_Slice_1	Kollmorgen	AKT2G-DN-008-000 8Ch. Dig. Input 24V, 3ms	N/A
5	DI_Slice_2	Kollmorgen	AKT2G-DNH-008-000 8Ch. Dig. Input 24V, 10µs	N/A
6	DO_Slice_1	Kollmorgen	AKT2G-DT-008-000 8-channel digital output terminal 24 V DC, 0.5 A	N/A
7	AI_Slice_1	Kollmorgen	AKT2G-AN-430-000 4Ch. Ana. Input +/-10V Diff., +/-20mA SingleEnded, 16 Bit	N/A
8	AO_Slice_1	Kollmorgen	AKT2G-AT-410-000 4Ch. Ana. Output 0-10V, 12bit	N/A
9	DO_SafeSlice_1	Kollmorgen	AKT2G-SDO-004-000, 4 Ch. Safety Output 24V, 0.5A	N/A
10	DI_SafeSlice_1	Kollmorgen	AKT2G-SDI-004-000, 4 Ch. Safety Input 24V	N/A
11	Device_2	0	Missing ESI for 0x7703052	N/A
12	Device_3	0	Missing ESI for 0xb583052	N/A

If a Safety device is added to a SafePLC2 project and the KAS IDE's ESI library does not have a corresponding ESI file for the device, then the Local log messages will indicate a problem with device creation. For example:

```
WARNING | Slave creation failed while trying to retrieve slave default
WARNING | Device_2: Slave creation failed while trying to retrieve
slave default Init Commands
WARNING | Device_2: Slave creation failed while trying to get
Distributed Clock(DC) details
WARNING | Slave creation failed while trying to retrieve slave default
PDOs
```

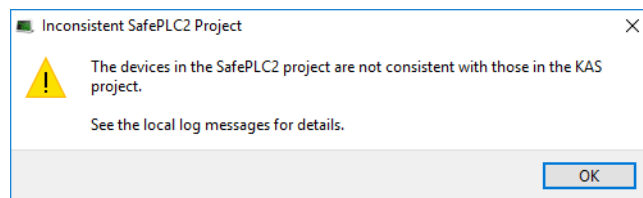
**To correct the problem:** Use the KAS IDE's EtherCAT-> **ESI Files** tab to add the ESI files for the missing devices.

**To avoid the problem:** Add all Safety devices from the KAS IDE, including their ESI files, and then launch SafePLC2 from the KAS IDE project tree. SafePLC2 will automatically read the Safety device topology from the KAS IDE.

#### *Topology mismatch with the imported SafePLC2 project*

If a SafePLC2 project file is imported into a KAS project, the KAS IDE will check the incoming Safety project's EtherCAT topology. Importing the project will fail if there is a topology mismatch and the Local logs will identify the mismatched nodes.

For example:



```
ERROR | SafePLC2 topology parser: device #1 - SafeModule #1 -
ConnectsTo element - KAS ID 0 does not match that of the device with
SafePLC2 ID 1 - 4
ERROR | SafePLC2 topology parser: device #1 - SafeModule #2 -
ConnectsTo element - KAS ID 0 does not match that of the device with
SafePLC2 ID 1 - 4
ERROR | SafePLC2 topology parser: device #1 - SafeModule #3 -
```

```
ConnectsTo element - KAS ID 8 does not match that of the device with
SafePLC2 ID 3 - 9
ERROR | SafePLC2 topology parser: device #1 - SafeModule #4 -
ConnectsTo element - KAS ID 9 does not match that of the device with
SafePLC2 ID 4 - 8
ERROR | SafePLC2 topology parser: device #2 - the EtherCAT ID of the
device with KAS ID 4 does not match that of "DI_Slice_2"
ERROR | SafePLC2 topology parser: device #3 - SafePLC2 ID (3) does not
match that of "DI_SafeSlice_1" (4)
ERROR | SafePLC2 topology parser: device #3 - the EtherCAT ID of the
device with KAS ID 9 does not match that of "DI_SafeSlice_1"
ERROR | SafePLC2 topology parser: device #4 - SafePLC2 ID (4) does not
match that of "DO_SafeSlice_1" (3)
ERROR | SafePLC2 topology parser: device #4 - the EtherCAT ID of the
device with KAS ID 8 does not match that of "DO_SafeSlice_1"
```

Another example:

```
ERROR | SafePLC2 topology parser: device #1 - no device with
identifier 11
ERROR | SafePLC2 topology parser: device #2 - no device with
identifier 10
ERROR | SafePLC2 topology parser: device #3 - no device with
identifier 18
ERROR | SafePLC2 topology parser: device #4 - no device with
identifier 19
```

#### NOTE

Any pre-existing SafePLC2 project inside the KAS project will be retained if importing the SafePLC2 project fails.

**To correct the problem:** Import the SafePLC2 project to the matching .kas project that was originally exported to SafePLC2.

#### *Duplicate FSoE Address*

When compiling, the KAS IDE checks to ensure the FSoE address is unique. A compiler warning message is generated if the address is not unique. For example:

```
EtherCAT: Warning: FSoE address 101 is duplicated for the following
devices: AKD2G_1, DI_SafeSlice_1, DO_SafeSlice_1
```

**To correct the problem:**

1. Change the FSoE address of the devices and set them to be unique to the network.  
Refer to the device installation manual for setting the FSoE address.
2. Rescan the EtherCAT network and repeat the steps described in [Working with SafePLC2 Projects](#).

#### *FSoE/PDO Connections do not match the EtherCAT topology.*

The compiler will generate warnings if the FSoE / PDO Connections defined by the Safety device topology from the SafePLC2 project do not match the EtherCAT Safety device topology defined by the KAS IDE. For example:

```
< compiler messages go here! >
```



**To correct the problem:** Use the KAS IDE's EtherCAT-> **Scan Devices** to discover and associate the physical Safety devices that are expected by the SafePLC2 project.

**To avoid the problem:** Scan and discover all physical EtherCAT Safety devices from the KAS IDE before launching SafePLC2 from the KAS IDE Project tree. SafePLC2 will automatically read the Safety device topology from the KAS IDE.

## 15.8 Connect Remotely

You can connect to a AKD PDMM or PCMM from an external network using [VPN](#) or other tunneling protocol. To do so you must open the following ports which are used by KAS to connect to a controller.

Port	Component/Protocol using the port
80	KAS IDE: Controller Web Server; Download PLC application; Runtime front end communication. This is necessary for the connection to the runtime. If port 80 is used by any other application, then the Simulator will use an alternate port available in the host computer for HTTP communication.
502	HMI and other Modbus devices to communicate with Modbus TCP.
2222	Ethernet/IP communication
4002	KAS IDE: Oscilloscope variable monitoring.
4003	KAS IDE: Communication with the PLC Virtual Machine. This is necessary for the connection to the runtime.
9900	KAS IDE: Communication with the runtime engine; WorkBenchviews. This is necessary for the connection to the runtime.
34962	Profinet communication
34964	Profinet communication
44848	Ethernet/IP communication

## 16 Annexes: Lists of Manuals, Content, and Support Information

This section provides lists of available manuals, tables, figures, how-to's, etcetera.

### 16.1 List of How-Tos

#### 16.1.1 PLC Code How-Tos

- [Declare an Array](#)
- [Control an SFC Child](#)
- [Draw SFC divergences](#)
- [Create SFC Parallel Branches](#)
- [Toggle a FBD Connection to make it Negative](#)
- [Change a Link in the Pipe Network](#)
- [Create a PLCopen Axis](#)
- [Read Output of a MC Function Block in ST](#)
- [Sort the Variables in the Dictionary](#)
- [Understand the Location Details in the Find and Replace window](#)
- [Start and Stop a PLC Application using HTTP Requests from the HMI](#)

#### 16.1.2 EtherCAT Fieldbus How-Tos

- [How To Create a Digitizing Axis](#)

#### 16.1.3 Advanced Motion How-Tos

- [Use Fast Inputs with PLCOpen](#)
- [Use Fast Inputs with Pipe Network Motion](#)
- [Implement the Torque Feed-forward](#)
- [Working With A Digitizing Axis in PLCopen](#)

#### 16.1.4 Run the Application How-Tos

- [Choose the Appropriate Level for Log Messages](#)
- [Plug a Probe in the Softscope](#)
- [Plug Motion Variables in the Softscope](#)
- [Plug PLC Variables in the Softscope](#)
- [How Do I Export the Oscilloscope Data?](#)
- [Set Breakpoints](#)
- [Activate Online Change](#)
- [Change Priority among Programs](#)
- [Specify the Duration of a Cycle](#)

#### 16.1.5 Hardware How-Tos







- [Download a new Firmware to my AKD Drive](#)
- [Check the NVRAM space is enough to store my retain variables](#)
- [Download your Application on the HMI device \(AKI\)](#)

### 16.2 List of KAS Manuals

Find below the available literature for the following components:

- [KAS IDE](#)
- [KVB](#)
- [HMI](#)
- [I/O](#)
- [AKD2G, AKD, & S300/S700](#)
- [PDMM / PCMM](#)








### 16.2.1 KAS IDE

Document Title	PDF	Description
<b>Release Notes</b>		The KAS version 3.07 Release Notes contain fixed limitations, known limitations, workarounds, and information on all hardware and software components that have been updated, changed or added in this release.
<b>Getting Started</b>		Covers the main steps to get your KAS system up and running.  <i>What does it contain?</i> <ul style="list-style-type: none"> <li>• HW Installation (Connection and Wiring): Wiring &amp; hardware details, connectors, system diagrams</li> <li>• HW Configuration: Basic configuration and settings needed to start the HW components (HMI + Industrial PC + Fieldbus + I/O)</li> <li>• SW Installation: KAS software setup</li> </ul>
<b>30 Minutes to Motion</b>		Covers the main topics to help you start quickly with KAS IDE. The objective is to familiarize you with the basic principles and the way the program works by creating a simple motion application project.  <i>What does it contain?</i> <ul style="list-style-type: none"> <li>• Key Features</li> <li>• Explore the Workspace</li> <li>• Build a motion project</li> </ul> <p>Almost every task that you perform in KAS falls under one of the following basic steps (which may not always be completed in the following order):</p> <ol style="list-style-type: none"> <li>1. Start Projects - Create a project from scratch, or modify an existing project.</li> <li>2. Add Components - Add elements to build your project necessary to control the motion part of your system.</li> <li>3. Build Output - Select a device and generate the application that you will deliver to users.</li> <li>4. Run Output - Make the output accessible to your end-users.</li> </ol>
<b>IDE User Manual</b>		Contains the content to help you with KAS IDE, except the topics included in the Reference Manuals.
<b>Reference Manual - PLC Library</b>		Contains Technical References on <b>PLC</b> Programming Languages and Library.
<b>Reference Manual - Motion Library</b>		Contains Technical References on <b>Motion</b> Library for Pipe Network and PLCopen.

### 16.2.2 KVB

KVB Guide	Description
<a href="#">Kollmorgen Visualization Builder™ Quick Start Guide</a>	Quick Start that covers the most important points to install and use Kollmorgen Visualization Builder, in order to configure HMI Panels and PC operated control applications.
<a href="#">Kollmorgen Visualization Builder™ User Manual</a>	Contains all the content to help you with Kollmorgen Visualization Builder.

### 16.2.3 HMI

HMI Part Number	Description	KVB	Manual
AKI2G-CDA-MOD-05T-000	Graphical Display 7" TFT LCD, Touchscreen	v2.20	
AKI2G-CDA-MOD-07T-000	Graphical Display 5" TFT LCD, Touchscreen	v2.20	
AKI2G-CDB-MOD-07T-000	Graphical Display 7" TFT LCD, Touchscreen	v2.20	
AKI2G-CDB-MOD-12T-000	Graphical Display 12" TFT LCD, Touchscreen	v2.20	
AKI-CDC-MOD-12T-000	Graphical Display 12.1" TFT LCD, Touchscreen	v2.0	
AKI-CDC-MOD-15T-000	Graphical Display 15.4" TFT LCD, Touchscreen	v2.0	
AKI-CDC-MOD-21T-000	Graphical Display 21.5" TFT LCD, Touchscreen	v2.0	







### 16.2.4 I/O

#### AKT2G (EtherCAT) Terminals

I/O Terminal Part Number	I/O Terminal Description
<a href="#">AKT2G-AC-FAN-001</a>	Fan cartridge for EtherCAT and Bus Terminals
<a href="#">AKT2G-AN-240-000</a>	2-channel input terminal PT100 (RTD) for resistance sensors, 16 bit, 2-, 3-wire system
<a href="#">AKT2G-AN-400-000</a>	4-channel thermocouple input terminal, preset to type K, with wire breakage detection, 16 bit
<a href="#">AKT2G-AN-430-000</a>	4-channel analog input, parameterisable, -10/0...+10 V, -20/0/+4...+20 mA, 16 bit
<a href="#">AKT2G-AT-410-000</a>	4-channel analog output terminal 0...10 V, 12 bit, 1-wire system
<a href="#">AKT2G-AT-425-000</a>	4-channel analog output terminal -10 V...+10 V, 12 bit, 4 x 2-wire system
<a href="#">AKT2G-BRC-000-000</a>	Brake Chopper Terminal
<a href="#">AKT2G-DN-002-000</a>	Up/down counter 24 V DC, 100 kHz, 32 bit counter depth
<a href="#">AKT2G-DN-008-000</a>	8-channel digital input terminal 24 V DC, filter 3.0 ms, 1-wire system
<a href="#">AKT2G-DNH-008-000</a>	8-channel digital input terminal 24 V DC, filter 10 µs, 1-wire system
<a href="#">AKT2G-DT-008-000</a>	8-channel digital output terminal 24 V DC, 0.5 A, 1-wire system
<a href="#">AKT2G-ECT-000-000</a>	EtherCAT Coupler for E-bus terminals
<a href="#">AKT2G-EM-000-000</a>	Bus end cover for E-bus terminals, cover for power and E-bus contacts, grey
<a href="#">AKT2G-ENC-180-000</a>	1-channel incremental encoder interface, 32 bit
<a href="#">AKT2G-ENC-190-000</a>	Incremental encoder interface with differential input, 16/32 bit

I/O Terminal Part Number	I/O Terminal Description
<a href="#">AKT2G-PSF-024-000</a>	Power supply terminal with fuse, 24 V DC
<a href="#">AKT2G-SDI-004-000</a>	4-channel digital input terminal, Safety, 24 V DC
<a href="#">AKT2G-SDO-004-000</a>	4-channel digital output terminal, Safety, 24 V DC, 0.5 A
<a href="#">AKT2G-SM-L15-000</a>	Stepper motor terminal, 24 V DC, 1.5 A, vector control
<a href="#">AKT2G-SM-L50-000</a>	Stepper motor terminal, 50 V DC, 5 A, vector control

*AKT (K-Bus) Terminals*

I/O Terminal Part Number	Description	Link	Replaced By
AKT-AN-200-000	2 Channel Thermocouple Input Module		<a href="#">AKT2G-AN-240-000</a>
AKT-AN-400-000	4 Channel Thermocouple Input Module		<a href="#">AKT2G-AN-400-000</a>
AKT-AN-410-000	4 Channel Analog Input Module, 0-10 VDC		<a href="#">AKT2G-AN-430-000</a>
AKT-AN-420-000	4 Channel Analog Input Module, 0-20 mA		<a href="#">AKT2G-AN-430-000</a>
AKT-AN-810-000	8 Channel Analog Input Module, 0-10 VDC		<a href="#">AKT2G-AN-430-000</a>
AKT-AN-820-000	8 Channel Analog Input Module, 0-20 mA		<a href="#">AKT2G-AN-430-000</a>
AKT-AT-220-000	2 Channel Analog Output Module, 0-20 mA		—
AKT-AT-410-000	4 Channel Analog Output Module, 0-10 VDC		—
AKT-AT-420-000	4 Channel Analog Output Module, 0-20 mA		—
AKT-AT-810-000	8 Channel Analog Output Module, 0-10 VDC		<a href="#">AKT2G-AT-410-000</a>
AKT-AT-820-000	8 Channel Analog Output Module, 0-20 mA		—
AKT-DN-004-000	4 Channel Digital Input Module, 24 VDC 3ms		<a href="#">AKT2G-DN-008-000</a>
AKT-DN-008-000	8 Channel Digital Input Module, 24 VDC 3ms		<a href="#">AKT2G-DN-008-000</a>
AKT-DNH-004-000	4 Channel Digital Input Module, 24 VDC 0.2ms		<a href="#">AKT2G-DNH-008-000</a>
AKT-DNH-008-000	8 Channel Digital Input Module, 24 VDC 0.2ms		<a href="#">AKT2G-DNH-008-000</a>
AKT-DT-004-000	4 Channel Digital Output Module, 24 VDC 0.5A		<a href="#">AKT2G-DT-008-000</a>
AKT-DT-008-000	8 Channel Digital Output Module, 24 VDC 0.5A		<a href="#">AKT2G-DT-008-000</a>
AKT-DT-2RT-000	2 Channel Relay Output Module, 230 V AC 2.0A Rel.2NO Pot.-Free		—
AKT-ECT-000-000	EtherCAT Bus Coupler		<a href="#">AKT2G-ECT-000-000</a>
AKT-EM-000-000	Standard-Bus End Terminal		<a href="#">AKT2G-EM-000-000</a>
AKT-IM-000-000	Isolation / Separation Terminal		—
AKT-PS-024-000	Power Supply, 24 VDC		<a href="#">AKT2G-PSF-024-000</a>
AKT-PSF-024-000	Fused Power Supply with diagnostics, 24 VDC		<a href="#">AKT2G-PSF-024-000</a>

I/O Terminal Part Number	Description	Link	Replaced By
AKT-SM-L15-000	Stepper Motor Terminal, 24 VDC, 1.5 A	 <a href="#">AKT2G-SM-L15-000</a>	<a href="#">AKT2G-SM-L15-000</a>
AKT-SM-L50-000	Stepper Motor Terminal, 50 VDC, 5 A	 <a href="#">AKT2G-SM-L50-000</a>	<a href="#">AKT2G-SM-L50-000</a>

**ⓘ IMPORTANT**

24-volt power is not passed through AKT-AN-200-000 and AKT-AN-400-000 thermocouple modules. To get 24VDC power to devices that need it (such as an AKT\_AT-410-000 Output module) there are two possible solutions.





- Place the module requiring 24VDC before the thermocouple module.
- Add a power feed module (AKT-PS-024-000 or AKT-PSF-024-000) after the thermocouple module.

**16.2.5 AKD2G, AKD, & S300/S700**

Drives Guide	Description
<a href="#">AKD PDMM User Manual</a>	Describes software installation, setup, and operation for the AKD PDMM drive. Includes basic topics and examples to help guide you in setting up and using the various features in the drive.
<a href="#">AKD PDMM Fault Card</a>	Describes AKD PDMM (including AKD) faults, warnings, error messages, and alarms. Provides cause and remedy instructions to help determine the specifics of the failure and to correct the underlying problem.
<a href="#">AKD Installation Manual</a>	Installation manual for AKD and AKD PDMM drives. Describes the AKD series of digital drives and includes mechanical, electrical, and software installation information needed to safely install AKD.
<a href="#">AKD2G Installation Manual</a>	Describes the AKD2G series of digital drives and includes mechanical, electrical, software and functional safety options.
<a href="#">AKD, AKD2G, S700 (in NA) Accessories Manual</a>	This manual describes the accessories for Kollmorgen digital drive systems and servo drive motors.
<a href="#">AKD EtherCAT Manual</a>	Describes the installation, setup, range of functions, and software protocol for the EtherCAT AKD product series,
<a href="#">AKD2G EtherCAT Manual</a>	AKD2G EtherCAT and CANopen Communications Manual describes the installation, setup, range of functions, and software protocol for the AKD2G product series.
<a href="#">S300 Reference Documentation</a>	Kollmorgen website that gives access to all S300 manuals.
<a href="#">S700 Reference Documentation</a>	Kollmorgen website that gives access to all S700 manuals.

**16.2.6 PDMM / PCMM**

Product Guide	Description
<a href="#">AKD PDMM Quick Start</a>	 Contains all information needed to safely install and setup an AKD drive

Product Guide	Description
<a href="#">AKD and AKD PDMM Installation Manual</a>	 Covers the most important points to install the drive hardware and software Provides instructions for basic drive setup and connection to a network
<a href="#">PCMM Installation Guide</a>	 Covers the most installation and setup of the controller.
<a href="#">AKD PDMM User Guide</a>	 Contains information on AKD drive parameters, EWW (Embedded Workbench Views), and using the AKD PDMM in the KAS IDE.
<a href="#">AKD PDMM Fault Card</a>	 Includes technical data and dimensional drawings of accessories such as cables, brake resistors, and mains supplies

## 17 Acronyms & Initialisms

Term	Definition	Description
AKA	Also Known As	Provides an alias to a name
AKD	Advanced Kollmorgen Drive	<a href="#">Drives</a>
AKI	Advanced Kollmorgen Interface	<a href="#">HMI</a>
AKT	Advanced Kollmorgen Terminal	<a href="#">Remote Input/Output Terminals</a>
ANSI	American National Standards Institute	ANSI is a private, nonprofit organization that oversees the development of voluntary consensus standards for products, services, processes, systems, and personnel
ASFB	Application Specific Function Block	Library that can be written to provide a specific application task
ASIC	Application-Specific Integrated Circuit	An ASIC is an integrated circuit (IC) customized for a particular use, rather than for general-purpose use. Modern ASICs often include entire 32-bit processors, memory blocks including ROM, RAM, EEPROM, Flash and other large building blocks
BiSS	Bi-directional Serial Synchronous interface	An open-source communication protocol for feedback devices. With BiSS, all of the computation for interpolation in regard to position occurs on the ASIC directly in the encoder
CAM	Computer-Aided Manufacturing	CAM means the use of a wide range of computer-based software tools that assist engineers and CNC machinists in the manufacture or prototyping of product components
CAN	Controller Area Network	CAN is a broadcast, differential serial bus standard developed for connecting electronic control units. Each node is able to send and receive messages, but not simultaneously.
CF	Compact Flash	CF is a mass storage device format used in portable electronic devices
CIP	Common Industrial Protocol	The Common Industrial Protocol allows complete integration of control with information, multiple CIP Networks, and Internet technologies
CRC	Cyclic Redundancy Check	A CRC is a type of function that takes as input a data stream of any length and produces as output a value of a certain fixed size. The term CRC is often used to denote either the function or the function's output. A CRC can be used as a checksum to detect accidental alteration of data during transmission or storage
CSV	Comma-Separated Values	CSV file format is a file type that stores tabular data



Term	Definition	Description
<b>DMA</b>	Dynamic Memory Allocation	DMA is the allocation of memory storage for use in a computer program during the run-time of that program. It can be seen also as a way of distributing ownership of limited memory resources among many pieces of data and code
<b>EDS</b>	Electronic Data Sheet	The Electronic Data Sheet is a file format that defines the communication behavior and object dictionary for the devices following the CANopen standard CiA 306. EtherCAT ESI files may contain references to EDS files.
<b>ENI</b>	EtherCAT Network Information	<p>A network configuration file in XML format, the ENI file describes the network topology, the initialization commands for each device, and commands which have to be sent cyclically. The ENI file is provided to the master, which sends commands according to this file.</p> <p>The KAS IDE creates the ENI file after a network discovery, which can be exported or imported. A scan and compile should be redone, if the network changes, in order to regenerate the ENI file.</p>
<b>ERP</b>	Enterprise Resource Planning	ERP integrates (or attempts to integrate) all data and processes of an organization into a unified system
<b>ESI</b>	EtherCAT Slave Information	<p>A device description in XML format. This is a fixed file provided by the supplier of a given EtherCAT device. The ESI file contains information about the device's functionality and settings.</p> <p>EtherCAT device vendors must provide an ESI file, which is used by the KAS IDE to compile the network information (e.g. process data structures, initialization commands) and create the ENI file.</p>
<b>FBD</b>	Function Block Diagram	A function block diagram describes a function between input variables and output variables. A function is described as a set of elementary blocks
<b>FFLD</b>	Free Form Ladder Diagram	Free Form Ladder logic is a method of drawing electrical logic schematics. It is now a very popular graphical language for programming Programmable Logic Controllers (PLCs). It was originally invented to describe logic made from relays. The name is based on the observation that programs in this language resemble ladders, with two vertical "rails" and a series of horizontal "rungs" between them
<b>FoE</b>	File over EtherCAT	This very simple protocol, similar to TFTP, enables access to any data structure in the device. Standardized firmware upload to devices is therefore possible, irrespective of whether or not they support TCP/IP
<b>FPGA</b>	Field-Programmable Gate Array	FPGA is a semiconductor device that can be configured by the customer or designer after manufacturing; hence the name "field-programmable"

Term	Definition	Description
<b>FSoE</b>	FailSafe over EtherCAT	The protocol FSoE was specified for the transmission of safety relevant data. It is used to send input information of safety sensors (such as safety light curtains or emergency stop buttons) to a safety logic controller. Based on these inputs, this controller computes the commands for the safe outputs (such as contactors or safety relevant drives) and thus controls the safety functionality of the machine
<b>GUI</b>	Graphical User Interface	A GUI is a type of user interface which allows people to interact with a computer and computer-controlled devices
<b>HMI</b>	Human-Machine Interfaces	Also known as computer-human interfaces (CHI), and formerly known as man-machine interfaces, they are usually employed to communicate with PLCs and other computers, such as entering and monitoring temperatures or pressures for further automated control or emergency response
<b>IC</b>	Integrated Circuits	Miniaturized electronic circuits (consisting mainly of semiconductor devices, as well as passive components) that have been manufactured in the surface of a thin substrate of semiconductor material
<b>IDE</b>	Integrated Development Environment	An integrated development environment is a type of computer software that assists computer programmers in developing software. IDEs normally consist of a source code editor, a compiler and/or interpreter, build-automation tools, and a debugger
<b>IDN</b>	Identification Number	An IDN preceded by the prefix "P", specifies a product specific (manufacturer) IDN in short-hand notation. The actual IDN number for a product-specific IDN, can be obtained by adding 32768 to the short-hand numeric value. For convenience, the actual IDN number is given in parentheses following the short hand notation. For example, P2 is a manufacturer-specific IDN whose actual IDN number is 32770
<b>IEC</b>	International Electrotechnical Commission	IEC is a not-for-profit, non-governmental international standards organization that prepares and publishes International Standards for all electrical, electronic and related technologies
<b>IEC 61131</b>		IEC standard for Programmable logic controllers (PLCs)
<b>IEC 61131-3</b>		IEC 61131-3 is the third part of the open international standard IEC 61131. The current (second) edition was published in 2003. IEC 61131-3 currently defines five programming languages for programmable control systems. It deals with programming languages and defines two graphical and two textual PLC programming language standards
<b>IL</b>	Instruction List	It is a low-level language and resembles assembly

Term	Definition	Description
<b>IPC</b>	Industrial PC	Industrial PC is the x86 PC-based computing platform for industrial applications. Industrial PC offers features different from the consumer PC on the reliability, compatibility, expansibility and long term supply. KAS IPC usually includes a touch-screen display as a combined input and output device.
<b>IRQ</b>	Interrupt Request	An interrupt request refers to the act of interrupting the bus lines used to signal an interrupt
<b>JTAG</b>	Joint Test Action Group	JTAG is used for accessing sub-blocks of integrated circuits, and is also useful as a mechanism for debugging embedded systems, providing a convenient "back door" into the system. When used as a debugging tool, an in-circuit emulator - which in turn uses JTAG as the transport mechanism - enables a programmer to access an on-chip debug module which is integrated into the CPU via the JTAG interface. The debug module enables the programmer to debug the software of an embedded system
<b>KAS</b>	Kollmorgen Automation Suite	Umbrella name for a software package including the KAS IDE and the KAS Runtime software
<b>KAS IDE</b>	Kollmorgen Automation Suite - Integrated Development Environment	The KAS IDE is the GUI View environment. It is a Windows integrated design environment (IDE) containing all the tools and editors (based on the different IEC 61131 languages) that users need during the entire life cycle of the machine
<b>KAS Runtime</b>	Kollmorgen Automation Suite - Runtime	The KAS Runtime is the engine that provides a soft PLC and a motion controller
<b>KVB IDE</b>	Kollmorgen HMI Development Environment	Kollmorgen Visualization Builder is an editor that allows the end-user to control the KAS Runtime
<b>LD</b>	Ladder Diagram	<a href="#">FFLD</a>
<b>LSB</b>	Least Significant Bit	Sometimes abbreviated as LSB, the least significant bit is the lowest bit in a series of numbers in binary; the LSB is located at the far right of a string. For example, in the binary number: 10111001, the least significant bit is the far right "1".
<b>MDI</b>	Multiple Document Interface	Graphical computer applications with an MDI are those whose windows reside under a single parent window (usually with the exception of modal windows), as opposed to all windows being separate from each other (single document interface).  <b>Advantages:</b> <ul style="list-style-type: none"> <li>- With MDI, a single menu bar and/or toolbar is shared between all child windows, reducing clutter and increasing efficient use of screen space</li> <li>- An application's child windows can be hidden/shown/minimized/maximized as a whole</li> <li>- Features such as "Tile" and "Cascade" can be implemented for the child windows</li> </ul>

Term	Definition	Description
<b>ML</b>	Motion Library	The Motion Library is the interface between the IEC61131-3 logical application and the motion engine. It gives access from IEC61131-3 to pipe and Pipe Blocks parameters and methods as well as to higher levels of functionalities such a homing, tensioning, dynamic correction, etc.
<b>MSB</b>	Most Significant Bit	Sometimes abbreviated as MSB, the most significant bit is the bit position in a binary number having the greatest value
<b>MDP</b>	Modular Device Profile	The EtherCAT Modular Device Profile defines the data structure organization for a device with subdivided substructures. The object dictionary structure, the PDO structure, and their corresponding indexes are defined by the MDP specification. This makes it possible for the EtherCAT master to support a variety of modular device types from different manufacturers. MDP supports devices with either physical plug-in modules or logical modules, enabling users to assemble scalable devices or select from a range of pre-built devices with various feature sets.
<b>NAT</b>	Network Address Translation	In computer networking, NAT is the process of modifying network address information in datagram (IP) packet headers while in transit across a traffic routing device for the purpose of remapping a given address space into another.
<b>NIC</b>	Network Interface Controller	A network interface controller (or card) is a hardware device that handles an interface to a computer network and allows a network-capable device to access that network
<b>NVRAM</b>	Non-Volatile Random Access Memory	NVRAM is the general name used to describe any type of random access memory which does not lose its information when power is turned off. This memory is in contrast to the most common forms of random access memory today, which both require continual power in order to maintain their data. NVRAM is a subgroup of the more general class of non-volatile memory types, the difference being that NVRAM devices offer random access, like hard disks. The best-known form of NVRAM memory today is flash memory
<b>OEM</b>	Original Equipment Manufacturer	A term that refers to containment-based re-branding, namely where one company uses a component of another company within its product, or sells the product of another company under its own brand. OEM refers to the company that originally manufactured the product
<b>OPC</b>	OLE for Process Control	OPC is the original name for an open standard to specify the communication of real-time plant data between control devices from different manufacturers
<b>PCI</b>	Peripheral Component Interconnect	The PCI specifies a computer bus for attaching peripheral devices to a computer motherboard

Term	Definition	Description
<b>PCMM</b>		Programmable controller which lets you control multiple EtherCAT slave drives and I/O. Essentially a AKD PDMM without an AKD.
<b>PD</b>	Programmable Drive	(Also known as Servo Amplifiers or Servo Drive) A Drive can be programmable, which means it has an open hardware and software architecture to make it ready for nearly all conceivable customer-specific modifications
<b>PDMM</b>	Programmable Drive Multi-axis Master	Programmable drive which lets you control multiple EtherCAT slave drives and I/O
<b>PDO</b>	Process Data Object	<p>PDO is a type of protocol frame used in some fieldbuses. A PDO contains one or more object dictionary entries, which define the application data transferred between devices.</p> <p>EtherCAT uses the same communication mechanisms (PDO and SDO) as CANopen. EtherCAT transfers the process data between the master and slave device cyclically. PDOs have several attributes to define their properties.</p> <ul style="list-style-type: none"> <li>• PDOs can be exclusive, meaning that no additional PDOs may be assigned to a device if an exclusive PDO is assigned.</li> <li>• If the PDO type is fixed (Fixed attribute = 1) then the PDO's content cannot be changed by users.</li> </ul>
<b>PID</b>	Proportional-Integral-Derivative	<p>A PID controller is a generic control-loop feedback mechanism widely used in industrial control systems.</p> <p>An "error" occurs when an event or a disturbance triggers off a change in the process variable.</p> <p>A PID controller attempts to correct the error between a measured process variable and a desired setpoint by calculating and then outputting a corrective action that can adjust the process accordingly</p>
<b>PLC</b>	Programmable Logic Controller	<p>A Programmable Logic Controller, PLC, or Programmable Controller is a digital computer used for automation of industrial processes, such as control of machinery on factory assembly lines.</p> <p>Used to synchronize the flow of inputs from (physical) sensors and events with the flow of outputs to actuators and events</p>
<b>PNE</b>	Pipe Network Engine	<p>The Pipe Network concept is an innovative solution to solve axis synchronization problems.</p> <p>It is based on Pipe Blocks representing the whole mechanical system by analogy</p>
<b>POU</b>	Programmable Organization Unit	An application is a list of programs. Programs are executed sequentially within the target cycle according to the order defined by the user and displayed in the Project View
<b>Qwt</b>	Qt Widgets	Qwt is a graphics extension to the Qt GUI application framework from Trolltech ASA

Term	Definition	Description
<b>RTC</b>	Real-Time Computing	RTC is the study of hardware and software systems which are subject to a "real-time constraint" (i.e., operational deadlines from event to system response)
<b>RTOS</b>	Real-Time Operating System	RTOS is a multitasking operating system intended for real-time applications
<b>S300</b>	Servostar 300 drive	<a href="#">Servo Drive</a>
<b>S700</b>	Servostar 700 drive	<a href="#">Servo Drive</a>
<b>SCADA</b>	Supervisory Control And Data Acquisition	SCADA systems are typically used to perform data collection and control at the supervisory level. Some SCADA systems only monitor without doing control, these systems are still referred to as SCADA systems
<b>SDO</b>	Service Data Object	The SDO protocol is used to read and write values across fieldbuses . The SDO data is defined by the object dictionary.  EtherCAT uses the same communication mechanisms (PDO and SDO) as CANopen. SDO data is non-cyclic and is applicable for non-deterministic data transfers.
<b>SFC</b>	Sequential Function Chart	It can be used to program processes that can be split into steps. The main components of SFC are: - Steps with associated actions - Transitions with associated logic conditions - Directed links between steps and transitions
<b>SPLC</b>	Software version of a PLC	Usually working on PC-based hardware
<b>ST</b>	Structured Text	A high-level language which is block structured and syntactically resembles Pascal
<b>TDI</b>	Tabbed Document Interface	TDI allows multiple documents to be contained within a single window, using tabs to navigate between them
<b>TMP</b>	Trapezoidal Motion Profile	This Pipe Block is a source block that frequently serves as a virtual master for a system composed of several pipes. Generally, a trapezoidal motion profile generator is used to generate a flow of values with a first derivative which produces a trapezoidal trajectory
<b>UDFB</b>	User-Defined Function Block	UDFB can be used as a sub-Function Block in another program of the application. It is described using FBD, LD, ST or IL language. Input/output parameters of a UDFB (as well as private variables) are declared in the variable editor as local variables of the UDFB
<b>UDP</b>	User Datagram Protocol	UDP is a network protocol used for the Internet. This protocol assumes that the Internet Protocol (IP) is used as the underlying protocol. This protocol provides a procedure for application programs to send messages to other programs with a minimum of protocol mechanism. The protocol is transaction oriented, and delivery and duplicate protection are not guaranteed.
<b>USB</b>	Universal Serial Bus	USB is a serial bus standard to interface devices

Term	Definition	Description
<b>UTF8</b>	Unicode Transformation Format (8-bit)	UTF-8 is a variable-length character encoding for Unicode. It is able to represent any character in the Unicode standard, yet the initial encoding of byte codes and character assignments for UTF-8 is backward-compatible with ASCII
<b>UU</b>	User Units	A coordinate value or length expressed in user units represents a coordinate value or length in the current user coordinate system. Thus, 10 user units represent a length of 10 units in the current user coordinate system.
<b>XML</b>	Extensible Markup Language	XML is a general-purpose markup language. It is classified as an extensible language because it allows its users to define their own tags
<b>VDK</b>	VisualDSP Kernel	Operating system supported by Blackfin microprocessors
<b>VLAN</b>	Virtual LAN	A VLAN is a group of hosts with a common set of requirements that communicate as if they were attached to the Broadcast domain, regardless of their physical location. A VLAN has the same attributes as a physical LAN, but it allows for end stations to be grouped together even if they are not located on the same network switch. Network reconfiguration can be performed using software instead of physically relocating devices
<b>XPe</b>	Windows XP Embedded	XPe is a componentized version of the Professional edition of Windows XP. An original equipment manufacturer is free to choose only the components needed, thereby reducing operating system footprint and also reducing attack area as compared with XP Professional. Unlike Windows CE, Microsoft's operating system for portable devices and consumer electronics, XP Embedded provides the full Windows API, and support for the full range of applications and device drivers written for Microsoft Windows. The system requirements state that XPe can run on devices with at least 32MB Compact Flash, 32MB RAM and a P-200 microprocessor
<b>WUI</b>	Web User Interface	WUI is the set of means by which people interact with a particular machine, device, computer program or other complex tool via the Web

## 18 Glossary

Terms in this Glossary are provided for informational purposes only and can describe features not included in your particular license.

Term	Definition
<b>Actuator</b>	A mechanical device for moving or controlling a mechanism or system. An actuator typically is a mechanical device which transforms an input signal (usually an electrical signal) into motion
<b>Bandwidth</b>	In computer networking, bandwidth often refers to a data rate measured in bits/s, for example, network throughput. The reason for the connection of data rate with the term bandwidth is that the limit to the data rate of a physical communication link is related to its bandwidth in hertz
<b>Black Channel</b>	A Black Channel allows transmission of safety data via the standard network line. Safety over EtherCAT frames, known as safety containers, contain safety-critical process data and additional information used to secure this data. The safety containers are transported as part of the communication's process data.
<b>Cam profiling</b>	The position of a slave axis is mathematically linked to the position of a master axis. A good example of this would be in a system where two rotating drums turn at a given ratio to each other. A more advanced case of electronic gearing is electronic camming. With electronic camming, a slave axis follows a profile which is a function of the master position. This profile need not be linear, but it must be a mathematical function
<b>CANopen</b>	CANopen is a communication protocol and device profile specification for embedded systems used in automation for fieldbuses working in real-time
<b>Caret</b>	The term caret is also sometimes used in graphical user interface terminology where it means a text insertion point indicator, frequently represented by a blinking vertical bar. In this context, it can be used interchangeably with the word <b>cursor</b> , although the latter term is often reserved for a mouse pointer
<b>Casting</b>	For Typecasting, <a href="#">Typecasting</a>
<b>Closed Loop</b>	The use of feedback devices (encoders, resolvers, interferometers, etc.) to allow a motor to position a user payload accurately.
<b>COM</b>	COM is the original name of the serial port interface. It does not only refer to physical ports, but also to virtual ports, such as ports created by bluetooth or USB-to-Serial adapters
<b>Contact</b>	A contactor is an electrically controlled switch (relay) used for switching a power circuit. A contactor is activated by a control input which is a lower voltage/current than that which the contactor is switching. Unlike a circuit breaker, a contactor is not intended to interrupt a short-circuit current
<b>Cycle Jitter</b>	The difference between the configured cycle time and the actual measured cycle time. Units in microseconds.



Term	Definition
<b>Datagram</b>	A datagram is a basic transfer unit in which the delivery arrival time and order are not guaranteed. A datagram consists of header and data areas. The source and destination addresses as well as a type field are found in the header of a datagram.
<b>DeviceNet</b>	DeviceNet is a communication protocol (based on Controller Area Network) used in the automation industry to interconnect control devices for data exchange. Typical applications are information exchange, safety devices, and large I/O control networks
<b>Drive</b>	In electrical engineering, a drive is an electronic device providing power to a servo or stepper motor, and controlling it through the current and timing in its coils
<b>Driver</b>	In computing and electronics, a driver is a software component allowing higher-level computer programs to interact with a computer hardware device. A driver typically communicates with the device through the computer bus or communications subsystem to which the hardware is connected
<b>Endian</b>	Big-endian and little-endian describe the order in which a sequence of bytes are stored in computer memory. Big-endian is an order in which the "big end" (most significant value in the sequence) is stored first (at the lowest storage address). Little-endian is an order in which the "little end" (least significant value in the sequence) is stored first  For example the decimal integer 56789652 (0x03628a94 in hexadecimal) is stored as follows: <ul style="list-style-type: none"> <li>• 0x03 0x62 0x8a 0x94 on big-endian</li> <li>• 0x94 0x8a 0x62 0x03 on little-endian</li> </ul> KAS applications can be downloaded to big-endian or little-endian processor targets
<b>Environment</b>	Environment objects are global objects that exist before the execution of the script. Typically, they are global objects of the KAS IDE that can be accessed from the script
<b>EtherCAT</b>	"Ethernet for Control Automation Technology" EtherCAT is an open, high-performance Ethernet-based fieldbus system. The development goal of EtherCAT was to apply Ethernet to automation applications which require short data update times (also called cycle times) with low communication jitter (for synchronization purposes) and low hardware costs
<b>Ethernet</b>	Ethernet is a large, diverse family of frame-based computer networking technologies that operate at many speeds for local area networks (LANs)
<b>EtherNet/IP</b>	An open industrial application layer protocol for industrial automation applications. The EtherNet/IP application layer protocol is based on the CIP layer

Term	Definition
<b>Fast Inputs</b>	The inputs are taken into account at each cycle depending on the system <b>periodicity</b> (for example each millisecond). Under certain circumstances it can be insufficient when more accuracy is needed, or if a quick response is required from the system. To fill the gap, a drive can have some Fast Input connections (generally one or two). When an event happens that triggers a Fast Input (e.g. when a sensor sends a <b>rising edge</b> ), the detection of a signal occurs faster (which can be 1000 times more accurate than the system periodicity). Then the <b>timestamp</b> associated with this input can be provided to the controller to take corrective action
<b>Feedback Device</b>	A process whereby some proportion of the output signal of a system is passed (fed back) to the input. In automation, a device coupled to each motor to provide indication of the motor's shaft angle, for use in commutating the motor and controlling its speed and position
<b>feed-forward</b>	This describes an element or pathway within a control system which passes a controlling signal from a source in the control system's external environment, often a command signal from an external operator, to a load elsewhere in its external environment
<b>Fieldbus</b>	A Fieldbus is an industrial network protocol used for distributed control (e.g. EtherCAT, CAN, Profinet). It is a way of connecting instruments in a plant design
<b>Flash Memory</b>	A Flash memory is a non-volatile computer storage chip that can be electrically erased and reprogrammed. In addition to being non-volatile, flash memory offers fast read access times, as fast as dynamic RAM, although not as fast as static RAM or ROM. Its mechanic shock resistance explain the popularity over hard disks in portable devices; so does its high durability, being able to withstand high pressure, temperature, immersion in water etc.
<b>Frame</b>	In networking dialect, a message is called a frame
<b>Front-end</b>	In software design, the front-end is the part of a software system that interacts directly with the user
<b>Homing</b>	The Homing procedure allows, based on a position measurement, to set a position offset to the motor in order to ensure it is physically at the home position. The homing offset is saved in the controller.
<b>Interrupt</b>	An interrupt is an asynchronous signal from hardware indicating the need for attention or a synchronous event in software indicating the need for a change in execution
<b>Jerk</b>	In physics, jerk is the rate of change of acceleration; more precisely, the derivative of acceleration with respect to time
<b>Latch</b>	The control word is used to activate the drive's latch status machine. The latch control word is processed independently of the EtherCAT bus cycle. The status word is used to return the drive's latch status
<b>MAC address</b>	A Media Access Control address (MAC address) is a quasi-unique identifier assigned to most network adapters or network interface cards (NICs) by the manufacturer for identification. If assigned by the manufacturer, a MAC address usually encodes the manufacturer's registered identification number

Term	Definition
<b>ModBus</b>	<p>ModBus is a serial communications protocol and is now the most commonly available means of connecting industrial electronic devices.</p> <p>ModBus is often used to connect a supervisory computer with a remote terminal unit in supervisory control and data acquisition (SCADA) systems.</p> <p>Versions of the ModBus protocol exist for serial port and Ethernet (it is widely used with TCP/IP over Ethernet)</p>
<b>Motion Bus</b>	A Motion bus is an industrial network protocol used for <b>real-time</b> distributed control (e.g. EtherCAT).
<b>Motion control</b>	Motion control is a sub-field of automation, in which the position and/or velocity of machines are controlled using some type of device such as a hydraulic pump, linear actuator, or an electric motor, generally a servo. Motion control is an important part of robotics and CNC machine tools; however, it is more complex than in the use of specialized machines, where the kinematics is usually simpler. The latter is often called General Motion Control (GMC). Motion control is widely used in the packaging, printing, textile and assembly industries
<b>Motor</b>	An actuator focused to a movement, converting electrical energy in a force or torque
<b>Non-volatile</b>	Information is stored in a specific memory to remain accessible even when the application has been powered off
<b>Online Change</b>	Applies to downloading PLC code changes while the application is running.
<b>Online Configuration Mode</b>	Applies to EtherCAT communication to the AKD drives in a special mode.
<b>Open Loop</b>	A positioning technique, typically utilizing stepper motors, in which the controller issues a sequence of commands to the motor without any absolute means of detecting if the move has in fact been made. When the load and move velocity and acceleration are appropriately defined, open loop positioning is capable of extended operation without losing steps.
<b>OpenGL</b>	OpenGL (Open Graphics Library) is a standard specification defining a cross-language, cross-platform API for writing applications that produce 2D and 3D computer graphics. The Softscope uses this API to implement graphical manipulations
<b>P-code</b>	<p>P-code machine or pseudo-code machine is a specification of a CPU whose instructions are expected to be executed in software rather than in hardware.</p> <p>Programs that have been translated to P-code are executed (interpreted) by a software program that emulates the behavior of the CPU specification</p>

Term	Definition
<b>PDO</b>	<p>PDO is a type of protocol frame used in some fieldbuses. A PDO contains one or more object dictionary entries, which define the application data transferred between devices.</p> <p>EtherCAT uses the same communication mechanisms (PDO and SDO) as CANopen. EtherCAT transfers the process data between the master and slave device cyclically. PDOs have several attributes to define their properties.</p> <ul style="list-style-type: none"> <li>• PDOs can be exclusive, meaning that no additional PDOs may be assigned to a device if an exclusive PDO is assigned.</li> <li>• If the PDO type is fixed (Fixed attribute = 1) then the PDO's content cannot be changed by users.</li> </ul>
<b>Periodicity</b>	The period of execution of a pipe is the time spent between two successive computations of set values for the same pipe. The period of execution of a pipe is specified by the PERIOD parameter of the input Pipe Block
<b>PLCopen</b>	A vendor -and product- independent worldwide association active in Industrial Control and aiming at standardizing PLC file formats based on XML
<b>Pragma</b>	<p>A compiler directive communicating additional "pragmatic" information.</p> <p>Pragmas are processed at compile time, not at run-time. They pass information to the compiler</p>
<b>Precedence</b>	<p>In arithmetic and algebra, when a number or expression is both preceded and followed by a binary operation, a rule is required for which operation must be applied first. From the earliest use of mathematical notation, multiplication took precedence over addition, whichever side of a number it appeared on. Thus <math>3 + 4 \times 5 = 5 \times 4 + 3 = 23</math>.</p> <p>To change the order of operations, we use parentheses (). Thus, if we want to force addition to precede multiplication, we write <math>(3 + 4) \times 5 = 35</math></p>
<b>Probe</b>	For Softscope -Probe, <a href="#">Softscope - Probe</a>
<b>Pulse</b>	When the step gets activated, the action is activated for a single execution, and possibly once again when the step is deactivated
<b>Reference Counting</b>	In computer science, reference counting is a technique of storing the number of references, pointers, or handles to a resource such as an object or block of memory. It is typically used as a means of deallocating objects which are no longer referenced
<b>Rising Edge</b>	A rising edge is the transition of a digital signal from low to high. It is also called positive edge
<b>Run-time</b>	In computer science, run-time (or run time) describes the operation of a computer program, the duration of its execution, from beginning to termination (compare compile time)
<b>Sensor</b>	A sensor is a type of transducer that converts one type of energy into another for various purposes including measurement or information transfer

Term	Definition
<b>Service Port</b>	UDP applications use <a href="#">datagram</a> sockets to establish host-to-host communications. An application binds a socket to its endpoint of data transmission, which is a combination of an IP address and a service port. A port is a software structure that is identified by the port number, a 16 bit integer value.
<b>Servo Drive</b>	A servo drive is a special electric amplifier used to power electric servo motors. It monitors feedback signals from the motor and continually adjusts for deviation from expected behavior
<b>Setpoint</b>	Setpoint is the target value that an automatic control system (for example a PID controller) aims to reach
<b>Softscope - Channel</b>	A Channel is used by the softscope to acquire the evolution of a variable which is plugged on it
<b>Softscope - Probe</b>	A device that uses onboard instruments to gather and relay a variety of measurement to controllers from remote locations. Probes can return their data over radio links or be physically tethered to controllers or another device, or to collect and return physical samples
<b>Softscope - Sampling</b>	To acquire the variable's evolution, samples are taken at fixed intervals. The accuracy to create the trace depends on the resolution of the acquisition. The sampling frequency must be higher than 2 times the highest frequency in the input signal. It is called the Nyquist frequency. Theoretically it is possible to reconstruct the input signal with more than 2 samples per period. In practice, 10 to 20 samples per period are recommended to be able to examine the signal thoroughly
<b>Softscope - Time-base</b>	The time-base allows you to set the speed at which all the lines for each channel are drawn, and is calibrated in milliseconds per division
<b>Softscope - Trace</b>	The trace is the resulting graph of a variable's evolution against time, with the more distant past on the left and the more recent past on the right
<b>Stepper Drive</b>	A stepper drive is a special electric amplifier used to power electric stepper motors without position feedback (open-loop). It controls the current to the various phases in a stepper motor to control the motor's position.
<b>Synchronization</b>	Combines an axis or axes group (as slave) with an axis as master so that the slave executes its path with synchronization to the progress of the master, meaning linked to a one-dimensional source for synchronization
<b>SynqNet</b>	SynqNet is a digital machine control network. Built on the 100BT physical layer, SynqNet provides a synchronous real-time connection between motion controllers, servo drives, stepper drives, I/O modules, and custom devices
<b>Tag</b>	In the HMI context, objects connected to tags can change values in a controller, and controller values can be reflected by changing object appearance in various ways. A tag has a symbolic name and can be of different data types. Tags can belong to a connected controller, be internal or belong to the system.
<b>Timestamp</b>	A timestamp is a sequence of characters denoting the date and/or time at which a certain event occurred

Term	Definition
<b>Torque</b>	Torque is the tendency of a force to rotate an object about an axis. Just as a force is a push or a pull, a torque can be thought of as a twist. The SI unit for torque is the newton metre (N.m).
<b>Typecasting</b>	In computer science, type conversion or typecasting refers to changing an entity of one data type into another. It is done to take advantage of certain features of type hierarchies. For instance, values from a more limited set, such as integers, can be stored in a more compact format and later converted to a different format enabling operations not previously possible, such as division with several decimal places' worth of accuracy. There are two types of conversion: implicit and explicit. The term for implicit type conversion is coercion. The most common form of explicit type conversion is known as casting. Explicit type conversion can also be achieved with separately defined conversion routines such as an overloaded object constructor

## 19 Licenses

### AjaxFileUpload – MPL License

See [www.mozilla.org/MPL/2.0/](http://www.mozilla.org/MPL/2.0/) for the Mozilla Public License.

### Apache log4net – Apache License

```

Apache License
Version 2.0, January 2004
http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensors" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

(a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

(b) You must cause any modified files to carry prominent notices stating that You changed the files; and

(c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

(d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those

```

notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

## bsdtar & libarchive 2 – BSD License

bsdtar.exe and libarchive2.dll for Windows

Copyright (c) 2003-2006 Tim Kientzle  
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer in this position and unchanged.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR(S) ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR(S) BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

## bzip2.dll – BSD License



```
bzip2.dll
```

```
Copyright © 1996-2007 Julian Seward
```

```
This program, bzip2, the associated library libbzip2, and all documentation, are copyright © 1996-2007 Julian Seward. All rights reserved.
```

```
Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
```

```
• Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
```

```
• The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
• Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
• The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.
```

```
THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
```

## DockPanel Suite – MIT License

```
DockPanel
```

```
The MIT License
```

```
Copyright (c) 2007 Weifen Luo (email: weifenluo@yahoo.com)
```

```
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
```

```
The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
```

```
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
```

## FileHelpers

```
-----
FileHelpers
-----
```

```
http://www.filehelpers.com - SF Home: http://sourceforge.net/projects/filehelpers
```

```
The FileHelpers are an easy to use library to import/export data from fixed length or delimited files.
```

```
If you want to start using the library go directly to the Quick Start Guide in the CHM.
```

```
-----
Who needs the File Helpers Library ?
-----
```

```
In almost every project there is a need to read/write data from/to a file of a specified format.
```

```
For example for log parsing, data warehouse and OLAP applications, communication between systems, file format transformations (for example from a fixed length to a CSV file).
```

```
This library aims to provide an easy and reliable way to accomplish this task.
```

```
-----
History
-----
```

```
Check The docs for the History (is hard to maintain two copies =)
```

```
-----
Licence
-----
```

```
FileHelpers Library is © Copyright 2005-2006 to Marcos Meli but it's source code and the binaries are free for commercial and non commercial use.
```

```
GPL license description is available here: http://www.opensource.org/licenses/lgpl-license.php
```

```
-----
Contact and Ideas
-----
```

```
-----
If you find that there is a feature that I must include, or you have a new idea (for the API, Source Code or Examples),
only let me know, sending an e-mail to marcos@filehelpers.com or entering the FileHelpers Forums at
```

```
http://www.filehelpers.com/Forums/
```

```
-----
Full Sources and Updates
-----
```

```
If you want to help in the development of the library please go to
```

```
http://sourceforge.net/project/showfiles.php?group_id=152382&package_id=169468
```

```
and download the AllInOne DevPack with the binaries of NUnit, NAnt and NDoc.
```

## jQuery Cookies

```
Copyright (c) 2005 - 2013 James Auldridge
```

```
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation
files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy,
modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the
Software is furnished to do so, subject to the following conditions:
```

```
The above copyright notice and this permission notice shall be included in all copies or substantial portions of the
Software.
```

```
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE,
ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
```

## jQuery File Tree – MIT License

```
JQueryFileTree
```

```
Copyright (c) 2008 A Beautiful Site, LLC.
```

```
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation
files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy,
modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the
Software is furnished to do so, subject to the following conditions:
```

```
The above copyright notice and this permission notice shall be included in all copies or substantial portions of the
Software.
```

```
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE,
ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
```

## JavaScript Graphics Library – GNU Lesser General Public License

```
GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007
```

```
Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.
```

```
Preamble
```

```
The GNU General Public License is a free, copyleft license for software and other kinds of works.
```

```
The licenses for most software and other practical works are designed to take away your freedom to share and change the
works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions
of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU
General Public License for most of our software; it applies also to any other work released this way by its authors. You
can apply it to your programs, too.
```

```
When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make
sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive
source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and
that you know you can do these things.
```

```
To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights.
Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it:
responsibilities to respect the freedom of others.
```

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

#### TERMS AND CONDITIONS

##### 0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

##### 1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

##### 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered

work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

### 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

### 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

### 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

### 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

#### 7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

#### 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or

rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

#### 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

#### 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

#### 11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

#### 12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

#### 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

## 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

## 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

## 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

## 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

## END OF TERMS AND CONDITIONS

## How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read

```
<http://www.gnu.org/philosophy/why-not-lgpl.html>
```

## Mongoose – MIT License

Copyright (c) 2004-2013 Sergey Lyubka

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

## MVVM Light Toolkit – MIT License

MVVM Light Toolkit

<http://mvvmlight.codeplex.com/>

License: The MIT License (MIT)

Copyright (c) 2009 - 2011 Laurent Bugnion

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

## pugixml – MIT License

Copyright (c) 2006-2012 Arseny Kapoulkine

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

## Qt – LGPL3 License

GNU LESSER GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright © 2007 Free Software Foundation, Inc.

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

This version of the GNU Lesser General Public License incorporates the terms and conditions of version 3 of the GNU General Public License, supplemented by the additional permissions listed below.

0. Additional Definitions.

As used herein, "this License" refers to version 3 of the GNU Lesser General Public License, and the "GNU GPL" refers to version 3 of the GNU General Public License.

"The Library" refers to a covered work governed by this License, other than an Application or a Combined Work as defined below.

An "Application" is any work that makes use of an interface provided by the Library, but which is not otherwise based on the Library. Defining a subclass of a class defined by the Library is deemed a mode of using an interface provided by the



Library.

A "Combined Work" is a work produced by combining or linking an Application with the Library. The particular version of the Library with which the Combined Work was made is also called the "Linked Version".

The "Minimal Corresponding Source" for a Combined Work means the Corresponding Source for the Combined Work, excluding any source code for portions of the Combined Work that, considered in isolation, are based on the Application, and not on the Linked Version.

The "Corresponding Application Code" for a Combined Work means the object code and/or source code for the Application, including any data and utility programs needed for reproducing the Combined Work from the Application, but excluding the System Libraries of the Combined Work.

1. Exception to Section 3 of the GNU GPL.

You may convey a covered work under sections 3 and 4 of this License without being bound by section 3 of the GNU GPL.

2. Conveying Modified Versions.

If you modify a copy of the Library, and, in your modifications, a facility refers to a function or data to be supplied by an Application that uses the facility (other than as an argument passed when the facility is invoked), then you may convey a copy of the modified version:

- a) under this License, provided that you make a good faith effort to ensure that, in the event an Application does not supply the function or data, the facility still operates, and performs whatever part of its purpose remains meaningful, or
- b) under the GNU GPL, with none of the additional permissions of this License applicable to that copy.

3. Object Code Incorporating Material from Library Header Files.

The object code form of an Application may incorporate material from a header file that is part of the Library. You may convey such object code under terms of your choice, provided that, if the incorporated material is not limited to numerical parameters, data structure layouts and accessors, or small macros, inline functions and templates (ten or fewer lines in length), you do both of the following:

- a) Give prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are covered by this License.
- b) Accompany the object code with a copy of the GNU GPL and this license document.

4. Combined Works.

You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications, if you also do each of the following:

- a) Give prominent notice with each copy of the Combined Work that the Library is used in it and that the Library and its use are covered by this License.
- b) Accompany the Combined Work with a copy of the GNU GPL and this license document.
- c) For a Combined Work that displays copyright notices during execution, include the copyright notice for the Library among these notices, as well as a reference directing the user to the copies of the GNU GPL and this license document.
- d) Do one of the following:
  - 0) Convey the Minimal Corresponding Source under the terms of this License, and the Corresponding Application Code in a form suitable for, and under terms that permit, the user to recombine or relink the Application with a modified version of the Linked Version to produce a modified Combined Work, in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.
  - 1) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (a) uses at run time a copy of the Library already present on the user's computer system, and (b) will operate properly with a modified version of the Library that is interface-compatible with the Linked Version.
- e) Provide Installation Information, but only if you would otherwise be required to provide such information under section 6 of the GNU GPL, and only to the extent that such information is necessary to install and execute a modified version of the Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version. (If you use option 4d0, the Installation Information must accompany the Minimal Corresponding Source and Corresponding Application Code. If you use option 4d1, you must provide the Installation Information in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.)

5. Combined Libraries.

You may place library facilities that are a work based on the Library side by side in a single library together with other library facilities that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities, conveyed under the terms of this License.
- b) Give prominent notice with the combined library that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

6. Revised Versions of the GNU Lesser General Public License.

The Free Software Foundation may publish revised and/or new versions of the GNU Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library as you received it specifies that a certain numbered version of the GNU Lesser General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.

If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.

## Qwt – GNU Lesser General Public License

GNU GENERAL PUBLIC LICENSE  
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>  
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

## 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

## 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

## 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

## 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

## 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

#### 7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but

permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

#### 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

#### 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

#### 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

#### 11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of

distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

#### 12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

#### 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

#### 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

#### 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

#### 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

#### 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

#### How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read

```
<http://www.gnu.org/philosophy/why-not-lgpl.html>.
```

## ZedGraph – LGPL License

For easy distribution, this archive provides the dll form of ZedGraph only. This distribution of the ZedGraph source code includes two versions:

Version 4.6.x is the .Net 1.1 compatible version  
Version 5.1.x is the .Net 2.0 compatible version

Complete source code for ZedGraph is available on sourceforge here:

```
http://sourceforge.net/projects/zedgraph/
```

A wiki providing help, samples, etc. is available here:

```
http://zedgraph.org
```

A tutorial on using ZedGraph is available here:

```
http://www.codeproject.com/csharp/zedgraph.asp
```

\*\*\*\*\*

GNU LESSER GENERAL PUBLIC LICENSE  
Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.  
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA  
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

#### GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for and what the program that uses the Library does).

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent



of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or inked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler,

kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this license and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

#### NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

## How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library `Frob' (a library for tweaking knobs) written by James Random Hacker.

```
<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice
```

That's all there is to it!

## Zlib1.dll – BSD License

zlib1.dll

Copyright © 1995-2004 Jean-loup Gailly and Mark Adler

This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

```
Jean-loup Gailly      Mark Adler
jloup@gzip.org       madler@alumni.caltech.edu
```

If you use the zlib library in a product, we would appreciate \*not\* receiving lengthy legal documents to sign. The sources are provided for free but without warranty of any kind. The library has been entirely written by Jean-loup Gailly and Mark Adler; it does not include third-party code.

If you redistribute modified sources, we would appreciate that you include in the file ChangeLog history information documenting your changes. Please read the FAQ for more information on the distribution of modified source versions.

## 20 Index

<b>#</b>	
<b>#ifdef</b> .....	210
<b>@</b>	
<b>@</b> .....	297
<b>1</b>	
<b>16#</b> .....	358
<b>2</b>	
<b>2#</b> .....	358
<b>2nd feedback</b> .....	761
<b>8</b>	
<b>8#</b> .....	358
<b>A</b>	
<b>About Window</b> .....	116
<b>acceleration</b> .....	529
<b>accelerator keys</b> .....	256, 258
<b>acronyms</b> .....	784
<b>actual position</b> .....	421, 469
pipe network .....	388
<b>adding</b>	
controller .....	118
drive .....	175
I/O .....	186-187
<b>address</b>	
IP address .....	119
<b>AKD drive</b>	
configure .....	176
creation .....	175
GUI .....	108
offline .....	460
online .....	460
setup wizard .....	178
toolbar .....	108
workbench .....	108
<b>AKD2G, digitizing axis</b> .....	155
<b>AKT-ECT-000</b>	
troubleshoot .....	770
<b>alias</b> .....	309, 373
<b>ALS format</b> .....	242
<b>animation</b> .....	71, 231, 234, 610
PLC cycle .....	464
UDFB .....	231
<b>architecture</b> .....	31
<b>auto</b> .....	243
completion .....	266, 268, 287

discovery .....	323
recovery .....	242, 247, 755
scan .....	323
start .....	471
autocompletion .....	266, 268, 287
AutoSave .....	247, 755
autostart .....	471
Axes Group .....	159, 164, 582
Axes Group, new .....	58, 159
Axis Velocity; velocity, axis .....	420
AXIS_GROUP_REF .....	58, 159
AXIS_NUM .....	153
AXIS_REF .....	407
<b>B</b>	
Backup Controller .....	507
Backup EtherCAT Devices .....	508
Backup, file .....	247
bandwidth .....	325
BISS .....	761
bookmarks .....	264
BOOL .....	364
boot	
PDMM .....	473
bootstrap .....	446
Brake Output .....	181
breakpoint .....	99, 220, 222
remove .....	222
set .....	222
buffer	
pipe network .....	394
PLCopen mode .....	400
button	
online change .....	611
softscope .....	535
BYTE .....	364
<b>C</b>	
C code .....	210
cam	
cam profile .....	56, 198
format .....	531
Cam on the fly .....	642
cam profile .....	642
transformation .....	199
CAN object .....	446
CANopen .....	441, 455
CANopen object .....	446
Capture Engine .....	378
cascade .....	248
case sensitive .....	92
change	
online change .....	607
channel .....	533
child SFC .....	370
circular file .....	87
Circular Move, precision requirements; Precision, Circular Move Input Requirements .....	587
clock synchronism .....	439, 614

<b>code</b>	
C code	210
color code	110
machine code	210
Normal	209, 610
Optimized	210, 610
P-code	209
<b>CoE</b>	<b>438</b>
<b>coil</b>	<b>296</b>
<b>cold start</b>	<b>214</b>
<b>collapse</b>	
FFLD network	294
<b>color</b>	
green	108, 110, 294, 523
grey	108
orange	110, 215
red	79, 108-109, 196, 229, 282, 515
<b>command position</b>	<b>421</b>
<b>comment</b>	
FBD	281
FFLD	295
pipe network	146
<b>Comment</b>	
FFLD	295
<b>Comment, FFLD</b>	<b>258</b>
<b>compare</b>	
PLC programs	612
<b>Compare Projects</b>	<b>243</b>
<b>compatibility</b>	<b>242</b>
<b>Compile</b>	<b>212</b>
<b>compiling mode</b>	
debug	209
release	209
<b>completion</b>	<b>266, 268, 287</b>
<b>conditional compiling</b>	<b>210</b>
<b>configure</b>	
AKD drive	176
EtherCAT XML	645
IO	312
<b>Configure via CoE Init-Commands</b>	<b>183</b>
<b>constant</b>	<b>309</b>
<b>constant expression</b>	<b>364</b>
<b>contact</b>	
ffld	296
<b>control word</b>	<b>458</b>
<b>controller</b>	
creation	118
log	85, 87
version	110
<b>Controller Performance, tip</b>	<b>660</b>
<b>Controller Properties</b>	<b>119, 216</b>
<b>controller; backup</b>	<b>507</b>
<b>Controller; restore</b>	<b>509</b>
<b>convention</b>	
variable naming	300
Windows standard	254
<b>coordinate system</b>	
unified	418
<b>copy</b>	<b>754</b>
<b>copyrights</b>	<b>2</b>

<b>coupler</b> .....	<b>750, 781</b>
<b>CPU load</b> .....	<b>547, 757, 764</b>
<b>creation</b>	
AKD drive .....	175
controller .....	118
controller wizard .....	118
pipe network .....	143
program .....	265
structure .....	302
variable .....	300, 304
<b>csv</b> .....	<b>531</b>
<b>curve</b>	
cam profile editor .....	530
softscope .....	533
<b>custom library</b> .....	<b>194</b>
online change .....	608
<b>cycle</b>	
animation .....	464
cycle time .....	325
motion .....	325
PLC .....	204
<b>Cycle Time</b> .....	<b>39, 54, 120, 658</b>
<b>CycleJitter</b> .....	<b>538, 549, 757</b>
<b>cyclic</b> .....	<b>520</b>
<b>D</b>	
<b>dashed line</b> .....	<b>469</b>
<b>data structure</b> .....	<b>407</b>
<b>data types</b> .....	<b>357</b>
<b>DC</b> .....	<b>439, 614</b>
<b>debug</b>	
PLC application .....	209
softscope .....	225
step-by-step .....	218
<b>defines</b> .....	<b>373</b>
<b>Derivation Order</b> .....	<b>541</b>
<b>device</b>	
EtherCAT .....	321
<b>diagnostics, ethercat</b> .....	<b>765</b>
<b>dictionary</b> .....	<b>67, 231</b>
<b>difference</b> .....	<b>612</b>
<b>digitizing axis</b> .....	<b>154, 450, 453, 640</b>
<b>Digitizing axis</b> .....	<b>636</b>
<b>DINT</b> .....	<b>364</b>
<b>directive</b> .....	<b>210</b>
compiler directive .....	210
<b>disclaimer</b> .....	<b>2</b>
<b>discovery</b> .....	<b>323</b>
<b>distributed clocks</b> .....	<b>439, 614</b>
<b>docking windows</b> .....	<b>239</b>
<b>dotted line</b> .....	<b>469</b>
<b>download</b>	
drive firmware .....	323
HMI .....	559
<b>drag and drop</b> .....	<b>266-267, 289, 754</b>
<b>drive</b>	
AKD creation .....	175
AKD GUI .....	108
AKD offline .....	460

AKD online .....	460
AKD setup wizard .....	178
configure .....	176
<b>duration</b>	
cycle .....	464
online change .....	607
<b>DWORD</b> .....	<b>364</b>
<b>E</b>	
<b>editor</b>	
cam profile .....	518
FBD .....	278
FFLD .....	291
HMI .....	554
IL .....	286
pipe network .....	514
SFC .....	272
ST .....	286
variable .....	74
<b>ENI File</b> .....	<b>444</b>
<b>enum,enumerated type</b> .....	<b>81</b>
<b>enumeration order</b> .....	<b>279</b>
<b>error</b>	
EtherCAT error management .....	765
EtherCAT error message .....	765
PLCopen errorID .....	423
<b>error handling</b>	
pipe network .....	632
<b>ESI file</b> .....	<b>125-126, 134, 324, 327, 332, 762</b>
<b>ESI File</b> .....	<b>444</b>
<b>ETG.5001 standard</b> .....	<b>331</b>
<b>EtherCAT</b>	
distributed clocks .....	439, 614
error management .....	765
error message .....	765
FoE .....	442
frame .....	325, 436
image .....	436, 645
master .....	437, 442
modes .....	446
online configuration mode .....	178, 252, 461, 738, 756
PDO .....	122, 441, 446
Process Image .....	436
profile .....	441
SDO .....	441
slave .....	122, 437, 442
status bootstrap .....	446
status operational .....	446
status preop .....	446
status safeop .....	446
topology .....	438
unsupported device .....	645
<b>EtherCAT Devices; backup</b> .....	<b>508</b>
<b>EtherCAT Devices; restore</b> .....	<b>509</b>
<b>EtherCAT Processing Unit Error Counter</b> .....	<b>765</b>
<b>execution order</b> .....	<b>279</b>
<b>expand</b>	
FFLD network .....	294



<b>export</b>	
program .....	55
softscope data .....	545
<b>F</b>	
<b>FailSafe over EtherCAT</b> .....	<b>328</b>
<b>faq</b> .....	<b>753</b>
<b>Fast input</b> .....	<b>378</b>
<b>Fault messages</b> .....	<b>108</b>
<b>FBD</b>	
editor .....	278
insert graphic .....	281
<b>feed-forward</b>	
torque- .....	615, 761
<b>feedback</b> .....	<b>761</b>
secondary .....	761
<b>feedback position</b> .....	<b>388, 533</b>
<b>FFLD</b>	
editor .....	291
Limitations .....	370
<b>filtering</b> .....	<b>91</b>
<b>find</b> .....	<b>92, 95</b>
case sensitive .....	92
find and replace .....	92
find next .....	95
find unused .....	93
<b>firewall</b> .....	<b>756</b>
<b>firmware</b>	
download protocol .....	442, 476
drive download .....	323
drive upgrade .....	323
PDMM upgrade .....	501
<b>FoE</b> .....	<b>442</b>
<b>forcing variable</b> .....	<b>106, 232</b>
<b>format</b>	
ALS project .....	242
KAS project .....	242
<b>Forwarded Rx Error Counter</b> .....	<b>765</b>
<b>frame EtherCAT</b> .....	<b>325, 436</b>
<b>Frame Loss</b> .....	<b>766</b>
<b>FSoE</b> .....	<b>328, 647, 772</b>
<b>FSoE Address</b> .....	<b>334</b>
<b>function</b> .....	<b>367</b>
<b>function block</b> .....	<b>367</b>
<b>G</b>	
<b>generator position</b> .....	<b>388</b>
<b>getting started</b> .....	<b>24, 779</b>
<b>global constant</b> .....	<b>310</b>
<b>glossary</b> .....	<b>792</b>
<b>green</b>	
AKD enable status .....	108
background .....	110
dashed rectangle .....	523
FFLD network header .....	294
<b>grey</b>	
AKD enable status .....	108
<b>grid</b> .....	<b>514</b>

<b>grid unit</b> .....	<b>514</b>
<b>GUI</b>	
AKD drive .....	108
KAS .....	49
<b>guideline</b>	
PLC program .....	369
project structure .....	691
setting units .....	146
<b>H</b>	
<b>hexadecimal</b> .....	<b>358, 364</b>
<b>HMI</b>	
add device .....	170
download .....	559
<b>Holding Registers</b> .....	<b>660</b>
<b>homing</b> .....	<b>391, 616</b>
<b>how to</b>	
list of- .....	778
<b>I</b>	
<b>I/O</b>	
adding I/O .....	186-187
configure .....	312
local .....	179, 489
mapping I/O .....	312
onboard .....	179
PDMM onboard .....	489
unmapping I/O .....	314
<b>I/O terminal</b>	
coupler .....	750, 781
isolation .....	750, 781
module .....	750, 781
stepper .....	750, 782
thermocouple .....	749, 781
<b>icon</b>	
controller toolbar .....	214, 251
debug toolbar .....	253
device toolbar .....	214, 251
ethercat toolbar .....	252
FBD editor .....	279
FFLD editor .....	295
main toolbar .....	250
online change toolbar .....	252
SFC editor .....	273
softscope .....	537
watch window .....	102
<b>ide</b> .....	<b>30</b>
<b>ifdef</b> .....	<b>210</b>
<b>IL</b>	
editor .....	286
<b>image EtherCAT</b> .....	<b>436, 645</b>
<b>import</b>	
import program .....	55
<b>initialization</b>	
motion .....	394
<b>input parameter</b> .....	<b>307</b>
<b>Input Registers</b> .....	<b>660</b>
<b>installation</b> .....	<b>24, 779</b>

<b>instance</b> .....	<b>115, 753</b>
<b>INT</b> .....	<b>364</b>
<b>intellisense</b> .....	<b>287</b>
<b>interpolation</b> .....	<b>521</b>
<b>IO</b>	
adding IO .....	186-187
configure .....	312
local .....	179, 489
mapping IO .....	312
onboard .....	179
PDMM onboard .....	489
unmapping IO .....	314
<b>IO terminal</b>	
coupler .....	750, 781
isolation .....	750, 781
module .....	750, 781
stepper .....	750, 782
thermocouple .....	749, 781
<b>IP address</b> .....	<b>54, 119, 503</b>
<b>isolation</b> .....	<b>750, 781</b>
<b>J</b>	
<b>jerk</b> .....	<b>402, 529</b>
<b>K</b>	
<b>KAS format</b> .....	<b>242</b>
<b>KDN</b> .....	<b>119, 702</b>
<b>KVB</b> .....	<b>554</b>
<b>L</b>	
<b>latch</b> .....	<b>296, 439</b>
<b>latency</b> .....	<b>629</b>
<b>level</b> .....	<b>87, 496</b>
<b>library</b>	
custom library .....	194
toolbox .....	66
<b>lifetime</b> .....	<b>392</b>
<b>limitations</b>	
acceleration .....	406
animation .....	231
breakpoint in SFC .....	222
breakpoint with online change .....	221
EtherCAT in Op state .....	215, 462
HMI variable mapping .....	171
intellisense .....	287
jerk .....	406
onboard IO .....	179
online change .....	608
online detection .....	324
PDMM onboard IO .....	489
PLC program .....	370
print preview .....	237
project files .....	235
replace .....	93
scan device .....	324
search and replace .....	93
SFC breakpoint .....	222

softscope .....	69, 225, 540
structure .....	358
UDFB .....	308
<b>line</b>	
dashed line .....	469
dotted line .....	469
normal line .....	469
solid line .....	469
<b>Link Loss/Device Fault</b> .....	<b>766</b>
<b>Link Lost Counter</b> .....	<b>765</b>
<b>LINT</b> .....	<b>364</b>
<b>list of</b>	
how to .....	778
literature .....	778
<b>literature</b>	
list of- .....	778
<b>local constant</b> .....	<b>311</b>
<b>local I/O</b> .....	<b>179, 489</b>
<b>local logs</b> .....	<b>85, 87</b>
<b>location</b>	
find and replace .....	93
library .....	194
project .....	118
<b>locking variable</b> .....	<b>106, 233</b>
<b>log</b>	
circular file .....	87
controller log .....	85, 87
filtering .....	91
level .....	87, 496
local logs .....	85, 87
log file .....	88
scrolling .....	91
source .....	87, 496
timestamp .....	85, 498
<b>LREAL</b> .....	<b>365</b>
<b>LREAL, data type</b> .....	<b>358</b>
<b>M</b>	
<b>machine code</b> .....	<b>210</b>
<b>manuals</b> .....	<b>24, 554, 751</b>
<b>mapping</b>	
HMI variable .....	559
I/O .....	312
onboard I/O .....	179
PDMM onboard I/O .....	489
<b>master</b> .....	<b>437, 442</b>
<b>MDI</b> .....	<b>248</b>
<b>MDP</b> .....	<b>324, 788</b>
<b>message</b>	
circular file .....	87
filtering .....	91
level .....	87, 496
local logs .....	85
log file .....	88
scrolling .....	91
source .....	87, 496
timestamp .....	85, 498
<b>MLPN_ACTIVATE</b> .....	<b>147</b>
<b>MLPN_CONNECT</b> .....	<b>147</b>

<b>MLPN_CREATE_OBJECTS</b> .....	148
<b>MLPN_DEACTIVATE</b> .....	148
<b>MLPN_POWER_OFF</b> .....	148
<b>MLPN_POWER_ON</b> .....	147
<b>MLPR_CREATE_PROFILES</b> .....	148, 160
<b>modbus</b> .....	172
<b>Modbus</b>	
master, multiple .....	658
<b>module</b> .....	750, 781
<b>modulo</b>	
Modulo Period .....	541
<b>motion</b>	
initialization .....	148, 160, 394
profile .....	380
restart .....	632
start .....	148, 160, 394
<b>MotionExecTime</b> .....	538, 549, 764
<b>multi-dimension</b> .....	301
<b>multiple</b> .....	115, 753
<b>N</b>	
<b>N</b>	
SFC step .....	276
<b>Name, node</b> .....	52
<b>New Axis</b> .....	58
<b>New Program dialog</b> .....	306
<b>node name</b> .....	52
<b>Normal code</b> .....	209, 610
<b>normal line</b> .....	469
<b>NVRAM</b> .....	746
AKD parameter .....	323
calculate space .....	746
<b>O</b>	
<b>Object Index</b> .....	336
<b>Object Name</b> .....	336
<b>octal</b> .....	358, 364
<b>offline</b>	
AKD drive .....	460
<b>onboard I/O</b> .....	179
<b>online</b>	
AKD drive .....	460
<b>online change</b> .....	607
breakpoint .....	221
difference .....	612
duration .....	607
revert .....	611
<b>online config</b> .....	252
<b>online configuration mode</b> .....	178, 252, 461, 738, 756
<b>Op</b> .....	446
<b>open from controller</b> .....	243
<b>Open Loop with Encoder</b> .....	183
<b>Open Loop with No Encoder</b> .....	183
<b>Optimized code</b> .....	210, 610
<b>option</b>	
PLC .....	209
<b>orange</b>	
background .....	110, 215

order in FBD .....	279
ordering variables .....	70
oscilloscope .....	533
output parameter .....	307
overload	
CPU .....	757
<b>P</b>	
P-code .....	209
P0	
SFC step .....	276
P1	
SFC step .....	276
Pack bits .....	676
panning	
cam profile editor .....	524
parameter	
input .....	307
output .....	307
Parameters dialog .....	306
password .....	513
paste .....	754
PDF .....	24, 554, 751
PDMM onboard I/O .....	489
PDO .....	122, 441, 446
PDO configuration .....	335
PDO Index .....	336
period .....	204
periodic .....	520
phase .....	204
Physical Link status .....	765
PID .....	376
pipe position .....	388
PLC	
cycle .....	204
option .....	209
PLC options dialog .....	251
PLCopen	
introduction .....	398
queuing .....	400
S-curve .....	402
Trapezoidal .....	404
PLCProgExecTime .....	538, 549, 764
plugging a probe .....	540
port .....	757, 777
Port 80 .....	777
ports, used by KAS .....	777
position	
actual position .....	388, 469
feedback position .....	388, 533
generator position .....	388
pipe position .....	388
reference position .....	388, 469
Position	
Actual .....	421
Command .....	421
Zero Offset .....	388
pou .....	367
power rail .....	280, 296

pragma	210, 294
pre-op	446
precision	309
preview	237
print	236
preview	237
project	237
setup	236
printed material	24
priority	204
private variable	307
probe	540
profile	
cam profile	56, 198
EtherCAT	441
motion profile	380
program	367
Program	
Limitations	370
Program Properties	306
proxy	756
pulse	
online change	608
<b>Q</b>	
queuing	400
quick start	25, 779
<b>R</b>	
read only	71
REAL	365
REAL, data type	358
RealTimeMargin	539, 549, 764
recovery	242, 247, 755
red	
AKD enable status	108
background	109
difference	229
line	282, 515
text	79, 196
reference manual	25, 779
reference position	388, 469
Registers, Holding	660
Registers, Input	660
registration	623
regulation	376
release	209
remote I/O	187, 749
remote version	110
replace all	95
replace next	95
Reset IDE Layout	249
restart	
motion	632
Restore Controller	509
Restore EtherCAT Devices	509
retain variable	359, 361
calculate space	746

starting application .....	214
variable editor .....	300, 304
<b>revert online change</b> .....	<b>611</b>
<b>RMS</b> .....	<b>184</b>
<b>root</b> .....	<b>513</b>
<b>root mean square</b> .....	<b>184</b>
<b>Rotary Switch</b>	
PDMM .....	503
<b>runtime</b> .....	<b>31</b>
<b>Rx Error Counter</b> .....	<b>765</b>
<b>Rx Invalid Frame Counter</b> .....	<b>765</b>
<b>RxPDO</b> .....	<b>447</b>

## S

<b>S-curve</b> .....	<b>402</b>
<b>S300</b> .....	<b>751, 782</b>
<b>S700</b> .....	<b>751, 782</b>
<b>safe-op</b> .....	<b>446</b>
<b>SafePLC2</b> .....	<b>53</b>
<b>Safety</b> .....	<b>647</b>
<b>save</b> .....	<b>243</b>
<b>scan</b> .....	<b>323</b>
<b>scope</b> .....	<b>225, 533</b>
<b>scrolling</b> .....	<b>91</b>
<b>SCU-1-EC</b> .....	<b>747</b>
<b>SDO</b> .....	<b>441</b>
AKD capture engine .....	619
update rate .....	759
<b>secondary feedback</b> .....	<b>761</b>
<b>servo axis</b> .....	<b>154</b>
<b>set number of input</b> .....	<b>193</b>
<b>setup</b>	
print .....	236
<b>SFC</b>	
breakpoint .....	222
child .....	370
editor .....	272
Limitations .....	370
Program size .....	370
timeout .....	761
when using SFC .....	265
<b>shortcut</b> .....	<b>256</b>
FBD .....	257
FFLD .....	258
graphic .....	263
SFC .....	261
ST .....	262
table .....	263
<b>simulation</b>	
EtherCAT slave .....	322
<b>simulator</b> .....	<b>32</b>
<b>SINT</b> .....	<b>364</b>
<b>slave</b> .....	<b>122, 437, 442</b>
<b>softscope</b> .....	<b>225, 533</b>
<b>solid line</b> .....	<b>469</b>
<b>sort variables</b> .....	<b>70</b>
<b>source</b> .....	<b>87, 496</b>
<b>source code</b> .....	<b>147</b>
<b>Spacebar</b> .....	<b>70, 75, 254</b>



<b>splitter</b>	
cam profile editor .....	519
softscope .....	533
<b>ST</b>	
editor .....	286
<b>start</b>	
motion .....	394
<b>Start</b> .....	<b>115, 753</b>
<b>state machine</b>	
application structure .....	694
CANopen .....	455
EtherCAT .....	446
online change .....	609
pipe network .....	148, 160, 393
PLCopen .....	430
<b>status bar</b> .....	<b>109, 217</b>
<b>status word</b> .....	<b>459</b>
<b>step-by-step debugging</b> .....	<b>218</b>
<b>stepper</b> .....	<b>750, 782</b>
<b>STRING</b> .....	<b>365</b>
<b>structure</b> .....	<b>302, 358</b>
creation .....	302
<b>subprogram</b> .....	<b>369</b>
<b>synchronization</b> .....	<b>439, 614</b>
<b>syntax</b>	
conditional compiling .....	210
edit variable .....	80
ST coloring .....	286
<b>system requirements</b> .....	<b>23</b>
<b>T</b>	
<b>tables</b> .....	<b>778</b>
<b>tag</b>	
IO mapping .....	645
<b>tasking</b> .....	<b>463</b>
<b>technical reference</b> .....	<b>25, 779</b>
<b>Temperature Limit</b> .....	<b>744-745</b>
<b>template</b> .....	<b>118-119, 702</b>
2 axes templates .....	702
select template .....	118
<b>Templates; Project Templates</b> .....	<b>702</b>
<b>thermocouple</b> .....	<b>749, 781</b>
<b>tile</b> .....	<b>248</b>
<b>TIME</b> .....	<b>365</b>
<b>time-frame</b> .....	<b>537</b>
<b>Time Scale Offset</b> .....	<b>537</b>
<b>timeout</b>	
SFC .....	761
<b>timestamp</b> .....	<b>85, 215, 498, 614</b>
<b>toolbar</b>	
AKD drive .....	108
FBD .....	279
FFLD .....	294
IDE .....	241
SFC .....	273
<b>toolbox</b> .....	<b>50</b>
<b>tooltip</b> .....	<b>217, 234, 266, 269</b>
<b>topology</b>	
discovery .....	124

EtherCAT .....	438
<b>torque feed-forward</b> .....	<b>615, 761</b>
scaling .....	761
<b>Touch Probe</b> .....	<b>378</b>
<b>TraceTimes</b> .....	<b>538</b>
<b>track selection</b> .....	<b>69</b>
<b>trademarks</b> .....	<b>2</b>
<b>troubleshooting</b> .....	<b>85, 498</b>
<b>tunnel</b> .....	<b>777</b>
<b>TxPDO</b> .....	<b>447</b>

## U

<b>UDFB</b> .....	<b>370</b>
animation .....	231
<b>UDINT</b> .....	<b>364</b>
<b>UDP</b> .....	<b>790</b>
<b>UINT</b> .....	<b>364</b>
<b>ULINT</b> .....	<b>364</b>
<b>undocking windows</b> .....	<b>239</b>
<b>unified coordinate system</b> .....	<b>418</b>
<b>unit per ulision</b> .....	<b>544</b>
<b>units</b> .....	<b>177</b>
<b>unmap</b>	
I/O .....	314
<b>unsupported EtherCAT device</b> .....	<b>645</b>
<b>unused</b>	
find variable .....	93
<b>upgrade</b>	
drive firmware .....	323
<b>user manual</b> .....	<b>25, 779</b>
<b>User Units</b> .....	<b>146, 157, 389</b>
<b>USINT</b> .....	<b>364</b>

## V

<b>variable</b> .....	<b>67, 74, 312, 360</b>
animation .....	231
create structure .....	302
creation .....	300, 304
dictionary .....	71, 73
FBD .....	280
forcing .....	106, 232
locking .....	106, 233
mapping I/O .....	312
mapping onboard I/O .....	179
mapping PDMM onboard I/O .....	489
monitoring .....	231
naming convention .....	300
plugging a probe .....	540
sorting .....	70
unmapping I/O .....	314
<b>variable selector</b>	
map I/O .....	317
<b>velocity</b> .....	<b>523, 529</b>
<b>Velocity compensation</b> .....	<b>421</b>
<b>Velocity Compensation Factor</b> .....	<b>421</b>
<b>Velocity Compensation Filter</b> .....	<b>421</b>
<b>versinfo.xml</b> .....	<b>120</b>
<b>Virtual Axis</b> .....	<b>154</b>

virtual machine .....	35, 55
Virtual machine .....	777
VPN .....	777
<b>W</b>	
warm start .....	214, 494
watch window .....	102
web server	
change IP address .....	503
upgrade firmware .....	501
Webserver	
log in; log in	
fail .....	757
window	
cascade .....	248
MDI .....	248
tile .....	248
wizard	
AKD setup .....	178
controller .....	118
WORD .....	364
Working Counter .....	767
Wrong/Missing Device .....	766
<b>X</b>	
X32 .....	170
X32, port	
X32 .....	644
XML	
configuration file .....	326
EtherCAT config .....	645
<b>Z</b>	
Zero Offset	
Pipe Network .....	388
zoom	
FBD .....	283
FFLD .....	297, 299
SFC .....	275
softscope .....	545

## About KOLLMORGEN

Kollmorgen is a leading provider of motion systems and components for machine builders. Through world-class knowledge in motion, industry-leading quality and deep expertise in linking and integrating standard and custom products, Kollmorgen delivers breakthrough solutions that are unmatched in performance, reliability and ease-of-use, giving machine builders an irrefutable marketplace advantage.



Join the [Kollmorgen Developer Network](#) for product support. Ask the community questions, search the knowledge base for answers, get downloads, and suggest improvements.

### North America KOLLMORGEN

201 West Rock Road  
Radford, VA 24141, USA

**Web:** [www.kollmorgen.com](http://www.kollmorgen.com)  
**Mail:** [support@kollmorgen.com](mailto:support@kollmorgen.com)  
**Tel.:** +1 - 540 - 633 - 3545  
**Fax:** +1 - 540 - 639 - 4162

### Europe

**KOLLMORGEN Europe GmbH**  
Pempelfurtstr. 1  
40880 Ratingen, Germany

**Web:** [www.kollmorgen.com](http://www.kollmorgen.com)  
**Mail:** [technik@kollmorgen.com](mailto:technik@kollmorgen.com)  
**Tel.:** +49 - 2102 - 9394 - 0  
**Fax:** +49 - 2102 - 9394 - 3155

### South America KOLLMORGEN

Avenida João Paulo Ablas, 2970  
Jardim da Glória, Cotia - SP  
CEP 06711-250, Brazil

**Web:** [www.kollmorgen.com](http://www.kollmorgen.com)  
**Mail:** [contato@kollmorgen.com](mailto:contato@kollmorgen.com)  
**Tel.:** +55 11 4615-6300

### China and SEA KOLLMORGEN

Room 302, Building 5, Lihpao Plaza,  
88 Shenbin Road, Minhang District,  
Shanghai, China.

**Web:** [www.kollmorgen.cn](http://www.kollmorgen.cn)  
**Mail:** [sales.china@kollmorgen.com](mailto:sales.china@kollmorgen.com)  
**Tel.:** +86 - 400 668 2802  
**Fax:** +86 - 21 6248 5367