31. Memory Map

This chapter explains the settings relevant to Memory Map.

31.1. Overview

Memory Map communication protocol is similar to IBM 3764R, and it is used when the memory data transferred seldom between two devices. When setting the two devices, one is set as Master, and another is Slave. Generally, Master and Slave do not communicate unless the data in the assigned address has changed. Once the data is synchronized, the communication will stop. The purpose of Memory Map is to keep the consistency of the assigned part of data between two devices (Master and Slave).

The corresponding addresses of Master and Slave devices should have the same property as MW (MB) address type. The size of MW (MB) in HMI is 10,000 words.

MB and MW indicate the same area of memory, for example, MB0~MBf correspond to the bits of MW0, MB10~MB1f correspond to MW1, as shown in the following table:

Device Type	Format	Range
MB	DDDDh	DDDD:0~4095 h:0~f(hex)
MW	DDDD	DDDD:0~9999

31.2. PIN Settings

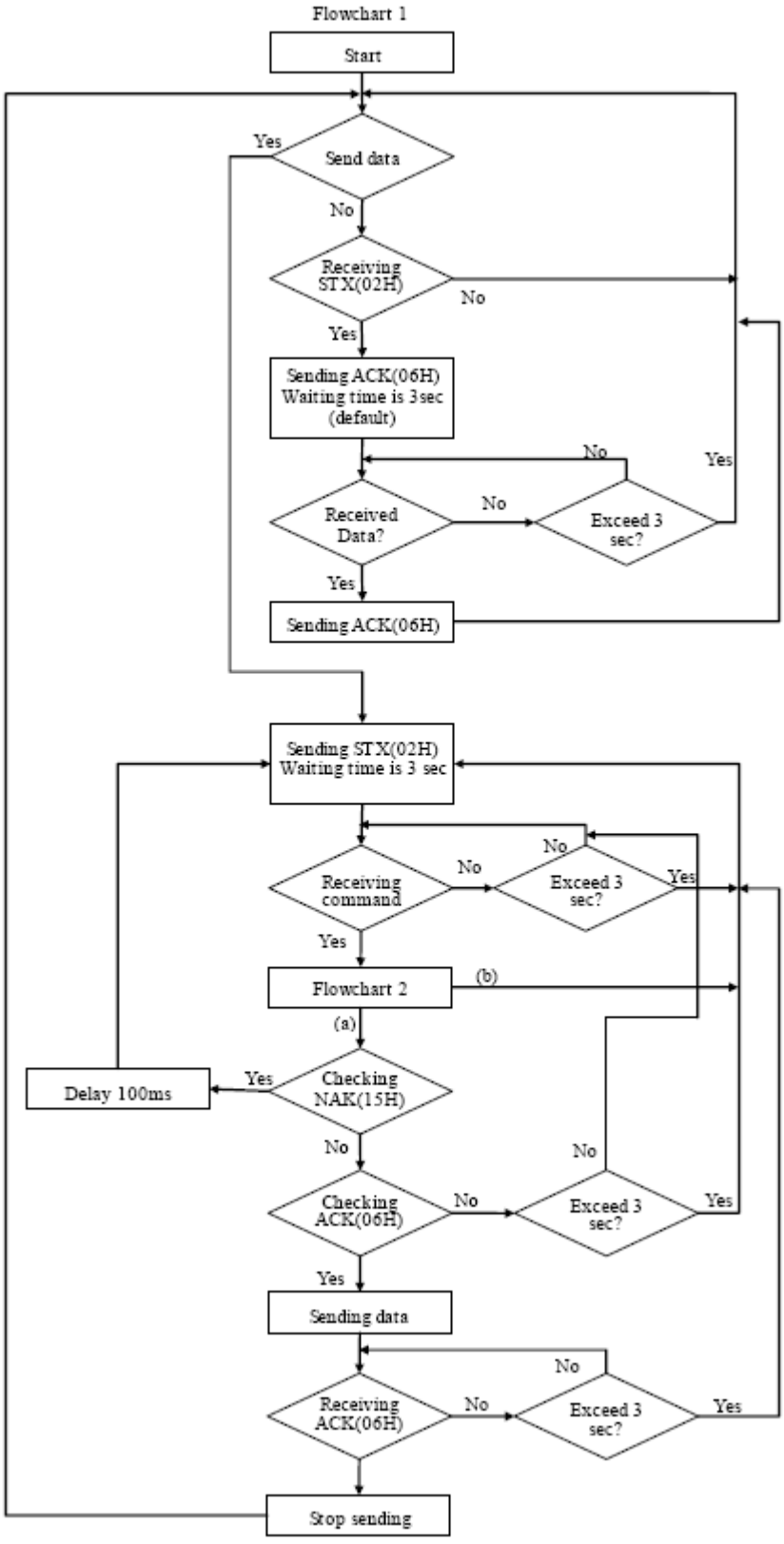
When using Memory Map communication protocol, the Master and Slave must have the same communication parameters. The wiring is shown in the following table:

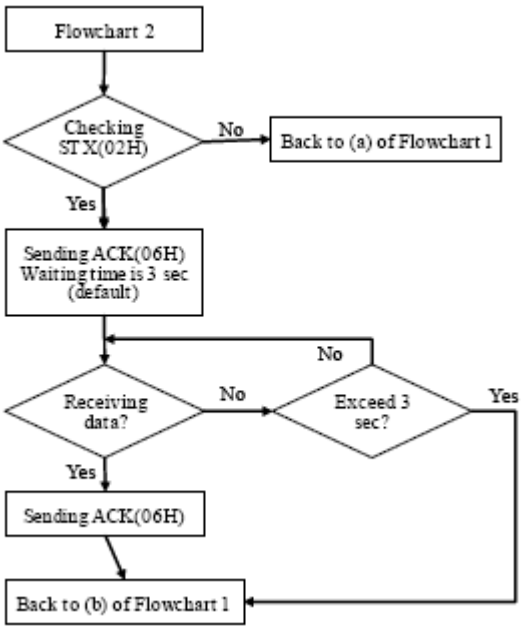
(the # will be distinct depends on the type of PLC or controller.)

COM Port	RS-232	
Device	Master	Slave
Pin Mapping	TX(#)	RX(#)
	RX(#)	TX(#)
	GND(#)	GND(#)

COM Port	RS-485 (4W)	
Device	Master	Slave
Pin Mapping	TX+(#)	RX+(#)
	TX-(#)	RX-(#)
	RX+(#)	TX+(#)
	RX-(#)	TX-(#)
	GND(#)	GND(#)

31.3. Communication Flowchart





Note

- Flowchart 2 works for Slave but not Master.
- STX: Start of Text, ACK: Acknowledge, NAK: Negative Acknowledge

31.4. Address Types

There are two address types, MB and MW.

The format of the commands that controls MB are listed in the following table:

MB Commands

Offset(byte)	Format	Description
0	0x02	The operating sign to MB
1	0x##	Address (Low byte)
2	0x##	Bit Address (High byte) For example: MB-18 = 1*16 + 2 = 18 = 0x12 and 0x00
3	0x00 (or 0x01)	The data in MB address. (Bit type, must be 0 or 1)
4, 5	0x10, 0x03	Stop sign
6	0x##	The checksum. Calculate XOR from offset 0 to 5.

The format of the commands that controls MW are listed in the following table:

MW Commands

Offset(byte)	Format	Description
0	0x01	The operating sign to MW
1 2	0x## 0x##	Address (Low byte) Bit Address (High byte) If the address includes 0x10, insert another 0x10 after it and all offsets after that are increased by 1. For example: 0x10, 0x04 will become 0x10,0x10,0x04
3	0x##	Number of sending bytes (To control a word, the number of bytes must be even). If the number of bytes is 0x10, insert another 0x10 after it and all offsets after that are increased by 1.
4 to 4+n-1	0x##(L),0x##(H) 0x##(L),0x##(H) …	The address that the first and second bytes correspond to is the initial address. “n” is the number of bytes. If the data includes 0x10, insert another 0x10 after it and the “Number of sending bytes” (offset 3) remains the same, but n = n + 1. Same thing applies to other 0x10 data.
4+n, 4+n+1	0x10 0x03	End sign
4+n+2	0x##	The checksum. Calculate XOR from all above.

31.4.1. Communication Examples

Example 1

If Master sets the data of MW-3 to 0x0a, Master will build communication with Slave immediately due to the data changed, so Slave will update its MW-3 to 0x0a, the procedure is:

1. Master sends STX(0x02h).
2. Slave receives STX(0x02h) from Master, and sends ACK(0x06h) to Master.
3. Master receives ACK(0x06h) from Slave.
4. Master sends 0x01,0x03,0x00,0x02,0x0a,0x00,0x10,0x03,0x19, as shown in the following table:

Offset(byte)	Format	Description
0	0x01	The operating sign for MW
1	0x03	Address(Low byte)
2	0x00	Bit Address (High byte)
3	0x02	The number of bytes sent (MW-3= two bytes).
4, 5	0x0a, 0x00	Data in MW-3 is 0x0a and 0x00
6, 7	0x10, 0x03	End sign
8	0x19	The checksum 0x01^0x03^0x00^0x02^0x0a^0x00^0x10^0x03=0x19

5. Slave receives data from Master and then sends ACK(0x06h).
6. Master receives ACK(0x06h) from Slave.

When finish communicating, Master sends the updated data in MW to Slave, and Slave synchronizes its MW data with Master.

Example 2

If the data includes 0x10; please notice the change in data format.

If MW-10 of Slave is set to 0x10, Slave will build communication with Master immediately, and Master will update its MW-10 to 0x10, the procedure is:

1. Slave sends STX(0x02h)
2. Master receives STX(0x02h) from Slave, and sends ACK(0x06h) to Slave.
3. Slave receives ACK(0x06h) from Master
4. Slave sends 0x01,0x10,0x10,0x00,0x02,0x10,0x10,0x00,0x10,0x03,0x10 as shown in the following table:

Offset(byte)	Format	Description
0	0x01	The operating sign to MW
1	0x10	Address(Low byte)
2	0x10	Insert 0x10
3	0x00	Bit Address (High byte)
4	0x02	The number of bytes sent (MW-10= two bytes).
5	0x10	0x10 is the low byte in MW-10
6	0x10	Insert 0x10
7	0x00	0x00 is the high byte
8	0x10	End sign
9	0x03	
10	0x10	The checksum, 0x01^0x10^0x10^0x00^0x02^0x10^0x10^0x00^0x10^0x03=0x10

5. Master receives data from Slave and sends ACK(0x06h) to Slave.
6. Slave receives ACK(0x06h) from Master.

Slave sends the updated data in MW to Master, and Master synchronizes its MW data with Slave.

31.5. Settings

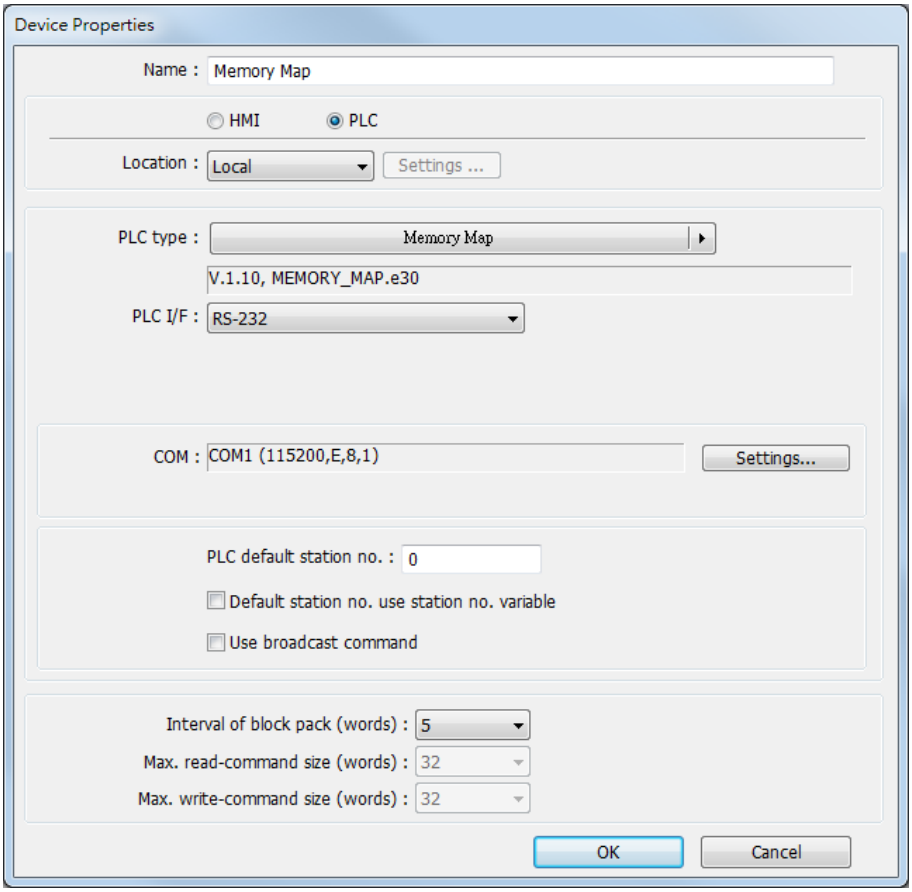
The following explains how to connect two HMIs using Memory Map protocol.

Note

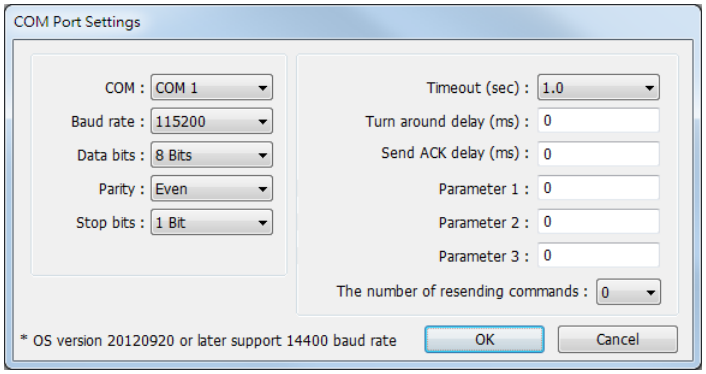
- If the type of these two HMIs are different, please create different project files, or, after setting the first HMI, directly change to the type of the second HMI in [Home] » [System Parameters] » [Model], and then compile and download the project to the second HMI.

31.5.1. Steps to Add a Memory Map Device

1. Launch EasyBuilder Pro, select [New], and select an HMI model.
2. Click [Home] » [System Parameters] » [Device] tab, then click [New] to add a new device.
3. In the [Name] field enter “Memory Map”, and then select [PLC], set the [Location] to [Local].
4. Set [PLC type] to [Memory Map], and set [PLC I/F] to [RS-232].



5. Click [Settings], and the setting is shown in the following figure.



6. After setting the COM port click [OK].
7. Click [OK] to finish setting.

Note

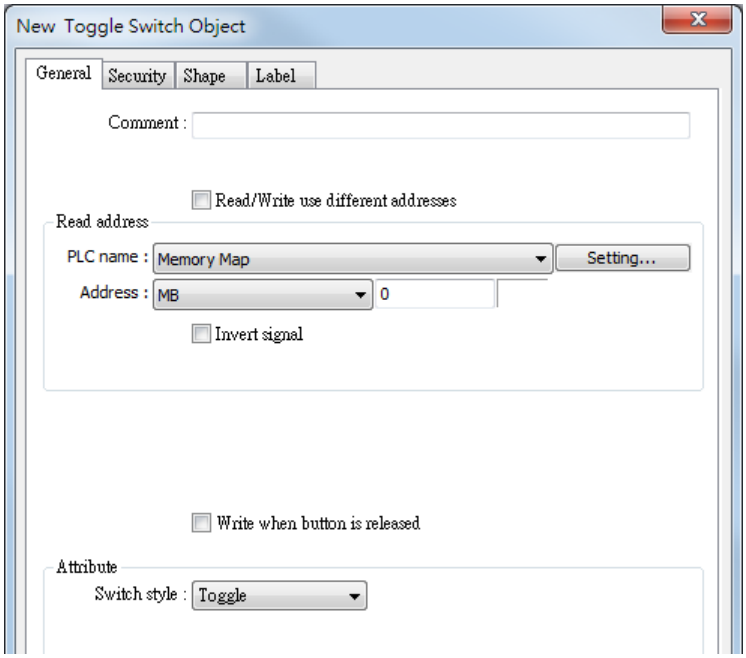
- Memory Map in MT500 is divided into [Memory Map_Master] and [MemoryMap_Slave]; please refer to the relevant manual.
- For eMT3000 and MT8000 Series, select [Memory Map] in the PLC type setting.
- [Data bit] must set to 8 bits.
- All the settings of the two HMIs must be the same.

31.5.2. Object Settings

Add two objects in window no. 10, a Toggle Switch and a Multi-state Switch:

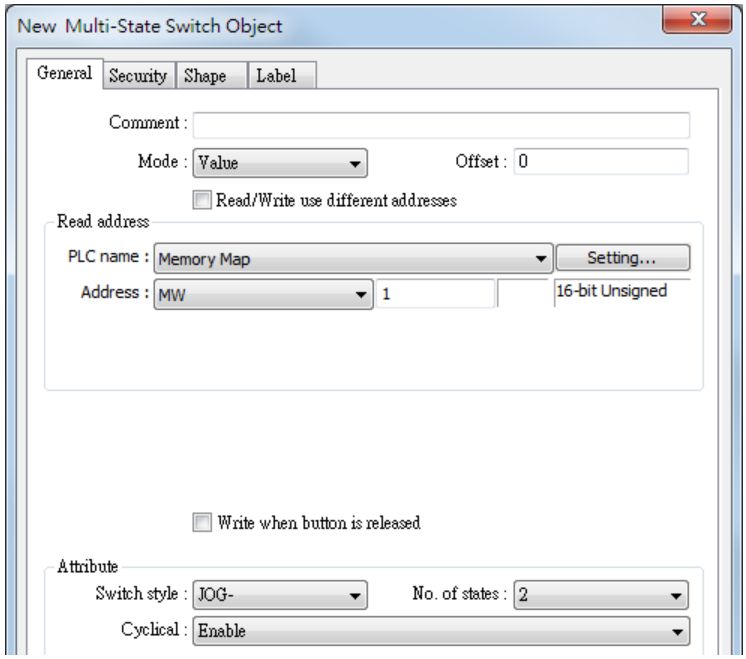
Create a Toggle Switch Object as shown in the following steps.

1. Set the [PLC name] of read address and write address to [Memory Map].
2. Set [Address] to MB-0.
3. Set [Switch style] to [Toggle]. (The picture and label of the object can be selected).



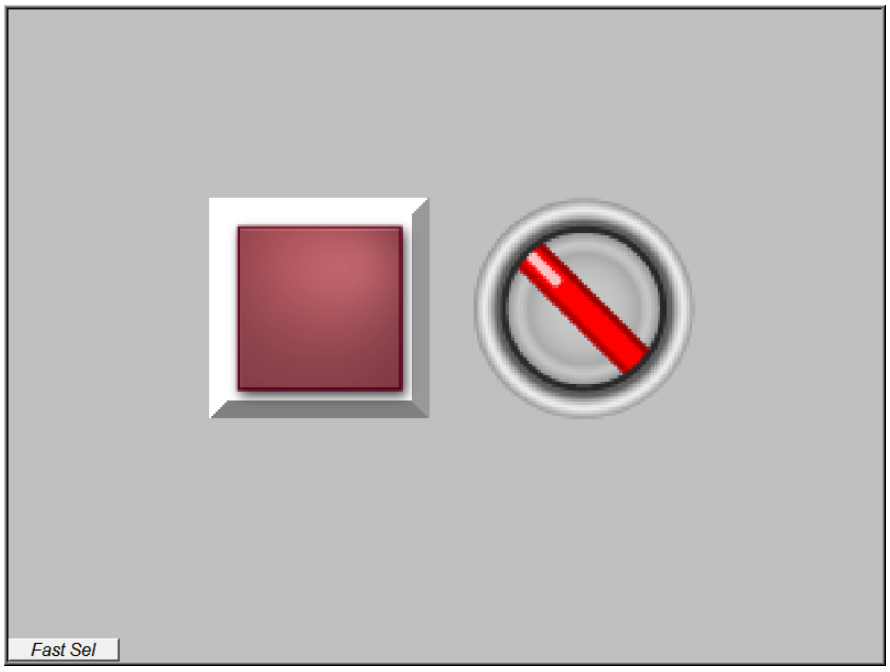
Create a Multi-state Object as shown in the following steps.

1. Set the [PLC name] of read address and write address to [Memory Map].
2. Set [Address] to MW-1.
3. Set [Cyclical] to [Enable]. (The picture and label of the object can be selected).



31.5.3. Executing the Settings

Compile and download the same project to HMI 1 and HMI 2.



When pressing the button in one of the HMIs, the status of another one will also be changed. The way to connect a HMI with a controller is similar to the example above. The data in the same addresses of the two devices are kept identical.