Reading A Parameter In The AKD Drive Over Ethernet IP version B. 2-28-2019.

There are several ways to read a parameter in the AKD drive over Ethernet IP.

It is important to note the advantages and disadvantages of each method.

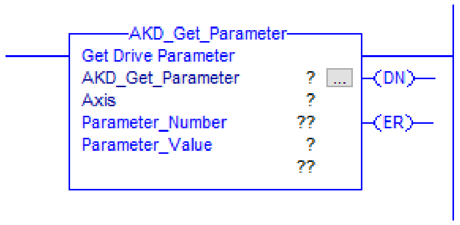
The AB Micrologix 1400 PLC supports only explicit messaging via the MSG block. For more details see the AKD Ethernet IP RSLogix Communications manual under

Appendix C: RSLogix 500 for examples and details.

Compactlogix and Contrologix PLCs support both explicit and implicit (I/O) messaging and our Add-On-Instruction library. The remainder of this article will apply to them.

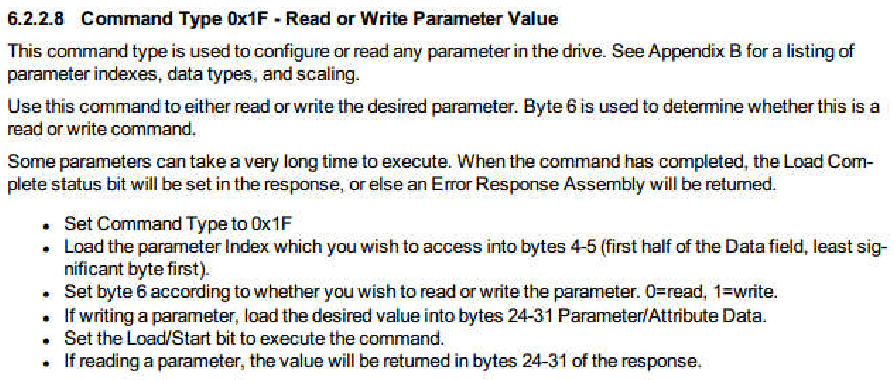
METHOD #1: Using Add On Instructions

The first method to read a parameter is the AKD_Get_Parameter Add-On-Instruction.



All Add-On-Instructions use the same Command and Response assembly bytes (bytes 0 through 32) therefore only 1 AOI can be executing at a time (for a given axis). If you command another AOI before the current one finishes you will get errors. The issue with this method is in order to query a parameter you can't be doing something else (like an AKD_Move, AKD_Jog, etc.) at the time same time.

The AKD_Get_Parameter or AKD_Set_Parameter AOIs use the following method internally. The key point is that they use the polled I/O to execute it.



6.2.2.8 Command Type 0x1F - Read or Write Parameter Value

This command type is used to configure or read any parameter in the drive. See Appendix B for a listing of parameter indexes, data types, and scaling.

Use this command to either read or write the desired parameter. Byte 6 is used to determine whether this is a read or write command.

Some parameters can take a very long time to execute. When the command has completed, the Load Complete status bit will be set in the response, or else an Error Response Assembly will be returned.

- Set Command Type to 0x1F
- Load the parameter Index which you wish to access into bytes 4-5 (first half of the Data field, least significant byte first).
- Set byte 6 according to whether you wish to read or write the parameter. 0=read, 1=write.
- If writing a parameter, load the desired value into bytes 24-31 Parameter/Attribute Data.
- Set the Load/Start bit to execute the command.
- If reading a parameter, the value will be returned in bytes 24-31 of the response.

Note there are also AKD_Set_Attribute and AKD_Get_Attribute add on instructions where a limited number of drive values can be set or queried but the Position Controller Attribute table is much more limited than the instance table. See the Position Controller Attribute list in the AKD Ethernet IP Communication manual. These have the same timing concerns mentioned above.

METHOD#2: Dynamic Mapping

The second method is to use dynamic mapping. Dynamic mapping allows you to put parameters you want to read or write data from/to on the RPI poll. These parameters get mapped to bytes 36-63 so they do not interfere with AOIs or the static mapping of the command and response assemblies. There is an application note on this (see the link below). The original method for mapping uses the EIP.CMDMAP and EIP.RSPMAP keywords in the Workbench terminal and it still valid. The AKD Ethernet IP Communications manual has examples for mapping using the Workbench terminal (see chapter 9). It is also possible to use the Workbench GUI to do dynamic mapping using a map table. This is also covered in the app. note below.

App Note:

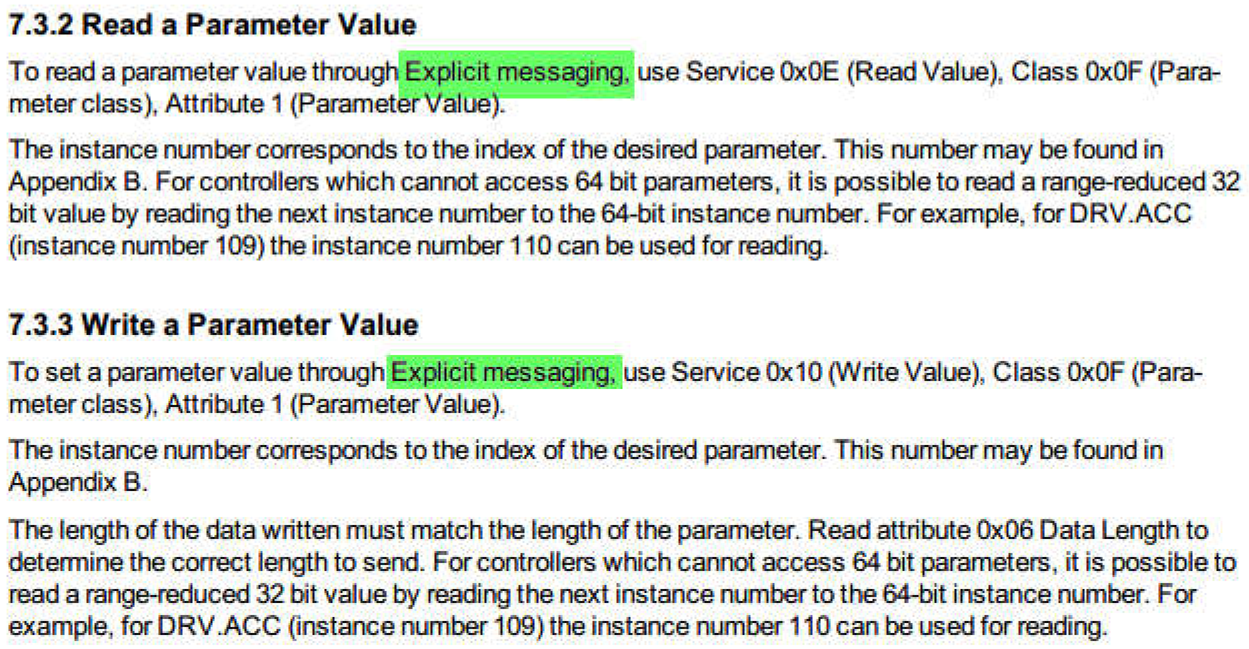
http://kdn.kollmorgen.com/content/akd-ethernet-ip-diagnostics-and-dynamic-mapping

METHOD#3: Using MSG (message) blocks.

The 3rd method is to do explicit messaging using the MSG block which is also independent of the polled IO.

Explicit Messaging using the MSG block uses the following method which is independent of the polled I/O. The disadvantage is this is on-demand and not on the RPI poll so the data is only updated every time the MSG is triggered.

The AKD EIP Communications manual describes the methods as follows:



7.3.2 Read a Parameter Value

To read a parameter value through Explicit messaging, use Service 0x0E (Read Value), Class 0x0F (Parameter class), Attribute 1 (Parameter Value).

The instance number corresponds to the index of the desired parameter. This number may be found in Appendix B. For controllers which cannot access 64 bit parameters, it is possible to read a range-reduced 32 bit value by reading the next instance number to the 64-bit instance number. For example, for DRV.ACC (instance number 109) the instance number 110 can be used for reading.

7.3.3 Write a Parameter Value

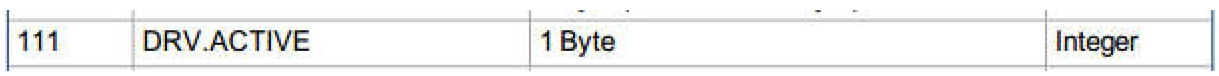
To set a parameter value through Explicit messaging, use Service 0x10 (Write Value), Class 0x0F (Parameter class), Attribute 1 (Parameter Value).

The instance number corresponds to the index of the desired parameter. This number may be found in Appendix B.

The length of the data written must match the length of the parameter. Read attribute 0x06 Data Length to determine the correct length to send. For controllers which cannot access 64 bit parameters, it is possible to read a range-reduced 32 bit value by reading the next instance number to the 64-bit instance number. For example, for DRV.ACC (instance number 109) the instance number 110 can be used for reading.

Compactlogix and Contrologix Explicit MSG Example: Read DRV.ACTIVE

From Appendix B: Parameter Listing which gives the instance number, parameter keyword, data size, and data type:

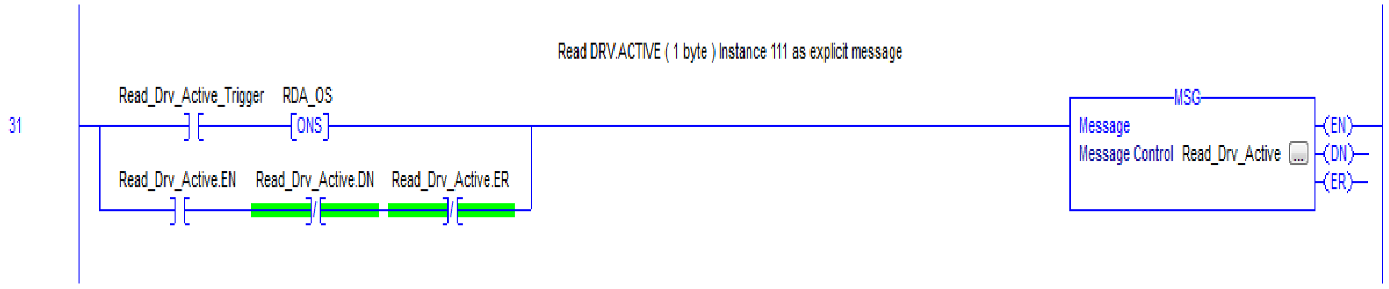


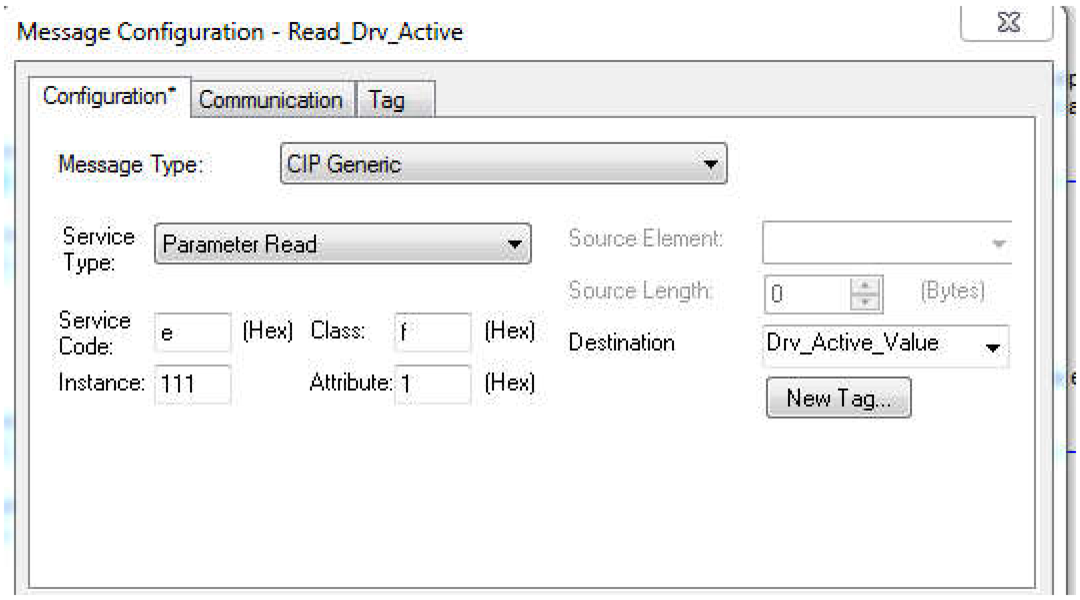
111	DRV.ACTIVE	1 Byte	Integer

Note the following table in regards to tag dimensions in the AB PLC versus in the AKD.

AKD Instance Data Type	AB PLC Tag Type (to copy the value into)
Command	SINT
1 Byte	SINT
1 Byte Signed	SINT
2 Byte	INT
2 Byte Signed	INT
4 Byte	DINT
4 Byte Signed	DINT
8 Byte	LINT
8 Byte Signed	LINT

* Tag Type REAL is not applicable with the AKD Ethernet IP parameters.





Message Configuration - Read_Drv_Active

Configuration* | Communication | Tag

Message Type: CIP Generic

Service Type: Parameter Read

Service Code: e (Hex) Class: f (Hex)

Instance: 111 Attribute: 1 (Hex)

Source Element:

Source Length: 0 (Bytes)

Destination: Drv_Active_Value

IMPORTANT!: In general, every read/write should be tested for data validity between the PLC and the AKD by the programmer and that the values do not rollover or are out of range.

In Summary:

The order of communication efficiency from most efficient to least is:

Dynamic Mapping->Explicit Messaging->AOI Read/Write

Dynamic Mapping and Explicit Messaging do not interfere with the execution of other AOIs (which use the static bytes of the command and response assemblies) for a given axis.

The order of ease of setup from easy to difficult is:

AOI Read/Write->Explicit Messaging->Dynamic Mapping

AOI Read/Writes are easy. Look up the instance number and trigger the AOI per best practices. Using the MSG block requires more steps. Dynamic Mapping requires mapping and byte consumption management and turning on the data transfer (map type=2 dynamic).

Note the order of efficiency versus ease of setup are opposite in order so perhaps there is a trade-off. Which method to use is application dependent.