Figure 1: MT6070iE Serial HMI

1. Introduction

The MT6070iE is a master by default and is able to connect to a single PLC slave via RS232 or RS485.

It is possible to configure the MT6070iE as a Modbus slave (either ASCII or RTU) so that the PLC can become the master.

This document will briefly describe the benefits and considerations of using the HMI as a slave and then describe how to configure the HMI as a slave as well as how to interact with the HMI from the PLC program. A sample program for the HMI called "HMI Configured as Modbus RTU Slave.mtp" (done in EBPRO) and a sample program for the PLC called "MT6070iE Configured as Modbus RTU Slave.PC6" (done in TRiLOGI) will be packaged with this document for download and will be referenced in this document for example purposes. The configuration process will be described for Modbus RTU only since both sample programs use Modbus RTU.

A. Benefits

The PLC can communicate with additional devices that are configured with the same protocol as the HMI (either Modbus ASCII or RTU).

The PLC CPU will not be interrupted frequently as the PLC program will dictate how often the PLC communicates with the HMI.

The PLC program can check every command to verify a successful response was received.

B. Considerations

HMI objects are not directly linked PLC I/O, register, or memory, so the PLC must be programmed to map the necessary data to the HMI.

The TRi_PLC protocol cannot be used when the HMI is a slave; so only integer data can be mapped to the HMI, but not string data.

The HMI cannot directly indicate whether communication has been lost.

2. HMI Configuration and Setup

A. Configure the MT6070iE as a Modbus RTU Slave:

When you start a new project with EBPRO the “System Parameter Settings” window will pop up. This is where you configure the MT6070iE protocol, which will be Modbus RTU as a slave. You can also get to this window by selecting “System Parameters” from the Edit menu. Do the following to setup the HMI protocol:

1. Select “New” so that the “Device Properties” window pops up, which allows a new device to be added – the PLC in this case.
2. Select “MODBUS Server” as the PLC type, which is used for Modbus RTU as a slave. Refer to Figure 1 below.
3. Choose the type of serial connection from the “PLC I/F” menu, which should either be RS232 or RS485 depending on how the HMI will be connected to the PLC.
4. Set the “PLC default station no.” to the ID currently configured for the PLC (default is 01).
5. Click on the “Settings” button beside the COM field to select the HMI COM port being used and to configure the baud settings. A new window will open where you need to select the COM port and set the baud rate, data bits, parity, and stop bits so that they match what is set in the PLC. The default baud settings for the PLC are 38400, 8 data bits, no parity, and 1 stop bit. Refer to Figure 2 below. For the MT6070iE, choose COM1 since that is the only option.
6. The setup is complete, so you can accept the changes and do any other necessary configuration.

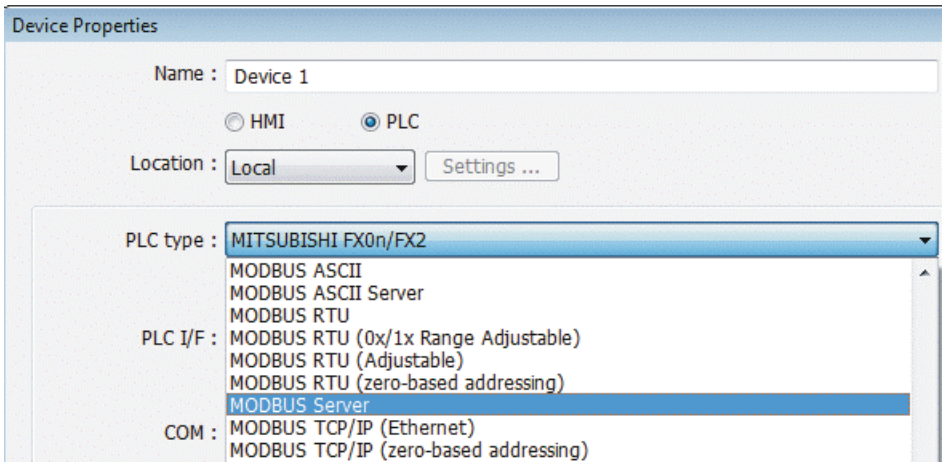

Figure 1: PLC Protocol Configuration in EBPRO

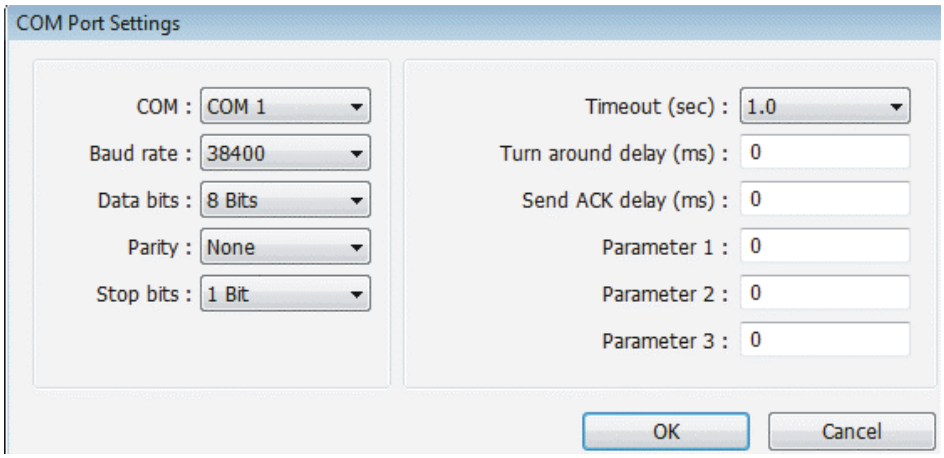

Figure 2: HMI Baud Rate Configuration in EBPRO

B. Word and Bit Addresses used in the HMI

Since the HMI is used a slave, any HMI objects that exchange data with the PLC use the HMI memory locations, which the PLC program will end up reading from and writing to.
The easiest way to map data between the HMI and PLC is to use the "LW" memory locations for data words (i.e. Mapping DM[] variables) and the LW_Bit locations for data bits (ie. Mapping digital I/O).

There are 9000 LW memory locations available to the user in the HMI memory, which are between 0 and 8999 (LW addresses 9000 and up are reserved for the HMI).

The LW_Bit locations are grouped in 16-bit words and shared with the LW locations. This means you need to be careful not to overlap the LW locations being used as words with the ones used as bits.

The LW-Bit addressing is as follows:

Address Range	LW Location	Bit Locations
0 to 15	0	0 to 15 of LW0
100 to 115	1	0 to 15 of LW1
200 to 215	2	0 to 15 of LW1
…		
899900 to 899915	8999	0 to 15 of LW8999

C. Configuring HMI Word and Bit Objects

The sample HMI program "HMI Configured as Modbus RTU Slave.mtp" will be referred to here.

If you open the ND_0 numeric display object properties (the object with the "DM[1]" label above it), you will see the address is set to LW 1. Refer to Figure 3 below. This is all you have to do in the HMI for that particular object because the address will be mapped to DM[1] in the PLC program, since the PLC will be the master.

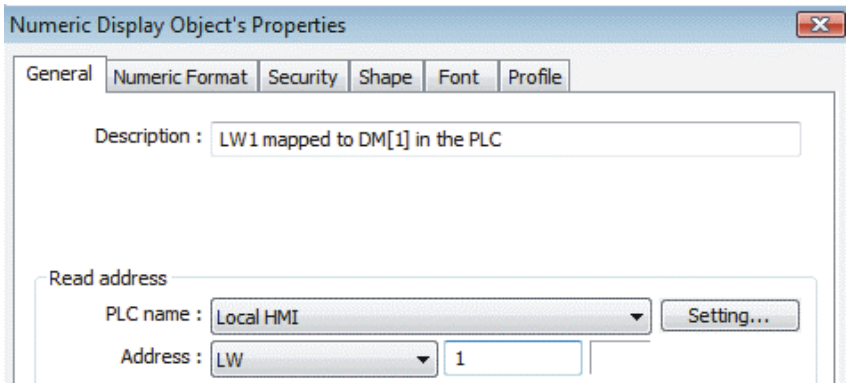

Figure 3: Object properties for ND_0 in HMI program

If you open the BL_0 bit lamp object properties (the object with the “In 1” label on it), you will see the address is set to LW_Bit 400100. Refer to Figure 4 below. As per the above address map table, this is using bit 0 of LW 4001. This is all you have to do in the HMI for that particular object because the address will be mapped to digital input #1 in the PLC program, since the PLC will be the master.

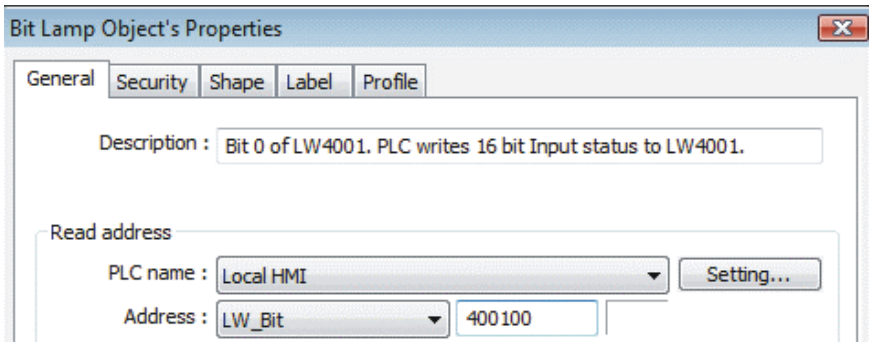

Figure 4: Object properties for BL_0 in HMI program

3. Mapping Data and I/O to the HMI in the PLC Program

Normally, the HMI is the master and the PLC is the slave and no PLC programming is necessary for communication between the HMI and PLC in that scenario. However, the PLC is the master in this case so it is the one that sends commands to read and write data in the HMI.

Since the LW memory locations are being used for data words and bits (LW_Bit) in the HMI, it is possible to use the WRITEMODBUS and READMODBUS commands for all communication because the LW memory is mapped to the 4x Modbus holding register locations that these commands can easily access.

Here is a memory map between the LW locations in the HMI, the Modbus offset address used in the PLC commands, and the data or I/O register it corresponds to.

LW Address in	LW Bit Range	Modbus Offset Address for	Data Register or I/O Range in

HMI		WRITEMODBUS and READMODBUS	the PLC
1	N/A	1	DM[1]
2 …. ….	N/A …. ….	2 …. ….	DM[2] …. ….
4000	N/A	4000	DM[4000]
4001	400100 to 400115	4001	INPUT[1] (1st 16 digital inputs)
4002	400200 to 400215	4002	OUTPUT[1] (1st 16 digital outputs)
5001	N/A	5001	ADC(1) (analog input #1)
5002	N/A	5002	ADC(2) (analog input #2)

NOTE: This memory map is only a suggestion and was used for the HMI and PLC sample programs provided with this write-up; however, a different map between the HMI and PLC could be created. This map could also be extended to include additional PLC registers and memory.

The PLC Code

The PLC sample program included with this document is "MT6070IE Configured as Modbus RTU Slave.PC6". Below are simple examples of PLC code based on the sample program code.

Example #1: Writing the Value of DM[1] to the HMI

You would use the following command in a TBASIC custom function to write the current value of DM[1] into the corresponding LW register in the HMI (per the above memory table).

```
WRITEMODBUS 11,1,1, DM[1]
```

The first parameter 11 means the command is in Modbus RTU and is being sent out of COM1 on the PLC.

The second parameter 1 means the PLC ID is 01. Normally you would put the ID of the slave device, not the PLC, but the MODBUS SLAVE protocol in the MMI requires that the ID be of the master device, which is the PLC in this case.

The third parameter 1 is the MMI memory address - LW1, which is what has been designated as DM[1] for this write-up and the accompanying sample programs.

The fourth parameter DM[1] is the data, which will be the value stored in DM[1].

Example #2: Reading the Value of DM[1000] from the HMI

It is also possible for a DM[] value to be entered on the HMI and read by the PLC. You would use the following command in a TBASIC custom function to read the current value of DM[1000] from the corresponding LW register in the HMI (per the above memory table) and write it into the actual DM[1000] address in the PLC.

```
DM[1000] = READMODBUS (11,1,1000)
```

The three parameters are the same as the first three parameters from the above WRITEMODBUS command, except the address is 1000 and is mapped to LW1000 in the HMI.

IMPORTANT NOTE:
When communicating with the HMI from the PLC, the device ID must be the ID of the master (source device), which is the PLC, instead of the slave (destination device).

Normally you would put the ID of the slave device, not the PLC, but the MODBUS SLAVE protocol in the MMI requires that the ID be of the master device, which is the PLC in this case.