



www.ProSoft-Technology.com

The purpose of this document is to aid in the configuration and setup of the communications between a ProSoft Technology Modbus communications module and an Endress+Hauser Promass flow meter.

This document assumes the user has a reasonable understanding Modbus, RS485 communications, and Rockwell Software's RSLogix product line.

The examples on the next few pages refer to an MVI56-MCM communicating with a Proline Promass 83.

Refer to the appropriate Endress+Hauser manual for setting up the RS485 Modbus communications (RTU/ASCII, baud rate, parity, node address, etc.) on the Promass unit. For this test, the manual used was the Device Functions Proline Promass 83.

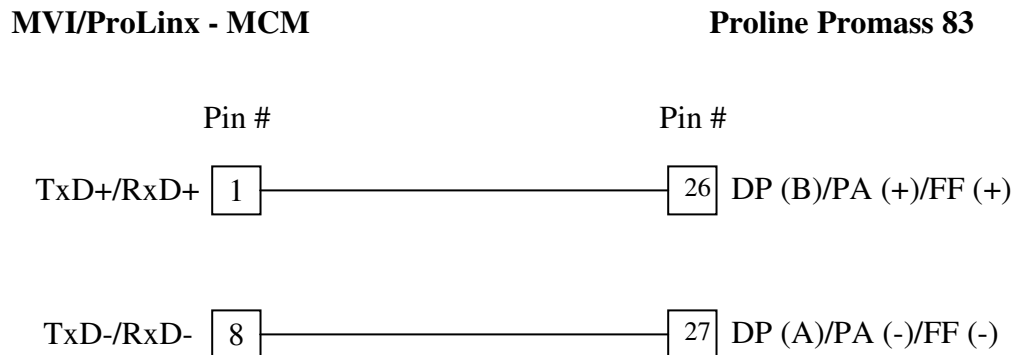
Refer to the ProSoft Technology manual that is specific to the MCM module being used. For this test, the MVI56-MCM user manual was used.

Refer to the manual of both products for wiring specifications. For this example, the MVI56-MCM user manual and the Proline Promass 83 manuals were used.

***Note – The ProSoft Technology MVI products have a hardware jumper that must be set to RS485 mode.**

Refer to the diagram below for wiring the MVI56-MCM to the Promass 83.

Figure 1



Reading From Promass

For this test, the sample MVI56-MCM ladder was used. The Promass 83 was setup for Modbus RTU, 19200 baud, Even parity and slave ID 247. Several items from the Promass 83 were polled, and they were:

Figure 2

Description	Modbus Address
Mass Flow	42007
Volume Flow	42009
Corr. Volume Flow	42011
Density	42013
Reference Density	42015
Temperature	42017

To read these values, a Modbus master command is used. The MVI56-MCM Modbus master port and command structure are detailed in figure 3 on the next page.

The following diagram lists the Controller Tags for the MVI56-MCM MCM.Port1 and MCM.P1Cmd[0] setup parameters. Refer to the MVI56-MCM manual for a complete description of all the fields.

Figure 3

- MCM.Port1		{...}	{...}		MCMPort1	This
+ MCM.Port1.Enabled	1		Decimal	INT	This	
+ MCM.Port1.Type	0		Decimal	INT	This	
+ MCM.Port1.FloatFlag	0		Decimal	INT	This	
+ MCM.Port1.FloatStart	0		Decimal	INT	This	
+ MCM.Port1.FloatOffset	0		Decimal	INT	This	
+ MCM.Port1.Protocol	0		Decimal	INT	This	
+ MCM.Port1.Baudrate	19200		Decimal	INT	This	
+ MCM.Port1.Parity	2		Decimal	INT	This	
+ MCM.Port1.DataBits	8		Decimal	INT	This	
+ MCM.Port1.StopBits	1		Decimal	INT	This	
+ MCM.Port1.RTSOn	0		Decimal	INT	This	
+ MCM.Port1.RTSOff	0		Decimal	INT	This	
+ MCM.Port1.MinResp	0		Decimal	INT	This	
+ MCM.Port1.UseCTS	0		Decimal	INT	This	
+ MCM.Port1.SlaveID	0		Decimal	INT	This	
+ MCM.Port1.BitInOffset	0		Decimal	INT	This	
+ MCM.Port1.WordInOffset	0		Decimal	INT	This	
+ MCM.Port1.OutOffset	0		Decimal	INT	This	
+ MCM.Port1.HoldOffset	0		Decimal	INT	This	
+ MCM.Port1.CmdCount	100		Decimal	INT	This	
+ MCM.Port1.MinCmdDelay	0		Decimal	INT	This	
+ MCM.Port1.CmdErrPtr	1100		Decimal	INT	This	
+ MCM.Port1.RespTO	1000		Decimal	INT	This	
+ MCM.Port1.Retry_Count	2		Decimal	INT	This	
+ MCM.Port1.ErrorDelayCntr	0		Decimal	INT	This	
+ MCM.Port2	{...}	{...}		MCMPort2	This	
- MCM.P1Cmd	{...}	{...}		MCMCmd[100]	This	
- MCM.P1Cmd[0]	{...}	{...}		MCMCmd	This	
+ MCM.P1Cmd[0].Enable	1		Decimal	INT	This	
+ MCM.P1Cmd[0].IntAddress	600		Decimal	INT	This	
+ MCM.P1Cmd[0].PollInt	0		Decimal	INT	This	
+ MCM.P1Cmd[0].Count	12		Decimal	INT	This	
+ MCM.P1Cmd[0].Swap	0		Decimal	INT	This	
+ MCM.P1Cmd[0].Device	247		Decimal	INT	This	
+ MCM.P1Cmd[0].Func	3		Decimal	INT	This	
+ MCM.P1Cmd[0].DevAddress	2006		Decimal	INT	This	

The image below shows how the Modbus master command will look to retrieve the appropriate data from the Promass 83. The values polled are from Figure 2.

Figure 4

[-] MCM.P1Cmd	{...}	{...}		MCMCmd[100]	Thi:
[-] MCM.P1Cmd[0]	{...}	{...}		MCMCmd	Thi:
[+] MCM.P1Cmd[0].Enable	1		Decimal	INT	Thi:
[+] MCM.P1Cmd[0].IntAddress	600		Decimal	INT	Thi:
[+] MCM.P1Cmd[0].PollInt	0		Decimal	INT	Thi:
[+] MCM.P1Cmd[0].Count	12		Decimal	INT	Thi:
[+] MCM.P1Cmd[0].Swap	0		Decimal	INT	Thi:
[+] MCM.P1Cmd[0].Device	247		Decimal	INT	Thi:
[+] MCM.P1Cmd[0].Func	3		Decimal	INT	Thi:
[+] MCM.P1Cmd[0].DevAddress	2006		Decimal	INT	Thi:

MCM.P1Cmd[0]

- Enable – a value of 1 is a continuously-enabled command
- IntAddress – 600 represents the address of the MVI56-MCM module (since MCM.ModDef.ReadStartReg is 600, we know that address 600 is the starting address for the first read commands. All other READ commands will have an IntAddress greater than 600, as to not overlap)**
- PollInt – Poll interval for this command
- Count – a value of 12 signifies that the command is to use 12 integers (whether they're read or write depends on the .Func)
- Swap – Typically only used with floats... this parameter can swap words, bytes, words and bytes. For the Promass 83, no swap was needed since the byte order was 1-0-3-2**
- Device – 247 is the slave ID of the Promass 83
- Func – 3 specifies Modbus function code 3, Read Multiple Holding Registers
- DevAddress – 2006 represents 42007 (there is an offset of 1)

The values read from the above command are floating point values and use two integers per float. These values are read in to MCM.ReadData of the MVI56-MCM ladder.

Remember that the MCM.ReadData elements are 16bit integers. Each value read from the table above, will consume two MCM.ReadData elements.

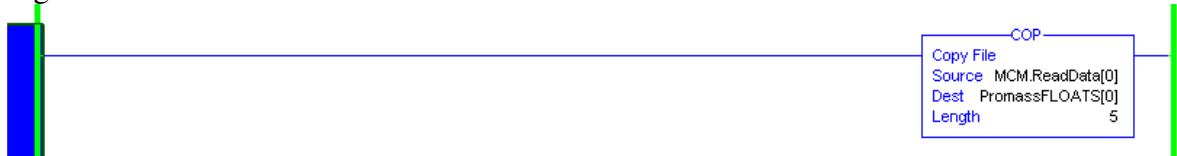
The image below shows how those values might look in integer format:

Figure 5

MCM.ReadData		{...}	
+ MCM.ReadData[0]	-4241	}	Mass Flow
+ MCM.ReadData[1]	17409		
+ MCM.ReadData[2]	-15507	}	Volume Flow
+ MCM.ReadData[3]	17669		
+ MCM.ReadData[4]	-22225	}	Corrected Volume Flow
+ MCM.ReadData[5]	16738		
+ MCM.ReadData[6]	7318	}	Density
+ MCM.ReadData[7]	16255		
+ MCM.ReadData[8]	21607	}	Reference Density
+ MCM.ReadData[9]	17017		
+ MCM.ReadData[10]	948	}	Temperature
+ MCM.ReadData[11]	17046		

A simple copy instruction in ladder is used to convert these values to their original floating point format.

Figure 6



The image below shows the result of the copy instruction.

Figure 7

PromassFLOATS		{...}	{...} Float	REAL[5]
PromassFLOATS[0]	519.77844		Float	REAL
PromassFLOATS[1]	2147.9597		Float	REAL
PromassFLOATS[2]	14.08493		Float	REAL
PromassFLOATS[3]	0.99647933		Float	REAL
PromassFLOATS[4]	62.327106		Float	REAL

Writing to Promass

The section below describes the steps necessary to write floating point and integer values to the Promass.

First, add two more commands as defined below.

Figure 8

+ MCM.P1Cmd[0]	{...}	{...}		MCMCmd
- MCM.P1Cmd[1]	{...}	{...}		MCMCmd
+ MCM.P1Cmd[1].Enable	2		Decimal	INT
+ MCM.P1Cmd[1].IntAddress	0		Decimal	INT
+ MCM.P1Cmd[1].PollInt	0		Decimal	INT
+ MCM.P1Cmd[1].Count	1		Decimal	INT
+ MCM.P1Cmd[1].Swap	0		Decimal	INT
+ MCM.P1Cmd[1].Device	247		Decimal	INT
+ MCM.P1Cmd[1].Func	16		Decimal	INT
+ MCM.P1Cmd[1].DevAddress	5100		Decimal	INT

The above example shows **MCM.P1Cmd[1]** writing one integer to holding register 45101 which is the Enable Low-Flow Cutoff.

- Enable – a value of 2 is a conditional command; the command will only execute when the data in WriteData[0] changes.

- IntAddress – 0 represents the address of the MVI56-MCM module (since MCM.ModDef.WriteStartReg is 0, we know that address 0 is the starting address for the first write command. All other write commands will have an IntAddress between 0 and 599, as to not overlap)*** If a bit-level function code (Read/Write Coils, Read Input Status, etc) is used, then the IntAddress is to the bit address of the MVI56-MCM. So if you want to send the value of WriteData[100].5 (bit 5), the IntAddress would be 1605.**

- PollInt – Poll interval for this command

- Count – a value of 1 signifies that the command is to use 1 integer (whether they're read or write depends on the .Func)

- Swap – Typically only used with floats... this parameter can swap words, bytes, words and bytes. For the Promass 83, no swap was needed since the byte order was 1-0-3-2**

- Device – 247 is the slave ID of the Promass 83

- Func – 16 specifies Modbus function code 16, Write Multiple Holding Registers

- DevAddress –5100 represents 45101 (there is an offset of 1)

The next example shows how to write a floating point value to the Promass.

Figure 9

[-] MCM.P1Cmd[2]	{...}	{...}		MCMCmd
+ MCM.P1Cmd[2].Enable	2		Decimal	INT
+ MCM.P1Cmd[2].IntAddress	1		Decimal	INT
+ MCM.P1Cmd[2].PollInt	0		Decimal	INT
+ MCM.P1Cmd[2].Count	2		Decimal	INT
+ MCM.P1Cmd[2].Swap	0		Decimal	INT
+ MCM.P1Cmd[2].Device	247		Decimal	INT
+ MCM.P1Cmd[2].Func	16		Decimal	INT
+ MCM.P1Cmd[2].DevAddress	5137		Decimal	INT

The above example shows **MCM.P1Cmd[2]** writing two integers to holding register 45138 which is the value for the Low-Flow Cutoff.

- Enable – a value of 2 is a conditional command; the command will only execute when the data in WriteData[1 or 2] changes.

- IntAddress – 1 represents the address of the MVI56-MCM module (this corresponds to WriteData[1])**

- PollInt – Poll interval for this command

- Count – a value of 2 signifies that the command is to use 2 integers (whether they're read or write depends on the .Func)

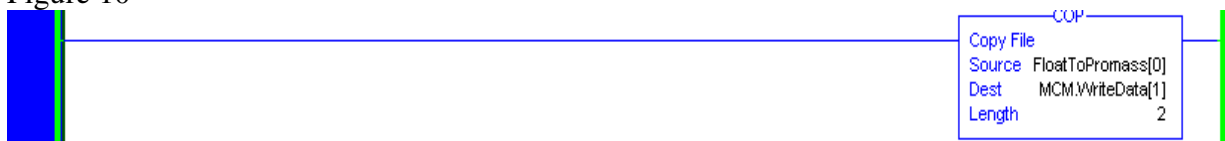
- Swap – Typically only used with floats... this parameter can swap words, bytes, words and bytes. For the Promass 83, no swap was needed since the byte order was 1-0-3-2**

- Device – 247 is the slave ID of the Promass 83

- Func – 16 specifies Modbus function code 16, Write Multiple Holding Registers

- DevAddress –5137 represents 45138 (there is an offset of 1)

To get the floating point value to the WriteData[1] area, a simple copy instruction is used
Figure 10



The image below shows a value of 2.5 in the FloatToPromass[0] tag.

Figure 11

- FloatToPromass	{...}	{...}	Float	REAL[5]
- FloatToPromass[0]	2.5		Float	REAL

The example below shows how a value of 2.5 looks in integer format, and is the result of the COPY instruction in figure 10

Figure 12

+ MCM.WriteData[1]	0		Decimal	INT
+ MCM.WriteData[2]	16416		Decimal	INT

For more information on this, or any other ProSoft Technology product, please call our Technical Services Department at 661.716.5100.