MODBUS.MSGLOG and MODBUS.MSGDUMP

When MODBUS.MSGLOG is set to 1 it enables the logging of the raw Modbus data received by the drive via the report given by MODBUS.MSGDUMP.

For a complete Modbus Specification please reference:

http://www.modbus.org

The AKD supports only the following function codes:

FC 03 (read holding registers) and FC 16 (Write Multiple Registers)

Example 1: Read Holding Registers:

The format (for read holding registers) in the MODBUS.MSGDUMP log is:

Bytes 0, 1: Transaction Identifier

Bytes 2,3: Protocol Identifier (always zero)

Bytes 4, 5: Length. Equal to the number of bytes which follow.

Byte 6: Unit Identifier

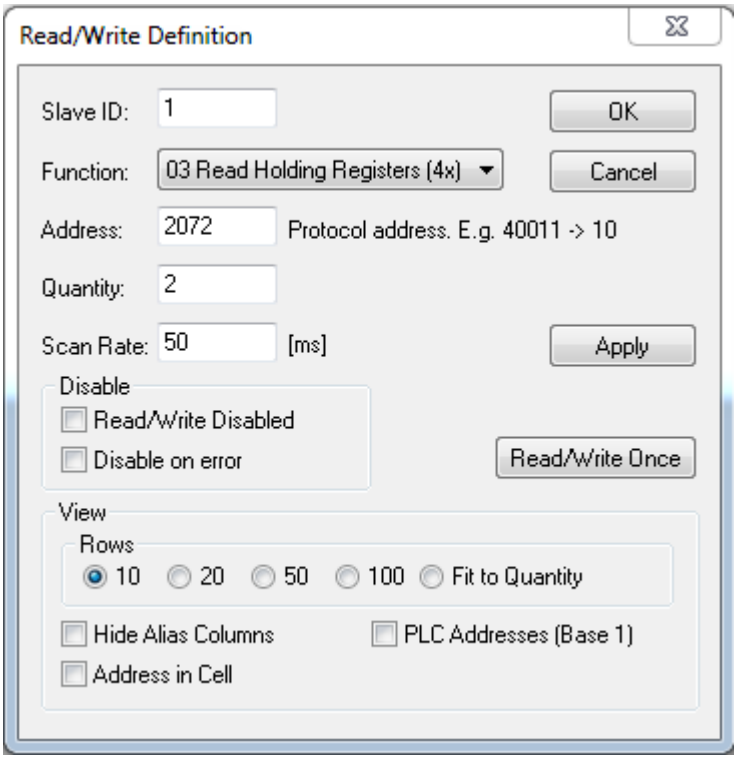
Byte 7: Function Code

Byte 8,9: Starting Address

Byte 10, 11: Number of words to read

For example:

I setup a poll for PL.FB_32 (position feedback) Modbus Registers 2072 and 2073. Note my Modbus master does not require offset or word swapping when addressing but some masters may require one, the other, or both.



The format (for read holding registers) using the data captured below:

Bytes 0, 1: Transaction Identifier = varies

Bytes 2,3: Protocol Identifier= 00 00

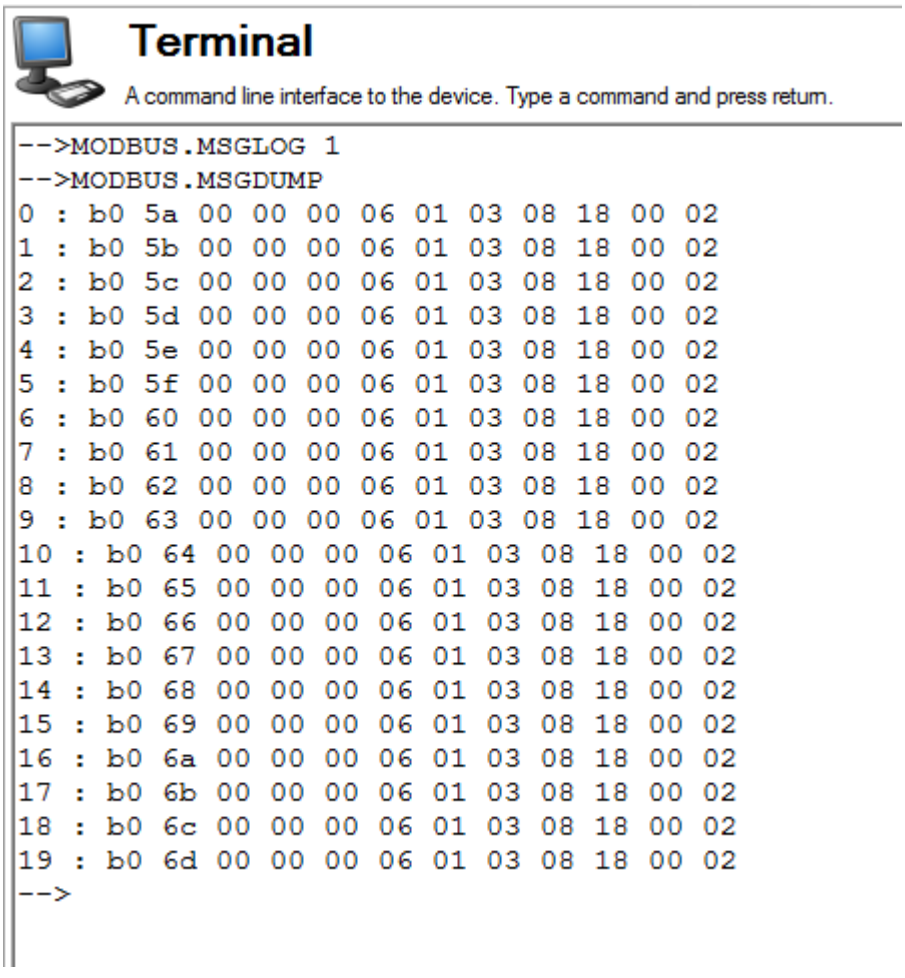
Bytes 4, 5: Length. Equal to the number of bytes which follow. In this case (1+1+ 2+ 2=6) or 00 06

Byte 6: Unit Identifier=01

Byte 7: Function Code=03

Byte 8,9: Starting Address=08 18 (hex)=2072 (dec)

Byte 10, 11: Number of words to read= 00 02 (2 registers)



Example 2: Write Multiple Registers (FC 16)

The format (for write multiple registers) in the MODBUS.MSGDUMP log is:

Bytes 0, 1: Transaction Identifier

Bytes 2,3: Protocol Identifier (always zero)

Bytes: 4,5: Data Length (00 0b) This is bytes to follow in the message. This is (7 + 2xnumber of words) .

Byte 6: Unit Identifier

Byte 7: Function Code

Bytes 8,9 Start Address

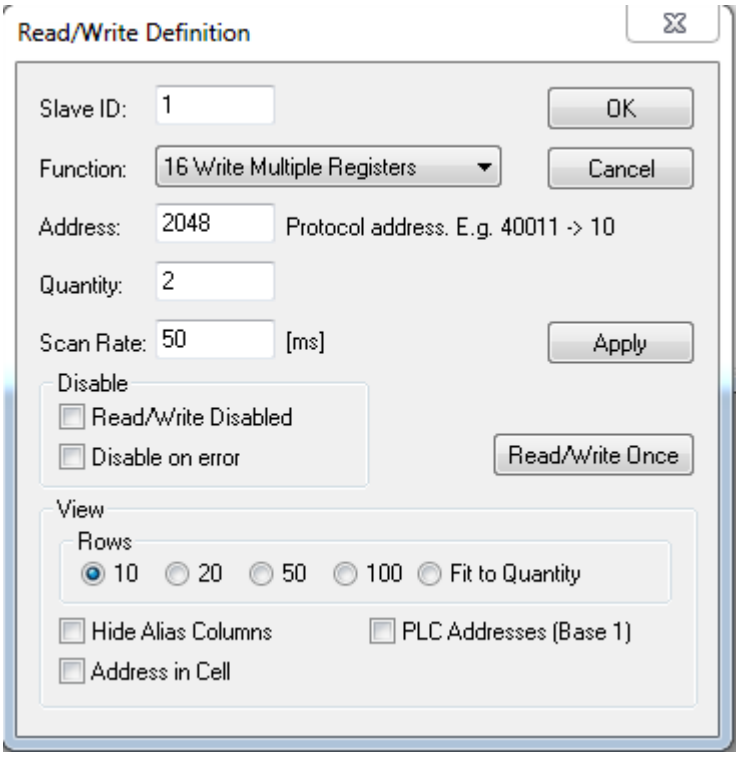
Bytes: 10,11: Number of words to write.

Byte 12: Byte Count of Data Field. Equal to number of bytes to follow (2 * number of words)

Byte 13, 14 Data value for first word

Byte 15, 16: Data value for second word if any, etc…

For example: I setup a poll to write to the HOME.DIST_32 parameter (Modbus Registers 2048 and 2049) and then set the value to 12345 (decimal).



	Alias	02040
0		
1		
2		
3		
4		
5		
6		
7		
8		0
9		12345

The format (for write multiple registers) and using the data captured below:

Bytes 0, 1: Transaction Identifier (xx xx (varies))

Bytes 2,3: Protocol Identifier (00 00)

Bytes: 4,5: Data Length (00 0b) This is bytes to follow in the message. This is (7 + 2xnumber of words) so 7 (for Unit ID, etc.) and + 2x2 (2 words to write in this case)=11

Byte 6: 1 Byte Unit Identifier (unit ID=1)

Byte 7: Function Code (FC=10 hex=16 dec Write Multiple Registers)

Bytes 8,9 Start Address=08 00 (hex)=2048 (dec)

Bytes: 10,11: Number of words to write= 00 02 (# of registers)

Byte 12: Byte Count of Data Field. Equal to number of bytes to follow (2 * number of words) in this case 2x2=4

Byte 13, 14 Data value for first word= 00 00 hex = 0 decimal.

Byte 15, 16: Data value for second word if any, etc…= 30 39 hex =12345 decimal.

Terminal

A command line interface to the device. Type a command and press return.

```
-->MODBUS.MSGLOG 1
-->MODBUS.MSGDUMP
0 : 0e b7 00 00 00 0b 01 10 08 00 00 02 04 00 00 30 39
1 : 0e b8 00 00 00 0b 01 10 08 00 00 02 04 00 00 30 39
2 : 0e b9 00 00 00 0b 01 10 08 00 00 02 04 00 00 30 39
3 : 0e ba 00 00 00 0b 01 10 08 00 00 02 04 00 00 30 39
4 : 0e bb 00 00 00 0b 01 10 08 00 00 02 04 00 00 30 39
5 : 0e bc 00 00 00 0b 01 10 08 00 00 02 04 00 00 30 39
6 : 0e bd 00 00 00 0b 01 10 08 00 00 02 04 00 00 30 39
7 : 0e be 00 00 00 0b 01 10 08 00 00 02 04 00 00 30 39
8 : 0e bf 00 00 00 0b 01 10 08 00 00 02 04 00 00 30 39
9 : 0e c0 00 00 00 0b 01 10 08 00 00 02 04 00 00 30 39
10 : 0e c1 00 00 00 0b 01 10 08 00 00 02 04 00 00 30 39
11 : 0e c2 00 00 00 0b 01 10 08 00 00 02 04 00 00 30 39
12 : 0e c3 00 00 00 0b 01 10 08 00 00 02 04 00 00 30 39
13 : 0e c4 00 00 00 0b 01 10 08 00 00 02 04 00 00 30 39
14 : 0e c5 00 00 00 0b 01 10 08 00 00 02 04 00 00 30 39
15 : 0e c6 00 00 00 0b 01 10 08 00 00 02 04 00 00 30 39
16 : 0e c7 00 00 00 0b 01 10 08 00 00 02 04 00 00 30 39
17 : 0e c8 00 00 00 0b 01 10 08 00 00 02 04 00 00 30 39
18 : 0e c9 00 00 00 0b 01 10 08 00 00 02 04 00 00 30 39
19 : 0e ca 00 00 00 0b 01 10 08 00 00 02 04 00 00 30 39
-->
```