Application Note

Configuring a Pro-face HMI to communicate with an MPiec controller over MODBUS/TCP

Applicable Product: MPiec controller

Yaskawa Electric America
2121 Norman Drive South
Waukegan, IL 60085
1-800-927-5292

Subject: Application Note	Product: MPiec	Doc#: AN.MCD.09.124
Title: Configuring a Pro-face HMI to communicate with an MPiec controller over MODBUS/TCP		

Application Overview

This application note describes the steps needed to configure a Pro-face HMI to communicate with an MPiec controller over MODBUS TCP. The set up steps for the two devices, HMI and the MPiec controller are shown in detail in this note.

Application Highlights:

Industry:	Automation
Major Features:	Quick and easy set up for communication using MODBUS/TCP

Products Used:

Component	Product and Model Number
Controller	MPiec
Software	MotionWorks IEC Express
HMI	Pro-face AGP3650

Application Hardware

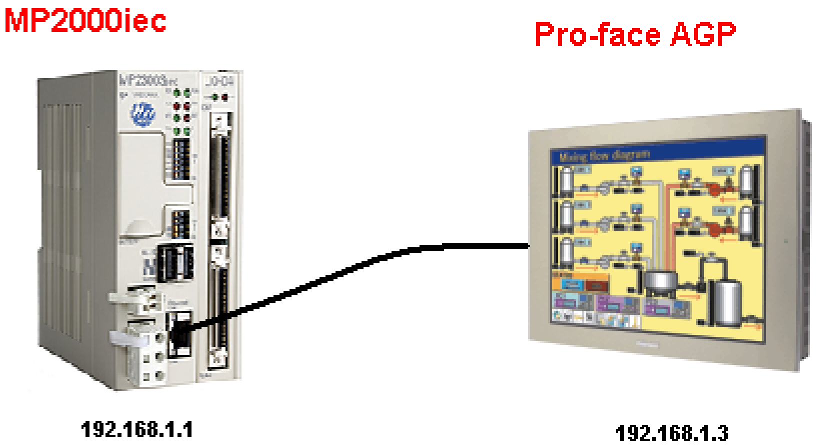
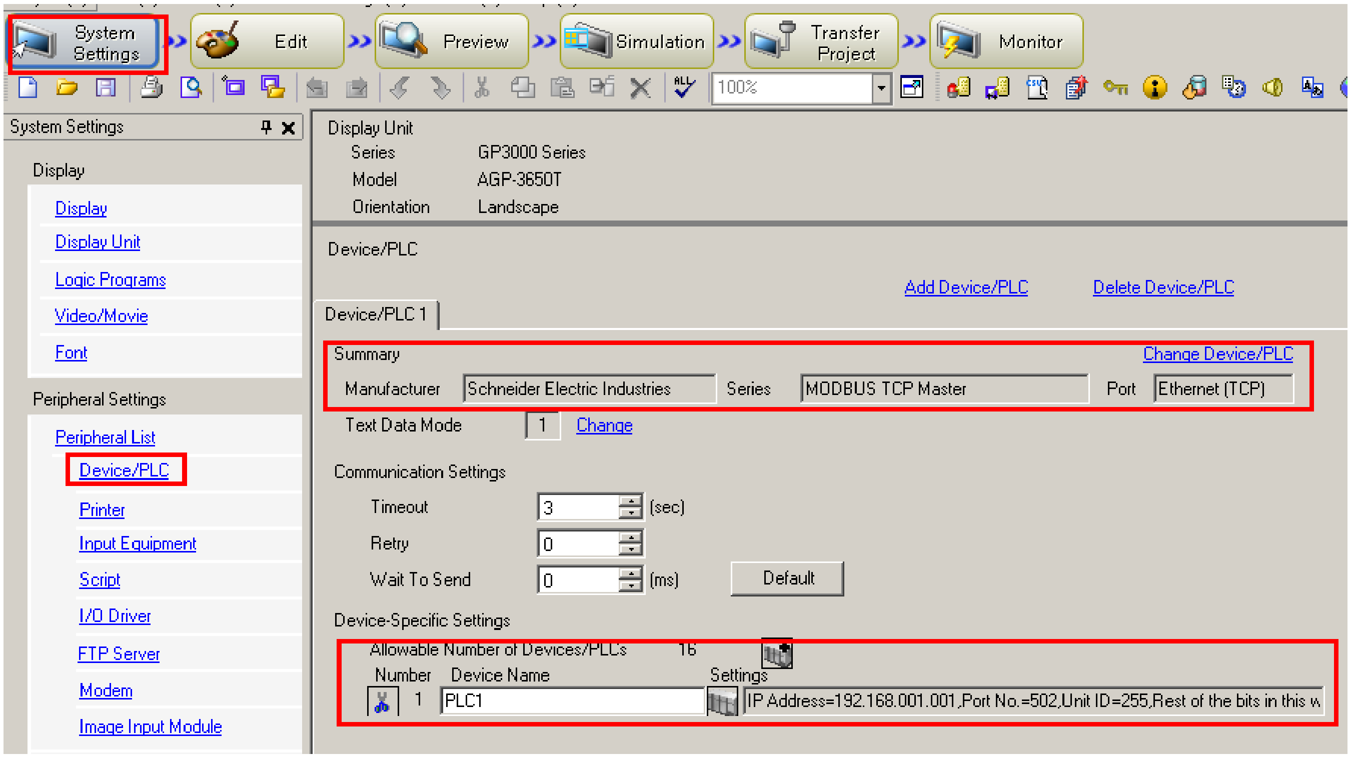


Figure 1: Hardware

Implementation Method of Core Operation

In order to set the HMI as a MODBUS client, select the manufacturer to be Schneider Electric so that the MODBUS TCP Master driver can be activated. The IP address of the MPiec controller is set up in this section as well. In this example, the IP address of the HMI is 192.168.1.3 and the controller IP address is 192.168.1.1



Display Unit
Series GP3000 Series
Model AGP-3650T
Orientation Landscape

Device/PLC

Summary
Manufacturer: Schneider Electric Industries
Series: MODBUS TCP Master
Port: Ethernet (TCP)
Text Data Mode: 1

Communication Settings
Timeout: 3 (sec)
Retry: 0
Wait To Send: 0 (ms)

Device-Specific Settings
Allowable Number of Devices/PLCs 16

Number	Device Name	Settings
1	PLC1	IP Address=192.168.001.001,Port No.=502,Unit ID=255,Rest of the bits in this w

Figure 2: Configuration of Pro-face as MODBUS master

Set up the user interface and variables on the HMI side. An example is shown in Figure 3.

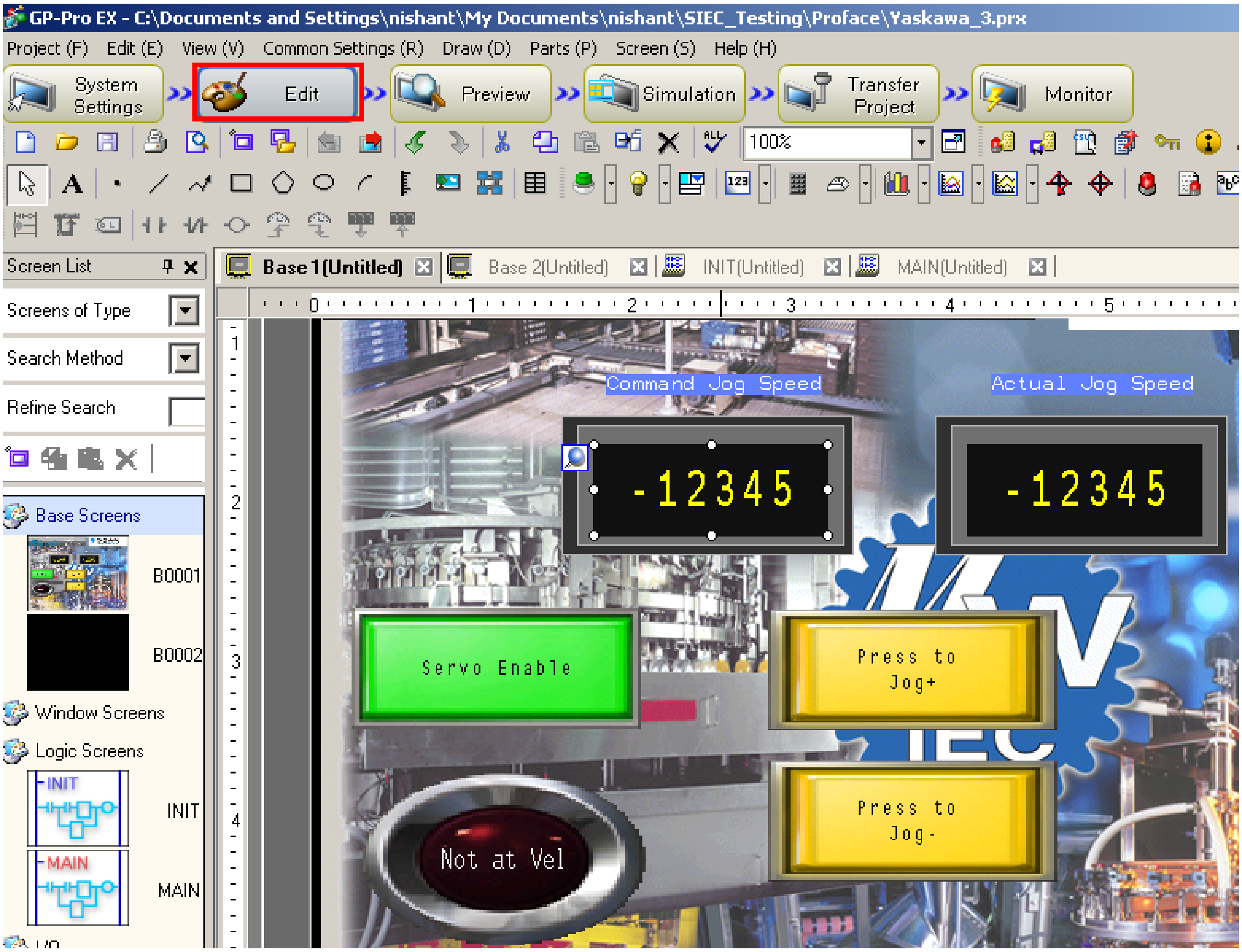
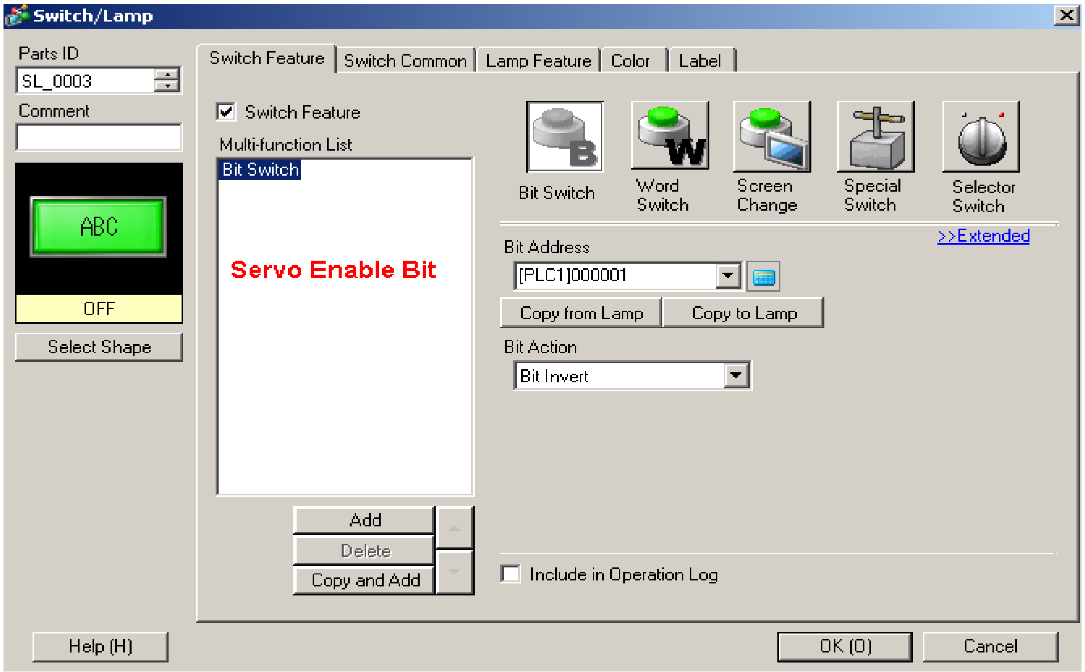


Figure 3: User Interface set up

Configure the variables such that the appropriate MODBUS addresses are allocated to the variables. Two such variable configurations are shown in figures 4 and 5. Figure 4 is an example of a coil set from the HMI. Figure 5 is an example of a 32 bit floating point data type used to send commanded speed to the MPiec controller.



Parts ID SL_0003

Comment

☑ Switch Feature

Multi-function List: Bit Switch

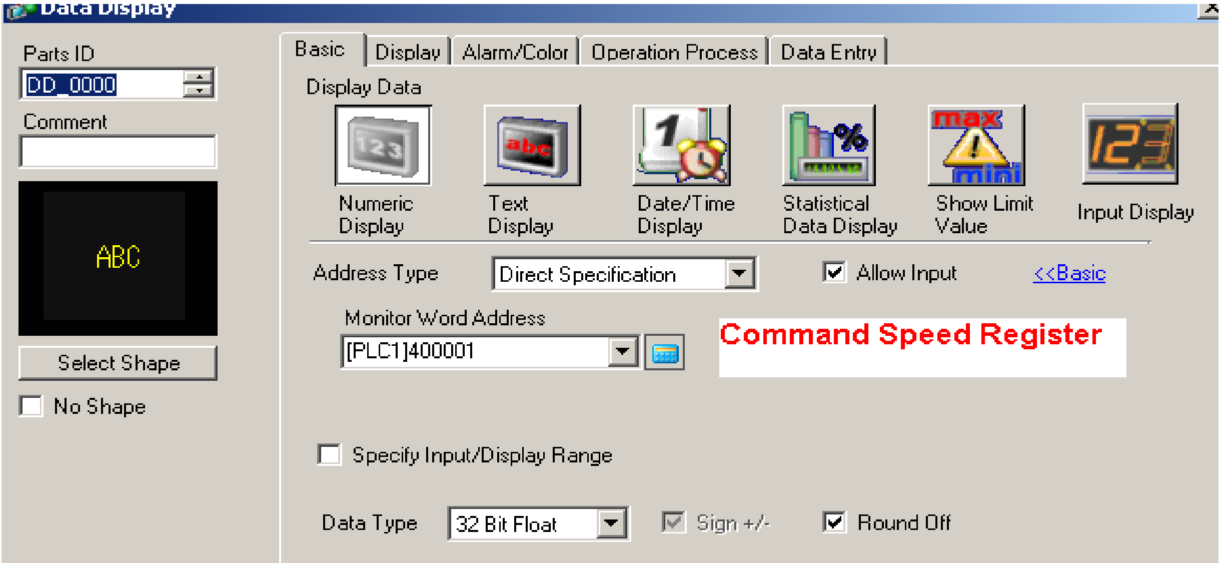
Servo Enable Bit

Bit Address [PLC1]000001

Bit Action Bit Invert

☐ Include in Operation Log

Figure 4: Set coil using FC 5



Parts ID DD_0000

Comment

☐ No Shape

Address Type Direct Specification

☑ Allow Input

Monitor Word Address [PLC1]400001

Command Speed Register

☐ Specify Input/Display Range

Data Type 32 Bit Float

☑ Sign +/-

☑ Round Off

Figure 5: Setting up floating point data (FC 6/16)

Figure 6 is an illustration of the various function codes supported by the MPiec controller when it is used as a server in MODBUS protocol. This diagram will be useful in planning variables before setting them up in the MP controller.

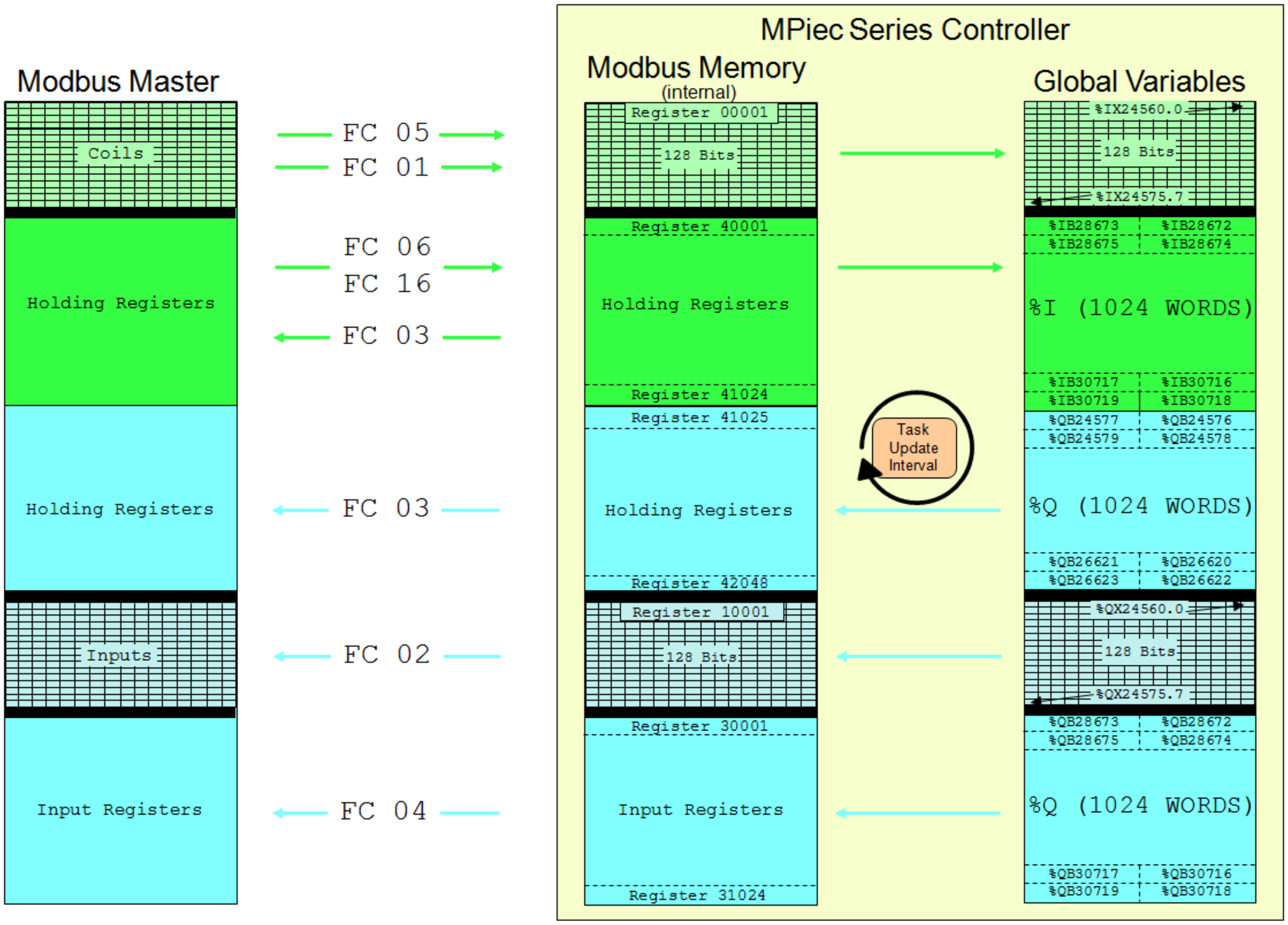
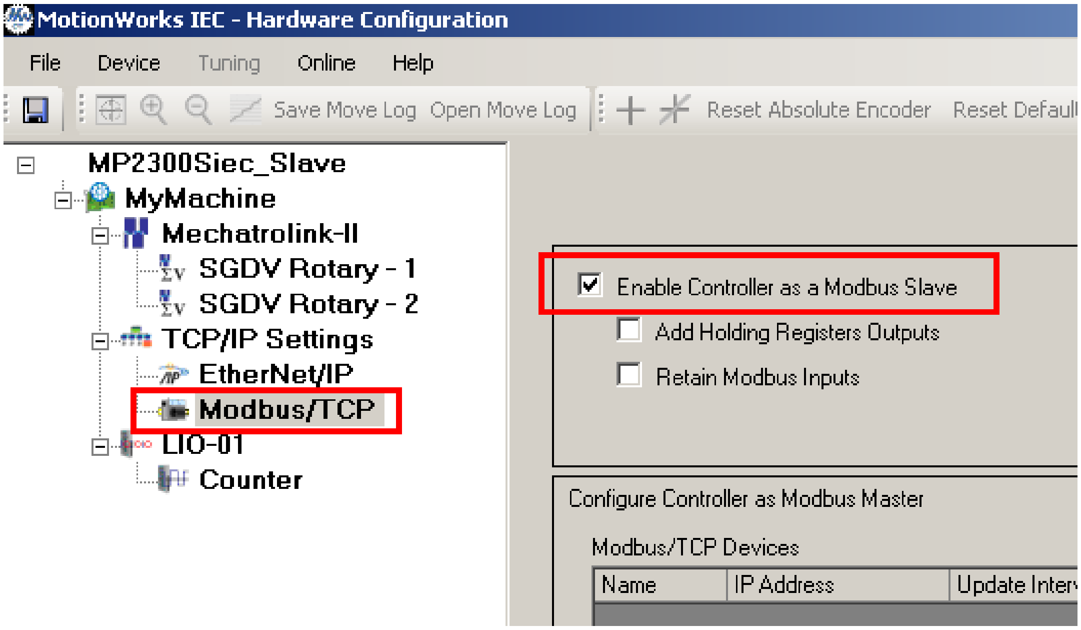


Figure 6: Supported function codes when the MPiec is a MODBUS server

To set up the MPiec controller as a server, open the Configuration tool. Once online with the controller, click the MODBUS option under the hardware tree. Select the option such which makes the controller a MODBUS server (Figure 7). Save the configuration. Cycle power on the whole system. The various MODBUS groups and the addresses of the groups can be seen in the global variables worksheet as shown in Figure 8.



☑ Enable Controller as a Modbus Slave

□ Add Holding Registers Outputs

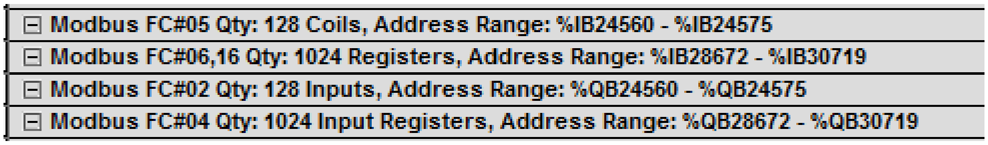
□ Retain Modbus Inputs

Configure Controller as Modbus Master

Modbus/TCP Devices

Name	IP Address	Update Inter

Figure 7: Hardware configuration on MPiec



⊟ Modbus FC#05 Qty: 128 Coils, Address Range: %IB24560 - %IB24575

⊟ Modbus FC#06,16 Qty: 1024 Registers, Address Range: %IB28672 - %IB30719

⊟ Modbus FC#02 Qty: 128 Inputs, Address Range: %QB24560 - %QB24575

⊟ Modbus FC#04 Qty: 1024 Input Registers, Address Range: %QB28672 - %QB30719

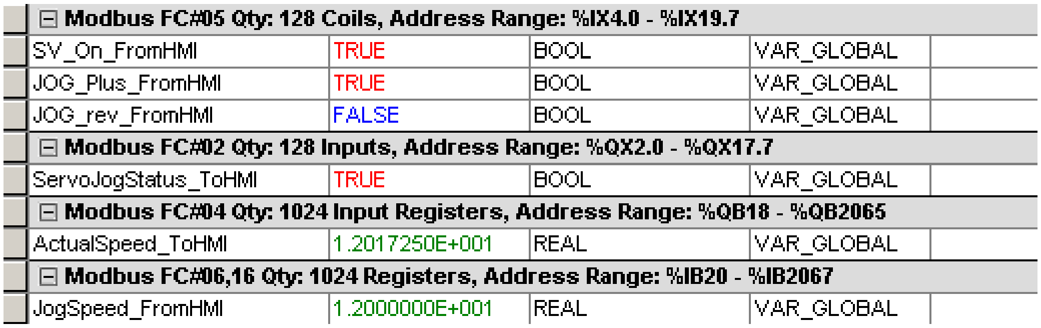
Figure 8: MODBUS groups in MotionWorks IEC

Create variables on the MPiec side and assign them addresses as per the group addresses that got created when the MPiec was configured as the MODBUS server

In this example, the various variables chosen are:

HMI Variable	MODBUS address		Function Code		MP address	MPiec variable
Servo Enable	000001	→	FC 5	→	%IX24560.0	SV_On_FromHMI
Press to Jog+	000002	→	FC 5	→	%IX24560.1	JOG_Plus_FromHMI
Press to Jog -	000003	→	FC 5	→	%IX24560.2	JOG_rev_FromHMI
Not at Vel	100001	←	FC 2	←	%QX24560.0	ServoJogStatus_toHMI
Command Jog Speed	400001	→	FC 6/16	→	%ID28672	JogSpeed_FromHMI
Actual Jog Speed	300001	←	FC 4	←	%QD28672	ActualSpeed_ToHMI

Compile the project. Download the project into the controller and cold start the controller. The program runs and communicates with the Pro-face HMI as can be seen from Figure 9.



⊟ Modbus FC#05 Qty: 128 Coils, Address Range: %IX4.0 - %IX19.7				
SV_On_FromHMI	TRUE	BOOL	VAR_GLOBAL	
JOG_Plus_FromHMI	TRUE	BOOL	VAR_GLOBAL	
JOG_rev_FromHMI	FALSE	BOOL	VAR_GLOBAL	
⊟ Modbus FC#02 Qty: 128 Inputs, Address Range: %QX2.0 - %QX17.7				
ServoJogStatus_ToHMI	TRUE	BOOL	VAR_GLOBAL	
⊟ Modbus FC#04 Qty: 1024 Input Registers, Address Range: %QB18 - %QB2065				
ActualSpeed_ToHMI	1.2017250E+001	REAL	VAR_GLOBAL	
⊟ Modbus FC#06,16 Qty: 1024 Registers, Address Range: %IB20 - %IB2067				
JogSpeed_FromHMI	1.2000000E+001	REAL	VAR_GLOBAL	

Figure 9: Online values from Pro-face - MPiec communication