TECHNICAL NOTE
MOTION PRODUCT AND ENGINEERING GROUP

YASKAWA
A World of Automation Solutions™

Subject: Differences between Modbus and Memobus
Product: MP9xx

Summary:
Yaskawa MP9xx products use Memobus communication. This protocol is very similar to Modbus. This document outlines the differences between the two protocols.

Further Documentation:

This document describes the difference between Memobus and Modbus. For additional detailed information, please refer to the following documents:

- SIE-C815-13.60E "Memobus Descriptive Information"
- SIEZ-C887-1.2 "Machine Controller MP9xx Ladder Programming Users Manual"

Memobus Basics:

When communicating with an MP9xx series controller and an HMI, the HMI should be the master, and the MP9xx should be the slave. Up to 63 slaves can be addressed using Memobus. The DEFAULT HMI Driver protocol settings are as follows:

Transmission rate: 19.2 kbps
Transmission mode: RTU (Remote Terminal Unit)
Data: 8 bits
Parity Check: Even parity
Stop bit: 1 bit
Error Check: CRC-16 (Cyclic Redundancy Check)
Transmission: Half duplex, asynchronous
MP9xx Port: port #2
Slave Address: 1

The MP9xx controller will use a message receive function called "MSG-RCV" in slave mode. This function accepts standard Modbus function codes in its messaging format. Note that physical hardware outputs are not supported by the Memobus protocol. This is for safety and to avoid conflicts due to unplanned writing of direct control outputs between the HMI and MP9xx ladder program.

The HMI Driver should support the following function codes for the MP9xx series controllers:

Function Code	**Register**	**Description**	**Max QTY per Query message**
01H	MBXXXXXX	Read Bit Condition	2000
02H	IBXXXXX	Read Physical Hardware Input	2000
03H	MWXXXXX	Read Contents of Holding Register	125
04H	IWXXXX	Read Contents of Physical Hardware Input Word	125
05H	MBXXXXXX	Write to Single Bit	1
06H	MWXXXXX	Write to Single Holding Register	1
08H	-	Loop-back test	-
09H	MWXXXXX	Read Contents of Holding Register (expanded)	252
0AH	IBXXXXX	Read Physical Hardware Input (expanded)	252
0BH	MWXXXXX	Write to Multiple Holding Register (expanded)	251
0DH	MWXXXXX	Discontinuous read of holding register (expanded)	
0EH	MWXXXXX	Discontinuous write of holding register (expanded)	
0FH	MBXXXXXX	Change conditions of multiple bits (expanded)	
10H	MWXXXXX	Write to Multiple Holding Register	100

Memobus Key Features and differences:

Memobus essentially uses the Modbus protocol. The difference lies in how the MP series controllers interprets the registers. HMI Memobus Drivers should be written with the following considerations:

1) The HMI Memobus driver for the MP9xx series controller should appear in the driver selection list labeled as “Memobus” and “Yaskawa MP Series Controllers”. Driver version number should also be readily available.

2) Bits (1 bit), Words (16 bits), and Long Words (32 bits), and Floating Point words (32 bits) take up the SAME address space (unlike straight Modbus).

Example: MB000001=1 is same as MW00000=1 is same as ML00000=1. This means that 0XXXX coils, 1XXXX input relays, and 3XXXX input registers do not exist.

3) Register numbers are to be mapped directly (no offset).

Example:

Memobus	Modbus
MW00000 =	40001
MW00040 =	40041

4) All bits are to be set up in reverse order, and expressed in hex.

Address range: MB000000 to MB32767F
Value range: on or off (example MB00000F=1 and MB00000F=0)
For example, MB000300 is register MW00030 bit 0 in the MP9**.
MP9** bit 0 = Modbus bit XXXXX.16
MP9** bit F = Modbus bit XXXXX.1 (Note: bit 15 is set up in the MP9** as hex bit F)
Example:

Bit numbers in MP9** register =	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Bit numbers in the Modbus tag=	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

Memobus Bit #:	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
MW00100:	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Modbus Bit #:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

The Memobus protocol supports individual bit read/write only from address 0 to 4095. Above that address, a bit write to the MP controller requires the HMI driver to (1) read the word which the bit is stored in, (2) mask and set that bit, and (3) rewrite that word to the MP controller. This sets up a danger of the HMI and the MP controller becoming out of sync and possibly over-writing if certain precautions are not taken. Programmers must consider this when assigning bits for read/write usage between the MP9xx and the HMI. It is important to organize all bits within a 16-bit word to be either a read or a write function only.

5) Word holding registers consist of 16 bits (two bytes) and are written directly. Address is expressed in decimal.

Address range: MW00000 to MW32767, Value range: -32768 to +32767

See chart below:

Memobus word:

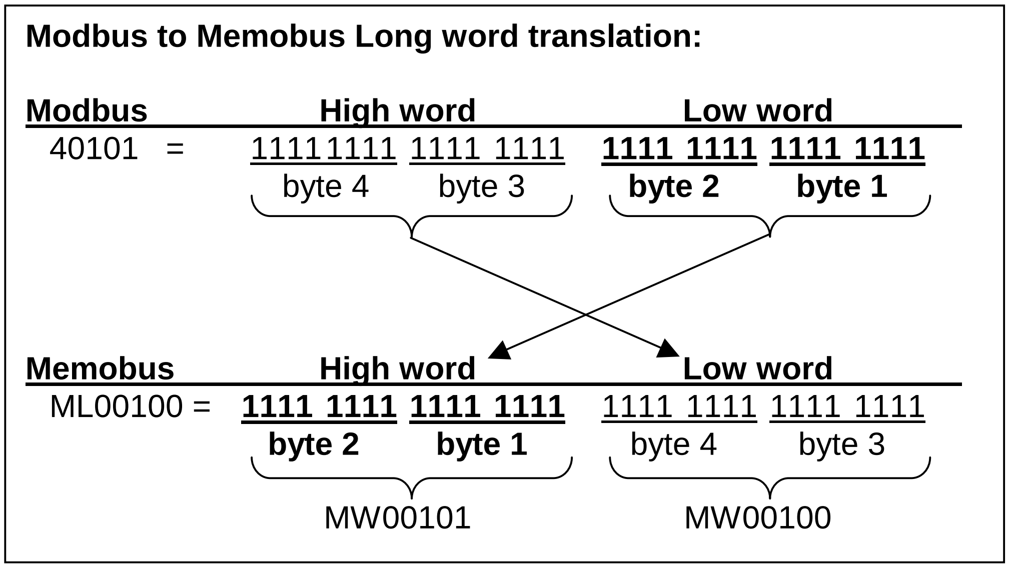
$$\text{MW00100} = \underbrace{1111\ \ 1111}_{\text{byte 2}}\ \ \underbrace{1111\ 1111}_{\text{byte 1}}$$

Long words consist of 32 bits (two words), high and low words are to be swapped. Address is expressed in decimal.

Address range: ML00000 to ML32766, Value range: -2147483648 to +2147483647

Example for MP930:
ML00100 consists of MW00100 (low word) and MW00101 (high word)

See chart below:



6) Floating point words are 32 bits and are in Intel 486 Single precision format (IEEE Floating Point). The address is expressed in decimal.

   Address range: MF00000 to MF32766, Value range: +/- (1.175E-38 to 3.402E+38),0

   Intel 486 single precision format: Bit 31=sign, bit 23-30 is biased exponent, bit 0-22 is significant.

   All MP Series products round to display real numbers (floating point) to display 7 digits (if the 8th digit is 5 or greater, then the 7 digit is rounded up).

7) Input registers are accessible for direct monitoring at the HMI. IB, IW, IF. Address is expressed in hex.

   Address range (MP940): IW0000 to IW07FF (2048 decimal)

   Address range (MP930): IW0000 to IW0FFF (4096 decimal) – Use this as the maximum value.

8) Output registers (physical output points) are not accessible in Memobus.

9) Field entries in HMI should be auto-configurable when the data type is detected after user entry.

Data type	Default # digits	Default Display
Bit	1	Bit
Word	6	Signed 16-Bit
Long word	11	Signed 32-Bit
Floating point word (real)	11	Floating point with exponent