

Totally Integrated Automation Portal								
AKD_PN_TG7_v01 [FB1]								
AKD_PN_TG7_v01 Properties								
General								
Name	AKD_PN_TG7_v01	Number	1	Type	FB	Language	SCL	
Numbering	automatic							
Information								
Title	AKD_PN_TG7_v01	Author	jcoleman	Comment	This function block supports Profinet communication with the AKD drive using Telegram 7. It is written in TIA V13.	Family	TG7	
Version	1.0	User-defined ID						
Name	Data type	Offset	Default value	Accessible from HMI	Visible in HMI	Setpoint	Comment	
▼ Input								
iID	HW_IO	0.0	16#0	False	False	False	Hardware identifier	
iEStop	Bool	2.0	false	False	False	False	immediate quick stop and disable	
iReset	Bool	2.1	false	False	False	False	reset drive faults	
iSW_Enable	Bool	2.2	false	False	False	False	enable/disable	
iStop	Bool	2.3	false	False	False	False	stop all movement	
iMT_Pause	Bool	2.4	false	False	False	False	pause active motion task	
iStart_Homing	Bool	2.5	false	False	False	False	start homing	
iStart_Jog1	Bool	2.6	false	False	False	False		
iStart_Jog2	Bool	2.7	false	False	False	False	start jog	
iMTNum	Word	4.0	16#0	False	False	False	motion task number 0x8000 = direct motion task	
iStart_Move	Bool	6.0	false	False	False	False	start motion task	
▼ Output								
oStatus	DWord	8.0	16#0	False	False	False	ZSW1	
oEnabled	Bool	12.0	false	False	False	False	Drive power on	
oError	Bool	12.1	false	False	False	False	fault in function block or communication	
oMoving	Bool	12.2	false	False	False	False	axis enabled and moving	
oHomed	Bool	12.3	false	False	False	False	axis homed	
oInPos	Bool	12.4	false	False	False	False	axis within in-position window	
oFault	Bool	12.5	false	False	False	False	drive fault	
oWarning	Bool	12.6	false	False	False	False	drive warning	
InOut								
▼ Static								
▼ TG7	"UDT_AKD_TG7"	14.0		False	False	False		
▼ Send	Struct	0.0		False	False	False		
STW1	Int	0.0	0	False	False	False	Control word 1	
SATZANW	Int	2.0	0	False	False	False	Motion task selection	
▼ Receive	Struct	4.0		False	False	False		
ZSW1	Int	0.0	0	False	False	False	Status word 1	
AKTSATZ	Int	2.0	0	False	False	False	Actual motion task selected	
wPNReadStatus	Word	22.0	16#0	False	False	False		
bPNReadError	Bool	24.0	false	False	False	False		
wPNWriteStatus	Word	26.0	16#0	False	False	False		
bPNWriteError	Bool	28.0	false	False	False	False		
bResetDone	Bool	28.1	false	False	False	False		
bHomeStarted	Bool	28.2	false	False	False	False		
bStartMoveDone	Bool	28.3	false	False	False	False		
bEstop	Bool	28.4	false	False	False	False		
wStartReset	Int	30.0	0	False	False	False		
wStartMove	Int	32.0	0	False	False	False		
wSDOStatus	Word	34.0	16#0	False	False	False		
▼ bySDODataArray	Array[0..63] of Byte	36.0		False	False	False		
bySDODataArray[0]	Byte	0.0	16#0	False	False	False		
bySDODataArray[1]	Byte	1.0	16#0	False	False	False		
bySDODataArray[2]	Byte	2.0	16#0	False	False	False		
bySDODataArray[3]	Byte	3.0	16#0	False	False	False		
bySDODataArray[4]	Byte	4.0	16#0	False	False	False		
bySDODataArray[5]	Byte	5.0	16#0	False	False	False		
bySDODataArray[6]	Byte	6.0	16#0	False	False	False		
bySDODataArray[7]	Byte	7.0	16#0	False	False	False		
bySDODataArray[8]	Byte	8.0	16#0	False	False	False		
bySDODataArray[9]	Byte	9.0	16#0	False	False	False		
bySDODataArray[10]	Byte	10.0	16#0	False	False	False		
bySDODataArray[11]	Byte	11.0	16#0	False	False	False		
bySDODataArray[12]	Byte	12.0	16#0	False	False	False		
bySDODataArray[13]	Byte	13.0	16#0	False	False	False		
bySDODataArray[14]	Byte	14.0	16#0	False	False	False		
bySDODataArray[15]	Byte	15.0	16#0	False	False	False		
bySDODataArray[16]	Byte	16.0	16#0	False	False	False		
bySDODataArray[17]	Byte	17.0	16#0	False	False	False		

Totally Integrated Automation Portal								
Name	Data type	Offset	Default value	Accessible from HMI	Visible in HMI	Setpoint	Comment	
bySDODataArray[18]	Byte	18.0	16#0	False	False	False		
bySDODataArray[19]	Byte	19.0	16#0	False	False	False		
bySDODataArray[20]	Byte	20.0	16#0	False	False	False		
bySDODataArray[21]	Byte	21.0	16#0	False	False	False		
bySDODataArray[22]	Byte	22.0	16#0	False	False	False		
bySDODataArray[23]	Byte	23.0	16#0	False	False	False		
bySDODataArray[24]	Byte	24.0	16#0	False	False	False		
bySDODataArray[25]	Byte	25.0	16#0	False	False	False		
bySDODataArray[26]	Byte	26.0	16#0	False	False	False		
bySDODataArray[27]	Byte	27.0	16#0	False	False	False		
bySDODataArray[28]	Byte	28.0	16#0	False	False	False		
bySDODataArray[29]	Byte	29.0	16#0	False	False	False		
bySDODataArray[30]	Byte	30.0	16#0	False	False	False		
bySDODataArray[31]	Byte	31.0	16#0	False	False	False		
bySDODataArray[32]	Byte	32.0	16#0	False	False	False		
bySDODataArray[33]	Byte	33.0	16#0	False	False	False		
bySDODataArray[34]	Byte	34.0	16#0	False	False	False		
bySDODataArray[35]	Byte	35.0	16#0	False	False	False		
bySDODataArray[36]	Byte	36.0	16#0	False	False	False		
bySDODataArray[37]	Byte	37.0	16#0	False	False	False		
bySDODataArray[38]	Byte	38.0	16#0	False	False	False		
bySDODataArray[39]	Byte	39.0	16#0	False	False	False		
bySDODataArray[40]	Byte	40.0	16#0	False	False	False		
bySDODataArray[41]	Byte	41.0	16#0	False	False	False		
bySDODataArray[42]	Byte	42.0	16#0	False	False	False		
bySDODataArray[43]	Byte	43.0	16#0	False	False	False		
bySDODataArray[44]	Byte	44.0	16#0	False	False	False		
bySDODataArray[45]	Byte	45.0	16#0	False	False	False		
bySDODataArray[46]	Byte	46.0	16#0	False	False	False		
bySDODataArray[47]	Byte	47.0	16#0	False	False	False		
bySDODataArray[48]	Byte	48.0	16#0	False	False	False		
bySDODataArray[49]	Byte	49.0	16#0	False	False	False		
bySDODataArray[50]	Byte	50.0	16#0	False	False	False		
bySDODataArray[51]	Byte	51.0	16#0	False	False	False		
bySDODataArray[52]	Byte	52.0	16#0	False	False	False		
bySDODataArray[53]	Byte	53.0	16#0	False	False	False		
bySDODataArray[54]	Byte	54.0	16#0	False	False	False		
bySDODataArray[55]	Byte	55.0	16#0	False	False	False		
bySDODataArray[56]	Byte	56.0	16#0	False	False	False		
bySDODataArray[57]	Byte	57.0	16#0	False	False	False		
bySDODataArray[58]	Byte	58.0	16#0	False	False	False		
bySDODataArray[59]	Byte	59.0	16#0	False	False	False		
bySDODataArray[60]	Byte	60.0	16#0	False	False	False		
bySDODataArray[61]	Byte	61.0	16#0	False	False	False		
bySDODataArray[62]	Byte	62.0	16#0	False	False	False		
bySDODataArray[63]	Byte	63.0	16#0	False	False	False		
▼ SDOWrite	WRREC			False	False	False		
▼ Input								
REQ	Bool		FALSE	False	False	False	REQ = 1: Transfer data record	
ID	HW_IO		16#0	False	False	False	HW-Id of the DP slave/PROFINET IO component	
INDEX	DInt		0	False	False	False	Data record number	
LEN	UInt		0	False	False	False	Maximum length in bytes of the data	
▼ Output								
DONE	Bool		FALSE	False	False	False	Function performed	
BUSY	Bool		FALSE	False	False	False	Function busy	
ERROR	Bool		FALSE	False	False	False	Error flag	
STATUS	DWord		DW#16#0	False	False	False	Function result/error message	
▼ InOut								
RECORD	Variant			False	False	False	Data record	
Static								
▼ SDORead	RDREC			False	False	False		
▼ Input								
REQ	Bool		FALSE	False	False	False	REQ = 1: Transfer data record	
ID	HW_IO		16#0	False	False	False	HW-Id of the DP slave/PROFINET IO component	
INDEX	DInt		0	False	False	False	Data record number	
MLEN	UInt		0	False	False	False	Maximum length in bytes of the data	
▼ Output								
VALID	Bool		FALSE	False	False	False	Function performed	
BUSY	Bool		FALSE	False	False	False	Function busy	
ERROR	Bool		FALSE	False	False	False	Error flag	
STATUS	DWord		DW#16#0	False	False	False	Function result/error message	
LEN	UInt		0	False	False	False	Length of the fetched data record	
▼ InOut								

Totally Integrated Automation Portal		
0057	#bPNWriteError := FALSE;	
0058		
0059	#TG7.Send.STW1.%X10 := TRUE;	
0060	IF #TG7.Receive.ZSW1.%X9 THEN // PLC has control	
0061		
0062	// Do a EStop	
0063	IF #iEStop THEN	
0064	#TG7.Send.STW1.%X2 := FALSE;	
0065	IF #TG7.Receive.ZSW1.%X13 THEN	
0066	#bEstop := TRUE;	
0067	END_IF;	
0068	END_IF;	
0069		
0070	// Do a reset	
0071	IF #iReset AND NOT #bResetStarted THEN	
0072	#wSDOStatus := 16;	
0073	#TG7.Send.STW1.%X7 := TRUE;	
0074	#bEstop := FALSE;	
0075	#bResetStarted := TRUE;	
0076	#wResetCounter := 0;	
0077	END_IF;	
0078	//IF ABS(#wStartReset - #TG7.Receive.ZSW2) > 2 THEN	
0079	// #TG7.Send.STW1.%X7 := FALSE;	
0080	//END_IF;	
0081	IF #bResetStarted THEN	
0082	#wResetCounter := #wResetCounter + 1;	
0083	END_IF;	
0084	IF #wResetCounter > 1 THEN	
0085	#TG7.Send.STW1.%X7 := FALSE;	
0086	#bResetStarted := FALSE;	
0087	#wResetCounter := 0;	
0088	END_IF;	
0089		
0090	// Enable the drive	
0091	IF #iSW_Enable AND NOT #bEstop AND NOT (#bSWEnabled AND #TG7.Receive.ZSW1.%X6 OR #TG7.Receive.ZSW1.%X3)	
THEN		
0092	IF #TG7.Receive.ZSW1.%X6 THEN // Switch on inhibited	
0093	#TG7.Send.STW1.%X1 := TRUE;	
0094	#TG7.Send.STW1.%X2 := TRUE;	
0095	ELSE	
0096	IF #TG7.Receive.ZSW1.%X0 THEN // Ready for switching on	
0097	#TG7.Send.STW1.%X0 := TRUE;	
0098	IF #TG7.Receive.ZSW1.%X1 THEN // Switched on	
0099	#TG7.Send.STW1.%X3 := TRUE;	
0100	IF #TG7.Receive.ZSW1.%X2 THEN // Operation	
0101	#bSWEnabled := TRUE;	
0102		
0103	// Do a Stop	
0104	IF #iStop THEN	
0105	#TG7.Send.STW1.%X4 := FALSE; // Stop active motion task	
0106	#TG7.Send.STW1.%X5 := FALSE; // Stop active motion task	
0107	#TG7.Send.STW1.%X8 := FALSE; // Stop jog 1	
0108	#TG7.Send.STW1.%X9 := FALSE; // Stop jog 2	
0109	#TG7.Send.STW1.%X11 := FALSE; // Stop homeing	
0110	//#TG7.Send.STW1.%X12 := FALSE; // Stop real time jogging	
0111	ELSE	
0112		
0113	// Do home	
0114	IF #iStart_Homing AND NOT #bHomingStarted THEN //start homing	
0115	#TG7.Send.STW1.%X11 := TRUE;	
0116	#bHomingStarted := TRUE; //set flag	
0117	#wHomingCounter := 0; //zero counter	
0118	ELSE	
0119	IF #bHomingStarted THEN // still homing, increment counter	
0120	#wHomingCounter := #wHomingCounter + 1;	
0121	END_IF;	
0122	IF #TG7.Receive.ZSW1.%X11 AND #TG7.Receive.ZSW1.%X13 AND #wHomingCounter > 5	
THEN		
0123	#TG7.Send.STW1.%X11 := FALSE; //if homed, stopped, and counter >5, then	
turn off homing bit		
0124	#bHomingStarted := FALSE; //reset flag	
0125	#wHomingCounter := 0; //zero counter	
0126	END_IF;	
0127		
0128	// Jogging1	
0129	IF #iStart_Jog1 THEN //do Jog1	
0130	#TG7.Send.STW1.%X9 := FALSE; //not Jog2	
0131	#TG7.Send.STW1.%X8 := TRUE;	
0132	// Jogging2	
0133	ELSIF #iStart_Jog2 THEN //or do Jog2	
0134	#TG7.Send.STW1.%X8 := FALSE; //not Jog1	
0135	#TG7.Send.STW1.%X9 := TRUE;	
0136		
0137	ELSE //otherwise turn off both jog bits	
0138	#TG7.Send.STW1.%X8 := FALSE;	
0139	#TG7.Send.STW1.%X9 := FALSE;	
0140		
0141	// Interrupt a move	

Totally Integrated Automation Portal		
0142	IF #iMT_Pause THEN	
0143	#TG7.Send.STW1.%X5 := FALSE; // intermediate stop	
0144	ELSE	
0145	#TG7.Send.STW1.%X5 := TRUE; // No intermediate stop	
0146		
0147	// Move	
0148	IF #iStart_Move AND NOT #bMoveStarted THEN	
0149	#TG7.Send.SATZANW := #iMTNum; // motion task number	
0150	#TG7.Send.STW1.%X4 := TRUE; // Do not reject traversing task	
0151	#TG7.Send.STW1.%X5 := TRUE; // No intermediat stop	
0152	#TG7.Send.STW1.%X6 := TRUE; // start the motion task	
0153		
0154	#bMoveStarted := TRUE;	
0155	#bMoveStartedDone := FALSE;	
0156	#wMoveCounter := 0;	
0157	ELSE	
0158	IF #bMoveStarted THEN	
0159	#wMoveCounter := #wMoveCounter + 1;	
0160	END_IF;	
0161	// Reset the rising edge of the start bit to be read for the next	
task		
0162		
0163	//if #bMoveStarted then - start counter	
0164	IF #TG7.Receive.ZSW1.%X12 OR #wMoveCounter > 62 THEN	
0165	#bMoveStarted := FALSE;	
0166	#bMoveStartedDone := TRUE;	
0167	#TG7.Send.STW1.%X6 := FALSE;	
0168	END_IF;	
0169	END_IF;	
0170	END_IF;	
0171	END_IF;	
0172	END_IF;	
0173	END_IF;	
0174	END_IF;	
0175	END_IF;	
0176	END_IF;	
0177		
0178	// If the axis gets disabled all the corresponding motion siganl have to be reset	
0179	ELSE	
0180	#bSWEnabled := FALSE;	
0181		
0182	#TG7.Send.STW1.%X0 := FALSE;	
0183	#TG7.Send.STW1.%X1 := FALSE;	
0184	#TG7.Send.STW1.%X2 := FALSE;	
0185	#TG7.Send.STW1.%X3 := FALSE;	
0186	#TG7.Send.STW1.%X4 := FALSE;	
0187	#TG7.Send.STW1.%X5 := FALSE;	
0188	#TG7.Send.STW1.%X6 := FALSE;	
0189	#TG7.Send.STW1.%X8 := FALSE;	
0190	#TG7.Send.STW1.%X9 := FALSE;	
0191	#TG7.Send.STW1.%X11 := FALSE;	
0192	#TG7.Send.STW1.%X12 := FALSE;	
0193	END_IF;	
0194	END_IF;	
0195	ELSE	
0196	#bPNWriteError := TRUE;	
0197	END_IF;	
0198		
0199	// Read drive faults and warning on occurens	
0200	IF #oFault THEN	
0201	// If alarm is not read yet: start read	
0202	IF NOT #wSDOStatus.%X6 AND NOT #bSDOStatusReadActive THEN	
0203	#wSDOStatus := 64;	
0204	#wSDOAddress := 16#09AD; // Read PNU 2477 (DRV.FAULT1)	
0205	#bSDOStatusReadActive := TRUE;	
0206	END_IF;	
0207		
0208	// If warning is not read yet: start read	
0209	ELSIF #oWarning THEN	
0210	IF NOT #wSDOStatus.%X5 AND NOT #bSDOStatusReadActive THEN	
0211	#wSDOStatus := 32;	
0212	#wSDOAddress := 16#0AE7; // Read PNU2791 (DRV.WARNING1)	
0213	#bSDOStatusReadActive := TRUE;	
0214	END_IF;	
0215	ELSE	
0216	IF NOT #bSDOStatusReadActive THEN	
0217	#wSDOStatus := 16;	
0218	END_IF;	
0219	END_IF;	
0220		
0221	// start the reading signal	
0222	IF #wSDOStatus >= 32 THEN	
0223	// Always clean out old values first	
0224	IF #bSDOWriteBusy THEN	
0225	#bSDOReq := FALSE;	
0226	ELSE	
0227	FILL_BLK(IN := 0,	
0228	COUNT := 64,	

Totally Integrated Automation Portal

```
0229         OUT => #bySDODataArray[0]);
0230         #bSDOReq := TRUE;
0231         #bSDOWriteDone := FALSE;
0232     END_IF;
0233     // Start the read cylce to the corresponding PNU
0234     IF #wSDOStatus = 32 OR #wSDOStatus = 64 THEN
0235         #bySDODataArray[0] := #wSDOStatus.%B0;
0236         #bySDODataArray[1] := 16#01;
0237         #bySDODataArray[2] := 16#00;
0238         #bySDODataArray[3] := 16#01;
0239
0240         #bySDODataArray[4] := 16#10;
0241         #bySDODataArray[5] := 16#01;
0242         #bySDODataArray[6] := #wSDOAddress.%B1;
0243         #bySDODataArray[7] := #wSDOAddress.%B0;
0244         #bySDODataArray[8] := 16#00;
0245         #bySDODataArray[9] := 16#00;
0246
0247         #SDOWrite(REQ := #bSDOReq,
0248                 ID := #iID,
0249                 INDEX := 47,
0250                 LEN := 14,
0251                 BUSY => #bSDOWriteBusy,
0252                 DONE => #bSDOWriteDone,
0253                 ERROR => #bSDOWriteError,
0254                 RECORD := #bySDODataArray);
0255         IF #bSDOWriteDone THEN
0256             #wSDOStatus := #wSDOStatus + 1;
0257         END_IF;
0258     END_IF;
0259
0260     // read the value back
0261     IF #wSDOStatus = 33 OR #wSDOStatus = 65 THEN
0262         #SDORead(REQ := #bSDOReq,
0263                 ID := #iID,
0264                 INDEX := 47,
0265                 MLEN := 14,
0266                 BUSY => #bSDOReadBusy,
0267                 VALID => #bSDOReadDone,
0268                 ERROR => #bSDOReadError,
0269                 RECORD := #bySDODataArray);
0270         IF #bSDOReadDone THEN
0271             IF #bySDODataArray[4] = 66 THEN // if the read signal is a Word use Byte 6&7
0272                 #wSDOReadValue.%B0 := #bySDODataArray[7];
0273                 #wSDOReadValue.%B1 := #bySDODataArray[6];
0274             ELSE // if the read signal is a DWord use Byte 8&9
0275                 #wSDOReadValue.%B0 := #bySDODataArray[9];
0276                 #wSDOReadValue.%B1 := #bySDODataArray[8];
0277             END_IF;
0278             IF #wSDOReadValue = 0 THEN // If the read value is zero repeat the read cycle
0279                 #wSDOStatus := #wSDOStatus - 1;
0280             ELSE
0281                 #wSDOStatus := #wSDOStatus + 1;
0282                 #bSDOStatusReadActive := FALSE;
0283             END_IF;
0284         END_IF;
0285     END_IF;
0286 END_IF;
0287 END_IF;
0288
0289 // check for rising edge for the input parameters
0290 IF NOT #iReset AND #bResetStarted THEN
0291     #bResetStarted := FALSE;
0292 END_IF;
0293 IF NOT #iStart_Homing AND #bHomingStarted THEN
0294     #bHomingStarted := FALSE;
0295 END_IF;
0296 IF NOT #iStart_Move AND #bMoveStarted AND #bMoveStartedDone AND NOT #TG7.Receive.ZSW1.%X12 THEN
0297     #bMoveStarted := FALSE;
0298     #bMoveStartedDone := FALSE;
0299 END_IF;
0300
0301 // fault detection/handling within the bloc
0302 IF #bPNReadError OR #bPNWriteError OR #bSDOReadError OR #bSDOWriteError THEN
0303     #oError := TRUE;
0304     IF #bPNReadError THEN
0305         #oStatus := #oStatus;
0306     END_IF;
0307     IF #bPNWriteError THEN
0308         #oStatus := #oStatus;
0309     END_IF;
0310 ELSIF #wSDOStatus > 15 THEN
0311     #oError := FALSE;
0312 END_IF;
0313
```

Symbol	Address	Type	Comment
#bEstop		Bool	
#bHomingStarted		Bool	

Totally Integrated Automation Portal			
Symbol	Address	Type	Comment
#bInitialize		Bool	
#bMoveStarted		Bool	
#bMoveStartedDone		Bool	
#bPNReadError		Bool	
#bPNWriteError		Bool	
#bResetStarted		Bool	
#bSDOReadBusy		Bool	
#bSDOReadDone		Bool	
#bSDOReadError		Bool	
#bSDOReq		Bool	
#bSDOStatusReadActive		Bool	
#bSDOWriteBusy		Bool	
#bSDOWriteDone		Bool	
#bSDOWriteError		Bool	
#bSWEnabled		Bool	
#bySDOdataArray		Array	
#iEStop		Bool	immediate quick stop and disable
#iID		HW_IO	Hardware identifier
#iMT_Pause		Bool	pause active motion task
#iMTNum		Word	motion task number 0x8000 = direct motion task
#iReset		Bool	reset drive faults
#iStart_Homing		Bool	start homing
#iStart_Jog1		Bool	
#iStart_Jog2		Bool	start jog
#iStart_Move		Bool	start motion task
#iStop		Bool	stop all movement
#iSW_Enable		Bool	enable/disable
#oEnabled		Bool	Drive power on
#oError		Bool	fault in function block or communication
#oFault		Bool	drive fault
#oHomed		Bool	axis homed
#oInPos		Bool	axis within in-position window
#oMoving		Bool	axis enabled and moving
#oStatus		DWord	ZSW1
#oStatus.%W0		Word	ZSW1
#oStatus.%W1		Word	ZSW1
#oWarning		Bool	drive warning
#SDORead		Multi_SFB	
#SDOWrite		Multi_SFB	
#TG7.Receive		Struct	
#TG7.Receive.ZSW1		Int	Status word 1
#TG7.Receive.ZSW1.%X0		Bool	Status word 1
#TG7.Receive.ZSW1.%X1		Bool	Status word 1
#TG7.Receive.ZSW1.%X2		Bool	Status word 1
#TG7.Receive.ZSW1.%X3		Bool	Status word 1
#TG7.Receive.ZSW1.%X6		Bool	Status word 1
#TG7.Receive.ZSW1.%X7		Bool	Status word 1
#TG7.Receive.ZSW1.%X9		Bool	Status word 1
#TG7.Receive.ZSW1.%X10		Bool	Status word 1
#TG7.Receive.ZSW1.%X11		Bool	Status word 1
#TG7.Receive.ZSW1.%X12		Bool	Status word 1
#TG7.Receive.ZSW1.%X13		Bool	Status word 1
#TG7.Receive.ZSW1.%X14		Bool	Status word 1
#TG7.Send		Struct	
#TG7.Send.SATZANW		Int	Motion task selection
#TG7.Send.STW1.%X0		Bool	Control word 1
#TG7.Send.STW1.%X1		Bool	Control word 1
#TG7.Send.STW1.%X2		Bool	Control word 1
#TG7.Send.STW1.%X3		Bool	Control word 1
#TG7.Send.STW1.%X4		Bool	Control word 1
#TG7.Send.STW1.%X5		Bool	Control word 1
#TG7.Send.STW1.%X6		Bool	Control word 1
#TG7.Send.STW1.%X7		Bool	Control word 1
#TG7.Send.STW1.%X8		Bool	Control word 1
#TG7.Send.STW1.%X9		Bool	Control word 1
#TG7.Send.STW1.%X10		Bool	Control word 1
#TG7.Send.STW1.%X11		Bool	Control word 1
#TG7.Send.STW1.%X12		Bool	Control word 1
#wHomingCounter		Word	
#wMoveCounter		Word	
#wPNReadStatus		Word	
#wPNWriteStatus		Word	
#wResetCounter		Word	
#wSDOAddress		Word	
#wSDOAddress.%B0		Byte	
#wSDOAddress.%B1		Byte	
#wSDOReadValue		Word	
#wSDOReadValue.%B0		Byte	
#wSDOReadValue.%B1		Byte	
#wSDOStatus		Word	
#wSDOStatus.%B0		Byte	
#wSDOStatus.%X5		Bool	
#wSDOStatus.%X6		Bool	

--	--	--