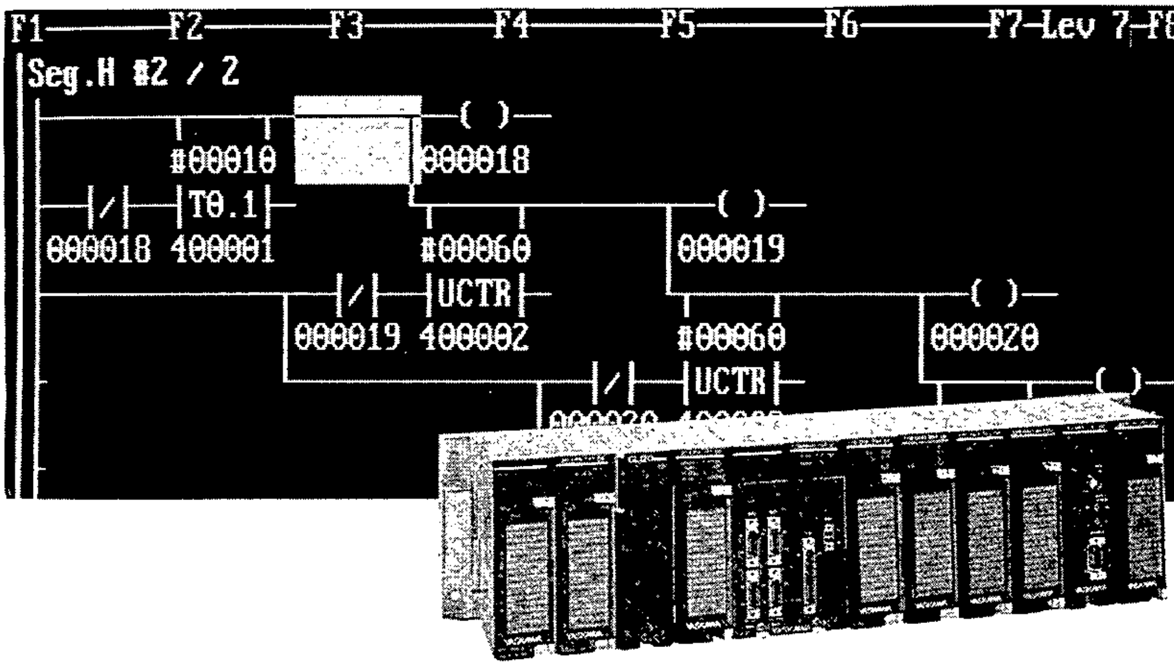
YASKAWA

MEMOCON Micro, GL120, GL130

# MEMOBUS

# USER'S MANUAL




YASKAWA

MANUAL NO. SIEZ-C825-70.13

# Manual Contents

This manual describes specifications and applications of the MEMOBUS industrial communications system for GL120, GL130, and Micro PLCs.

Please read this manual carefully and be sure you understand the information provided before attempting to use the MEMOBUS.

# Visual Aids

The following aids are used to indicate certain types of information for easier reference.

| | |
|---|---|
| ☞ | Indicates references for additional information. |
| IMPORTANT | Indicates important information that should be memorized. |
| ◀EXAMPLE▶ | Indicates application examples. |
| INFO | Indicates supplemental information. |
| ◆ SUMMARY | Indicates a summary of the important points of explanations. |
| **Note** | Indicates inputs, operations, and other information required for correct operation but that will not cause damage to the device. |
| TERMS | Indicates definitions of terms used in the manual. |

# NOTICE

The following conventions are used to indicate precautions in this manual. Failure to heed precautions provided in this manual can result in injury to people or damage to the products.

| | |
|---|---|
| ⚠ **WARNING** | Indicates precautions that, if not heeded, could possibly result in loss of life or serious injury. |
| ⚠ **Caution** | Indicates precautions that, if not heeded, could result in relatively serious or minor injury, damage to the product, or faulty operation. |

# CONTENTS


**CHAPTER 1 Introduction and Precautions** ........ **1-1**

1.1 Overview of Manual ........ 1-2
1.2 Precautions ........ 1-3
1.2.1 Safety Precautions ........ 1-3
1.2.2 Installation Precautions ........ 1-4
1.2.3 Wiring Precautions ........ 1-4
1.2.4 Application Safety Precautions ........ 1-5
1.2.5 MODEM Power ON/OFF Precautions ........ 1-6
1.2.6 Maintenance ........ 1-6
1.3 Using this Manual ........ 1-7

**CHAPTER 2 MEMOBUS Overview** ........ **2-1**

2.1 MEMOBUS Configurations ........ 2-2
2.1.1 What is the MEMOBUS? ........ 2-2
2.1.2 Masters and Slaves ........ 2-2
2.1.3 Typical 1:1 (Point-to-point) Configuration ........ 2-3
2.1.4 Typical 1:1 (Point-to-point) Configuration Connected by Modems ........ 2-3
2.1.5 Typical 1:N (Multi-drop) Configuration Connected by Modems ........ 2-4
2.2 Description of Functions ........ 2-5
2.2.1 MEMOBUS Application ........ 2-5
2.2.2 Transmission Procedure ........ 2-5
2.2.3 Address Numbers ........ 2-5
2.2.4 Transmission Parameters ........ 2-5
2.2.5 Transmission Mode ........ 2-6
2.3 System Configurations ........ 2-7
2.3.1 System Configuration Example ........ 2-7
2.3.2 Setup Procedures ........ 2-8

**CHAPTER 3 Functions and Specifications** ........ **3-1**

3.1 Functions ........ 3-2
3.2 Specifications ........ 3-3
3.2.1 Transmission Specifications ........ 3-3
3.2.2 Modem Specifications ........ 3-4

**CHAPTER 4 System Configuration** ........ **4-1**

4.1 Computer as Master ........ 4-2
4.1.1 Description ........ 4-2
4.1.2 Computer Master 1:1 Configurations ........ 4-3
4.1.3 Computer Master 1:1 Configuration Connected by Modems ........ 4-5
4.1.4 Computer Master 1:N Configuration Connected by Modems ........ 4-7
4.2 Programming Panel as Master ........ 4-9
4.2.1 Description ........ 4-9
4.2.2 PP Master 1:1 Configuration ........ 4-10
4.2.3 PP Master 1:1 Configuration Connected by Modems ........ 4-11
4.2.4 PP Master 1:N Configuration Connected by Modems ........ 4-12
4.3 Man-Machine Interface as Master ........ 4-13
4.3.1 Description ........ 4-13
4.3.2 MMI Master 1:1 Configuration ........ 4-14
4.3.3 MMI Master 1:1 Configurations Connected by Modems ........ 4-15
4.3.4 MMI Master 1:N Configuration Connected by Modems ........ 4-16

# CONTENTS


4.4 Programmable Controller as Master ........ 4-17
4.4.1 Description ........ 4-17
4.4.2 Micro Master ........ 4-17
4.5 Equipment Connections ........ 4-19
4.5.1 Two-core Twisted Cable Configuration ........ 4-19
4.5.2 Connections Between J2078 Modems ........ 4-21
4.5.3 J2078 Wiring ........ 4-21
4.5.4 Power ON/OFF Sequence ........ 4-24
4.6 Transmission Parameters ........ 4-25
4.6.1 What Are Transmission Parameters? ........ 4-25
4.6.2 Types of Transmission Parameters ........ 4-25
4.6.3 Setting Transmission Parameters ........ 4-26

**CHAPTER 5 Signal Transmission Procedures ........ 5-1**
5.1 Message Configuration ........ 5-2
5.1.1 Description ........ 5-2
5.1.2 Slave Address Number ........ 5-3
5.1.3 Function Codes ........ 5-3
5.1.4 Data ........ 5-4
5.1.5 Error Check ........ 5-4
5.2 Transmission Modes ........ 5-6
5.3 Slave Response ........ 5-7
5.3.1 Description ........ 5-7
5.3.2 Response Message in Normal Operation ........ 5-7
5.3.3 Response Messages Under Error Conditions ........ 5-7
5.3.4 No Response ........ 5-8
5.4 Signal Transmission Times ........ 5-11
5.4.1 Description ........ 5-11
5.4.2 Time Requirement for Each Block ........ 5-11

**CHAPTER 6 MEMOBUS Messages ........ 6-1**
6.1 Message Configurations for RTU Mode ........ 6-3
6.2 Basic RTU Mode Functions ........ 6-6
6.2.1 Reading Coil Status ........ 6-6
6.2.2 Reading the Status of Specific Coils ........ 6-7
6.2.3 Modifying Status of a Single Coil ........ 6-8
6.2.4 Modifying Status of Multiple Coils ........ 6-10
6.2.5 Reading Input Relay Status ........ 6-11
6.2.6 Reading Input Register Contents ........ 6-13
6.2.7 Reading Holding Register Contents ........ 6-14
6.2.8 Writing to a Single Holding Register ........ 6-16
6.2.9 Writing to Multiple Holding Registers ........ 6-17
6.2.10 Writing a Mask to Holding Registers ........ 6-19
6.2.11 Reading/Writing to Multiple Holding Registers ........ 6-20
6.3 RTU Mode Expansion Functions ........ 6-23
6.3.1 Reading Link Coil Status ........ 6-23

# CONTENTS


6.3.2 Modifying Status of a Single Link Coil ........ 6-24
6.3.3 Modifying Status of Multiple Link Coils ........ 6-25
6.3.4 Reading Link Register Contents ........ 6-27
6.3.5 Writing to a Single Link Register ........ 6-29
6.3.6 Writing to Multiple Link Registers ........ 6-30
6.3.7 Reading Constant Register Contents ........ 6-31
6.3.8 Writing to a Single Constant Register ........ 6-33
6.3.9 Writing to Multiple Constant Registers ........ 6-34
6.4 Special RTU Mode Functions ........ 6-37
6.4.1 Loopback Test ........ 6-37
6.4.2 Reading Event Counter ........ 6-38
6.4.3 Reading Communications Status ........ 6-39
6.4.4 Reading Equipment Status ........ 6-41
6.4.5 Reading the FIFO Register Contents ........ 6-42
6.4.6 Reading Special Reference Status ........ 6-44
6.4.7 Modifying Special Reference Status ........ 6-46
6.5 Example Communications Program for the RTU Mode ........ 6-48
6.5.1 CRC-16 Calculations ........ 6-48
6.5.2 Communications Program Example ........ 6-51
6.6 ASCII Mode Message ........ 6-56
6.6.1 Message Configuration ........ 6-56
6.6.2 Query Message Transmission ........ 6-59
6.6.3 Receiving the Response Message ........ 6-61
6.6.4 Message Example ........ 6-62
6.6.5 LRC Calculations ........ 6-65

**CHAPTER 7 Hardware Management ........ 7-1**
7.1 Module ........ 7-2
7.1.1 J2078 Modem ........ 7-2
7.1.2 CPU20 and CPU30 ........ 7-7
7.1.3 Micro ........ 7-9
7.2 RS-232C Cable ........ 7-11

**CHAPTER 8 Wiring ........ 8-1**
8.1 Wiring 2-core Twisted Cable on a Panel ........ 8-2
8.2 Wiring 2-core Twisted Cable off a Panel ........ 8-2
8.3 Grounding 2-core Twisted Cable ........ 8-3
8.4 RS-232C Cable ........ 8-3

# Introduction and Precautions

# 1



This chapter gives precautions and warnings concerning the use of this product and the manual. **You must read this chapter before reading the rest of the manual or using the product.**


**1.1 Overview of Manual ..................... 1-2**

**1.2 Precautions ............................. 1-3**

1.2.1 Safety Precautions ............................ 1-3
1.2.2 Installation Precautions ......................... 1-4
1.2.3 Wiring Precautions ............................ 1-4
1.2.4 Application Safety Precautions .................. 1-5
1.2.5 MODEM Power ON/OFF Precautions ............ 1-6
1.2.6 Maintenance ................................. 1-6

**1.3 Using this Manual ...................... 1-7**

# 1.1 Overview of Manual

- This manual describes how to use the MEMOBUS industrial communications system. Read this manual carefully to ensure the proper use of the MEMOBUS. Also, keep this manual in a safe place so that it can be used whenever necessary.

- Refer to the following manuals for related Peripheral Devices and Modules.

| Manual | Manual number | Contents |
|---|---|---|
| MEMOCON GL120, GL130 Hardware User's Manual | SIEZ-C825-20.1 | Describes the functions, specifications, and handling methods of GL120 and GL130 hardware. |
| MEMOCON Micro Hardware User's Manual | SIEZ-C825-10.1 | Descriptions of functions and specifications of Micro hardware. |
| MEMOCON Micro Software User's Manual | SIEZ-C825-10.11 | Descriptions of software programming for Micro. |
| MEMOCON GL120, GL130 Programming Panel P120 (MEMOSOFT) User's Manual | SIEZ-C825-60.7 | Describes the functions, specifications, and usage of the Programming Panel P120 (with built-in MEMOSOFT). |
| MEMOCON Micro, GL120, GL130 MEMOSOFT for DOS User's Manual | SIEZ-C825-60.10 | Describes the features and applications of DOS version of MEMOSOFT. |

- Thoroughly check the specifications and conditions or restrictions of the product before use.

# 1.2 Precautions

This section outlines general precautions that apply to using this manual and the product. **You must read this section first before reading the remainder of the manual.**

1.2.1 Safety Precautions ............ 1-3
1.2.2 Installation Precautions ............ 1-4
1.2.3 Wiring Precautions ............ 1-4
1.2.4 Application Safety Precautions ............ 1-5
1.2.5 MODEM Power ON/OFF Precautions ............ 1-6
1.2.6 Maintenance ............ 1-6



## 1.2.1 Safety Precautions

- MEMOCON was not designed or manufactured for use in devices or systems that concern people's lives. Users who intend to use the product described in this manual for special purposes such as devices or systems relating to transportation, medical, space aviation, atomic power control, or underwater use must contact Yaskawa Electric Corporation beforehand.

- This product has been manufactured under strict quality control guidelines. However, if this product is to be installed in any location in which a failure of MEMOCON involves a life and death situation or in a facility where failure may cause a serious accident, safety devices MUST be installed to minimize the likelihood of any accident.

## 1.2.2 Installation Precautions

Abide by the following precautions when installing MEMOCON systems.

⚠**Caution** The installation environment must meet the environmental conditions given in the product catalog and manuals. Using the MEMOCON in environments subject to high temperatures, high humidity, excessive dust, corrosive gases, vibration, or shock can lead to electrical shock, fire, or faulty operation. Do not use the MEMOCON in the following locations.

- Locations subject to direct sunlight or ambient temperatures not between 0 and 60 °C.
- Locations subject to relative humidity in excess of 95%, rapid changes in humidity, or condensation.
- Locations subject to corrosive or flammable gas.
- Locations that would subject the MEMOCON to direct vibration or shock.
- Locations subject to contact with water, oil, chemicals, etc.

⚠**Caution** Install the MEMOCON as described in this product manual. Improper installation can cause product failure, malfunctions, or Modules or other components to fall off.

⚠**Caution** Do not allow wire clippings or other foreign matter to enter the MEMOCON. Foreign matter can cause fires, product failure, or malfunctions.

## 1.2.3 Wiring Precautions

⚠**WARNING** Wiring must be performed by qualified personnel.

- Mistakes in wiring can cause fires, product failure, or malfunctions.

**Note** Insert the interface cables properly.

Insert the connectors of the various interface cables that are to be connected to MEMOCON into the communications ports and attach them properly. Improper insertion of interface cables may cause operational errors in the MEMOCON.

## 1.2.4 Application Safety Precautions

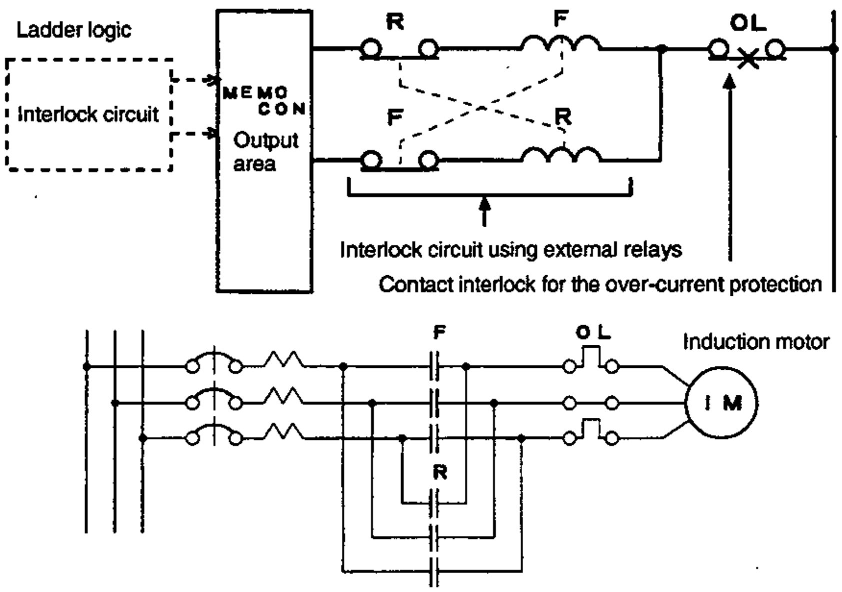
**External Interlocks for the MEMOCON**

Externally connect an interlock to the MEMOCON if there is any chance that MEMOCON failure could result in bodily harm as well as equipment or accessory damage.

⚠ **WARNING** Externally connect an emergency stop, interlock circuits, and other safety switches to the MEMOCON.

An example for reciprocal operation for forward and reverse motor operation is shown below.

An interlock circuit is generally written into the MEMOCON ladder logic program to ensure that forward and reverse signals are not simultaneously output. At the same time, an external interlock circuit must be provided using external relays for the same purpose.




⚠ **WARNING** Operations such as RUN, STOP, forced output, and program change during operation must be carried out with care. Operational errors may damage the machine or cause accidents.

## 1.2.5 MODEM Power ON/OFF Precautions

Be sure to follow the procedures regarding the power ON and OFF sequence given below.

**Note** Be very careful when turning power ON and OFF whenever you are using modems. Turning power to the slave ON and OFF with the modem turned ON will cause the modem to output spurious signals to the 2-core twisted pair cable for several tens of milliseconds. A transmission error is generated when these signal are sent to other modems.
In order to avoid this situation, we recommend a power supply sequencing procedure of turning the slave ON before turning the modem ON, turning the modem OFF before turning the slave OFF, or turning both OFF simultaneously.

## 1.2.6 Maintenance

**⚠ WARNING** Do not attempt to disassemble or modify the MEMOCON in any way. Doing so can cause fires, product failure, or malfunctions.

**⚠ Caution** Make sure that equipment power is turned OFF before mounting or removing the MEMOCON module.

# 1.3 Using this Manual

This manual is written for those who already have a basic knowledge of MEMOCON PLCs. We recommend reading the *MEMOCON GL120, GL130 Hardware User's Manual* before attempting to read this manual.

- **Meaning of Basic Terms**

In this manual, the following terms indicate the meanings as described below, unless otherwise specified.

- **PLC = Programmable (Logic) Controller**

- **PP = Programming Panel**

- **Description of Technical Terms**

The bold technical terms in this manual are briefly explained in the **Glossary** provided at the bottom of the page. An example is shown below.



TERMS

**Glossary**

The following types of terms are described.

- Specific sequence control terms required for explanation of functions.

- Terms that are specific to programmable controllers and electronic devices.

# MEMOBUS Overview

# 2

This chapter introduces MEMOBUS functions and necessary equipment.

2


**2.1 MEMOBUS Configurations ............... 2-2**

2.1.1 What is the MEMOBUS? ....................... 2-2

2.1.2 Masters and Slaves ............................. 2-2

2.1.3 Typical 1:1 (Point-to-point) Configuration .......... 2-3

2.1.4 Typical 1:1 (Point-to-point) Configuration Connected by Modems ..................................... 2-3

2.1.5 Typical 1:N (Multi-drop) Configuration Connected by Modems ..................................... 2-4

**2.2 Description of Functions .................. 2-5**

2.2.1 MEMOBUS Application ........................ 2-5

2.2.2 Transmission Procedure ......................... 2-5

2.2.3 Address Numbers ............................. 2-5

2.2.4 Transmission Parameters ........................ 2-5

2.2.5 Transmission Mode ............................ 2-6

**2.3 System Configurations ................... 2-7**

2.3.1 System Configuration Example ................... 2-7

2.3.2 Setup Procedures ............................. 2-8

# 2.1 MEMOBUS Configurations

This section describes MEMOBUS configurations.

2.1.1 What is the MEMOBUS? ........ 2-2
2.1.2 Masters and Slaves ........ 2-2
2.1.3 Typical 1:1 (Point-to-point) Configuration ........ 2-3
2.1.4 Typical 1:1 (Point-to-point) Configuration Connected by Modems ....... 2-3
2.1.5 Typical 1:N (Multi-drop) Configuration Connected by Modems .......... 2-4

## 2.1.1 What is the MEMOBUS?

1) The MEMOBUS is an industrial communications system for Yaskawa PLCs.

2) The MEMOBUS offers serial communications at a maximum rate of 19,200 baud via the widely used RS-232C standard (see *Table 3.2*). With its simple transmission procedures, the MEMOBUS facilitates the construction of highly reliable network systems.

## 2.1.2 Masters and Slaves

1) MEMOBUS systems are comprised of masters and slaves. Usually one master is connected to between 1 and 32 slaves.

2) A master can be a commercially available computer such as a personal computer. Any computer with an RS-232C port can be used as a master. Yaskawa also offers master devices for this purpose within its product lines. Yaskawa PLCs are used for slaves.

3) The following equipment can be used as masters:

- Commercially available computers (equipped with an RS-232C port).
- Yaskawa Programming Panels for PLCs: P120, P140, and P150.
- Yaskawa FA Monitors: ACGC4000-series and ACGC400-series Monitors.
- Yaskawa PLCs:
  MEMOCON: GL120 and GL130
  MEMOCON Micro
  MEMOCON-SC: GL70H, GL60H, GL60S, GL60S1, GL60S2, GL60S3, GL40S1, GL40S2, and GL40S3

4) The following PLCs can be used as slaves:

- Yaskawa PLCs:
  MEMOCON: GL120 and GL130
  MEMOCON Micro
  MEMOCON-SC: GL70H, GL60H, GL60S, GL60S1, GL60S2, GL60S3, GL40S1, GL40S2, and GL40S3

5) In addition to the equipment listed above, other Yaskawa PLCs can be used as either masters or slaves. Contact your nearest Yaskawa representative for further details on these PLCs.

6) These PLCs are fully interchangeable in the MEMOBUS system. Existing systems that include the GL120, GL130, and Micros can be expanded with ease with the MEMOBUS system.

7) This manual describes PLCs that serve as masters or slaves primarily with the GL120, GL130 and Micros. We also provide separate manuals for other Yaskawa PLCs with master and slave functions, such as the GL70H, GL60H, GL60S, GL60S1, GL60S2, and GL60S3.



## 2.1.3 Typical 1:1 (Point-to-point) Configuration

1) The simplest MEMOBUS system configuration consists of one master and one slave. *Figure 2.1* shows an example of this configuration.

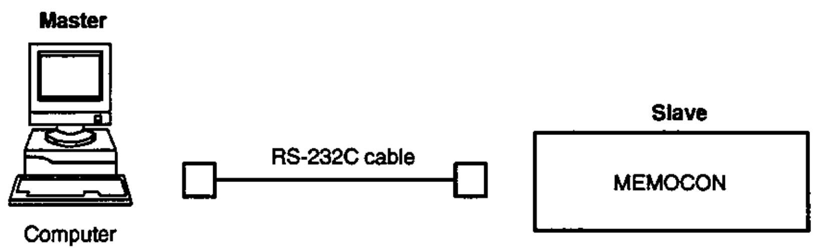



**Figure 2.1 Typical 1:1 MEMOBUS Configuration**

2) The RS-232C cable connecting the MEMOBUS master and slave is a maximum of15 meters in length. A suitable cable can be obtained directly from Yaskawa.

## 2.1.4 Typical 1:1 (Point-to-point) Configuration Connected by Modems

1) A modem is used whenever the distance between the master and the slave exceeds 15 meters. Yaskawa offers MEMOBUS Modems for this purpose, but commercially available modems may also be used in this configuration. The Yaskawa Modem uses a 2-core twisted cable for the transmission path, and thus offers a maximum distance of 4.5 kilometers between the modem on the master side and the modem on the slave side.

2) *Figure 2.2* shows an example of a master and slave configuration connected by modems.

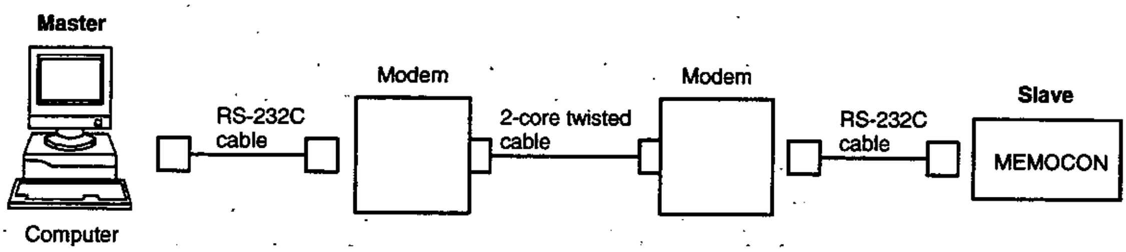



**Figure 2.2 Typical 1:1 MEMOBUS Configuration Connected by Modems**

## 2.1.5 Typical 1:N (Multi-drop) Configuration Connected by Modems

1) A MEMOBUS system normally has only one master, but may include a number of slaves. Modems are required when multiple slaves are connected to a single master, even if the distance between the slaves and master is under 15 meters. Up to 32 slaves can be connected to a single master.

2) *Figure 2.3* shows an example of a master and multiple slaves configuration connected by modems.

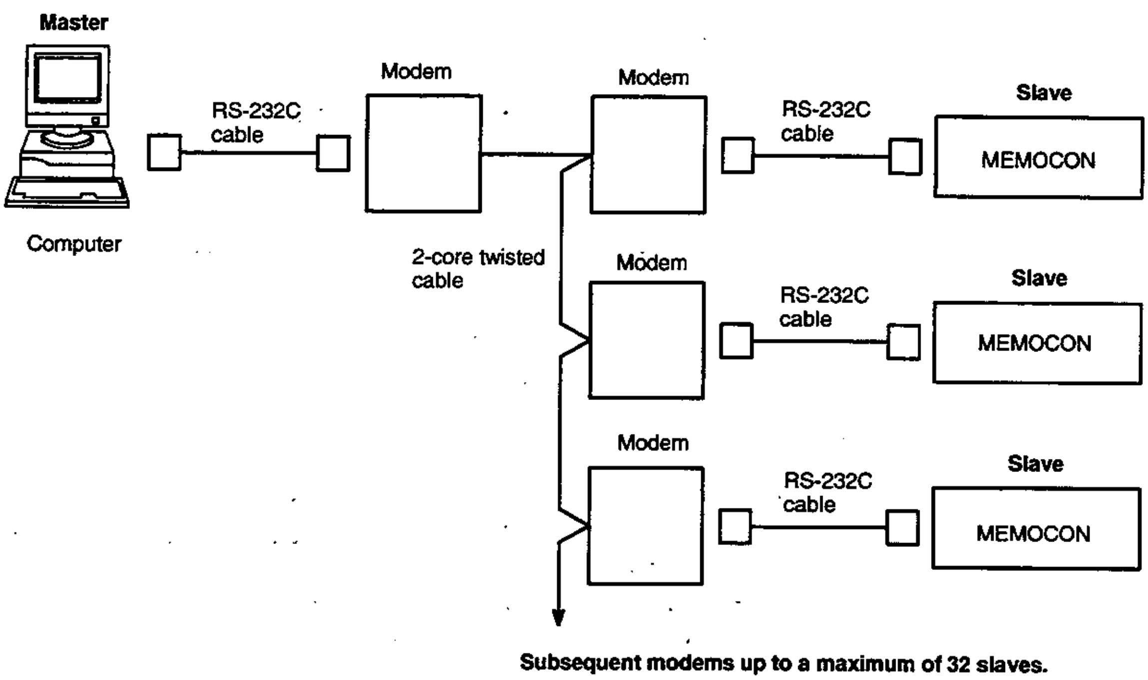



**Figure 2.3 Typical 1:N MEMOBUS Configuration Connected by Modems**

# 2.2 Description of Functions

This section describes the use, transmission procedures, address numbers as well as other functions of MEMOBUS.

2.2.1 MEMOBUS Application ........ 2-5
2.2.2 Transmission Procedure ........ 2-5
2.2.3 Address Numbers ........ 2-5
2.2.4 Transmission Parameters ........ 2-5
2.2.5 Transmission Mode ........ 2-6

## 2.2.1 MEMOBUS Application



The MEMOBUS facilitates centralized monitoring and control of a group of PLCs by currently installed computers.

## 2.2.2 Transmission Procedure

1) The MEMOBUS communicates using the Yaskawa transmission protocol. This protocol uses MEMOBUS messages and is simple to use and understand.

2) In a MEMOBUS system, communications are always initiated by the master, and the slaves then respond. Therefore, a program that controls communications using MEMOBUS messages must be prepared and loaded ahead of time into the computer that will be used as the master. Yaskawa Programming Panels and FA Monitors are already equipped with MEMOBUS message generating functionality, so that adding a simple program for designating functions is all that is needed to use this equipment as a master.

3) Yaskawa PLCs that are suitable as slaves are already equipped with slave functionality.

## 2.2.3 Address Numbers

With MEMOBUS communications, slaves are assigned address numbers in advance. The master specifies the address number in order to send query messages to slaves. Query messages are sent to all slaves at the same time, but only the slave with the right address number executes the command contained in the query message, organizes the necessary data, and returns a response message to the master. This procedure allows the master to read and modify the coil status and register contents of any slave.

## 2.2.4 Transmission Parameters

Various transmission parameters, such as baud rate and parity checks, can be set for MEMOBUS communications. The communications parameters must match on the equipment on both ends of the communications path.

## 2.2.5 Transmission Mode

1) Either the Remote Terminal Unit (RTU) mode or American National Standard Code for Information Interchange (ASCII) mode can be selected for MEMOBUS communications. The data length is eight bits in RTU mode, and seven bits in ASCII mode.

2) The transmission mode must match on the equipment on both ends of the communications path.

# 2.3 System Configurations

This section uses an actual example to describe a MEMOBUS system configuration.


2.3.1 System Configuration Example .................................. 2-7
2.3.2 Setup Procedures ................................................ 2-8


## 2.3.1 System Configuration Example

1) In this example, a communications program written in N88-BASIC is loaded on a personal computer made by NEC (PC-98) in order to read the contents of GL120 and GL130 holding registers.

2) *Figure 2.4* shows equipment used in the example system configuration.

- Master: Personal computer made by NEC (PC-98)
- Slave: GL120 or GL130
- Cable: JZMSZ-120W0200-03
  (2.5-m, RS-232C cable between the personal computer made by NEC (PC-98) and the GL120 or GL130 slave.)

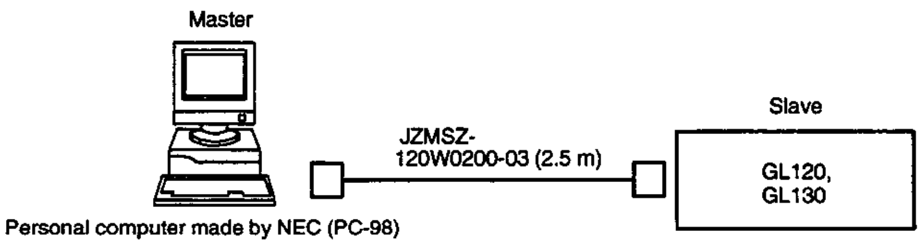



**Figure 2.4 System Configuration Example**

## 2.3.2 Setup Procedures

1) **Setting GL120 and GL130 Address Numbers**

With MEMOBUS communications, slaves must be assigned address numbers prior to communications. In the example, the address number for the GL120 or GL130 is set to 1. The method used to set address numbers is detailed in section *4.6 Transmission parameters.*

The quick way to set the address number is simply to set the address number for the GL120 or GL130 to 1. To do this, set DIP switch pin #1 for the MEMOBUS port for GL120 or GL130 to ON as shown in *Figure 2.5.* This sets the address number for the GL120 or GL130 to 1.

2) **Setting GL120 and GL130 Transmission Parameters**

Various transmission parameters, such as baud rate and parity check, must be set for MEMOBUS communications. In the example the transmission parameters are set as follows:

| | |
|---|---|
| • Baud rate: | 9,600 bps |
| • Parity check: | Provided |
| • Parity: | Even |
| • Number of stop bits: | 1 |
| • Data bit length: | 8 |

Details on individual transmission parameters as well as the setting method are given in section *4.6 Transmission Parameters.* By switching DIP switch pin #1 at the MEMOBUS port to ON as described above, however, the transmission values for the GL120 or GL130 can be preset to these default values.

3) **Setting the GL120 and GL130 Transmission Mode**

Either the RTU or ASCII transmission mode is used for MEMOBUS communications. The RTU mode is set in this example. Set DIP switch pin #2 for the MEMOBUS port for the

GL120 or GL130 to OFF as shown in *Figure 2.5.* This sets the transmission mode for the GL120 and GL130 to the RTU mode.

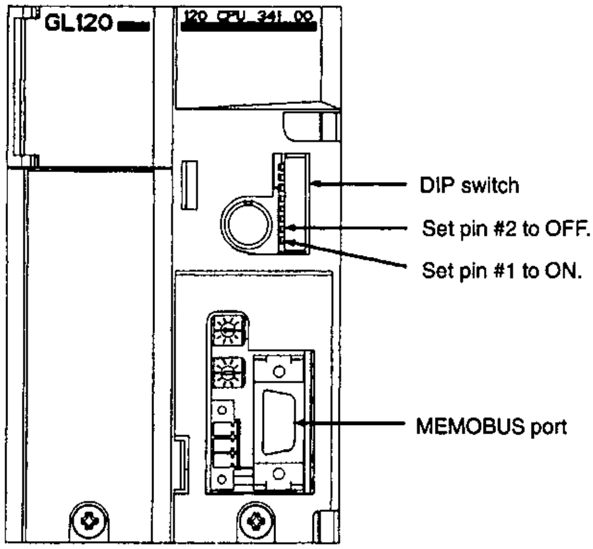



**Figure 2.5 Setting the Address Number and Transmission Parameters for the GL120**

4) **Setting Transmission Parameters for Personal Computer Made by NEC (PC-98)**

Set the transmission parameters for personal computer made by NEC (PC-98) to the same as the GL120 or GL130, that is:

- Baud rate: 9,600 bps
- Parity check: Provided
- Parity: Even
- Number of stop bits: 1
- Data bit length: 8

See the operating manual for the personal computer made by NEC (PC-98) for information on setting transmission parameters.

# Functions and Specifications

# 3

This chapter describes MEMOBUS functions and specifications.


**3.1 Functions ............................... 3-2**

**3.2 Specifications ............................ 3-3**

3.2.1 Transmission Specifications ...................... 3-3

3.2.2 Modem Specifications .......................... 3-4


3

# 3.1 Functions

This section summarizes the functions available in the MEMOBUS. Some of these functions are not available for Micros.

### Summary of MEMOBUS Functions

1) *Table 3.1* shows the functions available with the MEMOBUS. See chapter *6 MEMOBUS Messages* for details on specific functions.
   H indicates hexadecimal.

**Table 3.1 Summary of MEMOBUS Functions**

| Function | | Code | GL120, GL130 | Micro |
|---|---|---|---|---|
| Basic Functions | Read coil status | $01_H$ | Yes | Yes |
| | Read status of specific coils | $07_H$ | Yes | Yes |
| | Modify status of a single coil | $05_H$ | Yes | Yes |
| | Modify status of multiple coils | $0F_H$ | Yes | Yes |
| | Read input relay status | $02_H$ | Yes | Yes |
| | Read input register contents | $04_H$ | Yes | Yes |
| | Read holding register contents | $03_H$ | Yes | Yes |
| | Write to a single holding register | $06_H$ | Yes | Yes |
| | Write to multiple holding registers | $10_H$ | Yes | Yes |
| | Write a mask to the holding register | $16_H$ | Yes | Yes |
| | Read and write to multiple holding registers | $17_H$ | Yes | Yes |
| Extended functions | Read link coil status | $12_H$ | Yes | No |
| | Modify status of a single link coil | $19_H$ | Yes | No |
| | Modify status of multiple link coils | $1D_H$ | Yes | No |
| | Read link register contents | $15_H$ | Yes | No |
| | Write to a single link register | $1B_H$ | Yes | No |
| | Write to multiple link registers | $1F_H$ | Yes | No |
| | Read constant register contents | $13_H$ | Yes | No |
| | Write to a single constant register | $1A_H$ | Yes | No |
| | Write to multiple constant registers | $1E_H$ | Yes | No |
| Special functions | Loopback test | $08_H$ | Yes | Yes |
| | Read event counter | $0B_H$ | Yes | Yes |
| | Read status of communications | $0C_H$ | Yes | Yes |
| | Read equipment status | $11_H$ | Yes | Yes |
| | Read FIFO register contents | $18_H$ | Yes | Yes |
| | Read special reference status | $21_H$ | Yes | No |
| | Modify special reference status | $22_H$ | Yes | No |

# 3.2 Specifications

This section summarizes general specifications and grounding conditions for MEMOBUS transmission specifications and modems.

3.2.1 Transmission Specifications ........................................ 3-3
3.2.2 Modem Specifications ........................................ 3-4

## 3.2.1 Transmission Specifications

1) *Table 3.2* lists MEMOBUS transmission specifications.

**Table 3.2 MEMOBUS Transmission Specifications**

| Item | Specification |
|---|---|
| Communications Method | Half-duplex stop-start synchronization |
| Transmission Levels | Conform to RS-232C. |
| Protocol | MEMOBUS protocol or any other protocol |
| Baud Rate | 19,200/9,600/7,200/4,800/3,600/2,400/2,000/1,800/1,200/600/300/150 bps |
| Communications Mode | RTU mode or ASCII mode |
| Data Format | The following data format is used between the master and slaves, between the master and modems, and between modems and slaves:<br>1) Data bit length: 8 (RTU mode) or 7 (ASCII mode) bits<br>2) Parity check: Yes or No<br>3) Parity: Odd or even<br>4) Stop bits: 1 or 2 |
| Transmission Distance | 15 m (Can be extended to 4.5 km maximum by using Yaskawa modem. 500 m maximum for RS-422) |
| Transmission Error Detection | CRC-16 (RTU mode) or<br>LRC (ASCII mode) |
| Connector | D-sub connector (9-pin, femele) |

**Note** If you use the RS-422 MEMOBUS Module (model : JAMSC-120NOM27100), transmission levels conform to RS-422.

3

2) *Figure 3.1* shows the data format.

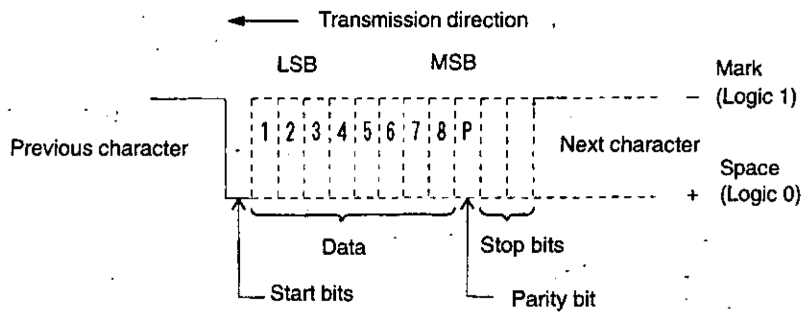



**Figure 3.1 Data Format**

## 3.2.2 Modem Specifications

1) Modems must be used when a master and slave are separated by more than 15 m, and when multiple slaves are connected to one master.

2) Yaskawa MEMOBUS Modems are available for this purpose, but commercially available modems can also be used.

3) *Table 3.3* gives specifications for Yaskawa Modems.

**Table 3.3 Modem Specifications**

| Item | Specification |
|---|---|
| Model | DISCT-J2078 |
| Modulation Method | FSK (50 kHz, 80 kHz) |
| Baud Rate | 19,200 bps max. |
| Transmission Distance | 4.5 km max. |
| Transmission Output Voltage | 6 Vp-p (100 kHz, 100 Ω load) |
| Receive Input Voltage | 100 mVp-p (min. at 100 kHz) |
| Power Supply | +15 VDC (+12 V) ±5%, 200 mA<br>–15 VDC (–12 V ) ±5%, 100 mA |
| Transmission Cable | 2-core twisted cable<br>RG-108/U or an equivalent cable<br>Characteristic impedance of 100 Ω at 100 kHz |
| Operating Temperature | 0 to 55 °C |
| Storage Temperature | –20 to 85 °C |
| Operating Humidity | 30% to 95% RH (without condensation) |
| Storage Humidity | 5% to 95% RH (without condensation) |
| Vibration Resistance | Conforms to JIS B3502.<br>10 to 57 Hz with a displacement amplitude of 0.075 mm<br>57 to 500 Hz with acceleration of 9.8 $m/s^2$ (1G) |
| Ground Resistance | 100 Ω max. |

4) All other general specifications conform to the general specifications for the GL120 and GL130.

# System Configuration

# 4

This chapter describes MEMOBUS hardware primarily through the MEMOBUS system configuration, connecting methods for individual equipment, and transmission parameter settings.


**4.1 Computer as Master .................... 4-2**

4.1.1 Description .................................. 4-2

4.1.2 Computer Master 1:1 Configurations .............. 4-3

4.1.3 Computer Master 1:1 Configuration Connected by Modems .................................. 4-5

4.1.4 Computer Master 1:N Configuration Connected by Modems .................................. 4-7

**4.2 Programming Panel as Master ............. 4-9**

4.2.1 Description .................................. 4-9

4.2.2 PP Master 1:1 Configuration ..................... 4-10

4.2.3 PP Master 1:1 Configuration Connected by Modems .. 4-11

4.2.4 PP Master 1:N Configuration Connected by Modems .. 4-12

**4.3 Man-Machine Interface as Master .......... 4-13**

4.3.1 Description .................................. 4-13

4.3.2 MMI Master 1:1 Configuration ................... 4-14

4.3.3 MMI Master 1:1 Configurations Connected by Modems 4-15

4.3.4 MMI Master 1:N Configuration Connected by Modems 4-16

**4.4 Programmable Controller as Master ........ 4-17**

4.4.1 Description .................................. 4-17

4.4.2 Micro Master ................................. 4-17

**4.5 Equipment Connections .................. 4-19**

4.5.1 Two-core Twisted Cable Configuration ............ 4-19

4.5.2 Connections Between J2078 Modems .............. 4-21

4.5.3 J2078 Wiring ................................ 4-21

4.5.4 Power ON/OFF Sequence ....................... 4-24

**4.6 Transmission Parameters ................. 4-25**

4.6.1 What Are Transmission Parameters? ............... 4-25

4.6.2 Types of Transmission Parameters ................. 4-25

4.6.3 Setting Transmission Parameters .................. 4-26

# 4.1 Computer as Master

This section describes system configurations when a computer is used as the master in a MEMOBUS system.

4.1.1 Description .................................................. 4-2
4.1.2 Computer Master 1:1 Configurations .......................... 4-3
4.1.3 Computer Master 1:1 Configuration Connected by Modems ........... 4-5
4.1.4 Computer Master 1:N Configuration Connected by Modems ........... 4-7

## 4.1.1 Description

1) A computer is the most common master used in MEMOBUS system configurations. Commercially available personal computers or other types of computers can be used as the master, i.e., any computer that has an RS-232C port can be used a master.

2) Computers that may be used as masters include the following:

- Personal computers made by NEC (PC-98) or compatible computers
- DOS (made by IBM) or compatible computers

Yaskawa PLCs used as slaves include the following:

- MEMOCON GL120 and GL130
- MEMOCON Micro

3) In order to use a personal computer as a master, a program that operates the computer as a MEMOBUS master using the format described in chapter *6 MEMOBUS Messages* must be prepared and loaded. This allows the personal computer to read and modify data at MEMOCON PLCs serving as slaves, including coils and registers.

4) Since MEMOCON PLCs serving as slaves are functionally equipped to be MEMOBUS slaves already, no programming needs to be created for this purpose.

5) Contact your nearest Yaskawa representative directly if you use any computer other than a personal computer made by NEC (PC-98) or DOS computer or compatibles for the master.

Three system configurations based on a computer master are described in the rest if this section.

## 4.1.2 Computer Master 1:1 Configurations

1) Computer master configurations with a master and slave are shown in *Figure 4.1*. Alternate slaves are shown for each master, but only one slave is actually connected in this type of configuration.

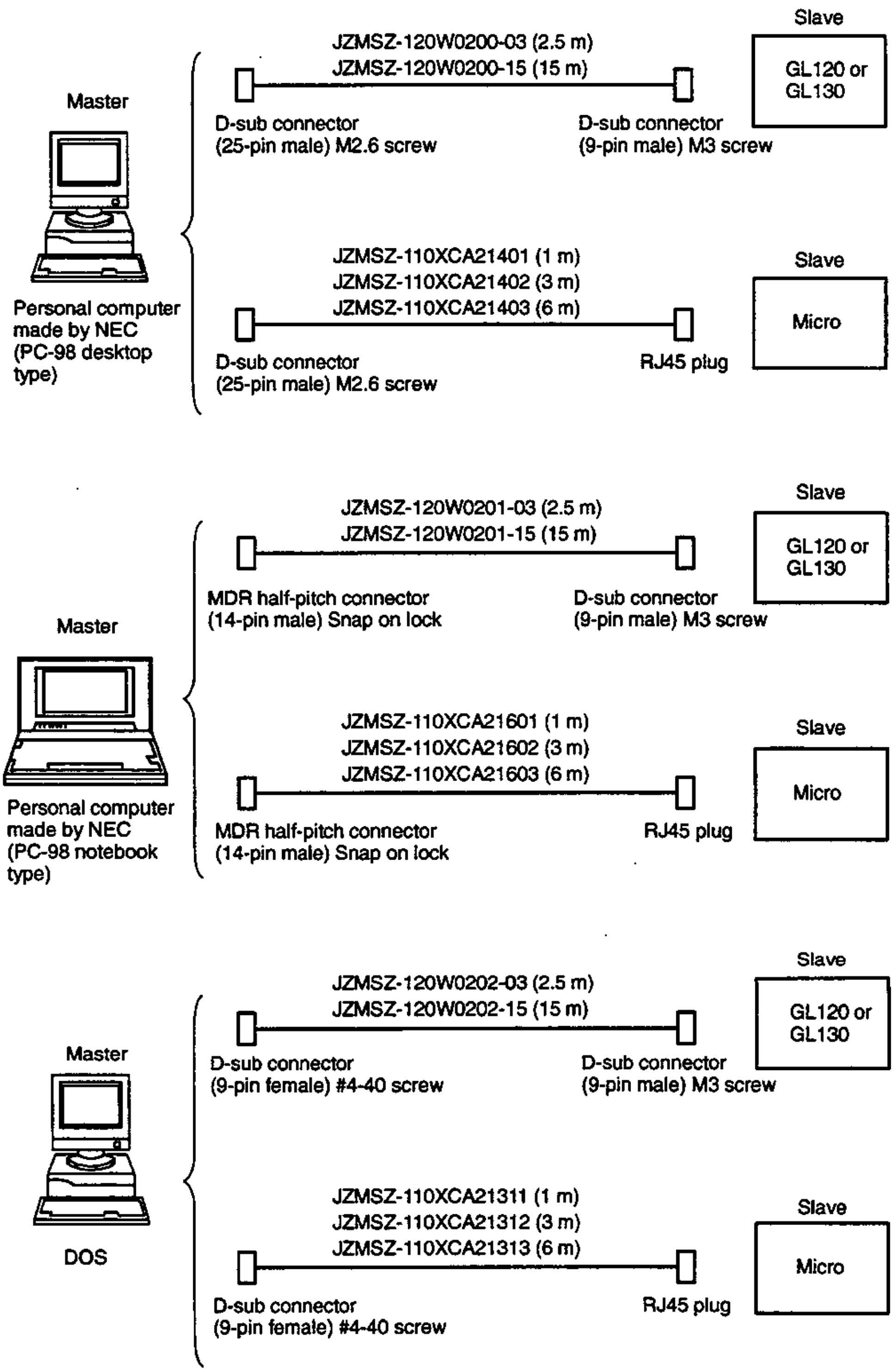



**Figure 4.1 Computer Master 1:1 Configurations**

2) Here the master and slave are connected by an RS-232C cable with a maximum length of 15 meters.

3) RS-232C cables are available from Yaskawa. Cable configurations vary depending on whether the master is a personal computer made by NEC (PC-98) desktop or notebook computer as well as a IBM AT personal computer or compatible computer, and whether the slave is a GL120, a GL130 or a Micro.

**Note** The RS-232C connectors listed in the examples above may differ depending on the type of personal computer. Be sure to check the connector specifications prior to selecting a cable.

## 4.1.3 Computer Master 1:1 Configuration Connected by Modems

1) *Figure 4.2* shows a computer master configuration where a master and slave are connected by modems. Alternate slaves and masters are shown at each end of the modem connection, but only one master and one slave are actually connected in this type of configuration.

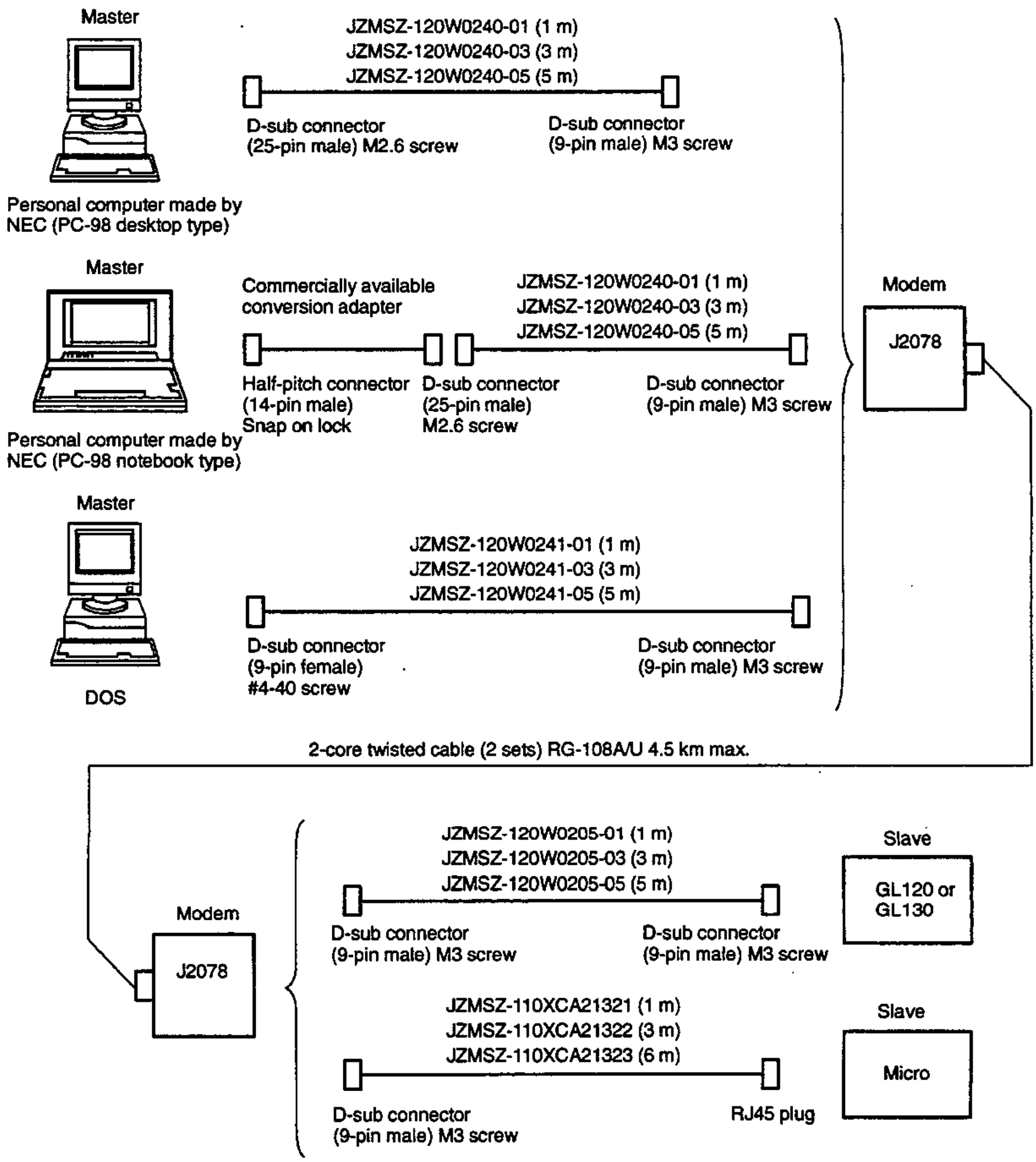



**Figure 4.2 Computer Master 1:1 Configurations Connected by Modems**

2) Modems are required if the distance between the master and the slave exceeds 15 meters. The modems used in this example are Yaskawa Modem (model: DISCT-J2078).

3) Two, shielded 2-core twisted cables are used to connect J2078 Modems.

   Recommended cable: RG108A/U or equivalent.

4) The maximum transmission distance may vary depending on the characteristics of the type of 2-core twisted cable used. The distance between the master and the slave can be up to 4.5 km with the RG-108A/U.

5) Commercially available modems may be used in these configurations, but be sure to contact your nearest Yaskawa representative when using such modems.

## 4.1.4 Computer Master 1:N Configuration Connected by Modems

1) *Figure 4.3* shows computer master configurations with a master and multiple slaves connected by modems. Alternate masters are shown, but only one master is actually connected in this type of configuration.

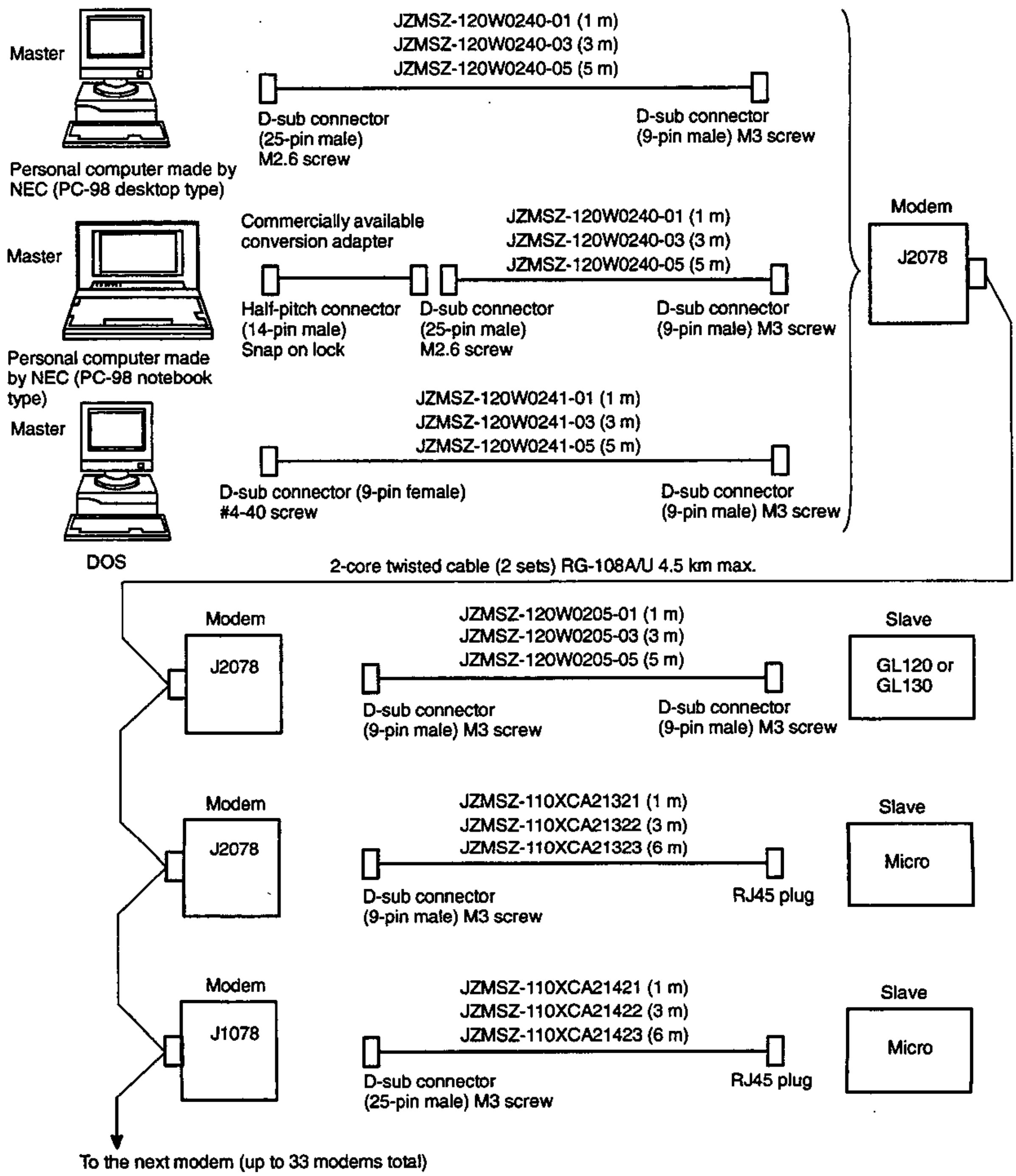



**Figure 4.3 Computer Master 1:N Configurations Connected by Modems**

2) Modems are required if there are two or more slaves even if the distance between them is less than 15 meters. Here J2078 Modems are used.

3) Up to 32 slaves can be connected to a single master. In other words, the number of modems that can be connected in a single network is 33 or less, including the one for the master.

4) The J2078 Modem is fully compatible with the DISCT-J1078 and other Yaskawa Modems, so all of these Modems can be used together on a single network. Therefore, existing systems using Yaskawa PLCs can easily be expanded by adding Modems if the system is already a MEMOBUS system. Non-MEMOBUS systems can be similarly expanded simply by upgrading to MEMOBUS.

# 4.2 Programming Panel as Master

This section describes configurations when P120, P140, P150, or other Yaskawa Programming Panels are used as the master.


4.2.1 Description .................................................... 4-9
4.2.2 PP Master 1:1 Configuration .................................... 4-10
4.2.3 PP Master 1:1 Configuration Connected by Modems ................. 4-11
4.2.4 PP Master 1:N Configuration Connected by Modems ................. 4-12


## 4.2.1 Description

1) The P120 Programming Panel is used for the GL120, GL130 and Micros, whereas the P140 and P150 are Programming Panels for other Yaskawa MEMOCON PLCs and the MEMOCON-SC Series.

2) These Programming Panels communicate with MEMOCONs through the MEMOBUS message protocol, and can be used as MEMOBUS masters.

3) Commercially available personal computers can be used as the programming device for the GL120, GL130, and Micros.

4) When the programming software MEMOSOFT for personal computers available through Yaskawa is loaded into personal computers, such as personal computer made by NEC (PC-98) or DOS computers and compatibles, the personal computer can be used as a GL120, GL130, or Micro Programming Panel. Such personal computers function as a Programming Panel master (PP master), and can be used to create and edit GL120, GL130, and Micro ladder logic programs just like the panels produced exclusively for programming.

5) Programming software MEMOSOFT provided through Yaskawa for personal computers (target PLCs: GL120, GL130) is as follows:

   - DOS (DOS operating system): FMSGL-AT3 (English version)

   - P120 Programming Panel: FMSGL-PP3E (English version)

   User's manuals for MEMOSOFT are as follows:

   - *MEMOCON GL120, GL130 Programming Panel P120 (MEMOSOFT) User's Manual* (SIEZ-C825-60.7)

   - *MEMOCON GL120, GL130 MEMOSOFT for DOS User's Manual* (SIEZ-C825-60.10)

6) The configurations when a Programming Panel master is a personal computer are the same as those for the computer masters described in section *4.1 Computer as Maste* (*Figure 4.1*, *Figure 4.2*, and *Figure 4.3*), so the structural diagrams are omitted here.

The following subsections describe the three types of configurations available when a P12( is used as a Programming Panel master.

## 4.2.2 PP Master 1:1 Configuration

1) *Figure 4.4* shows Programming Panel master configurations with a master and slave.

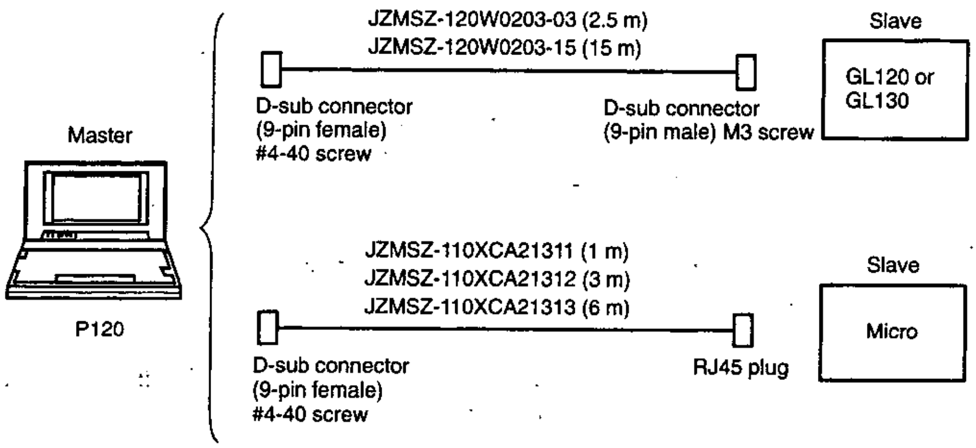



**Figure 4.4 Programming Panel 1:1 Configurations**

### 4.2.3 PP Master 1:1 Configuration Connected by Modems

1) *Figure 4.5* shows Programming Panel configurations when the master and slave are connected 1:1 by modem. Alternate slaves are shown, but only one slave is actually connected in this type of configuration.

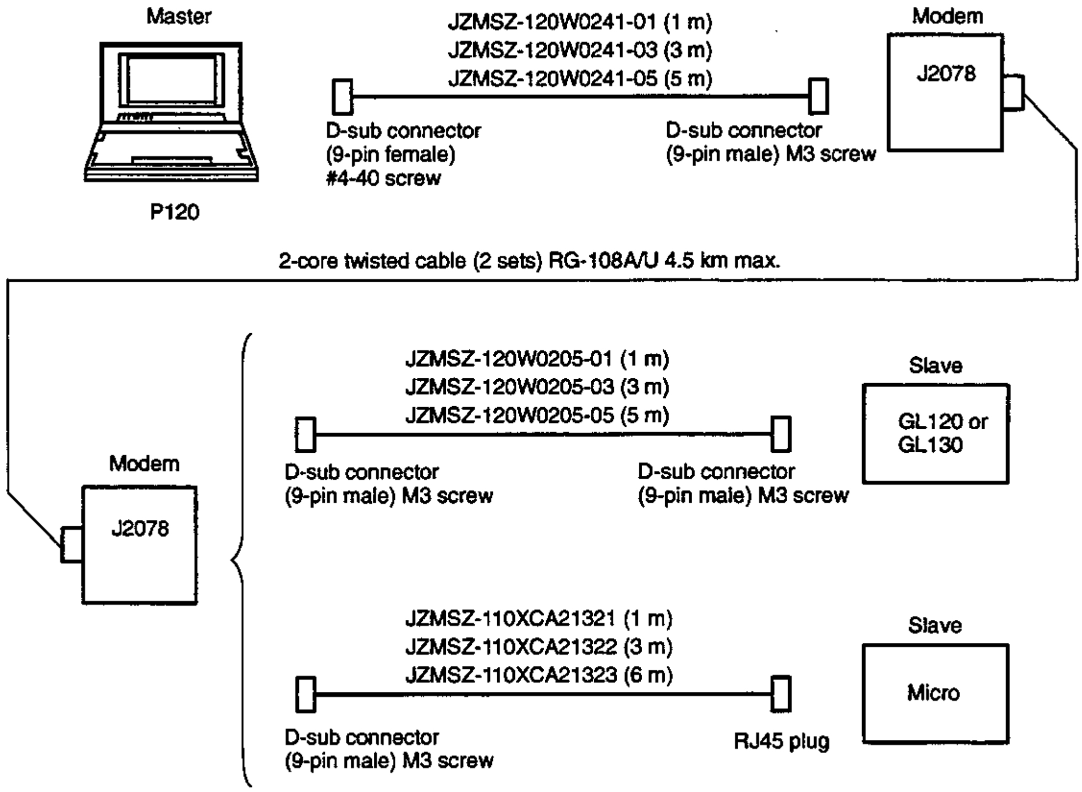



**Figure 4.5 Programming Panel Master 1:1 Configurations Connected by Modems**

**Note** The element status monitoring function of the P120 is not available when the P120 is used as a master.

## 4.2.4 PP Master 1:N Configuration Connected by Modems

1) *Figure 4.6* shows Programming Panel master configurations when a master and multiple slaves are connected by modems.

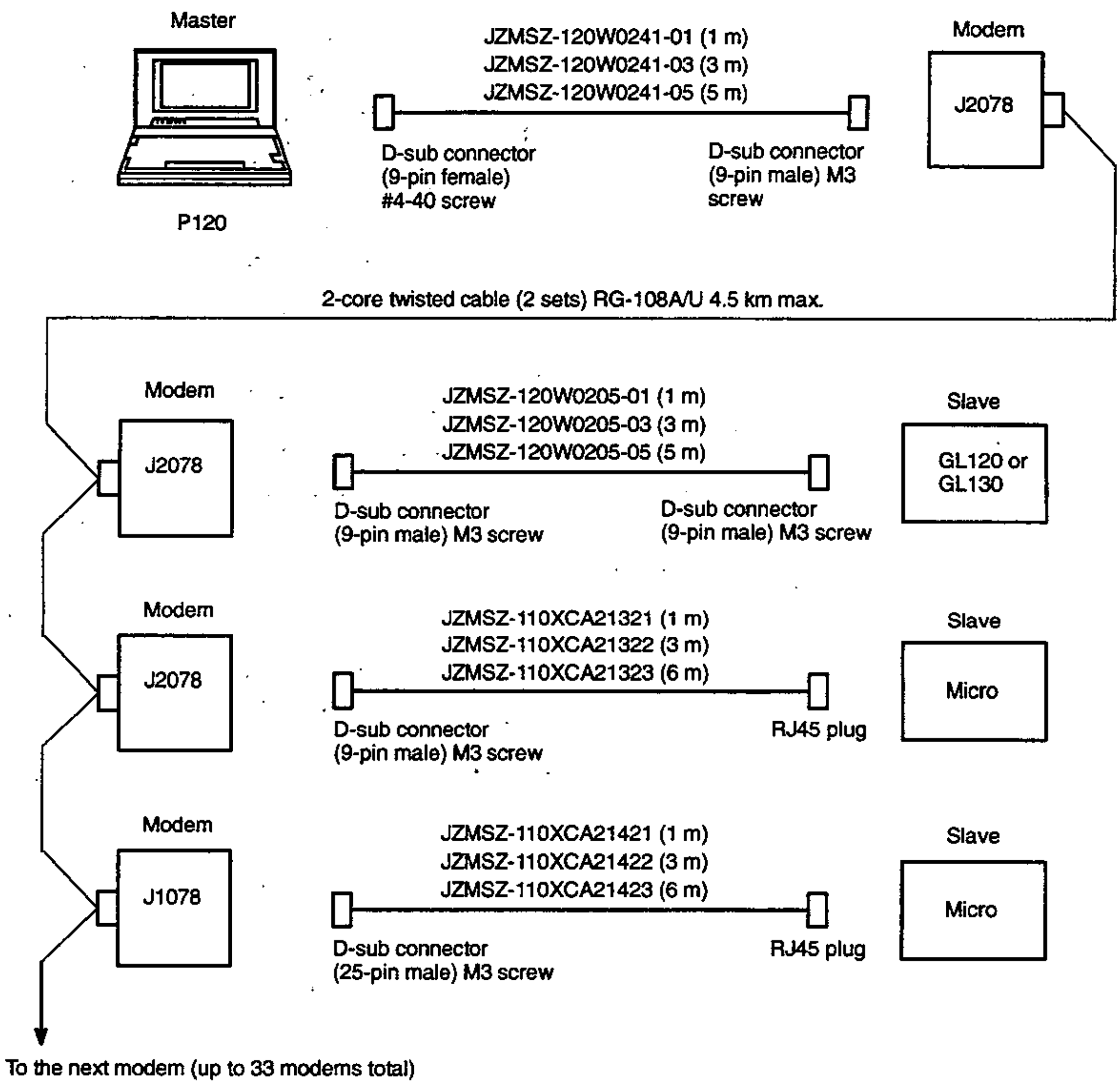



**Figure 4.6 Programming Panel Master 1:N Configuration Connected by Modems**

# 4.3 Man-Machine Interface as Master

This section describes configurations when a Man-Machine Interface (MMI) is used as a master.


4.3.1 Description ........ 4-13
4.3.2 MMI Master 1:1 Configuration ........ 4-14
4.3.3 MMI Master 1:1 Configurations Connected by Modems ........ 4-15
4.3.4 MMI Master 1:N Configuration Connected by Modems ........ 4-16


## 4.3.1 Description

1) Yaskawa offers ACGC4000-series and the ACGC400-series Man-Machine Interfaces.

2) The ACGC (Advanced Color Graphics Computer) Series feature FA monitoring devices that allow users with no knowledge of computers to create interactive MMI programs.

3) Since MMIs are equipped with MEMOBUS message generating functionality, it takes just a simple program to enable the MMI to be used as a master. This allows the MMI to display on screen various types of status, such as coils or registers from MEMOCON serving as slaves, and also allows status to be modified from the screen through the MMI.

4) User manuals are provided separately if more details about the ACGC Series are required.

5) An MMI is a type of computer, so its basic configurations as a master are the same as those for a computer master described in section *4.1 Computer as Master* in this chapter. The applicable RS-232C cable configurations, however, are different.

The following subsections describe the three types of configurations available when the ACGC Series is used as an MEMOBUS master.

## 4.3.2 MMI Master 1:1 Configuration

1) *Figure 4.7* shows MMI master configurations with a master and slave. Alternate slaves are shown, but only one slave is actually connected in this type of configuration.

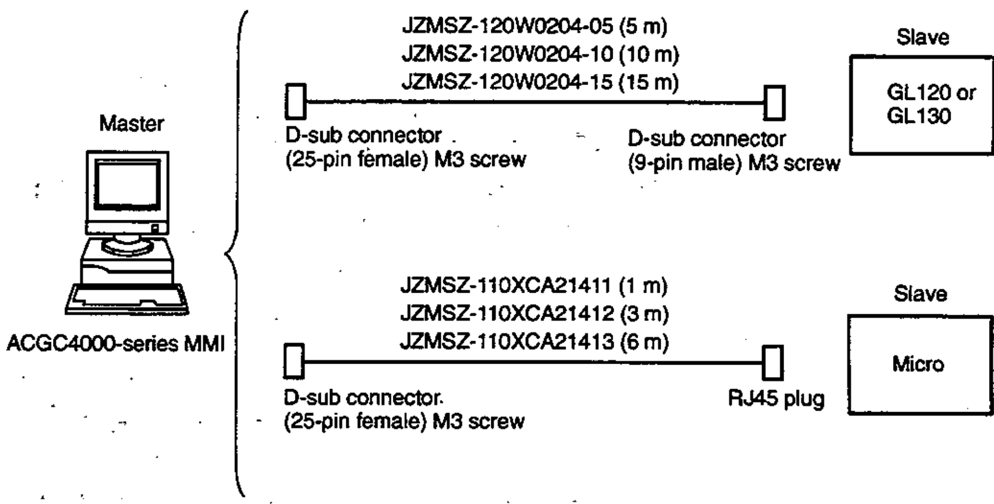



**Figure 4.7 MMI Master 1:1 Configuration**

## 4.3.3 MMI Master 1:1 Configurations Connected by Modems

1) *Figure 4.8* shows MMI master configurations where a master and slave are connected by modem. Alternate slaves are shown, but only one slave is actually connected in this type of configuration.

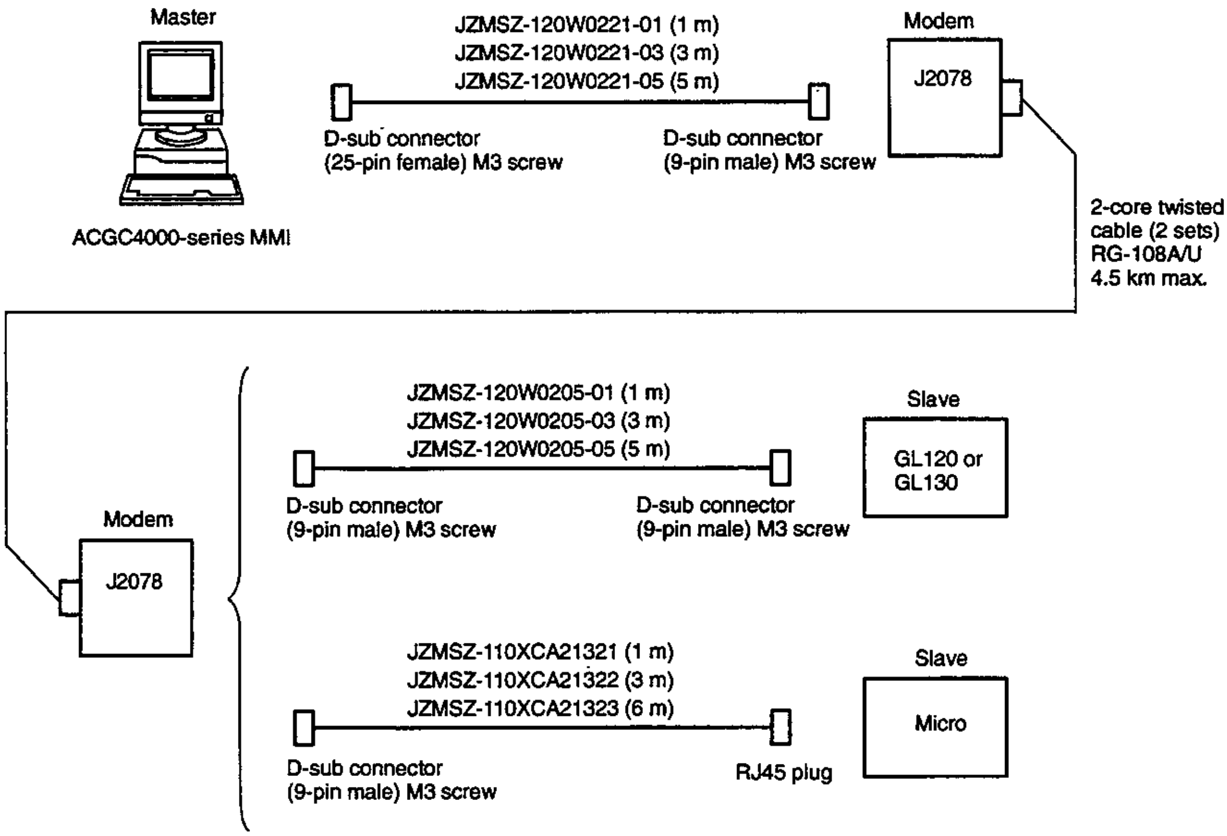



**Figure 4.8 MMI Master 1:1 Configuration Connected by Modems**

## 4.3.4 MMI Master 1:N Configuration Connected by Modems

1) *Figure 4.9* shows an MMI master configuration where a master and multiple slaves are connected by modems.

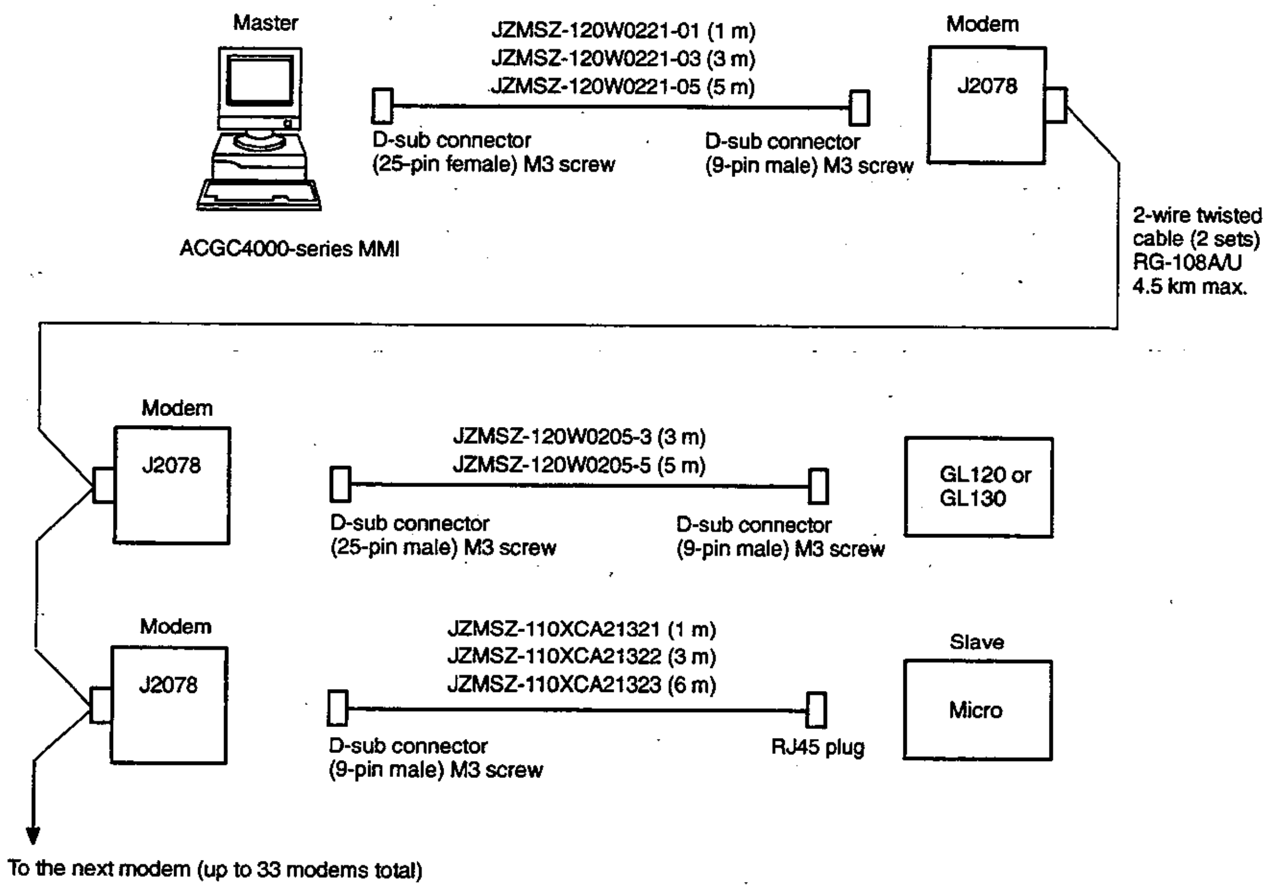



**Figure 4.9 MMI Master 1:N Configuration Connected by Modems**

2) Commercially available MMIs can be used in this configuration, but the users must prepare and load a program into the MMI that will operate the MMI as a MEMOBUS master using the format described in chapter *6 MEMOBUS Messages*.

# 4.4 Programmable Controller as Master

This section describes configurations when a Yaskawa MEMOCON Micro PLC is used as a master.

4.4.1 Description ........................................................ 4-17
4.4.2 Micro Master ........................................................ 4-17

## 4.4.1 Description

1) The GL120, GL130, and Micro PLCs all communicate through COMM instructions, and use these instructions to function as MEMOBUS masters.

2) COMM instructions enable the GL120, GL130, and Micro to freely exchange data with external equipment, such as MEMOCON PLCs, inverters, graphics control panels, and other equipment.

3) Yaskawa provides data sheets regarding data exchange with the MEMOCON PLCs, graphics control panels, and other equipment when the GL120 and GL130 are used as masters. Contact your nearest Yaskawa representative directly for further information. The explanation below describes configurations that use a Micro for the master.

4

## 4.4.2 Micro Master

1) *Figure 4.10* shows an example of a Micro master configuration that uses COMM instructions.

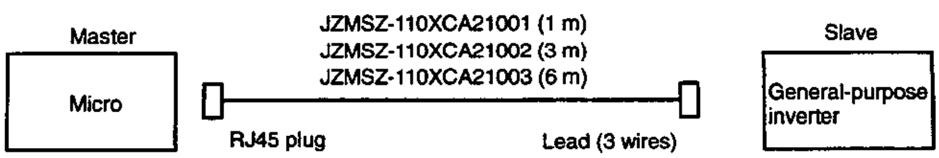



**Figure 4.10 Example of a Micro Master Configuration Using COMM Instructions**

2) *Figure 4.11* shows how the GL120, GL130, and other Yaskawa MEMOCON PLCs can be connected to the Micro master. The applicable cable configuration differs depending on whether the slave is a GL120, GL130, or another Yaskawa PLC even when they all share the same Micro master.

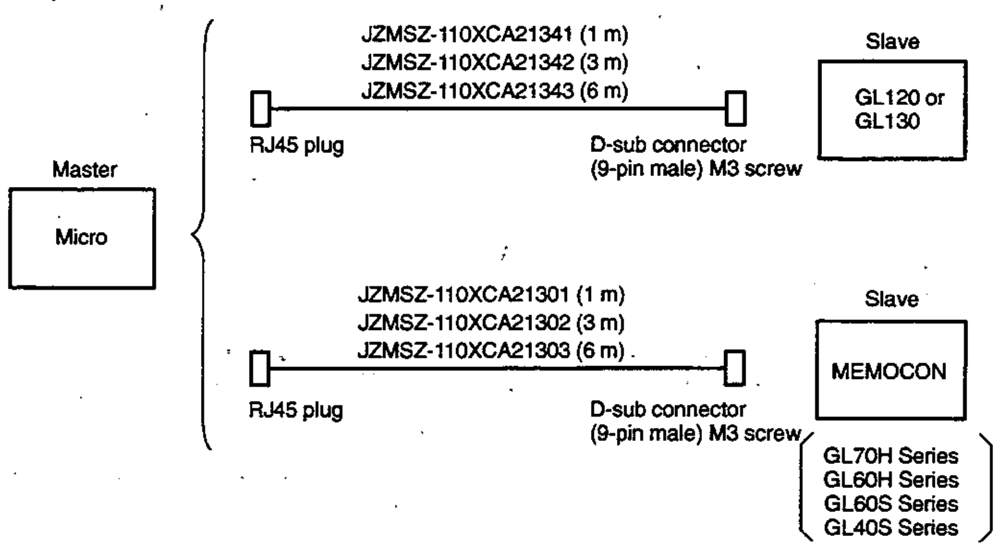



**Figure 4.11 Micro Master Connection to GL120, GL130, or Other MEMOCON PLCs**

# 4.5 Equipment Connections

This section describes the methods used to connect individual equipment primarily when using the Yaskawa J2078 Modem.


4.5.1 Two-core Twisted Cable Configuration ............................ 4-19
4.5.2 Connections Between J2078 Modems ............................ 4-21
4.5.3 J2078 Wiring ............................ 4-21
4.5.4 Power ON/OFF Sequence ............................ 4-24


## 4.5.1 Two-core Twisted Cable Configuration

1) Two, shielded 2-core twisted cables are needed to connect two J2078 Modems together.

2) Be sure to set up the 2-core twisted cable configuration as shown in *Figure 4.12.*

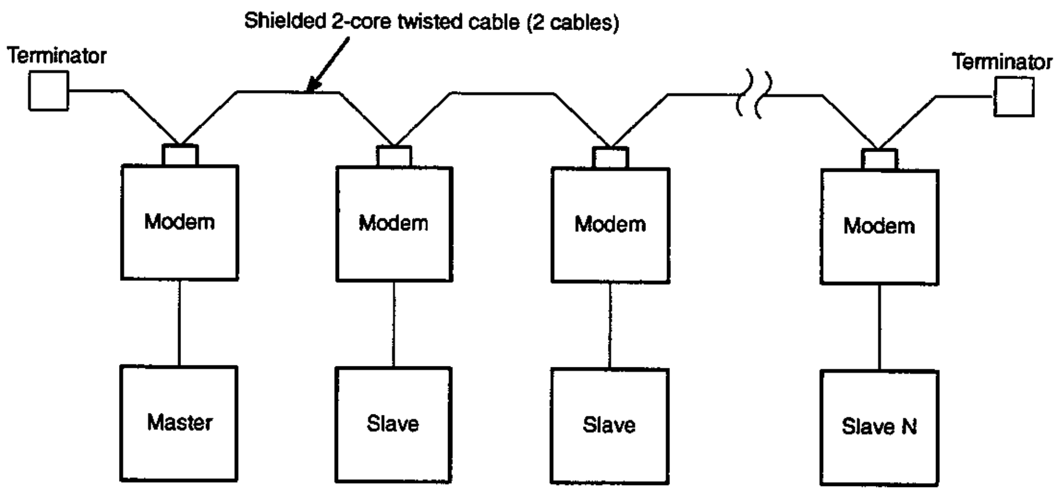



**Figure 4.12 Two-core Twisted Cable Configuration No. 1**

3) Terminators must be attached to both ends of the 2-core twisted cable.

4) Master placement is not important. This is shown in *Figure 4.13.*

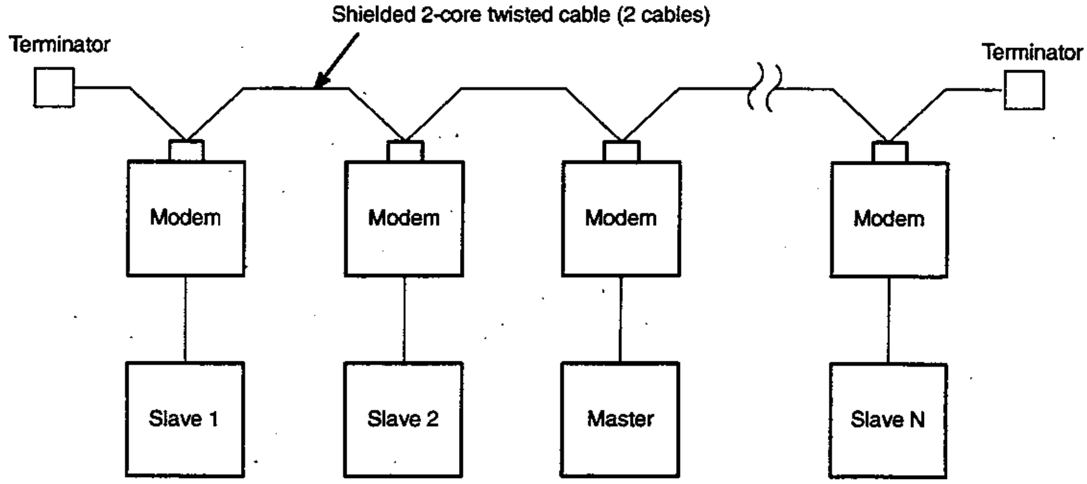



**Figure 4.13 Two-core Twisted Cable Configuration No. 2**

5) We recommend the following 2-core twisted cable:

- RG-108A/U or equivalent.

All cable manufacturers carry RG-108A/U cable, but you can also order it directly from Yaskawa.

6) The maximum distance for the MEMOBUS system with RG-108A/U cable is 4.5 km.

## 4.5.2 Connections Between J2078 Modems

1) The 2-core twisted cable is connected to the terminal board on the front panel of the J2078. No special equipment is needed for installation.

*Figure 4.14* shows terminal board connections, including the 2-core twisted cable.

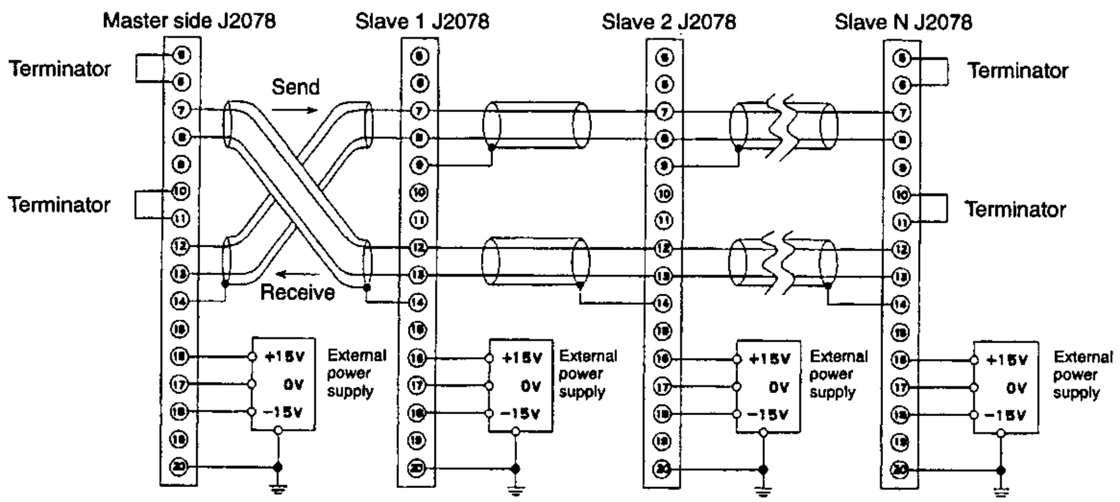



**Figure 4.14 J2078 Terminal Board Connections**

2) Connect the cable from the send terminal (SEND) of the J2078 terminal board on the master side to the receiver terminal (RECV) of the J2078 terminal boards of all the slaves, and connect the cable from the send terminal of all slave J2078 terminal boards to the receive terminal of the master side J2078.

3) The 2-core twisted cable is not polarized, so there is no need to worry about polarity when connecting the cable.

## 4.5.3 J2078 Wiring

1) Termination is required for the J2078 Modems at each end of the 2-core twisted cable.

2) Use jumper wires to connect terminals number 5 and 6 as well as terminals number 10 and 11 on the J2078 terminal boards located at both ends. This will connect the terminator (100 Ω 1/2 W) built into the J2078 to the 2-core twisted cable.

3) Do not connect anything between terminals number 5 and 6 or terminals number 10 and 11 on J2078 terminal boards anywhere between the two ends.

4) The shielded covering on the 2-core twisted cable must be grounded in order to block external noise interference. Therefore, connect terminal number 9 or 14 to the shielded covering of either end of the 2-core twisted cable. Also connect terminal number 20 to a ground with a resistance of 100 Ω max.

5) Terminals number 9, 14 and 20 are connected internally, so any one of these can be used to ground one end of the shielded covering.

6) Use an M3 screw to secure the crimp terminal to the terminal board for all wiring as shown in *Figure 4.15.* Where two, 2-core twisted cables are connected to J2078 Modems, (i.e., other than at the ones at the ends) overlap the crimp terminals as shown in *Figure 4.15,* and secure the overlapped terminals with the screw.

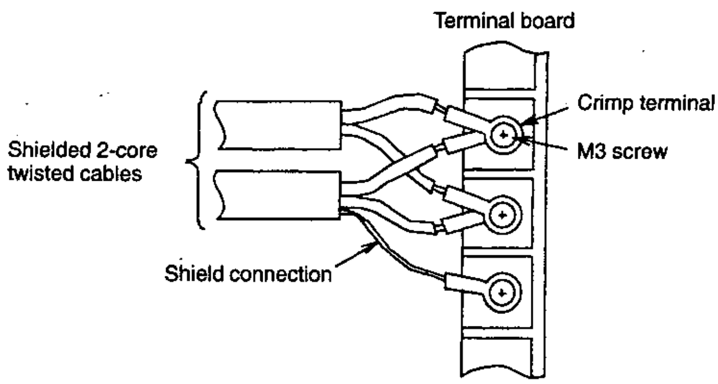
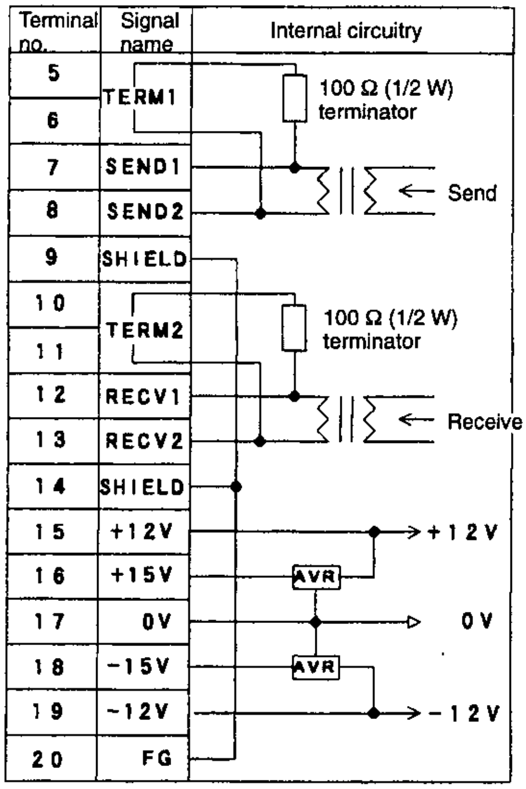



**Figure 4.15 Wiring to the J2078 Terminal Board**

7) The purpose for grounding the shield is explained in chapter *8 Wiring.*

8) *Figure 4.16* shows how J2078 Modems are internally connected to the terminal board.




Terminals number 1 through 4 are not used.

**Figure 4.16 Internal Connections on the J2078 Terminal Board**

9) J2078 Modems require an external ±15 V or ±12 V DC power supply. Connect the ±15 V power supply to terminals number 16 and 18, or connect the ±12 V power supply to terminals number 15 and 19. Power supply currents are as follows:

15 V or 12 V: 200 mA
–15 V or –12 V: 100 mA

## 4.5.4 Power ON/OFF Sequence

Be sure to follow the procedures for the power ON and OFF sequence given below.

**Note** Be very careful when turning power ON and OFF whenever you are using modems. Turning power to the slave ON and OFF with the modem turned ON will cause the modem to output spurious signals to the 2-core twisted cable for several tens of milliseconds. A transmission error is generated when these signal are sent to other modems. To avoid this situation, we recommend a power supply sequencing procedure of turning the slave ON before turning the modem ON, turning the modem OFF before turning the slave OFF, or turning both OFF simultaneously.

*Figure 4.17* shows an example of a power ON and OFF circuit.

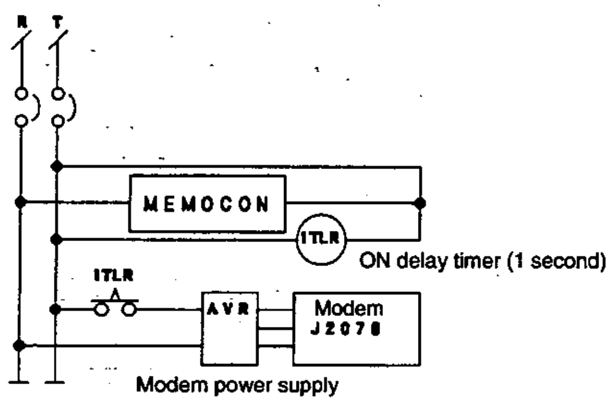



**Figure 4.17 Power Supply ON and OFF Circuit**

Nothing will happen if the slave is turned ON and OFF while the modem is turned ON if the RS-232C cable between the modem and the slave is disconnected. The modem may briefly (about 1 ms) output spurious signals to the 2-wire twisted cable when the modem is turned ON or OFF.

# 4.6 Transmission Parameters

This section describes the transmission parameters and setting method.


4.6.1 What Are Transmission Parameters? .............................. 4-25
4.6.2 Types of Transmission Parameters ................................ 4-25
4.6.3 Setting Transmission Parameters ................................. 4-26


## 4.6.1 What Are Transmission Parameters?

With MEMOBUS communications, transmission specifications like the baud rate can be set to match the specifications of the user's system. As well, unique addresses must be set for each piece of equipment for 1:N communications. These settings are referred to as transmission parameters. Except for address numbers, the transmission parameters for all equipment in a single network must be set to the same values. Failure to do so makes it impossible to communicate properly.

## 4.6.2 Types of Transmission Parameters

Transmission parameters are listed below.

1) **Address Numbers**

MEMOCON serving as slaves must be given unique addresses.

The setting range may vary depending on the equipment. The range for the GL120, GL130, and Micro is any number between 1 and 247.

**Note** Do not set the same address number for two or more slaves because both will return a response message at the same time and a transmission error will be generated. This is important because the system will accept duplicate settings.

2) **Baud Rate**

This parameter determines the transmission speed. The baud rate can be set in steps from 150 bps (bits per second) to 19,200 bps as shown in *Table 3.2*, but the normal setting is 9,600 bps.

3) **Parity Check Provided/Not Provided**

The parity check is used to detect data errors, and can be set to check or not to check parity. If "Provided" is selected, a parity bit is added to the end of the data. "Provided" is the default setting.



4

4) **Types of Parity Check**

Either even parity or odd parity can be selected when a parity check is specified. Even parity is the default setting.

5) **Number of Stop Bits**

The number of stop bits is used to detect data breaks, and can be set to either one or two bits. The selected number of bits is added to the end of the data. One bit is the default setting.

6) **Transmission Mode**

Either the RTU or the ASCII mode can be specified as the transmission mode. The data length is 8 bits in the RTU mode, and 7 bits in the ASCII mode.

7) **Port Delay Timer**

This delays the transmission time for the response message of the slave by the amount of time set in this parameter.

**Note** Except for address numbers, transmission parameters for all equipment in a single network must be set the same. Failure to set the same values will make it impossible to communicate properly.

## 4.6.3 Setting Transmission Parameters

The method used to set transmission parameters differs with the type of equipment. The method below is used to set transmission parameters for the GL120, GL130, and Micro.

1) **Method for Setting GL120 or GL130 Transmission Parameters**

a) Either preset default settings or any settings can be selected for the GL120 or GL130 address numbers and transmission parameters.

b) Turning ON DIP switch pin #1 at the GL120 or GL130 MEMOBUS port selects the preset initial settings. Turning OFF the same DIP switch allows any settings to be made. Changing the DIP switch settings changes the transmission parameters.

c) The preset default settings are given in the following table.

| Item | RTU mode | ASCII mode |
|---|---|---|
| Address | 1 | 1 |
| Baud Rate | 9,600 | 2,400 |
| Parity Check | Provided | Provided |
| Parity | Even | Even |
| Number of Stop Bits | 1 | 1 |
| Data Bit Length | 8 | 7 |
| Port Delay Timer | 0 (ms) | |

d) Use MEMOSOFT to set any transmission parameters. All transmission parameters can be set in the port setting window that can be opened from the configuration setting window. Refer to the following manuals for more details on the procedures.

- *MEMOCON GL120, GL130 Programming Panel P120 (MEMOSOFT) User's Manual* (SIEZ–C825–60.7)

- *MEMOCON Micro, GL120, GL130 MEMOSOFT (DOS) User's Manual* (SIEZ–C825–60.10)

2) **Method for Setting Micro Transmission Parameters**

The default settings for transmission parameters listed in the table below are stored on flash ROM. These values can be modified through the Hand–held Programmer.

| Item | Default Setting | |
|---|---|---|
| | COMM 1 | COMM2* |
| Address | 1 | |
| Baud Rate | 9600 | |
| Parity Check | Provided | |
| Parity | Even | |
| Number of Stop Bits | 1 | |
| Data Bit Length | 8 (RTU mode) | |
| Port Delay Timer | 10 (ms) | |

**Note** Only the COMM 2 port can be used for the CPU512/612.

4

# Signal Transmission Procedures

# 5

This chapter describes basic software, including procedures, used to send signals with MEMOBUS.


**5.1 Message Configuration .................. 5-2**

5.1.1 Description .................................. 5-2

5.1.2 Slave Address Number ......................... 5-3

5.1.3 Function Codes ............................... 5-3

5.1.4 Data ......................................... 5-4

5.1.5 Error Check .................................. 5-4

**5.2 Transmission Modes ..................... 5-6**

**5.3 Slave Response ......................... 5-7**

5.3.1 Description .................................. 5-7

5.3.2 Response Message in Normal Operation ............ 5-7

5.3.3 Response Messages Under Error Conditions ......... 5-7

5.3.4 No Response .................................. 5-8

**5.4 Signal Transmission Times ................ 5-11**

5.4.1 Description .................................. 5-11

5.4.2 Time Requirement for Each Block ................ 5-11

# 5.1 Message Configuration

This section describes the slave address number, function code, data, and error check components that comprise the query message configuration.

| | | |
|---|---|---|
| 5.1.1 | Description | 5-2 |
| 5.1.2 | Slave Address Number | 5-3 |
| 5.1.3 | Function Codes | 5-3 |
| 5.1.4 | Data | 5-4 |
| 5.1.5 | Error Check | 5-4 |

## 5.1.1 Description

1) When signals are sent between the master and slaves, communications are controlled through the program on the master side. The master always initiates signal transmission, and slaves respond to the signal from the master.

2) To initiate signal transmission, the master sends a query message to the slaves in a set format. The slaves receive the query message, and check the address number in the message. Only the slave with the matching address number decodes the query message, organizes the required data, and then returns a response message to the master also in a set format.

3) Query messages are comprised of the slave address, function code, data, and error check components sent in exactly this order. *Figure 5.1* shows the configuration of a query message.

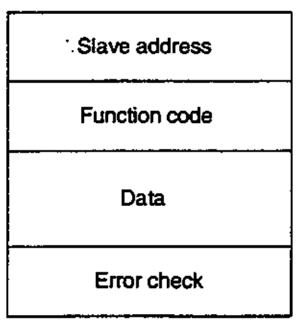



**Figure 5.1 Query Message Configuration**

The individual components of the query message are explained next.

## 5.1.2 Slave Address Number

1) The address number preset for each slave is used as the slave address number. The setting range may vary depending on the equipment, but the range for the GL120, GL130, and Micro is any number between 1 and 247.

2) The master specifies the address number and sends a query message to the slave it wishes to communicate with.

**Note** Do not set the same address number for two or more slaves because both return a response message at the same time and a transmission error will be generated. This is important because the system will accept duplicate settings.

## 5.1.3 Function Codes

1) The function codes specify the operation that slaves execute. *Table 5.1* lists the function codes.

**Table 5.1 Summary of Function Codes**

| Function code | Function | GL120, GL130 | Micro |
|---|---|---|---|
| $01_H$ | Read coil status | Yes | Yes |
| $02_H$ | Read input relay status | Yes | Yes |
| $03_H$ | Read holding register contents | Yes | Yes |
| $04_H$ | Read input register contents | Yes | Yes |
| $05_H$ | Modify status of a single coil* | Yes | Yes |
| $06_H$ | Write to a single holding register* | Yes | Yes |
| $07_H$ | Read status of specific coils | Yes | Yes |
| $08_H$ | Loopback test | Yes | Yes |
| $0B_H$ | Read event counter | Yes | Yes |
| $0C_H$ | Read communications status | Yes | Yes |
| $0F_H$ | Modify status of multiple coils* | Yes | Yes |
| $10_H$ | Write to multiple holding registers | Yes | Yes |
| $11_H$ | Read equipment status | Yes | Yes |
| $12_H$ | Read link coil status | Yes | No |
| $13_H$ | Read constant register contents | Yes | No |
| $15_H$ | Read link register contents | Yes | No |
| $16_H$ | Write a mask to the holding register* | Yes | Yes |
| $17_H$ | Read and write to multiple holding registers | Yes | Yes |
| $18_H$ | Read FIFO register contents | Yes | Yes |
| $19_H$ | Modify status of a single link coil* | Yes | No |
| $1A_H$ | Write to a single constant register* | Yes | No |
| $1B_H$ | Write to a single link register* | Yes | No |
| $1D_H$ | Modify status of multiple link coils* | Yes | No |
| $1E_H$ | Write to multiple constant registers* | Yes | No |
| $1F_H$ | Write to multiple link registers* | Yes | No |
| $21_H$ | Read special reference status | Yes | No |
| $22_H$ | Modify special reference status* | Yes | No |

2) The function codes listed in *Table 5.1* are expressed in hexadecimal.

3) Items in the function column of the table that are marked with an asterisk (*) are those for which the slave address can be set to 0, i.e., items that can be broadcast. When this is done, all slaves connected to the master accept and execute the query message.

4) A response message will not be returned with the slave address set to 0. Therefore another query message will have to be sent from the master to read modified locations in order to confirm slave operation.

## 5.1.4 Data

1) After receiving the function code, the slave organizes the data required in order to execute the function. The data required varies by function code.

2) See chapter *6 MEMOBUS Messages* for further details about the required data.

3) The coil, input relay, input register, holding register, and other numbers are set using relative numbers rather than by reference numbers. The GL120 and GL130 extract the lower-place 5 digits of each reference number (lower-place 4 digits for the link coils and link registers), and subtract one from the 5-digit number (from the 4-digit number with the link coils and link registers) to generate the relative number. The Micro extracts the lower-place 4 digits of each reference number, and subtracts one from the 4-digit number to generate the relative number.

4) This relationship between reference and relative numbers is shown in *Table 5.2*.

**Table 5.2 Relationship Between Reference and Relative Numbers**

| Item | Reference number | | Relative number | |
|---|---|---|---|---|
| | GL120, GL130 | Micro | GL120, Gl130 | Micro |
| Coils | 0xxxxx | 0xxxx | 0xxxxx -000001 | 0xxxx -00001 |
| Input relays | 1xxxxx | 1xxxx | 1xxxxx -100001 | 1xxxx -10001 |
| Input registers | 3xxxxx | 3xxxx | 3xxxxx -300001 | 3xxxx -30001 |
| Holding registers | 4xxxxx | 4xxxx | 4xxxxx -400001 | 4xxxx -40001 |
| Constant registers | 7xxxxx | – | 7xxxxx -700001 | – |
| Link coils | Duxxxx | – | Duxxxx -D00001 | – |
| Link registers | Ruxxxx | – | Ruxxxx- R00001 | – |

## 5.1.5 Error Check

1) Error check data is sent at the end of a message to detect message errors (bit changes) in the signal transmission process.

2) The MEMOBUS system uses either the RTU or the ASCII mode for transmission. (See section *5.2 Transmission Modes* for details.) The method used to detect errors depends on which transmission mode is selected. A cyclic redundancy check (CRC-16) is used for the RTU mode, while a longitudinal redundancy check (LRC) is used for the ASCII mode.

3) See chapter *6 MEMOBUS Messages* for further details on these detection methods.

# 5.2 Transmission Modes

This section describes the RTU and ASCII modes used in the MEMOBUS system.

**Description**

1) Either the Remote Terminal Unit (RTU) mode or American National Standard Code for Information Interchange (ASCII) mode can be selected in the MEMOBUS system by setting the transmission mode in the transmission parameters.

2) The transmission mode used depends mainly on the processing capacity (speed) of the master. In other words, the RTU mode is used when the transmission processing speed of the master is relatively fast, and the ASCII mode is used when the transmission speed of the master is relatively slow.

3) The procedure for sending signals is identical for both modes, but there are differences between the modes. For one, the ASCII mode requires message start and end marks. For another, data is sent and received in the RTU mode in an 8-bit decimal format, while the same data sent and received in the ASCII mode requires two ASCII 7-bit characters. The error detection method is also different for the two modes. *Table 5.3* shows the differences between the RTU and ASCII modes.

**Table 5.3 Differences Between the RTU and ASCII Modes**

| Item | RTU mode | ASCII mode |
|---|---|---|
| Data Length | 8 bits (decimal) | 7 bits (ASCII) |
| Message Start Mark | Not required | : (Colon) |
| Message End Mark | Not required | CR LF (Carriage return, line feed) |
| Message Length | N | 2N + 1 |
| Data Time Interval | 24 bits period or less | Less than 1 second |
| Error Detection | CRC-16 (Cyclic redundancy check) | LRC (Longitudinal redundancy check) |

**Note** When sending a query message from the master, do not insert a longer time interval between data comprising a single message than the time intervals listed in *Table 5.3*. With a time interval longer than that stipulated, the slave is unable to recognize the end of a query message, and cannot properly accept the message.

# 5.3 Slave Response

This section describes individual response messages for both normal and error operations as well as no response situations.

5.3.1 Description ........ 5-7
5.3.2 Response Message in Normal Operation ........ 5-7
5.3.3 Response Messages Under Error Conditions ........ 5-7
5.3.4 No Response ........ 5-8

## 5.3.1 Description

1) A slave that receives a query message from the master stores the message in the receive buffer until an error check is performed. If the results of the check are normal, the query message in the receive buffer is transferred to a buffer from which the message will be executed.
If there is an error in the query message, then the query message is ignored, and the ERR indicator will light up to show the type of error.

2) The slave checks the execution buffer at the end of a scan, and decodes and executes the contents of the buffer if a valid query message is present there. Then the slave prepares and sends a response message to the master.

## 5.3.2 Response Message in Normal Operation

1) If there is no problem with the query message, the slave prepares and sends a response message appropriate for the respective query.

2) The contents of response messages vary with the respective query message.

3) See chapter *6 MEMOBUS Messages* for further details on response messages.



## 5.3.3 Response Messages Under Error Conditions

1) If there is a logic problem (such as a non-existent function code) with the contents of a query message, then the slave prepares and sends a response message like that shown in *Figure 5.2* without actually executing the query message.

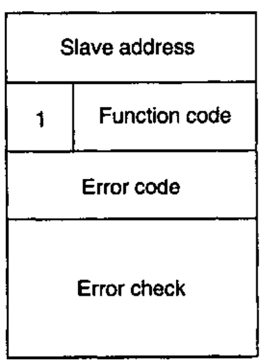



**Figure 5.2 Response Message to an Erroneous Query Message**

2) The response message in *Figure 5.2* is comprised of the slave address, function code, error code, and error check. The slave address is the same as the query message slave address and the function code has a 1 in the most significant bit position. Error codes are listed in *Table 5.4*.

**Table 5.4 Error Codes**

| Error code | Meaning |
|---|---|
| $01_H$ | Incorrect function code (non-existent function code). |
| $02_H$ | Incorrect coil, input relay, or register number (outside the range). |
| $03_H$ | Incorrect coil, input relay, or register count (outside the range). |

3) The master checks the most significant bit position of the function code in the query message in order to determine whether the query message sent was properly executed or not. In an error condition, the master can get details about the error from the error code that follows.

## 5.3.4 No Response

1) Slaves ignore the query message and will not return a response message in any of the following situations:

- Transmission parameters do not match.
- Slave address does not match.
- A transmission error due to noise or other factors is detected in the query message.
- The time interval between data comprising a single message is longer than stipulated.
- The result of the error check does not agree with the received data.

2) If a 0 is specified in a function that can specify 0 for the slave address, then all slaves execute the function, but no slave returns a response message.

3) Be sure to equip the master with a timer for monitoring response messages. If a response message is not returned within the time limit on the timer after a query message requiring a response message is sent, then some error has occurred. If this should happen, retry the message again. Set the time limit for the timer between 3 and 5 seconds.
The time limit on the timer for monitoring response messages is usually 3 to 5 seconds. However, after starting the CPU, the station will be unable to response for 15 seconds max. Considering this, it is recommended to set the timer to 20 seconds. Refer to *5)* for details.

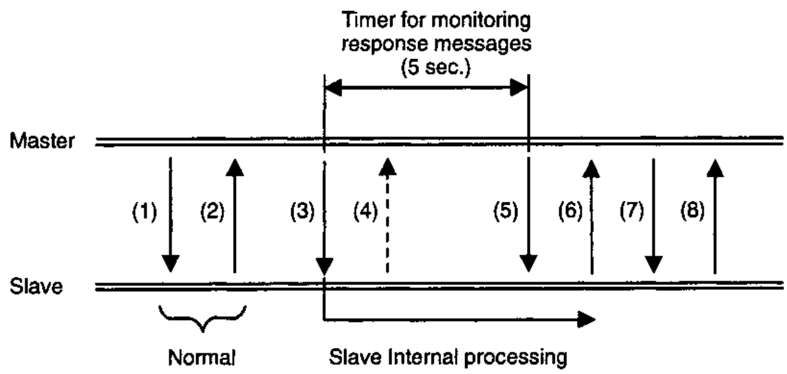
4) If a response message from the slave is disrupted by factors like those listed above, then the master will not receive a proper response message, and a transmission error is generated. The query message must be retried when the master detects a transmission error.

5) Set the time limit on the timer for monitoring response messages according to the following conditions:

- Allow 5 seconds if the master judges that an error exists in the response message from the slave

- Allow 20 seconds if the master cannot judge that an error exists in the response message from the slave

a) The following diagram explains the time limit on the timer for monitoring response messages.




(1) The master sends a query message to a slave.

(2) The slave sends a response message for the query message.

(3) The master sends another query message to the slave.

(4) The slave cannot send another response message to the master within 5 seconds because of the no–response time imposed by the starting up of the CPU.

(5) The master resends the query message to the slave after the timer has reached the time limit.

(6) The slave sends a response message for the query message (3) to the master.

(7) The master sends a third query message to the slave.

(8) The slave sends a response message for the query message (5) to the master.

If a response message is delayed because of the no–response time imposed the starting up of the CPU, the response message does not correspond to the query message. To avoid such problems, consider the following to decide the time limit on the timer for monitoring response messages.

- The master can judge whether the response message (8) is for the query message (5) or (7) in the example.

↓

The master ignores the response message (8), and waits for the response message for the query message (7). And so, the master executes the next processing correctly. In this case, set the time limit on the timer to 5 seconds because a processing time of 3 to 5 seconds is required when the CPU module is in running.

- The master cannot judge if the response message (8) is for the query message (5) or (7) in the example.

↓

A malfunction occurs in the MEMOBUS communications system. Set the time limit on the timer for monitoring the response message to 20 seconds, because 15 seconds are required to start up the CPU.

# 5.4 Signal Transmission Times

This section gives details on time requirements for sending signals between masters and slaves.

5.4.1 Description ................................................ 5-9
5.4.2 Time Requirement for Each Block ................................ 5-9

## 5.4.1 Description

1) The time requirements for sending signals between a master and a slave in the MEMO-BUS system are calculated based on the items listed below:

   - Transmission processing time of the master for query messages.
   - Delay time of the modem on the master side.
   - Transmission time of the query message.
   - Processing time of the slave.
   - Delay time of the modem on the slave side.
   - Transmission time of the response message.
   - Processing time of the master for response messages.

2) In order to calculate the total time requirement when multiple slaves are connected to the port of one master, calculate the time requirement for each individual slave, and then add them all together.

The above items are explained in more detail next.

## 5.4.2 Time Requirement for Each Block

1) **Transmission Processing Time of the Master for Query Messages**

   a) This is the time that it takes a master, such as a computer, to prepare a query message at the MEMOBUS port.

   b) The time depends on the processing time of the respective master. With a PLC master, the time depends on the scan time of the MEMOCON, and is normally one scan.

5

2) **Delay Time of the Modem on the Master Side**

a) This is the time that it takes the modem on the master side to get back a clear to sen (CTS) signal once the request to send (RTS) signal is received from the master.

b) With a Yaskawa J2078 Modem, the time can generally be ignored because it is les than 5 ms. There is no delay time if modems are not used.

3) **Transmission Time of the Query Message**

a) This is the time that it takes the communications port of the master to send the quer message.

b) The time depends on the length of the query message and the baud rate, and is calcu lated using the equation below.

$$\text{Transmission time} = \frac{\text{No. of characters in the query message} \times \text{the No. of bits per character} \times 1000}{\text{Baud rate}} \text{ (ms}$$

c) The number of bits per character is the number of data bits (8 or 7) plus the number o start bits (1), the number of stop bits (1 or 2), and the number of parity bits (1 or 0)

4) **Processing Time of the Slave**

a) This is the time that it takes for a slave to process the message once a query messag is received from the master, and then to prepare a response message to the master a the MEMOBUS port.

b) The time is related to the MEMOCON scan time, the number of items, such as th coils or registers specified in the query message, and the number of items that can b processed in one scan by the MEMOCON.

c) Since all functions can be processed in one scan with the GL120, GL130, and Micro the time is one scan for these PLCs.

5) **Delay Time of the Modem on the Slave Side**

a) This is the time that it takes for the modem on the slave side to get back a clear to senc (CTS) signal once the request to send (RTS) signal is received from the slave.

b) With a Yaskawa J2078 Modem, the time can generally be ignored because it is less than 5 ms.

c) There is no delay time if modems are not used.

6) **Transmission Time of the Response Message**

a) This is the time that it takes the communications port of the slave to send the response message. The time is calculated with the same equation used for query messages.

$$\text{Transmission time} = \frac{\text{No. of characters in the response message} \times \text{the No. of bits per character} \times 1000}{\text{Baud rate}} \text{ (ms)}$$

b) Here the number of bits per character is the number of data bits (8 or 7) plus the number of start bits (1), the number of stop bits (1 or 2), and the number of parity bits (1 or 0).

7) **Processing Time of the Master for Response Messages**

a) This is the time that it takes a master, such as a computer, to process the response message once it is received from the slave.

b) The time depends on the processing time of the respective master.

c) With a PLC master, the time depends on the scan time of the MEMOCON, and is normally one or two scans.

# MEMOBUS Messages

# 6

This chapter describes the MEMOBUS message configuration for each function in the RTU and ASCII transmission modes.


**6.1 Message Configurations for RTU Mode ..... 6-3**

**6.2 Basic RTU Mode Functions ............... 6-6**

6.2.1 Reading Coil Status ............ 6-6
6.2.2 Reading the Status of Specific Coils ............ 6-7
6.2.3 Modifying Status of a Single Coil ............ 6-8
6.2.4 Modifying Status of Multiple Coils ............ 6-10
6.2.5 Reading Input Relay Status ............ 6-11
6.2.6 Reading Input Register Contents ............ 6-13
6.2.7 Reading Holding Register Contents ............ 6-14
6.2.8 Writing to a Single Holding Register ............ 6-16
6.2.9 Writing to Multiple Holding Registers ............ 6-17
6.2.10 Writing a Mask to Holding Registers ............ 6-19
6.2.11 Reading/Writing to Multiple Holding Registers ...... 6-20

**6.3 RTU Mode Expansion Functions ........... 6-23**

6.3.1 Reading Link Coil Status ............ 6-23
6.3.2 Modifying Status of a Single Link Coil ............ 6-24
6.3.3 Modifying Status of Multiple Link Coils ............ 6-25
6.3.4 Reading Link Register Contents ............ 6-27
6.3.5 Writing to a Single Link Register ............ 6-29
6.3.6 Writing to Multiple Link Registers ............ 6-30
6.3.7 Reading Constant Register Contents ............ 6-31
6.3.8 Writing to a Single Constant Register ............ 6-33
6.3.9 Writing to Multiple Constant Registers ............ 6-34

**6.4 Special RTU Mode Functions .............. 6-37**

6.4.1 Loopback Test ............ 6-37
6.4.2 Reading Event Counter ............ 6-38
6.4.3 Reading Communications Status ............ 6-39

6.4.4 Reading Equipment Status ........ 6-41
6.4.5 Reading the FIFO Register Contents ........ 6-42
6.4.6 Reading Special Reference Status ........ 6-44
6.4.7 Modifying Special Reference Status ........ 6-46

**6.5 Example Communications Program for the RTU Mode ........ 6-48**
6.5.1 CRC-16 Calculations ........ 6-48
6.5.2 Communications Program Example ........ 6-51

**6.6 ASCII Mode Message ........ 6-56**
6.6.1 Message Configuration ........ 6-56
6.6.2 Query Message Transmission ........ 6-59
6.6.3 Receiving the Response Message ........ 6-61
6.6.4 Message Example ........ 6-62
6.6.5 LRC Calculations ........ 6-65

# 6.1 Message Configurations for RTU Mode

- This section provides a general description of RTU mode configurations.

**Message Configuration**

1) *Figure 6.1* shows the message configuration for the RTU mode.

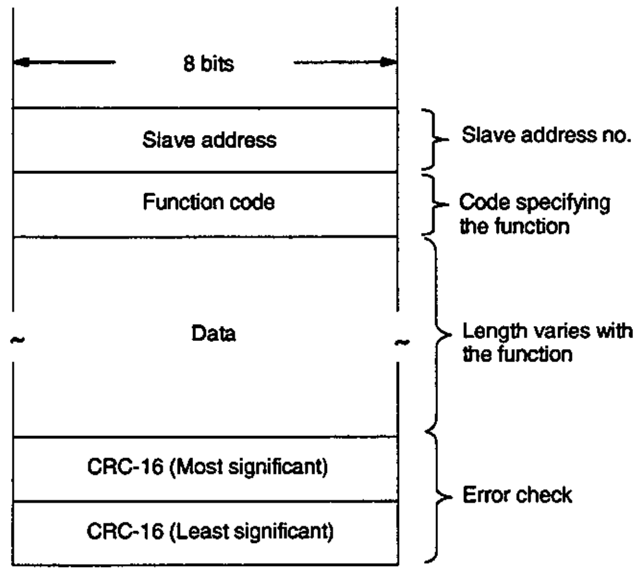



**Figure 6.1 RTU Mode Message Configuration**

2) The number of coils and registers handled by a single message is limited in each function. The maximum numbers of units that can be handled in a single message are given in *Table 6.1*.

**Table 6.1 Maximum Number of Units Handled in a Single Message in RTU Mode**

| Function | | Function code | Maximum no. of units handled in a single message | |
|---|---|---|---|---|
| | | | GL120, GL130 | Micro |
| Basic functions | Read coil status | $01_H$ | 2,000 | 1024/1536 *5 |
| | Read Status of specific coils | $07_H$ | 8 | 8 |
| | Modify status of a single coil | $05_H$ | 1 | 1 |
| | Modify status of multiple coils | $0F_H$ | 800 | 800 |
| | Read input relay status | $02_H$ | 2,000 | 256/512 *5 |
| | Read input register contents | $04_H$ | 125 | 32/48 *5 |
| | Read holding register contents | $03_H$ | 125 | 125 |
| | Write to a single holding register | $06_H$ | 1 | 1 |
| | Write to multiple holding registers | $10_H$ | 100 | 100 |
| | Write a mask to the holding register | $16_H$ | 1 | 1 |
| | Read and write to multiple holding registers | $17_H$ | 255 *2 | 255 *2 |
| Expansion functions | Read link coil status | $12_H$ | 2,000 | – |
| | Modify status of a single link coil | $19_H$ | 1 | – |
| | Modify status of multiple link coils | $1D_H$ | 800 | – |
| | Read link register contents | $15_H$ | 125 | – |
| | Write to a single link register | $1B_H$ | 1 | – |
| | Write to multiple link registers | $1F_H$ | 100 | – |
| | Read constant register contents | $13_H$ | 125 | – |
| | Write to a single constant register | $1A_H$ | 1 | – |
| | Write to multiple constant registers | $1E_H$ | 100 | – |
| Special functions | Loopback test | $08_H$ | *1 | *1 |
| | Read event counter | $0B_H$ | *1 | *1 |
| | Read communications status | $0C_H$ | *1 | *1 |
| | Read equipment status | $11_H$ | *1 | *1 |
| | Read FIFO register contents | $18_H$ | 31 | 31 |
| | Read special reference status | $21_H$ | *3 | – |
| | Modify special reference status | $22_H$ | *4 | – |

*1 See the items for each function.
*2 Read 125 and write 100.
*3 1,984 coil types and 124 registers.
*4 800 coil types and 100 registers.
*5 Varies with the model of Micro.

3) The GL120, GL130, and Micro process all these functions in one scan.

4) The message length varies with data length, and the data length and contents vary with the applicable function.

5) The message length for various types of equipment is given in *Table 6.2*.

**Table 6.2 Message Length of Functions in RTU Mode**

| Function | | Code | Query message | | | | Response message | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Minimum | | Maximum | | Minimum | | Maximum | |
| | | | GL120 GL130 | Micro | GL120 GL130 | Micro | GL120 GL130 | Micro | GL120 GL130 | Micro |
| Basic functions | Read coil status | $01_H$ | 8 | 8 | 8 | 8 | 6 | 6 | 255 | 133/197*1 |
| | Read status of specific coils | $07_H$ | 4 | 4 | 4 | 4 | 5 | 5 | 5 | 5 |
| | Modify status of a single coil | $05_H$ | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 |
| | Modify status of multiple coils | $0F_H$ | 10 | 10 | 109 | 109 | 8 | 8 | 8 | 8 |
| | Read input relay status | $02_H$ | 8 | 8 | 8 | 8 | 6 | 6 | 255 | 37/69*1 |
| | Read input register contents | $04_H$ | 8 | 8 | 8 | 8 | 7 | 7 | 255 | 13/17*1 |
| | Read holding register contents | $03_H$ | 8 | 8 | 8 | 8 | 7 | 7 | 255 | 255 |
| | Write to a single holding register | $06_H$ | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 |
| | Write to multiple holding registers | $10_H$ | 11 | 11 | 209 | 209 | 8 | 8 | 8 | 8 |
| | Write a mask to the holding register | $16_H$ | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 |
| | Read and write to multiple holding registers | $17_H$ | 15 | 15 | 213 | 213 | 7 | 7 | 255 | 255 |
| Expansion functions | Read link coil status | $12_H$ | 8 | – | 8 | – | 6 | – | 255 | – |
| | Modify status of a single link coil | $19_H$ | 8 | – | 8 | – | 8 | – | 8 | – |
| | Modify status of multiple link coils | $1D_H$ | 10 | – | 109 | – | 8 | – | 8 | – |
| | Read link register contents | $15_H$ | 8 | – | 8 | – | 7 | – | 255 | – |
| | Write to a single link register | $1B_H$ | 8 | – | 8 | – | 8 | – | 8 | – |
| | Write to multiple link registers | $1F_H$ | 11 | – | 209 | – | 8 | – | 8 | – |
| | Read constant register contents | $13_H$ | 8 | – | 8 | – | 7 | – | 255 | – |
| | Write to a single constant register | $1A_H$ | 8 | – | 8 | – | 8 | – | 8 | – |
| | Write to multiple constant registers | $1E_H$ | 11 | – | 209 | – | 8 | – | 8 | – |
| Special functions | Loopback test | $08_H$ | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 |
| | Read event counter | $0B_H$ | 4 | 4 | 4 | 4 | 8 | 8 | 8 | 8 |
| | Read communications status | $0C_H$ | 4 | 4 | 4 | 4 | 75 | 75 | 75 | 75 |
| | Read equipment status | $11_H$ | 4 | 4 | 4 | 4 | 14 | 14 | 14 | 14 |
| | Read FIFO register contents | $18_H$ | 6 | 6 | 6 | 6 | 7 | 7 | 67 | 67 |
| | Read special reference status | $21_H$ | 9 | – | 9 | – | 7 | – | 254 | – |
| | Modify special reference status | $22_H$ | 12 | – | 210 | – | 9 | – | 9 | – |

*1 Varies with model of Micro.

6

# 6.2 Basic RTU Mode Functions

This section describes basic RTU mode functions in the MEMOBUS system.

6.2.1 Reading Coil Status ........ 6-6
6.2.2 Reading the Status of Specific Coils ........ 6-7
6.2.3 Modifying Status of a Single Coil ........ 6-8
6.2.4 Modifying Status of Multiple Coils ........ 6-10
6.2.5 Reading Input Relay Status ........ 6-11
6.2.6 Reading Input Register Contents ........ 6-13
6.2.7 Reading Holding Register Contents ........ 6-14
6.2.8 Writing to a Single Holding Register ........ 6-16
6.2.9 Writing to Multiple Holding Registers ........ 6-17
6.2.10 Writing a Mask to Holding Registers ........ 6-19
6.2.11 Reading/Writing to Multiple Holding Registers ........ 6-20

## 6.2.1 Reading Coil Status

### 1) Function

The status of a series of coils is read starting from a specified coil number up to the number of coils specified. The slave organizes data for the status of groups of 8 coils in numerical order, and returns a response message. If the number of coils is a multiple of 8, then the MSB of the last data represents the status of the last coil. If the number of coils is not a multiple of 8, then the unused bits in the last data on the MSB side all become zeroes.

- Function code: $01_H$.

- No. of units handled by a single message: 2,000 for GL120 or GL130; 2,000 for Micro.

### 2) Example Message

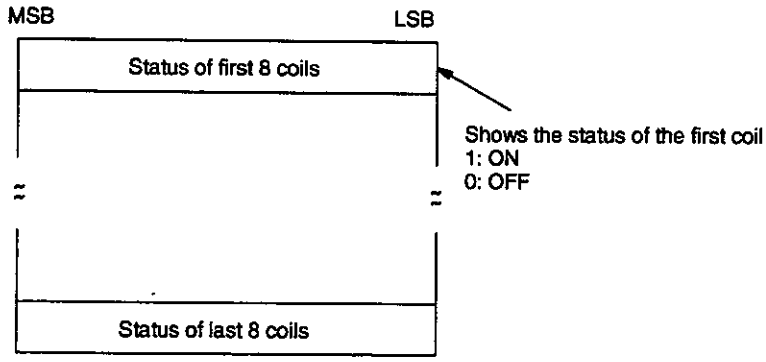
In this example, the status is read for a total of 37 coils numbered 000020 to 000056 (00020 to 00056 for Micros) from slave 2.

◀EXAMPLE▶ **Query Message**

| | | | |
|---|---|---|---|
| Slave address | | $02_H$ | |
| Function code | | $01_H$ | |
| Start no. | MSB | $00_H$ | |
| | LSB | $13_H$ | |
| No. of Units | MSB | $00_H$ | 1 to 2,000 |
| | LSB | $25_H$ | |
| CRC-16 | MSB | $0C_H$ | |
| | LSB | $27_H$ | |

**Note Start No.**
First coil number for GL120 or GL130: 000001
First coil number for Micro: 00001

### Data Read Order and Meaning




### Response Message (Normal Operation)

| | | |
|---|---|---|
| Slave address | | $02_H$ |
| Function code | | $01_H$ |
| No. of data | | $05_H$ |
| Status of first 8 coils | | $CD_H$ |
| Status of next 8 coils | | $6B_H$ |
| Status of next 8 coils | | $B2_H$ |
| Status of next 8 coils | | $0E_H$ |
| Status of next 8 coils | | $1B_H$ |
| CRC-16 | MSB | $04_H$ |
| | LSB | $FF_H$ |

(No. of data: (N+7)/8 (see note))

**Note** N is the number of coils, and (N + 7)/8 is the maximum integer value.

### Response Message (Error Operation)

| | | |
|---|---|---|
| Slave address | | $02_H$ |
| Function code +$80_H$ | | $81_H$ |
| Error code | | $03_H$ |
| CRC-16 | MSB | $F0_H$ |
| | LSB | $51_H$ |

## 6.2.2 Reading the Status of Specific Coils

1) **Function**

The status of a specified group of 8 coils (000001 to 000008) is read. The slave organizes the coil status data, and returns a response message.

- Function code: $07_H$.
- Maximum no. of units handled by a single message: 8 for GL120 or GL130; 8 for Micro.

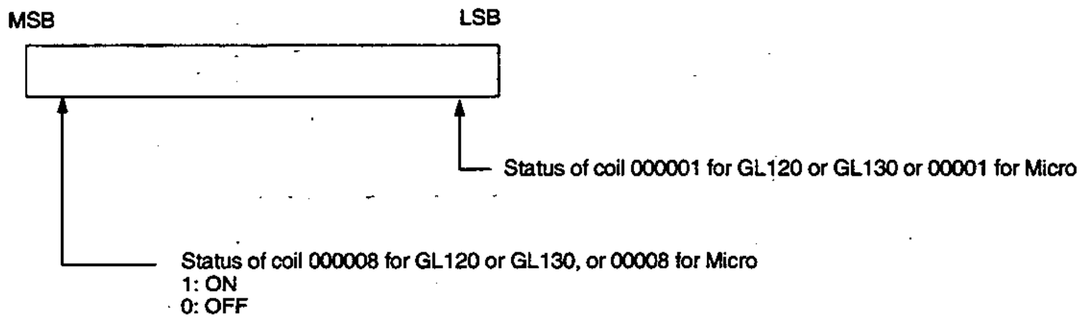
2) **Example Message**

In this example, the status is read for specified coils numbered 000001 to 000008 (00001 to 00008 for the Micro) from slave 1.

◀EXAMPLE▶ **Query Message**

| | | |
|---|---|---|
| Slave address | | $01_H$ |
| Function code | | $07_H$ |
| CRC-16 | MSB | $41_H$ |
| | LSB | $E2_H$ |

**Data Read Order and Meaning**




**Response Message (Normal Operation)**

| | | |
|---|---|---|
| Slave address | | $01_H$ |
| Function code | | $07_H$ |
| Status of 8 coils | | $03_H$ |
| CRC-16 | MSB | $62_H$ |
| | LSB | $31_H$ |

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | $01_H$ |
| Function code + $80_H$ | | $87_H$ |
| Error code | | $03_H$ |
| CRC-16 | MSB | $03_H$ |
| | LSB | $F1_H$ |

## 6.2.3 Modifying Status of a Single Coil

1) **Function**

A specified coil status is changed to a specified state (ON or OFF). The master specifies ON or OFF, and sends a query message.

- Function code: $05_H$.
- Maximum no. of units handled by a single message: 1 for GL120 or GL130; 1 for Micro.

2) **Example Message**

In this example, the status is modified for slave 3 coil 000173 (00173 for the Micro) (173 = $AD_H$)

◀EXAMPLE▶

**Query Message**

| | | |
|---|---|---|
| Slave address | | $03_H$ |
| Function code | | $05_H$ |
| Specified no. | MSB | $00_H$ |
| | LSB | $AC_H$ |
| Specified status | MSB | $FF_H$ |
| | LSB | $00_H$ |
| CRC-16 | MSB | $4D_H$ |
| | LSB | $F9_H$ |

(Specified status: $FF00_H$: ON, $0000_H$: OFF)

**Note** **Specified No.**
Coil no. for GL120 or GL130: 000001
Coil no. for Micro: 00001

**Response Message (Normal Operation)**

| | | |
|---|---|---|
| Slave address | | $03_H$ |
| Function code | | $05_H$ |
| Specified no. | MSB | $00_H$ |
| | LSB | $AC_H$ |
| Specified status | MSB | $FF_H$ |
| | LSB | $00_H$ |
| CRC-16 | MSB | $4D_H$ |
| | LSB | $F9_H$ |

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | $03_H$ |
| Function code +$80_H$ | | $85_H$ |
| Error code | | $03_H$ |
| CRC-16 | MSB | $A3_H$ |
| | LSB | $51_H$ |

**Note**

(1) If the slave address is set to 0, all slaves will execute the instruction, however no slave will return a response message after executing the instruction. Make sure that the master reads the coil status once again in order to confirm that the instruction was executed.

(2) The coil that you wish to modify should always be set in the Disable mode. The instruction can be executed in the Enable mode, but be very careful because the coil status may be overwritten despite the internal logic decoding results.

## 6.2.4 Modifying Status of Multiple Coils

### 1) Function

The status of a series of coils starting from a specified coil number up to the number o coils specified is modified to a specified state (ON or OFF). The master organizes data specifying ON or OFF for groups of 8 coils in numerical order, and sends a query mes sage. If the number of coils is a multiple of 8, then the MSB of the last data specifies the status of the last coil, but if the number of coils is not a multiple of 8, then unused bits are generated for the last data on the MSB side. The unused bits, lower are ignored on the slave side.

- Function code: $0F_H$.
- Maximum no. of units handled by a single message: 800 for GL120 or GL130; 800 fo Micro.

### 2) Example Message

In this example, the status is modified for a total of 10 coils numbered 000020 to 000029 (00020 to 00029 for the Micro) in slave 1 ($20 = 14_H$, $10 = 0A_H$).

◀EXAMPLE▶

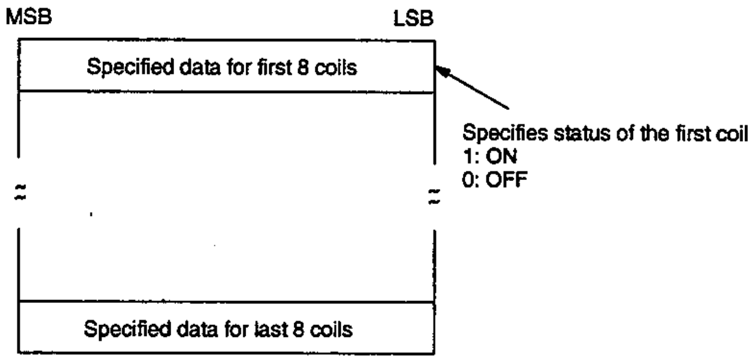
**Query Message**

| | | | |
|---|---|---|---|
| Slave address | | $01_H$ | |
| Function code | | $0F_H$ | |
| Start no. | MSB | $00_H$ | |
| | LSB | $13_H$ | |
| No. of units | MSB | $00_H$ | 1 to 800 |
| | LSB | $0A_H$ | |
| No. of data | | $02_H$ | (N+7)/8 (see note a) |
| Specified data for first 8 coils | | $00_H$ | |
| Specified data for next 8 coils | | $00_H$ | |
| CRC-16 | MSB | $E7_H$ | |
| | LSB | $9B_H$ | |

**Note** (a) N is the number of coils, and (N + 7)/8 is the maximum integer value.

(b) **Start No.**
First coil number for GL120 or GL130: 000001
First coil number for Micro: 00001

**Specified Data Order and Meaning**




**Response Message (Normal Operation)**

| | | |
|---|---|---|
| Slave address | | $01_H$ |
| Function code | | $0F_H$ |
| Start no. | MSB | $00_H$ |
| | LSB | $13_H$ |
| No. of units | MSB | $00_H$ |
| | LSB | $0A_H$ |
| CRC-16 | MSB | $24_H$ |
| | LSB | $09_H$ |

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | $01_H$ |
| Function code $+80_H$ | | $8F_H$ |
| Error code | | $03_H$ |
| CRC-16 | MSB | $04_H$ |
| | LSB | $31_H$ |

**Note** (1) If the slave address is set to 0, all slaves will execute the instruction, however no slave will return a response message after executing the instruction. Make sure that the master reads the coil status once again in order to confirm that the instruction was executed. The coil that you wish to modify should always be set in the Disable mode.

(2) The instruction can be executed in the Enable mode, but be very careful because the coil status may be overwritten despite the internal logic decoding results.



## 6.2.5 Reading Input Relay Status

1) **Function**

The status of a series of input relays is read starting from a specified input relay number up to the number of relays specified. The slave organizes data for the status in groups of 8 relays in numerical order, and returns a response message. If the number of relays is a multiple of 8, then the MSB of the last data represents the status of the last relay, but if the number of relays is not a multiple of 8, then the unused bits in the last data on the MSB side all become zeroes.

- Function code: $02_H$.

- Maximum no. of units handled by a single message: 2,000 for GL120 or GL130; 2,000 for Micro.

2) **Example Message**

In this example, the status is read for a total of 22 input relays numbered 100197 to 100218 (10197 to 10218 for Micros) from slave 2 (197 = $C5_H$, 22 = $16_H$).

◀**EXAMPLE**▶

**Query Message**

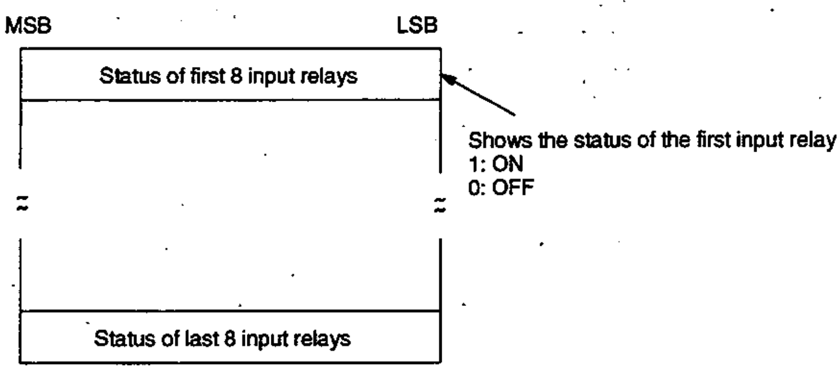
| | | | |
|---|---|---|---|
| Slave address | | $02_H$ | |
| Function code | | $02_H$ | |
| Start no. | MSB | $00_H$ | |
| | LSB | $C4_H$ | |
| No. of units | MSB | $00_H$ | 1 to 2,000 |
| | LSB | $16_H$ | |
| CRC-16 | MSB | $B8_H$ | |
| | LSB | $0A_H$ | |

**Note** **Start No.**

First input relay number for GL120 or GL130: 100001

First input relay number for Micro: 10001

**Data Read Order and Meaning**




**Response Message (Normal Operation)**

| | | | |
|---|---|---|---|
| Slave address | | $02_H$ | |
| Function code | | $02_H$ | |
| No. of data | | $03_H$ | (N+7)/8 (see note) |
| Status of first 8 input relays | | $AC_H$ | |
| Status of next 8 input relays | | $DB_H$ | |
| Status of next 8 input relays | | $35_H$ | |
| CRC-16 | MSB | $22_H$ | |
| | LSB | $BB_H$ | |

**Note** N is the number of coils, and (N + 7)/8 is the maximum integer value.

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave Address | | $02_H$ |
| Function code +80H | | $82_H$ |
| Error code | | $03_H$ |
| CRC-16 | MSB | $F0_H$ |
| | LSB | $A1_H$ |

## 6.2.6 Reading Input Register Contents

1) **Function**

The contents of a series of input registers is read starting from a specified input register number up to the number of registers specified. The slave splits the contents of the input registers into the most significant 8 bits and the least significant 8 bits, and then organizes the input register data in numerical order before returning a response message.

- Function code: $04_H$.
- Maximum no. of units handled by a single message: 125 for GL120 or GL130; 125 for Micro.

2) **Example Message**

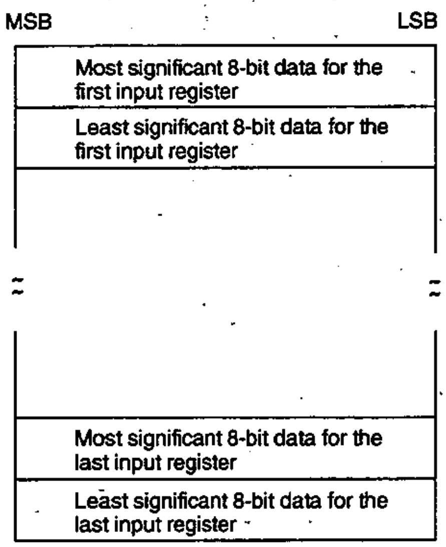
In this example, the contents are read for slave 21 input register 300009 (30009 for Micros) (21 = $15_H$).

◀EXAMPLE▶ **Query Message**

| | | | |
|---|---|---|---|
| Slave address | | $15_H$ | |
| Function code | | $04_H$ | |
| Start no. | MSB | $00_H$ | |
| | LSB | $08_H$ | |
| No. of units | MSB | $00_H$ | 1 to 125 |
| | LSB | $01_H$ | |
| CRC-16 | MSB | $B3_H$ | |
| | LSB | $1C_H$ | |

**Note Start No.**
First input register number for GL120 or GL130: 300001
First input register number for Micro: 30001

**Data Read Order and Meaning**




**Response Message (Normal Operation)**

| | | | |
|---|---|---|---|
| Slave address | | $15_H$ | |
| Function code | | $04_H$ | |
| No. of data | | $02_H$ | No. of input registers × 2 |
| Input register contents | MSB | $05_H$ | |
| | LSB | $39_H$ | |
| CRC-16 | MSB | $4A_H$ | |
| | LSB | $71_H$ | |

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | $15_H$ |
| Function code +$80_H$ | | $84_H$ |
| Error code | | $03_H$ |
| CRC-16 | MSB | $43_H$ |
| | LSB | $05_H$ |

## 6.2.7 Reading Holding Register Contents

1) **Function**

The contents of a series of holding registers is read starting from a specified holding register number up to the number of registers specified. The slave splits the contents of the holding register into the most significant 8 bits and the least significant 8 bits, and then organizes the holding register data in numerical order before returning a response message.

- Function code: $03_H$.
- Maximum no. of units handled by a single message: 125 for GL120 or GL130; 125 for Micro.

2) **Example Message**

In this example, the contents are read for a total of three holding registers numbered 400108 to 400110 (40108 to 40110 for Micros) from slave 2 (108 = $6C_H$).

◀EXAMPLE▶

**Query Message**

| | | |
|---|---|---|
| Slave address | | $02_H$ |
| Function code | | $03_H$ |
| Start no. | MSB | $00_H$ |
| | LSB | $6B_H$ |
| No. of units | MSB | $00_H$ |
| | LSB | $03_H$ |
| CRC-16 | MSB | $74_H$ |
| | LSB | $24_H$ |

No. of units: 1 to 125

**Note Start No.**
First holding register number for GL120 or GL130: 400001
First holding register number for Micro: 40001

**Data Read Order and Meaning**

| MSB | LSB |
|---|---|
| Most significant 8-bit data for the first holding register | |
| Least significant 8-bit data for the first holding register | |
| ⋮ | |
| Most significant 8-bit data for the last holding register | |
| Least significant 8-bit data for the last holding register | |

**Response Message (Normal Operation)**

| | | |
|---|---|---|
| Slave address | | $02_H$ |
| Function code | | $03_H$ |
| No. of data | | $06_H$ |
| First holding register contents | MSB | $02_H$ |
| | LSB | $2B_H$ |
| Next holding register contents | MSB | $00_H$ |
| | LSB | $00_H$ |
| Next holding register contents | MSB | $00_H$ |
| | LSB | $63_H$ |
| CRC-16 | MSB | $50_H$ |
| | LSB | $48_H$ |

No. of data: No. of input registers × 2

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | $02_H$ |
| Function code $+80_H$ | | $83_H$ |
| Error code | | $03_H$ |
| CRC-16 | MSB | $F1_H$ |
| | LSB | $31_H$ |

## 6.2.8 Writing to a Single Holding Register

1) **Function**

Specified data is written to a specified holding register. The master splits and organizes the write data into the most significant 8 bits and the least significant 8 bits, and then sends a query message.

- Function code: $06_H$
- Maximum no. of units handled by a single message: 1 for GL120 or GL130; 1 for Micro.

2) **Example Message**

In this example, the master writes to slave 5 holding register 400136 (40136 for the Micro) ($136 = 88_H$)

◀EXAMPLE▶ **Query Message**

| | | | |
|---|---|---|---|
| Slave address | | $05_H$ | |
| Function code | | $06_H$ | |
| Specified no. | MSB | $00_H$ | |
| | LSB | $87_H$ | |
| Write data | MSB | $03_H$ | } Write data |
| | LSB | $9E_H$ | |
| CRC-16 | MSB | $B9_H$ | |
| | LSB | $3F_H$ | |

**Note Specified No.**
Holding register no. for GL120 or GL130: 400001
Holding register no. for Micro: 40001

**Data Write Order and Meaning**

| MSB LSB |
|---|
| Most significant 8-bit write data |
| Least significant 8-bit write data |

**Response Message (Normal Operation)**

| | | |
|---|---|---|
| Slave address | | $05_H$ |
| Function code | | $06_H$ |
| Specified no. | MSB | $00_H$ |
| | LSB | $87_H$ |
| Write data | MSB | $03_H$ |
| | LSB | $9E_H$ |
| CRC-16 | MSB | $B9_H$ |
| | LSB | $3F_H$ |

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | $05_H$ |
| Function code +$80_H$ | | $86_H$ |
| Error code | | $03_H$ |
| CRC-16 | MSB | $43_H$ |
| | LSB | $A0_H$ |

**Note** If the slave address is set to 0, all slaves will execute the instruction, however no slave will return a response message after executing the instruction. Make sure that the master reads the coil status once again in order to confirm that the instruction was executed.

## 6.2.9 Writing to Multiple Holding Registers

1) **Function**

Specified data is written to a series of holding registers starting from a specified holding register number up to the number of registers specified. The master splits the write data into the most significant 8 bits and the least significant 8 bits, and then organizes the write data in numerical order before sending a query message.

- Function code: $10_H$.

- Maximum no. of units handled by a single message: 100 for GL120 or GL130; 100 for Micro.

6

2) **Example Message**

In this example, data is written to holding registers 400136 and 400137 (40136 and 40137 for Micros) at slave 1 (136 = $88_H$).

◀EXAMPLE▶ **Query Message**

| | | | |
|---|---|---|---|
| Slave address | | $01_H$ | |
| Function code | | $10_H$ | |
| Start no. | MSB | $00_H$ | |
| | LSB | $87_H$ | |
| No. of units | MSB | $00_H$ | 1 to 100 |
| | LSB | $02_H$ | |
| No. of data | | $04_H$ | No. of holding registers × 2 |
| Data written to the first holding register | MSB | $00_H$ | |
| | LSB | $0A_H$ | |
| Data written to the next holding register | MSB | $01_H$ | |
| | LSB | $02_H$ | |
| CRC-16 | MSB | $1A_H$ | |
| | LSB | $7A_H$ | |

**Note Start No.**
First holding register number for GL120 or GL130: 400001
First holding register number for Micro: 40001

**Data Write Order and Meaning**

| MSB LSB |
|---|
| Most significant 8-bit write data for the first holding register |
| Least significant 8-bit write data for the first holding register |
| ⋮ |
| Most significant 8-bit write data for the last holding register |
| Least significant 8-bit write data for the last holding register |

**Response Message (Normal Operation)**

| | | |
|---|---|---|
| Slave address | | $01_H$ |
| Function code | | $10_H$ |
| Start no. | MSB | $00_H$ |
| | LSB | $87_H$ |
| No. of units | MSB | $00_H$ |
| | LSB | $02_H$ |
| CRC-16 | MSB | $F1_H$ |
| | LSB | $E1_H$ |

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | $01_H$ |
| Function code $+80_H$ | | $90_H$ |
| Error code | | $03_H$ |
| CRC-16 | MSB | $0C_H$ |
| | LSB | $01_H$ |

**Note** If the slave address is set to 0, all slaves will execute the instruction, however no slave will return a response message after executing the instruction. Make sure that the master reads the coil status once again in order to confirm that the instruction was executed.

## 6.2.10 Writing a Mask to Holding Registers

1) **Function**

A specified bit pattern is written to the contents of a specified holding register. With an AND mask, each bit in the holding register is ANDed with the corresponding bit in the AND mask, and with an OR mask, each bit in the holding register is ORed with the corresponding bit in the OR mask. The holding register is modified accordingly.

- Function code: $16_H$.

- Maximum no. of units handled by a single message: 1 for GL120 or GL130; 1 for Micro.

6

2) **Example Message**

In this example, an AND mask is written to slave 5 holding register 400136 (40136 for the Micro) ($136 = 88_H$).

◀EXAMPLE▶ **Query Message**

| | | |
|---|---|---|
| Slave address | | $05_H$ |
| Function code | | $16_H$ |
| Specified no. | MSB | $00_H$ |
| | LSB | $87_H$ |
| AND mask data | MSB | $05_H$ |
| | LSB | $05_H$ |
| OR mask data | MSB | $00_H$ |
| | LSB | $00_H$ |
| CRC-16 | MSB | $53_H$ |
| | LSB | $26_H$ |

**Note Specified No.**
Holding register no. for GL120 or GL130: 400001
Holding register no. for Micro: 40001

**Mask Data Order and Meaning**

| MSB | LSB |
|---|---|
| Most significant 8-bit mask data | |
| Least significant 8-bit mask data | |

**Response Message (Normal Operation)**

| | | |
|---|---|---|
| Slave address | | $05_H$ |
| Function code | | $16_H$ |
| Specified no. | MSB | $00_H$ |
| | LSB | $87_H$ |
| AND mask data | MSB | $05_H$ |
| | LSB | $05_H$ |
| OR mask data | MSB | $00_H$ |
| | LSB | $00_H$ |
| CRC-16 | MSB | $53_H$ |
| | LSB | $26_H$ |

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | $05_H$ |
| Function code $+80_H$ | | $96_H$ |
| Error code | | $03_H$ |
| CRC-16 | MSB | $4E_H$ |
| | LSB | $60_H$ |

## 6.2.11 Reading/Writing to Multiple Holding Registers

1) **Function**

The contents of a series of holding registers is read starting from a specified holding register number up to the number of registers specified. At the same time, the function writes the respective data to a series of holding registers starting from a specified holding register number up to the number of registers specified. The master splits the write data into the most significant 8 bits and the least significant 8 bits, and then organizes the holding register data in numerical order before sending a query message. On the other side, the slave splits the contents of the holding register into the most significant 8 bits and the least significant 8 bits, and then organizes the holding register data in numerical order before returning a response message.

- Function code: $17_H$.
- Maximum no. of units handled by a single message: 125 read and 100 write for GL120 or GL130; and 125 read and 100 write for Micro.

### 2) Example Message

In this example, the contents are read for a total of three holding registers numbered 400108 to 400110 (40108 to 40110 for Micros) from slave 2, and data is written to holding registers 400136 and 400137 (40136 and 40137 for the Micro) (108 = $6C_H$, 136 = $88_H$).

◀EXAMPLE▶ **Query Message**

| | | | |
|---|---|---|---|
| Slave address | | $02_H$ | |
| Function code | | $17_H$ | |
| Read start no. | MSB | $00_H$ | |
| | LSB | $6B_H$ | |
| No. of read units | MSB | $00_H$ | 1 to 125 |
| | LSB | $03_H$ | |
| Write start no. | MSB | $00_H$ | |
| | LSB | $87_H$ | |
| No. of write units | MSB | $00_H$ | 1 to 100 |
| | LSB | $02_H$ | |
| No. of data | | $04_H$ | No. of write holding registers × 2 |
| Write data to the first holding register | MSB | $00_H$ | |
| | LSB | $0A_H$ | |
| Write data to the next holding register | MSB | $01_H$ | |
| | LSB | $02_H$ | |
| CRC-16 | MSB | $59_H$ | |
| | LSB | $0A_H$ | |

**Note** (a) **Read Start No.**
First holding register number for GL120 or GL130: 400001
First holding register number for Micro: 40001

(b) **Write Start No.**
First holding register number for GL120 or GL130: 400001
First holding register number for Micro: 40001

6

### Data Read and Write, Order and Meaning

MSB … LSB

| |
|---|
| Most significant 8-bit data for the first holding register |
| Least significant 8-bit data for the first holding register |
| : |
| Most significant 8-bit data for the last holding register |
| Least significant 8-bit data for the last holding register |

### Response Message (Normal Operation)

| | | |
|---|---|---|
| Slave address | | $02_H$ |
| Function code | | $17_H$ |
| No. of data | | $06_H$ |
| First holding register contents | MSB | $02_H$ |
| | LSB | $2B_H$ |
| Next holding register contents | MSB | $00_H$ |
| | LSB | $00_H$ |
| Next holding register contents | MSB | $00_H$ |
| | LSB | $63_H$ |
| CRC-16 | MSB | $50_H$ |
| | LSB | $B7_H$ |

No. of data: No. of read holding registers × 2

### Response Message (Error Operation)

| | | |
|---|---|---|
| Slave address | | $02_H$ |
| Function code $+80_H$ | | $97_H$ |
| Error code | | $03_H$ |
| CRC-16 | MSB | $FE_H$ |
| | LSB | $31_H$ |

# 6.3 RTU Mode Expansion Functions

This section describes link coils, constant registers and other expansion functions as well as message examples applicable to the RTU mode.


6.3.1 Reading Link Coil Status ........................................ 6-23
6.3.2 Modifying Status of a Single Link Coil ........................................ 6-24
6.3.3 Modifying Status of Multiple Link Coils ........................................ 6-25
6.3.4 Reading Link Register Contents ........................................ 6-27
6.3.5 Writing to a Single Link Register ........................................ 6-29
6.3.6 Writing to Multiple Link Registers ........................................ 6-30
6.3.7 Reading Constant Register Contents ........................................ 6-31
6.3.8 Writing to a Single Constant Register ........................................ 6-33
6.3.9 Writing to Multiple Constant Registers ........................................ 6-34


## 6.3.1 Reading Link Coil Status

1) **Function**

The status of a series of link coils is read starting from a specified link coil number up to the number of coils specified. The slave organizes data for the status of groups of 8 link coils in numerical order, and returns a response message. If the number of link coils is a multiple of 8, then the MSB of the last data represents the status of the last link coil, but if the number of link coils is not a multiple of 8, then the unused bits in the last data on the MSB side all become zeroes.

- Function code: $12_H$.

- No. of units handled by a single message: 2,000 (GL120 or GL130)

2) **Example Message**

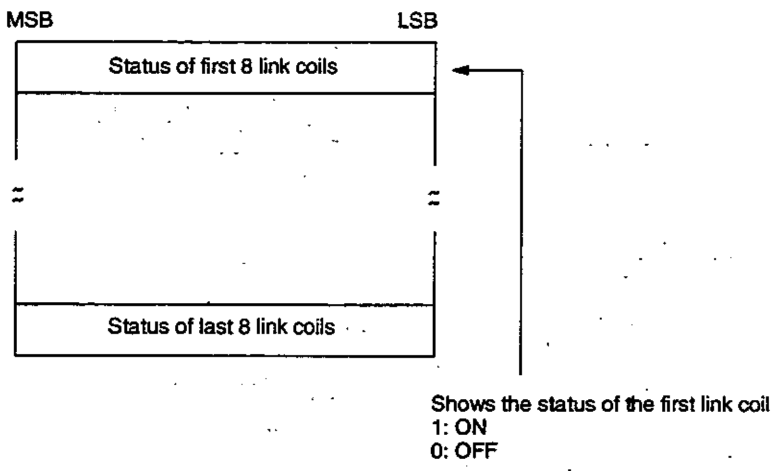
In this example, the status is read for a total of 37 link coils numbered D10020 to D10056 from slave 2 (GL120 or GL130) (10020 = $2724_H$, 37 = $25_H$).

◀EXAMPLE▶ **Query Message**

| | | | |
|---|---|---|---|
| Slave address | | $02_H$ | |
| Function code | | $12_H$ | |
| Start no. | MSB | $27_H$ | |
| | LSB | $23_H$ | |
| No. of units | MSB | $00_H$ | 1 to 2,000 |
| | LSB | $25_H$ | |
| CRC-16 | MSB | $83_H$ | |
| | LSB | $5F_H$ | |

**Note Start No.**
First link coil number: D00001

**Data Read Order and Meaning**




**Response Message (Normal Operation)**

| | | | |
|---|---|---|---|
| Slave address | | $02_H$ | |
| Function code | | $12_H$ | |
| No. of data | | $05_H$ | } (N+7)/8 (see note) |
| Status of first 8 link coils | | $CD_H$ | |
| Status of next 8 link coils | | $6B_H$ | |
| Status of next 8 link coils | | $B2_H$ | |
| Status of next 8 link coils | | $DE_H$ | |
| Status of next 8 link coils | | $1B_H$ | |
| CRC-16 | MSB | $7B_H$ | |
| | LSB | $FE_H$ | |

**Note** N is the number of coils, and (N + 7)/8 is the maximum integer value.

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | $02_H$ |
| Function code $+80_H$ | | $92_H$ |
| Error code | | $02_H$ |
| CRC-16 | MSB | $3C_H$ |
| | LSB | $A1_H$ |

## 6.3.2 Modifying Status of a Single Link Coil

1) **Function**

A specified link coil status is changed to a specified status (ON or OFF). The master specifies ON or OFF status, and sends a query message.

- Function code: $19_H$.
- Maximum no. of units handled by a single message: 1 (GL120 or GL130).

2) **Example Message**

In this example, the status is modified for slave 3 (GL120 or GL130) link coil D10173 (10173 = $27BD_H$).

◀EXAMPLE▶ **Query Message**

| | | |
|---|---|---|
| Slave address | | $03_H$ |
| Function code | | $19_H$ |
| Specified no. | MSB | $27_H$ |
| | LSB | $BC_H$ |
| Specified status | MSB | $FF_H$ |
| | LSB | $00_H$ |
| CRC-16 | MSB | $97_H$ |
| | LSB | $4A_H$ |

Specified status: ON with FF00, OFF with 0000

**Note Specified No.**
Link coil no.: D00001

**Response Message (Normal Operation)**

| | | |
|---|---|---|
| Slave address | | $03_H$ |
| Function code | | $19_H$ |
| Specified no. | MSB | $27_H$ |
| | LSB | $BC_H$ |
| Specified status | MSB | $FF_H$ |
| | LSB | $00_H$ |
| CRC-16 | MSB | $97_H$ |
| | LSB | $4A_H$ |

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | $03_H$ |
| Function code +$80_H$ | | $99_H$ |
| Error code | | $03_H$ |
| CRC-16 | MSB | $AB_H$ |
| | LSB | $91_H$ |

6

**Note**

(1) If the slave address is set to 0, all slaves will execute the instruction, however no slave will return a response message after executing the instruction. Make sure that the master reads the link coil status once again in order to confirm that the instruction was executed.

(2) The link coil that you wish to modify should always be set in the Disable mode. The instruction can be executed in the Enable mode, but be very careful because the link coil status may be overwritten despite the internal logic decoding results.

## 6.3.3 Modifying Status of Multiple Link Coils

1) **Function**

The status of a series of link coils starting from a specified link coil number up to the number of coils specified is modified to a specified status (ON or OFF). The master organizes data specifying ON or OFF status for groups of 8 link coils in numerical order, and sends a query message. If the number of link coils is a multiple of 8, then the MSB of the last data specifies the status of the last link coil, but if the number of link coils is not a multiple of 8, then unused bits are generated in the last data on the MSB side. The unused bits, however, are ignored on the slave side.

- Function code: $1D_H$.

- Maximum no. of units handled by a single message: 800 (GL120 or GL130).

2) **Example Message**

In this example, the status is modified for a total of 10 link coils numbered D10020 to D10029 at slave 1 (GL120 or GL130) ($10020 = 2724_H$, $10 = 0A_H$).

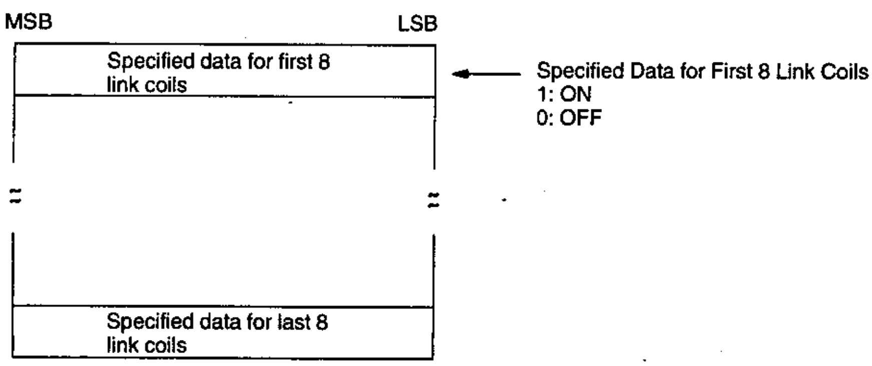
◀**EXAMPLE**▶ **Query Message**

| | | | |
|---|---|---|---|
| Slave address | | $01_H$ | |
| Function code | | $1D_H$ | |
| Start no. | MSB | $27_H$ | |
| | LSB | $23_H$ | |
| No. of units | MSB | $00_H$ | 1 to 800 |
| | LSB | $0A_H$ | |
| No. of data | | $02_H$ | (N+7)/8 (see note b) |
| Specified data for first 8 link coils | | $00_H$ | |
| Specified data for next 8 link coils | | $00_H$ | |
| CRC-16 | MSB | $35_H$ | |
| | LSB | $BC_H$ | |

**Note** (a) **Start No.**
First link coil number: D00001

(b) N is the number of link coils, and (N + 7)/8 is the maximum integer value.

**Data Read Order and Meaning**



**Response Message (Normal Operation)**

| | | |
|---|---|---|
| Slave address | | $01_H$ |
| Function code | | $1D_H$ |
| Start no. | MSB | $27_H$ |
| | LSB | $23_H$ |
| No. of units | MSB | $00_H$ |
| | LSB | $0A_H$ |
| CRC-16 | MSB | $96_H$ |
| | LSB | $B1_H$ |

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | $01_H$ |
| Function code $+80_H$ | | $9D_H$ |
| Error code | | $02_H$ |
| CRC-16 | MSB | $C9_H$ |
| | LSB | $51_H$ |

**Note** (1) If the slave address is set to 0, all slaves will execute the instruction, however no slave will return a response message after executing the instruction. Make sure that the master reads the link coil status once again in order to confirm that the instruction was executed. The link coil that you wish to modify should always be set in the Disable mode.

(2) The instruction can be executed in the Enable mode, but be very careful because the link coil status may be overwritten despite the internal logic decoding results.

## 6.3.4 Reading Link Register Contents

1) **Function**

The contents of a series of link registers is read starting from a specified link register number up to the number of registers specified. The slave splits the contents of the link register into the most significant 8 bits and the least significant 8 bits, and then organizes the link register data in numerical order before returning a response message.

- Function code: $15_H$.

- Maximum no. of units handled by a single message: 125 (GL120 or GL130).

2) **Example Message**

In this example, the contents are read for a total of three link registers numbered R10001 to R10003 from slave 2 (GL120 or GL130) ($10001 = 2711_H$).

◀EXAMPLE▶

**Query Message**

| | | |
|---|---|---|
| Slave address | | $02_H$ |
| Function code | | $15_H$ |
| Start no. | MSB | $27_H$ |
| | LSB | $10_H$ |
| No. of units | MSB | $00_H$ |
| | LSB | $03_H$ |
| CRC-16 | MSB | $47_H$ |
| | LSB | $4A_H$ |

No. of units: 1 to 125

**Note Start No.**

First link register number: R00001

**Data Read Order and Meaning**

| MSB LSB |
|---|
| Most significant 8-bit data for the first link register |
| Least significant 8-bit data for the first link register |
| ≈ |
| Most significant 8-bit data for the last link register |
| Least significant 8-bit data for the last link register |

**Response Message (Normal Operation)**

| | | |
|---|---|---|
| Slave address | | $02_H$ |
| Function code | | $15_H$ |
| No. of data | | $06_H$ |
| First link register contents | MSB | $12_H$ |
| | LSB | $34_H$ |
| Next link register contents | MSB | $56_H$ |
| | LSB | $78_H$ |
| Next link register contents | MSB | $9A_H$ |
| | LSB | $BC_H$ |
| CRC-16 | MSB | $FC_H$ |
| | LSB | $95_H$ |

No. of data: No. of link registers × 2

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | $02_H$ |
| Function code $+80_H$ | | $95_H$ |
| Error code | | $03_H$ |
| CRC-16 | MSB | $FF_H$ |
| | LSB | $51_H$ |

## 6.3.5 Writing to a Single Link Register

1) **Function**

Specified data is written to a specified link register. The master splits and organizes the write data into the most significant 8 bits and the least significant 8 bits, and then sends a query message.

- Function code: $1B_H$.
- Maximum no. of units handled by a single message: 1 (GL120 or GL130).

2) **Example Message**

In this example, the master writes to slave 5 (GL120 or GL130) link register R10136 (10136 = $2798_H$).

◀EXAMPLE▶

**Query Message**

| | | | |
|---|---|---|---|
| Slave address | | $05_H$ | |
| Function code | | $1B_H$ | |
| Specified no. | MSB | $27_H$ | |
| | LSB | $97_H$ | |
| Write data | MSB | $03_H$ | Write data |
| | LSB | $9E_H$ | |
| CRC-16 | MSB | $5E_H$ | |
| | LSB | $4C_H$ | |

**Note Specified No.**
Link register no.: R00001

**Data Write Order and Meaning**

| MSB | LSB |
|---|---|
| Most significant 8-bit write data | |
| Least significant 8-bit write data | |

**Response Message (Normal Operation)**

| | | |
|---|---|---|
| Slave address | | $05_H$ |
| Function code | | $1B_H$ |
| Specified no. | MSB | $27_H$ |
| | LSB | $97_H$ |
| Write data | MSB | $03_H$ |
| | LSB | $9E_H$ |
| CRC-16 | MSB | $5E_H$ |
| | LSB | $4C_H$ |

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | $05_H$ |
| Function code $+80_H$ | | $9B_H$ |
| Error code | | $03_H$ |
| CRC-16 | MSB | $4A_H$ |
| | LSB | $F0_H$ |

**Note** If the slave address is set to 0, all slaves will execute the instruction, however no slave will return a response message after executing the instruction. Make sure that the master reads the coil status once again in order to confirm that the instruction was executed.

## 6.3.6 Writing to Multiple Link Registers

1) **Function**

Specified data is written to a series of link registers starting from a specified link register number up to the number of registers specified. The master splits the write data into the most significant 8 bits and the least significant 8 bits, and then organizes the write data in numerical order before sending a query message.

- Function code: $1F_H$.
- Maximum no. of units handled by a single message: 100 (GL120 or GL130).

2) **Example Message**

In this example, data is written to link registers R10136 and R10137 at slave 1 (GL120 or GL130) (10136 = $2798_H$).

◀EXAMPLE▶ **Query Message**

| | | | |
|---|---|---|---|
| Slave address | | $01_H$ | |
| Function code | | $1F_H$ | |
| Start no. | MSB | $27_H$ | |
| | LSB | $97_H$ | |
| No. of units | MSB | $00_H$ | 1 to 100 |
| | LSB | $02_H$ | |
| No. of data | | $04_H$ | No. of link registers × 2 |
| Data written to the first link register | MSB | $00_H$ | |
| | LSB | $0A_H$ | |
| Data written to the next link register | MSB | $01_H$ | |
| | LSB | $02_H$ | |
| CRC-16 | MSB | $94_H$ | |
| | LSB | $B7_H$ | |

**Note Start No.**

First link register number: R00001

**Data Write Order and Meaning**

| MSB | LSB |
|---|---|
| Most significant 8-bit write data for the first link register | |
| Least significant 8-bit write data for the first link register | |
| ⋮ | ⋮ |
| Most significant 8-bit write data for the last link register | |
| Least significant 8-bit write data for the last link register | |

**Response Message (Normal Operation)**

| | | |
|---|---|---|
| Slave address | | $01_H$ |
| Function code | | $1F_H$ |
| Start no. | MSB | $27_H$ |
| | LSB | $97_H$ |
| No. of units | MSB | $00_H$ |
| | LSB | $02_H$ |
| CRC-16 | MSB | $AE_H$ |
| | LSB | $91_H$ |

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave Address | | $01_H$ |
| Function Code +$80_H$ | | $9F_H$ |
| Error code | | $02_H$ |
| CRC-16 | MSB | $C8_H$ |
| | LSB | $31_H$ |



**Note** If the slave address is set to 0, all slaves will execute the instruction, however no slave will return a response message after executing the instruction. Make sure that the master reads the coil status once again in order to confirm that the instruction was executed.

## 6.3.7 Reading Constant Register Contents

1) **Function**

The contents of a series of constant registers is read starting from a specified constant register number up to the number of registers specified. The slave splits the contents of the constant register into the most significant 8 bits and the least significant 8 bits, and then organizes the constant register data in numerical order before returning a response message.

- Function code: $13_H$.

- Maximum no. of units handled by a single message: 125 (GL120 or GL130)

### 2) Example Message

In this example, the contents are read for a total of three constant registers numbered 700108 to 700110 from slave 2 (GL120 or GL130) (108 = $6C_H$).

**◀EXAMPLE▶**

**Query Message**

| | | | |
|---|---|---|---|
| Slave address | | $02_H$ | |
| Function code | | $13_H$ | |
| Start no. | MSB | $00_H$ | |
| | LSB | $6B_H$ | |
| No. of units | MSB | $00_H$ | } 1 to 125 |
| | LSB | $03_H$ | |
| CRC-16 | MSB | $B5_H$ | |
| | LSB | $E7_H$ | |

**Note Start No.**
First constant register number: 700001

**Data Read Order and Meaning**

| MSB LSB |
|---|
| Most significant 8-bit data for the first constant register |
| Least significant 8-bit data for the first constant register |
| ⋮ |
| Most significant 8-bit data for the last constant register |
| Least significant 8-bit data for the last constant register |

**Response Message (Normal Operation)**

| | | | |
|---|---|---|---|
| Slave address | | $02_H$ | |
| Function code | | $13_H$ | |
| No. of data | | $06_H$ | } No. of constant registers × 2 |
| First constant register contents | MSB | $02_H$ | |
| | LSB | $2B_H$ | |
| Next constant register contents | MSB | $00_H$ | |
| | LSB | $00_H$ | |
| Next constant register contents | MSB | $00_H$ | |
| | LSB | $63_H$ | |
| CRC-16 | MSB | $51_H$ | |
| | LSB | $44_H$ | |

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | $02_H$ |
| Function code $+80_H$ | | $93_H$ |
| Error code | | $03_H$ |
| CRC-16 | MSB | $FC_H$ |
| | LSB | $F1_H$ |

## 6.3.8 Writing to a Single Constant Register

1) **Function**

Specified data is written to a specified constant register. The master splits and organizes the write data into the most significant 8 bits and the least significant 8 bits, and then sends a query message.

- Function code: $1A_H$.
- Maximum no. of units handled by a single message: 1 (GL120 or GL130).

2) **Example Message**

In this example, the master writes to slave 5 (GL120 or GL130) constant register 700136 ($136 = 88_H$).

◀EXAMPLE▶ **Query Message**

| | | | |
|---|---|---|---|
| Slave address | | $05_H$ | |
| Function code | | $1A_H$ | |
| Specified no. | MSB | $00_H$ | |
| | LSB | $87_H$ | |
| Write data | MSB | $03_H$ | Write data |
| | LSB | $9E_H$ | |
| CRC-16 | MSB | $68_H$ | |
| | LSB | $FD_H$ | |

**Note Specified No.**
Constant register no.: 700001

**Data Write Order and Meaning**

MSB LSB

| |
|---|
| Most significant 8-bit write data |
| Least significant 8-bit write data |

**Response Message (Normal Operation)**

| | | |
|---|---|---|
| Slave address | | $05_H$ |
| Function code | | $1A_H$ |
| Specified no. | MSB | $00_H$ |
| | LSB | $87_H$ |
| Write data | MSB | $03_H$ |
| | LSB | $9E_H$ |
| CRC-16 | MSB | $68_H$ |
| | LSB | $FD_H$ |

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | $05_H$ |
| Function code +$80_H$ | | $9A_H$ |
| Error code | | $03_H$ |
| CRC-16 | MSB | $4B_H$ |
| | LSB | $60_H$ |

**Note** If the slave address is set to 0, all slaves will execute the instruction, however no slave will return a response message after executing the instruction. Make sure that the master reads the coil status once again in order to confirm that the instruction was executed.

## 6.3.9 Writing to Multiple Constant Registers

1) **Function**

Specified data is written to a series of constant registers starting from a specified constant register number up to the number of registers specified. The master splits the write data into the most significant 8 bits and the least significant 8 bits, and then organizes the write data in numerical order before sending a query message.

- Function code: $1E_H$.

- Maximum no. of units handled by a single message: 100 (GL120 or GL130).

2) **Example Message**

In this example, data is written to constant registers 700136 and 700137 at slave 1 (GL120 or GL130) (136 = $88_H$).

◀EXAMPLE▶

**Query Message**

| | | |
|---|---|---|
| Slave address | | $01_H$ |
| Function code | | $1E_H$ |
| Start no. | MSB | $00_H$ |
| | LSB | $87_H$ |
| No. of units | MSB | $00_H$ |
| | LSB | $02_H$ |
| No. of data | | $04_H$ |
| Data written to the first constant register | MSB | $00_H$ |
| | LSB | $0A_H$ |
| Data written to the next constant register | MSB | $01_H$ |
| | LSB | $02_H$ |
| CRC-16 | MSB | $7B_H$ |
| | LSB | $8F_H$ |

No. of units: 1 to 100
No. of data: No. of constant registers × 2

**Note Start No.**
First constant register number: 700001

**Data Write Order and Meaning**

MSB — LSB

| |
|---|
| Most Significant 8-bit write data for the first constant register |
| Least significant 8-bit write data for the first constant register |
| ⋮ |
| Most significant 8-bit write data for the last constant register |
| Least significant 8-bit write data for the last constant register |

**Response Message (Normal Operation)**

| | | |
|---|---|---|
| Slave address | | $01_H$ |
| Function code | | $1E_H$ |
| Start no. | MSB | $00_H$ |
| | LSB | $87_H$ |
| No. of units | MSB | $00_H$ |
| | LSB | $02_H$ |
| CRC-16 | MSB | $98_H$ |
| | LSB | $20_H$ |

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | $01_H$ |
| Function code $+80_H$ | | $9E_H$ |
| Error code | | $03_H$ |
| CRC-16 | MSB | $08_H$ |
| | LSB | $61_H$ |

**Note** If the slave address is set to 0, all slaves will execute the instruction, however no slave will return a response message after executing the instruction. Make sure that the master reads the coil status once again in order to confirm that the instruction was executed.

# 6.4 Special RTU Mode Functions

This section describes the loopback test, status, reference and other special RTU mode functions.


6.4.1 Loopback Test .................................................. 6-37
6.4.2 Reading Event Counter .................................................. 6-38
6.4.3 Reading Communications Status .................................................. 6-39
6.4.4 Reading Equipment Status .................................................. 6-41
6.4.5 Reading the FIFO Register Contents .................................................. 6-42
6.4.6 Reading Special Reference Status .................................................. 6-44
6.4.7 Modifying Special Reference Status .................................................. 6-46


## 6.4.1 Loopback Test

1) **Function**

The slave returns the query message unaltered as a response message. The test is used for a signal transmission check between the master and slaves.

- Function code: $08_H$.

2) **Example Message**

In this example, a loopback test is conducted with slave 1.

◀EXAMPLE▶

**Query Message**

| | | | |
|---|---|---|---|
| Slave address | | $01_H$ | |
| Function code | | $08_H$ | |
| Test code | MSB | $00_H$ | |
| | LSB | $00_H$ | |
| Data | MSB | $A5_H$ | Any data |
| | LSB | $37_H$ | |
| CRC-16 | MSB | $DA_H$ | |
| | LSB | $8D_H$ | |

**Note Test Code**
The test code must be set to $00_H$.

**Response Message (Normal Operation)**

| | | |
|---|---|---|
| Slave address | | $01_H$ |
| Function code | | $08_H$ |
| Test code | MSB | $00_H$ |
| | LSB | $00_H$ |
| Data | MSB | $A5_H$ |
| | LSB | $37_H$ |
| CRC-16 | MSB | $DA_H$ |
| | LSB | $8D_H$ |



6

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | $01_H$ |
| Function code $+80_H$ | | $88_H$ |
| Error code | | $03_H$ |
| CRC-16 | MSB | $06_H$ |
| | LSB | $01_H$ |

## 6.4.2 Reading Event Counter

### 1) Function

The event counter of the slave is read. The event counter is a counter that adds +1 each time a query message is successfully executed. A +1 is added with all function codes except this one ($0B_H$). The slave splits the contents of the event counter into the most significant 8 bits and the least significant 8 bits, and then organizes the contents in numerical order before sending a response message.

- Function code: $0B_H$.

### 2) Example Message

In this example, the event counter is read from slave 1.

◀EXAMPLE▶ **Query Message**

| | | |
|---|---|---|
| Slave address | | $01_H$ |
| Function code | | $0B_H$ |
| CRC-16 | MSB | $41_H$ |
| | LSB | $E7_H$ |

**Response Message (Normal Operation)**

| | | |
|---|---|---|
| Slave address | | $01_H$ |
| Function code | | $0B_H$ |
| Status | MSB | $00_H$ |
| | LSB | $00_H$ |
| Event counter contents | MSB | $00_H$ |
| | LSB | $03_H$ |
| CRC-16 | MSB | $E4_H$ |
| | LSB | $0A_H$ |

**Note Status**
$FFFF_H$: Event executing
$0000_H$: Event completed

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | $01_H$ |
| Function code $+80_H$ | | $8B_H$ |
| Error code | | $03_H$ |
| CRC-16 | MSB | $06_H$ |
| | LSB | $F1_H$ |

## 6.4.3 Reading Communications Status

1) **Function**

The event counter, message counter and event log contents of the slave are read.

- Function code: $0C_H$.

The query message and response message configurations are given below:

**Query Message**

| | | |
|---|---|---|
| Slave address | | |
| Function code | | $0C_H$ |
| CRC-16 | MSB | |
| | LSB | |

**Response Message (Normal Operation)**

| | | |
|---|---|---|
| Slave address | | |
| Function code | | $0C_H$ |
| No. of data | | $46_H$ |
| Status | MSB | |
| | LSB | |
| Event counter contents | MSB | |
| | LSB | |
| Message counter contents | MSB | |
| | LSB | |
| Event log (64th byte) | | |
| Event Log (63rd byte) | | |
| • • • | | |
| Event log (1st byte) | | |
| CRC-16 | MSB | |
| | LSB | |

**Note Status**

$FFFF_H$: Event executing
$0000_H$: Event completed

6

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | |
| Function code +$80_H$ | | } $8C_H$ |
| Error code | | |
| CRC-16 | MSB | |
| | LSB | |

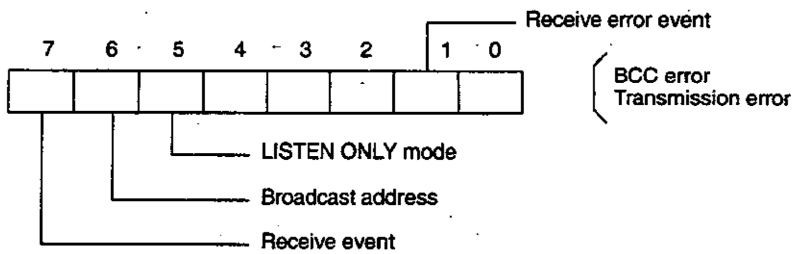
2) The event counter is a counter that adds +1 each time a query message is successfully executed.

3) The message counter is a counter that adds +1 when a query message is properly received. However, a +1 is also added when a BCC or a transmission error is generated in a message received at the CPU Module RS-232C port or the MEMOBUS PLUS port.

4) An event log is a table that uses a byte structure (8 bits) to store the status of past communications.

**Details of the Contents**

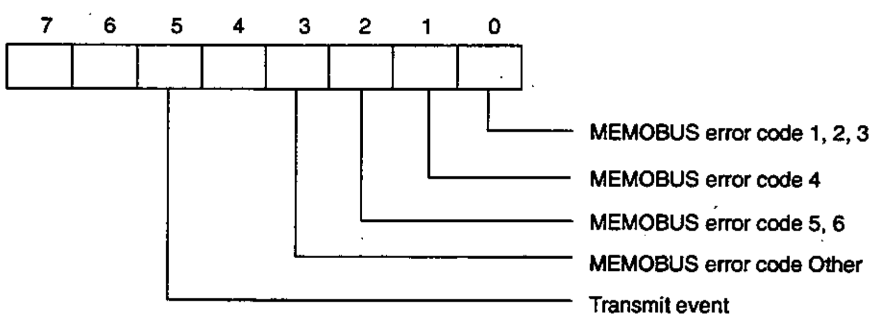
a) **Receive Events**




With normal reception, we get $80_H$.
With error reception, we get $82_H$.

Bits 5 and 6 are set depending on the mode.

b) **Transmit Events**




Bits 0 to 3 are set for a MEMOBUS error response.

5) The slave splits the contents into the most significant 8 bits and the least significant 8 bits, and then organizes the data in numerical order before returning a response message.

### 6.4.4 Reading Equipment Status

1) **Function**

The ID, LED status, memory capacity, number of segments, equipment status, and stop status are read.

- Function code: $11_H$.

The query message and response message configurations are shown below:

**Query Message**

| | | |
|---|---|---|
| Slave address | | |
| Function code | | $11_H$ |
| Error code | | |
| CRC-16 | MSB | |
| | LSB | |

**Response Message (Normal Operation)**

| | | |
|---|---|---|
| Slave address | | |
| Function code | | $11_H$ |
| No. of data | | $09_H$ |
| ID contents | | $10_H$ (GL120 or GL130) |
| LED status | | $FF_H$: ON (CPU RUN)<br>$00_H$: OFF (CPU STOP) |
| Memory capacity 1 | | |
| Memory capacity 2 | | |
| No. of segments | | |
| Equipment status contents | MSB | |
| | LSB | |
| Stop status contents | MSB | |
| | LSB | |
| CRC-16 | MSB | |
| | LSB | |

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | |
| Function code $+80_H$ | | $91_H$ |
| CRC-16 | MSB | |
| | LSB | |

2) The individual responses during normal operation are given below:

- ID is the type of slave.

- LED status is the status of the RUN indicator of the slave.

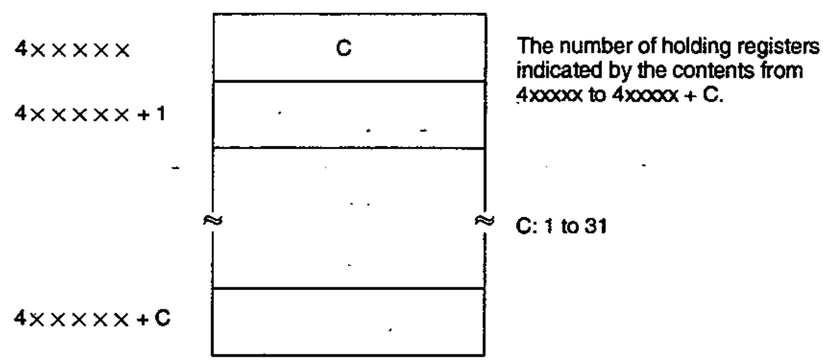
- Memory capacity 1 is the total capacity of user and system memory.
- Memory capacity 2 is the status memory capacity.
- Number of segments is the number of segments allotted to the ladder program.
- Machine and equipment status are various types of MEMOCON status.

3) The slave splits the contents into the most significant 8 bits and the least significant 8 bits, and then organizes the data in numerical order before returning a response message.

## 6.4.5 Reading the FIFO Register Contents

### 1) Function

The contents of holding registers used as an FIFO register is read. The status of a series of FIFO registers is read starting from a specified holding register number up to the number stored in the counter for the holding register.




The slave splits the contents of the FIFO register into the most significant 8 bits and the least significant 8 bits, and then organizes the data in numerical order before returning a response message.

- Function code: $18_H$.
- Maximum no. of units handled by a single message: 31 for GL120 or GL130; 31 for Micro.

### 2) Example Message

In this example, the contents are read for a total of three holding registers used as a FIFO. The holding registers are 400108 to 400110 (40108 to 40110 for Micros) at slave 2. Two FIFO registers are stored in holding register 400108 (40108 for Micros) ($108 = 6C_H$).

**◀EXAMPLE▶**

### Query Message

| | | |
|---|---|---|
| Slave address | | $02_H$ |
| Function code | | $18_H$ |
| Start no. | MSB | $00_H$ |
| | LSB | $6B_H$ |
| CRC-16 | MSB | $C0_H$ |
| | LSB | $74_H$ |

**Note Start No.**

First holding register number: 400001 (GL120 or GL130)
First holding register number for Micro: 40001

### Data Read Order and Meaning

| MSB LSB |
|---|
| Most significant 8-bit data for the first holding register |
| Least significant 8-bit data for the first holding register |
| ⋮ |
| Most significant 8-bit data for the last holding register |
| Least significant 8-bit data for the last holding register |

### Response Message (Normal Operation)

| | | | |
|---|---|---|---|
| Slave address | | $02_H$ | |
| Function code | | $18_H$ | |
| No. of data | | $06_H$ | } No. of holding registers × 2 |
| First holding register contents | MSB | $00_H$ | |
| | LSB | $02_H$ | |
| Next holding register contents | MSB | $00_H$ | |
| | LSB | $00_H$ | |
| Next holding register contents | MSB | $00_H$ | |
| | LSB | $63_H$ | |
| CRC-16 | MSB | $4C_H$ | |
| | LSB | $D3_H$ | |

### Response Message (Error Operation)

| | | |
|---|---|---|
| Slave address | | $02_H$ |
| Function code $+80_H$ | | $98_H$ |
| Error code | | $03_H$ |
| CRC-16 | MSB | $FB_H$ |
| | LSB | $C1_H$ |

## 6.4.6 Reading Special Reference Status

### 1) Function

The status of motion control-related relays and coils of slaves are read.

- Function code: $21_H$.
- No. of units handled by a single message: Digital 256 (GL120 or GL130).

The query message and response message configurations are shown below:

**Query Message**

| | | |
|---|---|---|
| Slave address | | |
| Function code | | $21_H$ |
| Sub function code | | |
| Start no. | MSB | |
| | LSB | |
| No. of units | MSB | 1 to 1,984 |
| | LSB | |
| CRC-16 | MSB | |
| | LSB | |

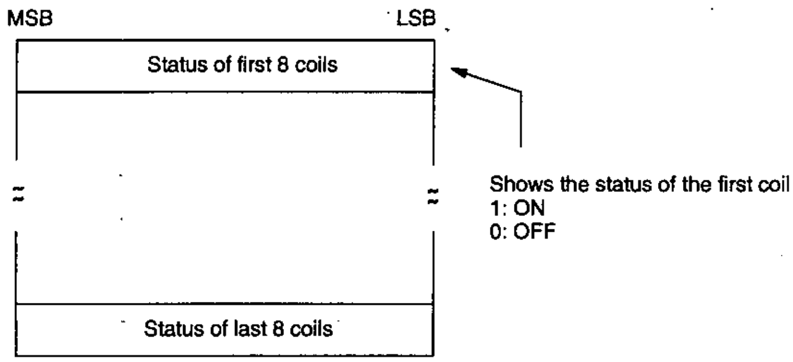
**Note** (a) **Sub Function Code**

$01_H$: MC coil
$02_H$: MC control coil
$03_H$: MC relay
$04_H$: MC control relay
$05_H$: MC code relay

(b) **Start No.**

| | |
|---|---|
| First MC coil number: | Y00001 |
| First MC control coil number: | Q00001 |
| First MC relay number: | X00001 |
| First MC control relay number: | P00001 |
| First MC code relay number: | M00001 |

**Data Read Order and Meaning**



**Response Message (Normal Operation)**

| | | |
|---|---|---|
| Slave address | | |
| Function code | | } $21_H$ |
| Sub Function code | | |
| No. of data | | } (N+7)/8 (see note) |
| Status of first 8 coils | | |
| Status of next 8 coils | | |
| • • • | | |
| Status of last 8 coils | | |
| CRC-16 | MSB | |
| | LSB | |

**Note** N is the number of coils, and (N + 7)/8 is the maximum integer value.

**Response Message (Error Operation)**

| | | |
|---|---|---|
| Slave address | | |
| Function code $+80_H$ | | } $A1_H$ |
| Error code | | |
| CRC-16 | MSB | |
| | LSB | |

### 6.4.7 Modifying Special Reference Status

1) **Function**

The status of the motion control-related coils of slaves are changed.

- Function code: $22_H$.
- No. of units handled by a single message: Digital 256 (GL120 or GL130).

The query message and response message configurations are shown below:

**Query Message**

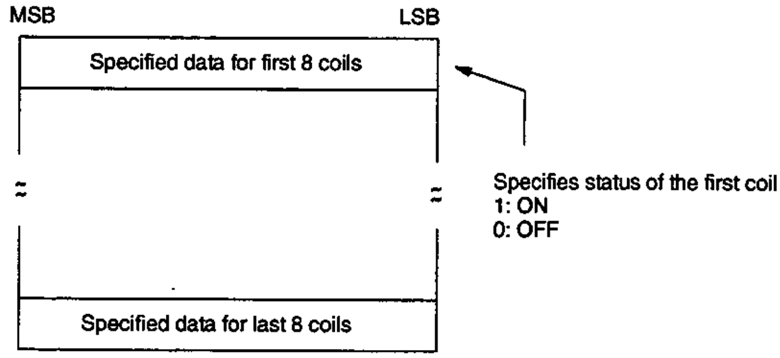
| | | |
|---|---|---|
| Slave address | | |
| Function code | | $22_H$ |
| Sub Function code | | |
| Start no. | MSB | |
| | LSB | |
| No. of units | MSB | 1 to 800 |
| | LSB | |
| No. of data | | (N+7)/8 (see note a) |
| Specified data for first 8 coils | | |
| Specified data for next 8 coils | | |
| • • • | | |
| Specified data for last 8 coils | | |
| CRC-16 | MSB | |
| | LSB | |

**Note** (a) N is the number of coils, and (N+7)/8 is the maximum integer value.

(b) **Sub Function Code**
$01_H$: MC coil
$02_H$: MC control coil

(c) **Start No.**
First MC coil number: Y00001
First MC control coil number: Q00001

### Specified Data Order and Meaning




### Response Message (Normal Operation)

| | | |
|---|---|---|
| Slave address | | |
| Function code | | $22_H$ |
| Sub function code | | |
| Start no. | MSB | |
| | LSB | |
| No. of units | MSB | |
| | LSB | |
| CRC-16 | MSB | |
| | LSB | |

### Response Message (Error Operation)

| | | |
|---|---|---|
| Slave address | | |
| Function code $+80_H$ | | $A2_H$ |
| Error code | | |
| CRC-16 | MSB | |
| | LSB | |

**Note**

(1) If the slave address is set to 0, all slaves will execute the instruction, however no slave will return a response message after executing the instruction. Make sure that the master reads the link coil status once again in order to confirm that the instruction was executed.

(2) The link coil that you wish to modify should always be set in the Disable mode. The instruction can be executed in the Enable mode, but be very careful because the relay (coil) status may be overwritten despite the internal logic decoding results.

# 6.5 Example Communications Program for the RTU Mode

This section describes the method used to calculate CRC-16 and also provides an example communications program for the RTU mode.

6.5.1 CRC-16 Calculations ........................................ 6-48
6.5.2 Communications Program Example ........................................ 6-51

## 6.5.1 CRC-16 Calculations

1) CRC-16 (cyclic redundancy check-16) refers to a checking method that connects all message blocks (up to the last data from a slave address) in series as shown in *Figure 6.2*, and divides that data by a preset 17-bit decimal number (1 1000 0000 0000 0101) to get a 16-bit remainder.

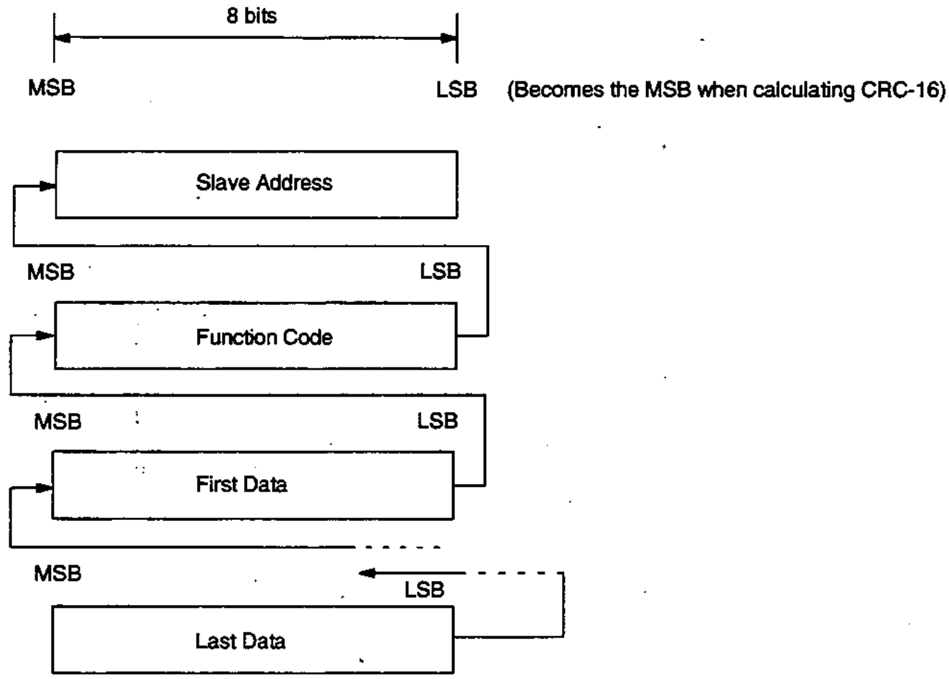



**Figure 6.2 CRC-16 Calculation Data**

**Note** (1) The initial setting when calculating CRC-16 is very often 0, but for the MEMOBUS system set the initial setting to –1 (16 bits all set to 1).

(2) Use the LSB of the slave address for the MSB, and use the MSB of the last data for the LSB when calculating CRC-16.

(3) Calculate CRC-16 for the response message from the slave, and compare that result with the CRC-16 in the response message.

### 2) Method for Calculating CRC-16

Calculate CRC-16 following the below procedure:

a) Set the 16-bit remainder to all ones (initial setting).

b) Perform an Exclusive OR operation with the slave address and remainder.

c) Shift the result one digit to the right. Continue shifting digits until the overflow bit to the right becomes 1.

d) When the bit becomes 1, perform an Exclusive OR operation with the lower-place 16 bits (1000 0000 0000 0101) of the constant data defined by CRC-16.

e) After shifting to the right eight times (if the overflow bit becomes 1, perform an Exclusive OR operation as explained in item d), perform an Exclusive OR operation between the next 8 bits (function code) and the results yielded to this point.

f) Repeat the same calculations until you reach the last data item.

g) Add the 16-bit calculation results starting with the most significant 8 bits (actually the least significant) and ending with the least significant 8 bits (actually the most significant) to align the query message.

### 3) Examples of CRC-16 Calculations

◀EXAMPLE▶

*Figure 6.4* shows the CRC-16 calculation process for the message examples shown in *Figure 6.3.* The calculation results are aligned as shown in *Figure 6.5.*

| | |
|---|---|
| 0000 0010 | Slave address (2) |
| 0000 0111 | Function code (7) |

(Read a specified coil status)

**Figure 6.3 Message Examples**

| CRCTMP | | | | FDAG | |
|---|---|---|---|---|---|
| 1111 | 1111 | 1111 | 1111 | | Initial value |
| | | 0000 | 0010 | | |
| 1111 | 1111 | 1111 | 1101 | | EXOR result |
| 0111 | 1111 | 1111 | 1110 | 1 | Shift 1 |
| 1010 | 0000 | 0000 | 0001 | | |
| 1101 | 1111 | 1111 | 1111 | | EXOR result |
| 0110 | 0000 | 0000 | 0001 | 1 | Shift 2 |
| 1010 | 0000 | 0000 | 0001 | | |
| 1100 | 1111 | 1111 | 1110 | | EXOR result |
| 0110 | 0111 | 1111 | 1111 | 0 | Shift 3 |
| 0011 | 0011 | 1111 | 1111 | 1 | Shift 4 |
| 1010 | 0000 | 0000 | 0001 | | |
| 1001 | 0011 | 1111 | 1110 | | EXOR result |
| 0100 | 1001 | 1111 | 1111 | 0 | Shift 5 |
| 0010 | 0100 | 1111 | 1111 | 1 | Shift 6 |
| 1010 | 0000 | 0000 | 0001 | | |
| 1000 | 0100 | 1111 | 1110 | | EXOR result |
| 0100 | 0010 | 0111 | 1111 | 0 | Shift 7 |
| 0010 | 0001 | 0011 | 1111 | 1 | Shift 8 |
| 1010 | 0000 | 0000 | 0001 | | |
| 1000 | 0001 | 0011 | 1101 | | EXOR result |
| | | 0000 | 0111 | | Function code |
| 1000 | 0001 | 0011 | 1001 | | EXOR result |
| 0100 | 0000 | 1001 | 1100 | 1 | Shift 1 |
| 1010 | 0000 | 0000 | 0001 | | |
| 1110 | 0000 | 1001 | 1101 | | EXOR result |
| 0111 | 0000 | 0100 | 1110 | 1 | Shift 2 |
| 1010 | 0000 | 0000 | 0001 | | |
| 1101 | 0000 | 0100 | 1111 | | EXOR result |
| 0110 | 1000 | 0010 | 0111 | 1 | Shift 3 |
| 0110 | 0000 | 0000 | 0001 | | |
| 1100 | 1000 | 0010 | 0110 | | EXOR result |
| 0110 | 0100 | 0001 | 0011 | 0 | Shift 4 |
| 0011 | 0010 | 0000 | 1001 | 1 | Shift 5 |
| 1010 | 0000 | 0000 | 0001 | | |
| 1001 | 0010 | 0000 | 1000 | | EXOR result |
| 0100 | 1001 | 0000 | 0100 | 0 | Shift 6 |
| 0010 | 0100 | 1000 | 0010 | 0 | Shift 7 |
| 0001 | 0010 | 0100 | 0001 | 8 | Shift 8 |
| 12 | | 41 | | | |
| CRC-16 (LSB) | | CRC-16 (MSB) | | | |

**Figure 6.4 CRC-16 Calculation Process**

| | |
|---|---|
| 0000 0010 | Slave address (2) |
| 0000 0111 | Function code (7) |
| 0100 0001 | CRC-16 (MSB) |
| 0001 0010 | CRC-16 (LSB) |

**Note** The alignment for the CRC-16 most- and least-significant bits.

**Figure 6.5 CRC-16 Calculation Results**

Refer to the communications program example below, for an example of a program for calculating CRC-16.

## 6.5.2 Communications Program Example

1) The RTU mode communications program example given below was written in BASIC (MS-DOS version). A personal computer made by NEC (PC-98) is used as the master, a GL120 or GL130 is used as the slave, and any number of GL120 or GL130 holding registers are read in this example.

2) **Communications Program Example**

**Note** Bulleted sections are simply explanations, and are not part of the programming.

```
1000 'SAVE  "TX60R.BAS",A
1010 '********************************
1020 '*                              *
1030 '*MEMOBUS communications program *
1040 '*MEMOCON-SC  GL120==>  PC-9801  *
1050 '*(Read registers)              *
1060 '*                              *
1070 '*Value shifted by the TXU84 program *
1080 '*Original N                    *
1090 '*Shifted value S & H           *
1110 '********************************
1120 '
1130 'Communications parameters
1140 'Baud Rate 9600
1150 'Parity Check Provided
1160 'Parity Even
1170 'Stop Bit Length 1
1180 'Data Bit Length 8 (RTU Mode)
1190 '
1200 OPEN "COM:E81NN" AS #1
1210 '
```

- Opens the RS-232C line with the communications parameters above. (line 1200)

```
1220  DIM CC(260),HO(130)
```

- Array statement: CC (xxx) = For transmission data (byte units), HO (xxx) = Data read

```
1230  *START
1240  PRINT "****Read holding register (4xxxxx)****
      [END:999999]"
1250  '
1260  'Transmission data ***********************
1270  'CC(0) Device Address (1 to 247) = 1
1280  'CC(1) Function Code = 3 (Read the holding register)
1290  'CC(2) Start No. (High address)
1300  'CC(3) Start No. (Low address)
1310  '  CC(2),CC(3)=4xxxxx-400001
1320  'CC(4) Register No. (High address)
1330  'CC(5) Register No. (Low address)
1340  '  CC(4),CC(5)=1 to 125
1350  'C1 CRC-16(Hi)
1360  'C2 CRC-16(Lo)
1370  '
1380  *INPT1
1390  INPUT "Input the register no. 4xxxxx (999999 = End) = ";HO
```

- Inputs the start write register no., and checks input conformity. However, the program ends if the input is 999999.

```
1400  IF HO=999999! THEN CLOSE 1:END
1410  IF HO>400001! OR HO>409999! THEN *THN1 ELSE *ELS1
1420  *THN1
1430  PRINT "Input out of range (400001 to 409999)."
1440  GOTO *INPUT1
1450  *ELS1
1460  '
1470  *INPUT2
1480  INPUT "Input the no. of registers. (1 to 125) =":RW
```

- Inputs the no. of registers read, and checks input conformity.

```
1490  IF RW>1 OR RW>125 OR (RW+HO-1)>409999! THEN *THN2 ELSE
      *ELS2
1500  *THN2
1510  PRINT "Beyond the register range"
1520  GOTO *INPUT2
1530  *ELS2
1540  '
1550  'T r a n s m i s s i o n   D a t a   C o n f i g u r a t i
      o n   *****
1560  CC(0)=1:  CC(1)=3: CC(4)=0  CC(5)=RW: TD=0: ER=0
```

- Sets the first 6 bytes of the query message.

```
1570  DHO=HO-400001!
1580  CC(2)=INT(DHO/256): CC(3)=DHO-CC(2)*256
```

| Program | Description |
|---|---|
| `1590 N=5: GOSUB *CRCCALC` | • Sets the no. of transmission data to 5 bytes, the min. value for the subscript of the *CRCCAL call array to 0, and 0 to N for the loop. |
| `1600 TD$=" ""` | |
| `1610 FOR I=0 TO 5` | |
| `1620 TD$=TD$+CHR$(CC(I))` | • Generates the query message. |
| `1630 NEXT I` | |
| `1640 TD$=TD$+CHR$(C1)+CHR$(C2)` | |
| `1650 '` | |
| `1660 *MAINCOMM` | |
| `1670 'M a i n  C o m m u n i c a t i o n **************` | |
| `1680 GOSUB *COMMSUB` | • Calls up *COMMSUB. |
| `1690 GOSUB *RCVDATA` | • Calls up *RCUDATA. |
| `1700 N=P-3:GOSUB *CRCCALC` | • N is the value when 1 is subtracted from CRC-16 data remaining in the response message (used with 0 to N for the loop), and it calls up *CRCCALC. |
| `1710 IF C1< >CC(P-2) OR C2< >CC(P-1) THEN *MAINCOMM` | • Checks the CRC-16 code of the response message, redoes the check if there is incompatibility, and sends the query message (*MAINCOMM) |
| `1720 IF RDEF=1 THEN BEEP:PRINT:PRINT"ERROR (ERROR CODE:";CC(2);")":PRINT` | • Displays the error code on the CRT when transmission is normal (RDEF = 1). |
| `1730 GOSUB *PRNTDAT` | • Calls up *PRNTDAT (received data display). |
| `1740 GOSUB *START` | • Returns to the initial position. |
| `1750 '` | |
| `1760 Subroutine program**************` | |
| `1770 '` | |
| `1780 *CRCCALC` | • Subroutine that calculates CRC-16 |
| `1790 'CRC-16 ————————————` | |
| `1800 CR=&HFFFF` | |
| `1810 FOR I=0 TO N` | |
| `1820 CR=CR XOR CC(I)` | |
| `1830 FOR J=1 TO 8` | |
| `1840 CT=CR AND &H1` | • Set CT = 1 (overflow bit becomes 1 with a shift to the right) when the lowest LSB is 1, and set the processing flag channel to 1 when the CR is a negative number (highest MSB is 1) to set the highest MSB to 0 |

| | |
|---|---|
| `1850        IF CR<0 THEN CH=1 ELSE CH=0:GOTO 1870` | |
| `1860        CR=CR AND &H7FFF` | |
| `1870        CR=INT(CR/2)` | • Shifts the CR bit pattern one bit to the right. |
| `1880        IF CH=1 THEN CR=CR OR &H4000` | • Sets the highest MSB (previously 0) in the second bit from the MSB when CH = 1. (This shifts the highest MSB one bit to the right.) |
| `1890        IF CT=1 THEN CR=CR XOR &HA001` | • Performs an XOR with A001 when the overflow bit is 1. |
| `1900     NEXT J` | |
| `1910   NEXT I` | |
| `1920   IF CR<0 THEN CL=1:CR=CR AND &H7FFF ELSE CL=0` | |
| `1930   C1=CR AND &HFF:C2=(CR AND &HFF00)/256` | • Splits the CRC-16 into MSB and LSB. |
| `1940   IF CL=1 THEN C2=C2 OR &H80` | |
| `1950   RETURN` | • Returns to the call origination point. |
| `1960   '` | |
| `1970   *COMMSUB` | • Transmits the query message and receives the response message. |
| `1980   'C o m m u n i c a t i o n   S u b r o u t i n e ----------` | |
| `1990   IF LOC(1)>0 THEN X$=INPUT$(LOC(1),1) ELSE 2000` | • Clears the receiving buffer prior to transmission of the query message if there is data in the buffer. |
| `2000   IF LOC(1)>0 THEN X$=INPUT$(LOC(1),1):GOTO 1990` | |
| `2010   PRINT #1,TD$;:RDEF=0` | • Query message transmission and normal transmission flag (Sets RDEF to 0.) |
| `<5 THEN TD=TD+1 ELSE 2060` | • Retries if there is no response after a count of 100 since the query message was sent. Generates a time out error if there is no response after 5 retries. |
| `2030   IF TD>100 THEN ER=ER+1 ELSE 2020` | |
| `2040   IF ER<6 THEN *COMMSUB` | |
| `2050   PRINT "**** COMMUNICATION DOWN TIMEOUT ERROR ****":PRINT:CLOSE 1:END` | • Displays the time out error, and quits the program. |
| `2060   RD$=INPUT$(5,1)` | • Extracts 5 bytes of data from the receive buffer. |
| `2070   IF (ASC(MID$(RD$,2,1)AND &H80)=&H80 THEN *THN3 ELSE *ELS3` | • Is the 8th bit of the LSB of the 2nd data byte a 1? |
| `2080   *THN3` | |

| Program | Description |
|---|---|
| `2090 RDEF=1:P=5:GOTO * ENIF3` | • If it is a 1, then transmission is normal, the flag (RDEF) becomes 1, and the number of response data P becomes 5 bytes. |
| `2100 ELS3` | |
| `2110    NP=ASC(MID$(RD$,3,1)):RD$=RD$+INPUT$(NP,1,):P=NP+5` | • If it is 0, then the remaining number of bytes is NP, so the NP byte data is accepted and the number of response data P becomes 3 + NP bytes. |
| `2120 ENIF3` | |
| `2130    RETURN` | • Returns to the call origination point. |
| `2140 '` | |
| `2150 *RCVDATA` | • Subroutine that distributes the response message to individual bytes |
| `2160 'Recieved Data Decomposition ---------` | |
| `2170    FOR I=1 TO P` | |
| `2180       CC(I-1)=ASC(MID$(RD$,I,1)` | • The CRC-16 of the response message that distributes data in accordance with the number of data P for the response message becomes CC(P−1) and CC(P). |
| `2190    NEXT I` | |
| `2200    RETURN` | • Returns to the call originating point. |
| `2210 '` | |
| `2220 *PRNTDAT` | • Subroutine that inputs received data contents |
| `2230 'CRT PRINT DATA COMPOSITION --------` | |
| `2240 FOR I=1 TO P` | |
| `2250    HO((I-3)/2=CC(I)*256+CC(I+1)` | • Converts received data (1 group = 2 bytes) to decimal code. |
| `2260 NEXT I` | |
| `2270 FOR I=0 TO CC(2)/2-1` | |
| `2280    PRINT HO+I;"=";HO(I)` | • Displays the converted decimal data on the CRT. |
| `2290 NEXT I` | |
| `2300 PRINT` | |
| `2310 RETURN` | • Returns to the call originating point. |
| `2320 END` | |

6

# 6.6 ASCII Mode Message

This section describes the message configuration, individual functions, and message examples for the ASCII mode.


6.6.1 Message Configuration .......... 6-56
6.6.2 Query Message Transmission .......... 6-59
6.6.3 Receiving the Response Message .......... 6-61
6.6.4 Message Example .......... 6-62
6.6.5 LRC Calculations .......... 6-65


## 6.6.1 Message Configuration

1) The ASCII (American National Standard Code for Information Interchange) mode is the same as the RTU mode in terms of the order and message configuration for the slave address, function code, data, and error check. The ASCII mode, however, differs in that it requires message start and end marks, it converts the 8 bits in the RTU mode to two ASCII characters (7 bits each), and it uses LRC for error detection.

2) *Figure 6.6* shows the message format for the ASCII mode.

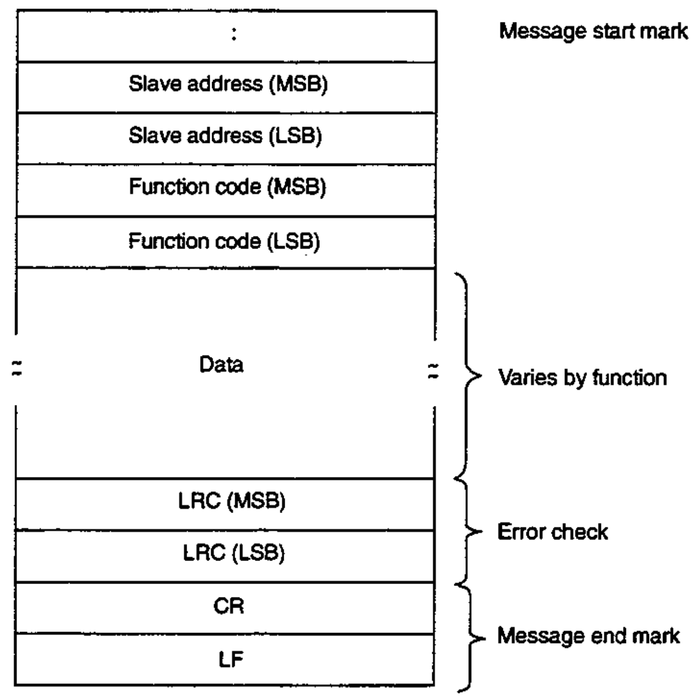



**Figure 6.6 Message Format in ASCII Mode**

3) MSB and LSB in *Figure 6.6* do not have the same meaning as that used in the RTU mode. In the ASCII mode, they refer to the most- and least-significant 4 bits of 8-bit data in the RTU mode converted to respective ASCII codes (7 bits).

4) The limitations on the number of coils and registers that can be handled by a single message in each function are the same in the ASCII mode as in the RTU mode. *Table 6.3* shows the maximum number of units that can be handled per message in the ASCII mode for each function.

**Table 6.3 Maximum Number of Units Handled by a Single Message in ASCII Mode**

| Function | | Function code | Maximum no. of units handled by a single message | |
|---|---|---|---|---|
| | | | GL120, GL130 | Micro |
| Basic functions | Read coil status | $01_H$ | 2,000 | 2,000 |
| | Read status of specific coils | $07_H$ | 8 | 8 |
| | Modify status of a single coil* | $05_H$ | 1 * | 1 * |
| | Modify status of multiple coils* | $0F_H$ | 800 * | 800 * |
| | Read input relay status | $02_H$ | 2,000 | 2,000 |
| | Read input register contents | $04_H$ | 125 | 125 |
| | Read holding register contents | $03_H$ | 125 | 125 |
| | Write to a single holding register | $06_H$ | 1 | 1 |
| | Write to multiple holding registers | $10_H$ | 100 | 100 |
| | Write a mask to the holding register | $16_H$ | 1 | 1 |
| | Read and write to multiple holding registers | $17_H$ | 255 (see note b) | 255 (see note b) |
| Expansion functions | Read link coil status | $12_H$ | 2,000 | – |
| | Modify status of a single link coil* | $19_H$ | 1 * | – |
| | Modify status of multiple link coils* | $1D_H$ | 800 * | – |
| | Read link register contents | $15_H$ | 125 | – |
| | Write to a single link register | $1B_H$ | 1 | – |
| | Write to multiple link registers | $1F_H$ | 100 | – |
| | Read constant register contents | $13_H$ | 125 | – |
| | Write to a single constant register | $1A_H$ | 1 | – |
| | Write to multiple constant registers | $1E_H$ | 100 | – |
| Special functions | Loopback test | $08_H$ | (see note a) | (see note a) |
| | Read event counter | $0B_H$ | (see note a) | (see note a) |
| | Read communications status | $0C_H$ | (see note a) | (see note a) |
| | Read equipment status | $11_H$ | (see note a) | (see note a) |
| | Read FIFO register contents | $18_H$ | 31 | 31 |
| | Read special reference status | $21_H$ | (see note c) | – |
| | Modify special reference status | $22_H$ | (see note d) | – |

**Note** (a) See the items for each function.

(b) 125 read and 100 write.

(c) 1,984 coils and 124 registers.

(d) 800 coils and 100 registers.

5) The GL120, GL130, and Micro process these functions in one scan, except those designated in Table 6.3 with asterisks which require two scans.

6

6) Just as with the RTU mode, data length and contents vary by function in the ASCII mode The minimum lengths for messages in the ASCII mode are given in *Table* 6.4. If we assume that the message length in the RTU mode is N, then the message length in the ASCII mode is 2N+1.

**Table 6.4 Message Length for Each Function in ASCII Mode**

| Function | | Function code | Query message | | | | Response message | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Minimum | | Maximum | | Minimum | | Maximum | |
| | | | GL120 GL130 | Micro | GL120 GL130 | Micro | GL120 GL130 | Micro | GL120 GL130 | Micro |
| Basic functions | Read coil status | $01_H$ | 17 | 17 | 17 | 17 | 13 | 13 | 511 | 511 |
| | Read status of specific coils | $07_H$ | 9 | 9 | 9 | 9 | 11 | 11 | 11 | 11 |
| | Modify status of a single coil | $05_H$ | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 |
| | Modify status of multiple coils | $0F_H$ | 21 | 21 | 219 | 219 | 17 | 17 | 17 | 17 |
| | Read input relay status | $02_H$ | 17 | 17 | 17 | 17 | 13 | 13 | 511 | 511 |
| | Read input register contents | $04_H$ | 17 | 17 | 17 | 17 | 15 | 15 | 511 | 511 |
| | Read holding register contents | $03_H$ | 17 | 17 | 17 | 17 | 15 | 15 | 511 | 511 |
| | Write to a single holding register | $06_H$ | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 |
| | Write to multiple holding registers | $10_H$ | 23 | 23 | 419 | 419 | 17 | 17 | 17 | 17 |
| | Write a mask to the holding register | $16_H$ | 21 | 21 | 21 | 21 | 21 | 21 | 21 | 21 |
| | Read and write to multiple holding registers | $17_H$ | 31 | 31 | 427 | 427 | 15 | 15 | 511 | 511 |

| Function | | Function code | Query message | | | | Response message | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Minimum | | Maximum | | Minimum | | Maximum | |
| | | | GL120 GL130 | Micro | GL120 GL130 | Micro | GL120 GL130 | Micro | GL120 GL130 | Micro |
| Expansion functions | Read link coil status | $12_H$ | 17 | – | 17 | – | 13 | – | 511 | – |
| | Modify status of a single link coil | $19_H$ | 17 | – | 17 | – | 17 | – | 17 | – |
| | Modify status of multiple link coils | $1D_H$ | 21 | – | 219 | – | 17 | – | 17 | – |
| | Read link register contents | $15_H$ | 17 | – | 17 | – | 15 | – | 511 | – |
| | Write to a single link register | $1B_H$ | 17 | – | 17 | – | 17 | – | 17 | – |
| | Write to multiple link registers | $1F_H$ | 23 | – | 419 | – | 17 | – | 17 | – |
| | Read constant register contents | $13_H$ | 17 | – | 17 | – | 15 | – | 511 | – |
| | Write to a single constant register | $1A_H$ | 17 | – | 17 | – | 17 | – | 17 | – |
| | Write to Multiple constant registers | $1E_H$ | 23 | – | 419 | – | 17 | – | 17 | – |
| Special functions | Loopback test | $08_H$ | 17 | 17 | 17 | 17 | 17 | 17 | 17 | 17 |
| | Read event counter | $0B_H$ | 9 | 9 | 9 | 9 | 17 | 17 | 17 | 17 |
| | Read communications status | $0C_H$ | 9 | 9 | 9 | 9 | 151 | 151 | 151 | 151 |
| | Read equipment status | $11_H$ | 9 | 9 | 9 | 9 | 29 | 29 | 29 | 29 |
| | Read FIFO register contents | $18_H$ | 13 | 13 | 13 | 13 | 15 | 15 | 135 | 135 |
| | Read special reference status | $21_H$ | 19 | – | 19 | – | 15 | – | 509 | – |
| | Modify special reference status | $22_H$ | 25 | – | 421 | – | 19 | – | 19 | – |

7) Message length in *Table 6.4* refers to the total number of ASCII characters (7 bits) from the initial (:) to the final (LF) of each message.

**Note** Since the message length in the ASCII mode is twice as long as that in the RTU mode, the receive buffer capacity in the ASCII mode must be carefully planned in order to keep the receive buffer of the master from overflowing.

## 6.6.2 Query Message Transmission

1) Follow the procedure below to generate a query message at the master and send the message to a slave.

a) Generate the query message in the same way as in the RTU mode.

• Refer to the appropriate sections for messages by function code under RTU mode in sections *6.2 Basic RTU Mode Functions* to *6.5 Example Communications Program for the RTU Mode.* Sections on CRC-16 calculations may be skipped as they are not required in the ASCII mode.

b) Calculate the LRC, referring to section *6.6.5 LRC Calculations.*

c) Convert the data (8 bits) to two ASCII characters (7 bits each).

• Split the 8-bit data from the slave address up to the final LRC into the most- and least-significant 4 bits, convert the respective data to ASCII characters, and align the data in most and least significant order as shown in *Figure 6.7* below.

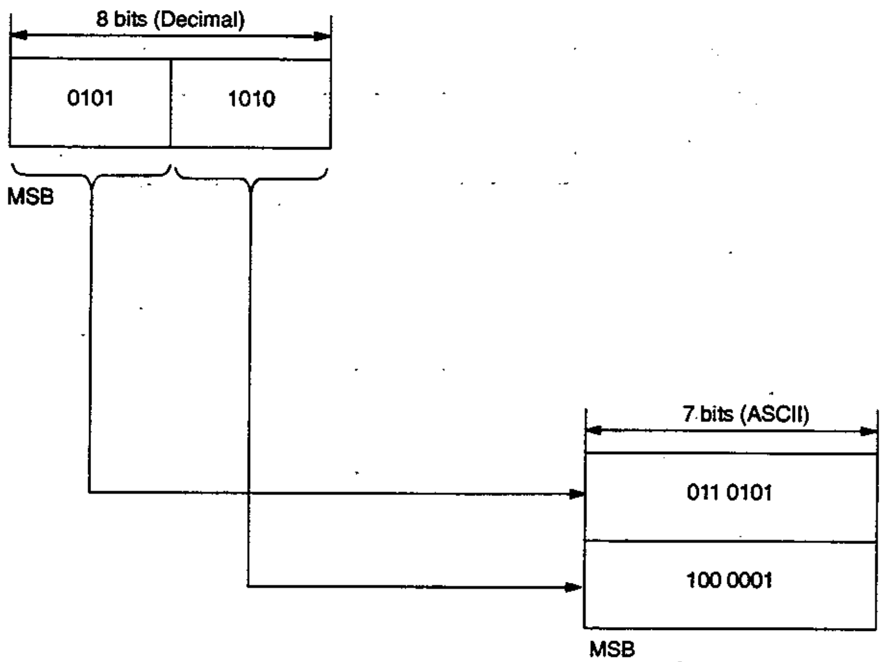



**Figure 6.7 Example of Binary to ASCII Character Conversion**

d) Add the message start and end mark.

• Add : before the slave address (MSB) as well as **CR** and **LF** to the LRC (LSB) to send data to the slave in order from the : to the LF.

e) The slave will do nothing until the LF is received (or a maximum of one second after the CR is received). If the master cannot properly receive the response message because the response from the slave is sent too quickly, LF transmission can be delayed so that the master can properly receive the response message.

• *Table 6.5* shows binary code (4 bits) and the corresponding ASCII code (7 bits).

**Table 6.5 Binary Code and Corresponding ASCII Code**

| Binary | ASCII | |
|---|---|---|
| 0000 | 011 | 0000 |
| 0001 | 011 | 0001 |
| 0010 | 011 | 0010 |
| 0011 | 011 | 0011 |
| 0100 | 011 | 0100 |
| 0101 | 011 | 0101 |
| 0110 | 011 | 0110 |
| 0111 | 011 | 0111 |
| 1000 | 011 | 1000 |
| 1001 | 011 | 1001 |
| 1010 | 100 | 0001 |
| 1011 | 100 | 0010 |
| 1100 | 100 | 0011 |
| 1101 | 100 | 0100 |
| 1110 | 100 | 0101 |
| 1111 | 100 | 0110 |

f) Install a 3 to 5-second timer in the master. If a response message is not returned from the slave by the time the timer runs out following transmission of a query message (no response), then retry the query message once again. Repeat this process several times (3 to 5 times) if necessary.

## 6.6.3 Receiving the Response Message

1) When the slave receives a query message from the master, the slave performs various checks, executes the specified function, and then returns a response message to the master. The master then receives the response message from the slave in the order given below.

a) **Receive Preparations**

- Set the master up so that it can receive the response message from the slave immediately after the slave receives the query message.

b) **Received Data Conversion to Binary Code**

- Make sure that the leading character in the response message from the slave is a : after all data including the LF is received, and then convert the individual ASCII characters to binary code (4 bits). Arrange each 4-bit segment into MSB and LSB positions as shown in *Figure 6.8.*

- Here we need convert only up to the LSB of the LRC to binary code. You can check to make sure that CR and LF are present, but there is no need for this data.

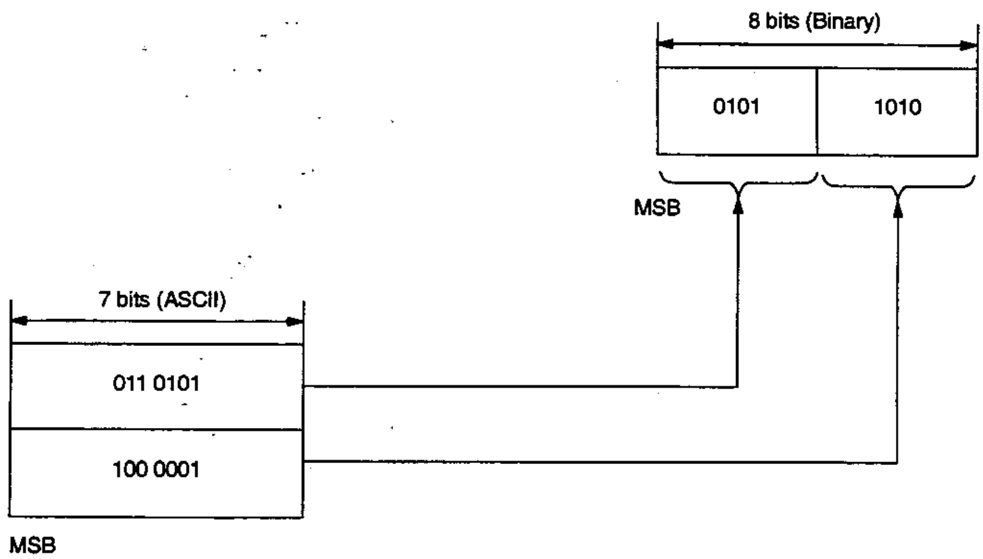



**Figure 6.8 Example of ASCII Character Conversion to Binary Code**

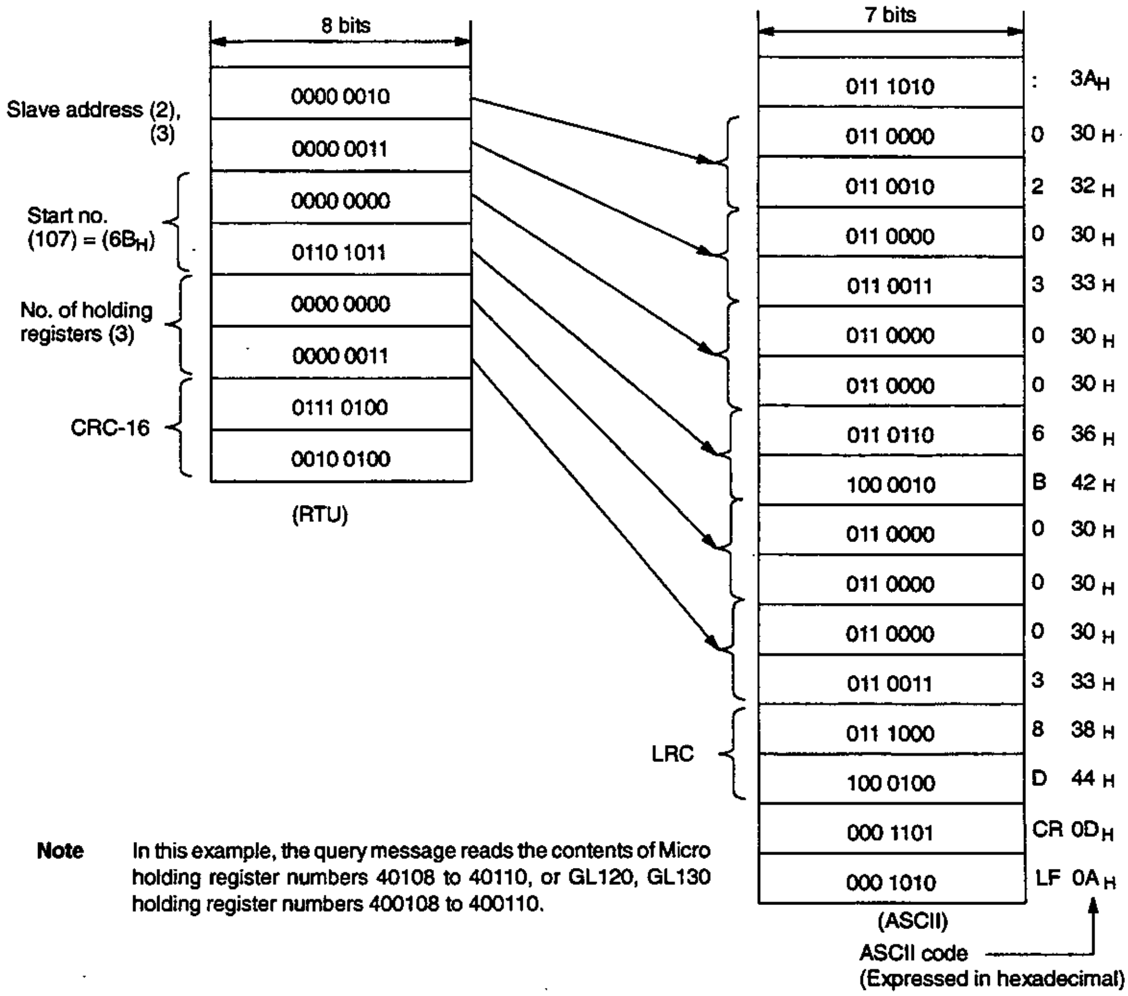
c) **Error Checking**

- Add all 8-bit data up to the LRC from the slave address (ignoring any overflow) to confirm that the results total 0.

## 6.6.4 Message Example

1) Since ASCII mode messages are exactly like RTU mode messages, refer to section *6.2 Basic RTU Mode Functions*, section *6.3 RTU Mode Expansion Functions,* and section *6.4 Special RTU Mode Functions* for further details on ASCII mode messages.

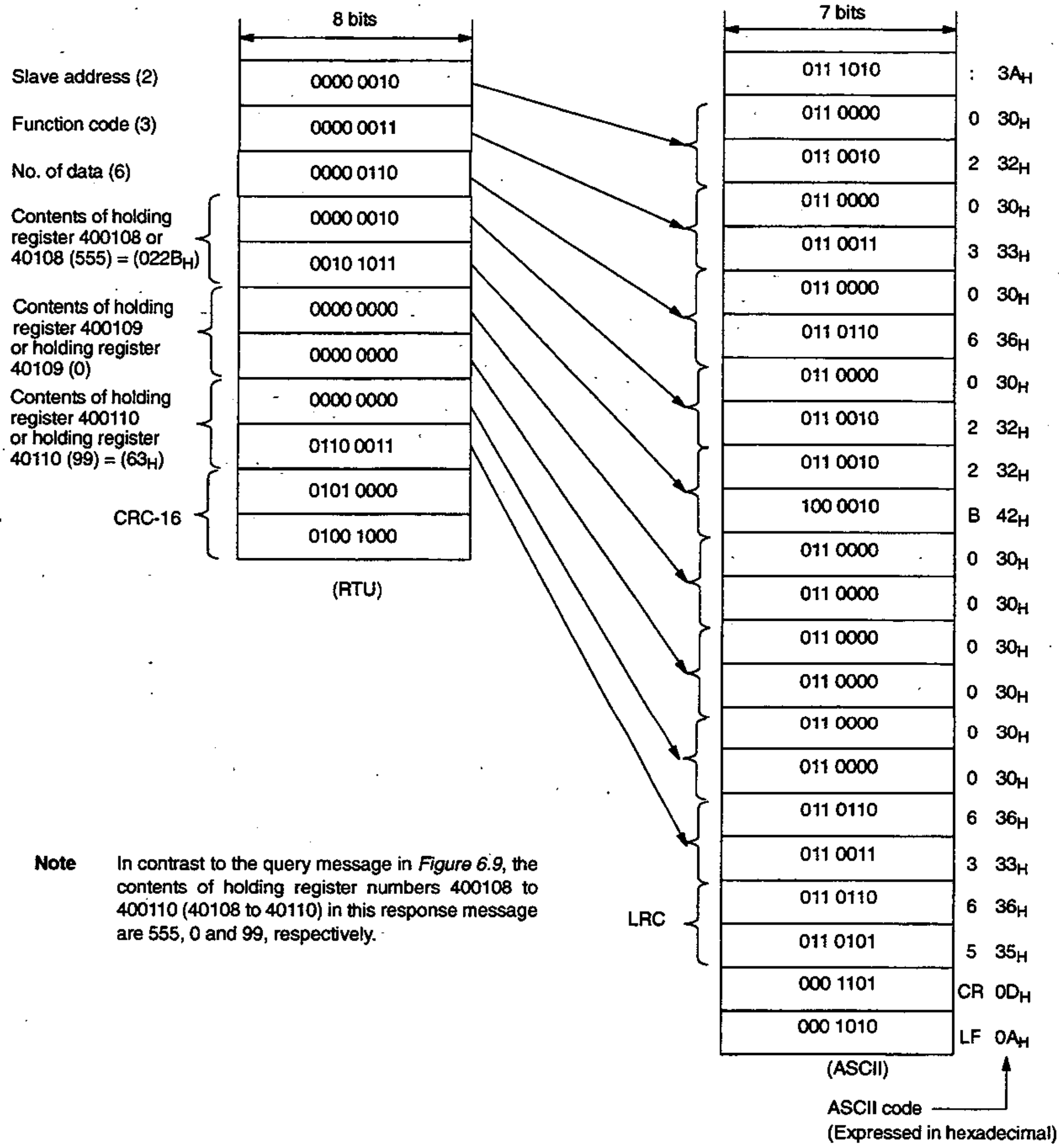
2) *Figure 6.9* shows the same query message generated in the ASCII mode that was generated in the RTU mode in section *6.2.7 Reading Holding Register Contents.* An example of the ASCII mode query message is shown in *Figure 6.10.*




**Note** In this example, the query message reads the contents of Micro holding register numbers 40108 to 40110, or GL120, GL130 holding register numbers 400108 to 400110.

**Figure 6.9 Example of a Query Message in ASCII Mode**

6




**Note** In contrast to the query message in *Figure 6.9*, the contents of holding register numbers 400108 to 400110 (40108 to 40110) in this response message are 555, 0 and 99, respectively.

**Figure 6.10 Example of a Response Message in ASCII Mode**

## 6.6.5 LRC Calculations

1) LRC (longitudinal redundancy check) refers to a double complement checking method that results from adding all data up to the last data from the slave address (ignoring any overflow). If the received data is normal, then the results of adding all data including the LRC will be 0.

2) LRC calculations are performed prior to binary to ASCII conversion, that is, it is performed in the same situation as the RTU mode (except for CRC-16). The :, CR, LF and other indicators are not included in the calculation.

3) *Figure 6.11* shows an example of an LRC7 calculation using the query message from the previous item. (See *Figure 6.9.*)

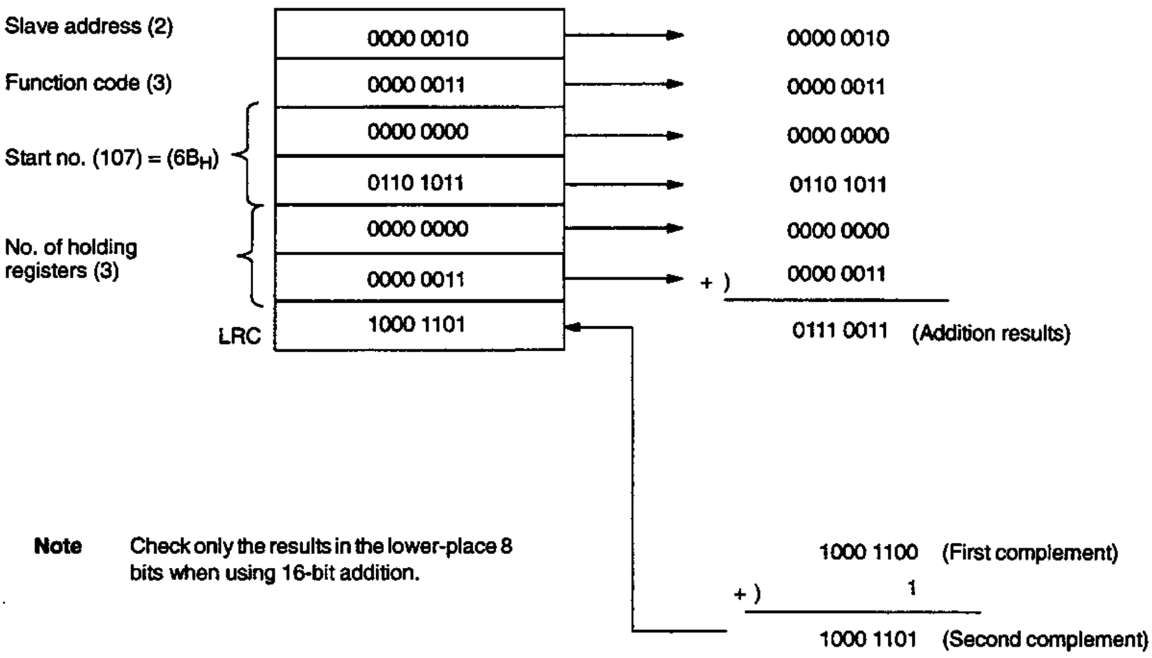



**Figure 6.11 Example of an LRC Calculation**

# Hardware Management

# 7

This chapter describes procedures for the J2078 Modem, Micros, and RS-232C cables that comprise the MEMOBUS system.


**7.1 Module .................................. 7-2**

7.1.1 J2078 Modem ................................ 7-2

7.1.2 CPU20 and CPU30 ............................ 7-7

7.1.3 Micro ...................................... 7-9

**7.2 RS-232C Cable .......................... 7-11**


7

# 7.1 Module

This section describes procedures for the J2078 Modem, CPU20, CPU30, and Micros, that comprise the MEMOBUS system.


7.1.1 J2078 Modem .................................................. 7-2
7.1.2 CPU20 and CPU30 .................................................. 7-7
7.1.3 Micro .................................................. 7-9


## 7.1.1 J2078 Modem

1) **Description**

*Figure 7.1* shows the external appearance of the J2078 Modem.

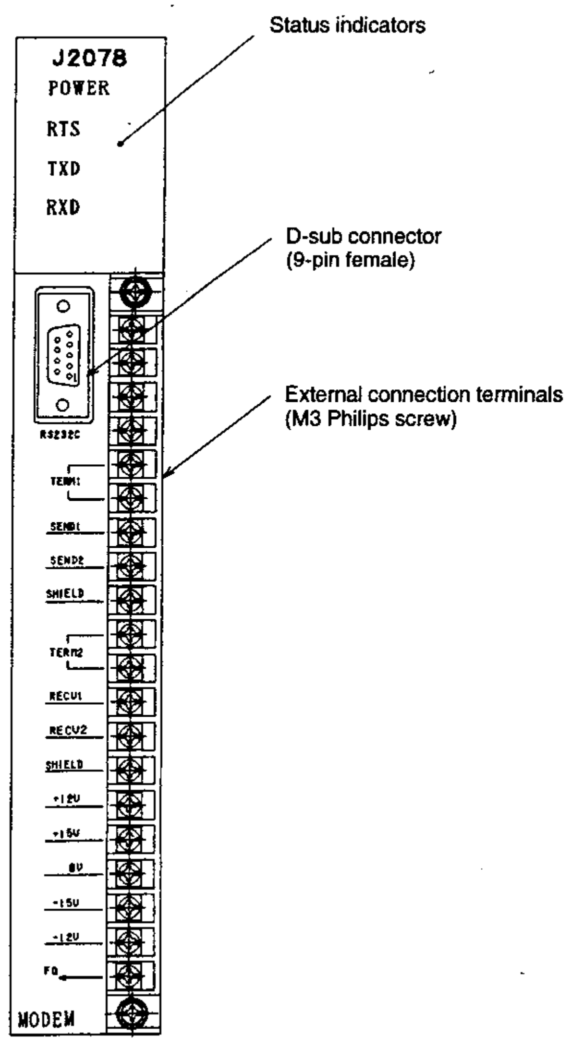



**Figure 7.1 External Appearance of the J2078**

The J2078 comprises a modulator that converts the RS-232C digital signal into 50 and 80 kHz analog signals and a demodulator that performs the reverse process. A block diagram of the J2078 is shown in *Figure 7.2*.

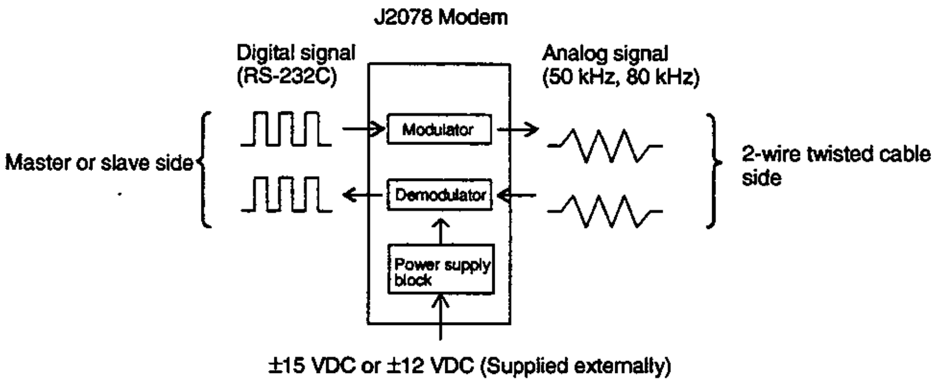



**Figure 7.2 Block Diagram of the J2078**

With the digital signal, logic 1 (–12 V) is a mark, and logic 0 (+12 V) is a space. With the analog signal, 1 corresponds to 50 kHz, while 0 corresponds to 80 kHz.

## 2) Mounting Method

The J2078 can be mounted in the slot on the 2000-series I/O Mounting Base for Yaskawa PLCs. It occupies a single slot. *Figure 7.3* shows the basic points of mounting the J2078 to the Mounting Base. Remove the connector cover attached to the J2078, align the guide posts on the J2078 to the guide holes on the Mounting Base, and then push the J2078 straight in. Secure the unit to the Mounting Base with the mounting screws (M4) on the J2078.

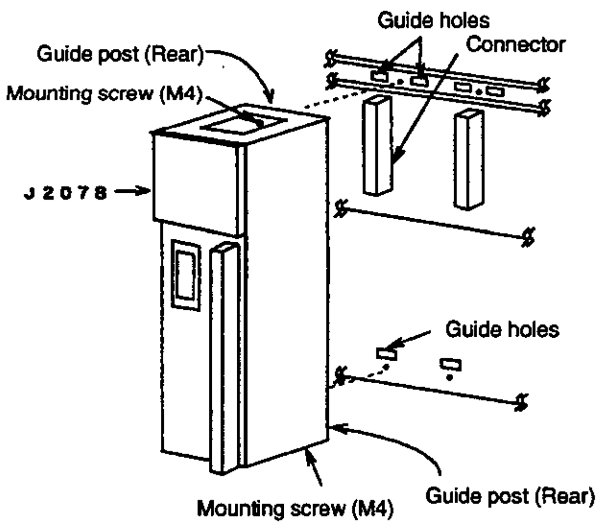



**Figure 7.3 Mounting the J2078 to the Mounting Base**

With an optional Mounting Bracket, the J2078 can be mounted to a control panel. The J2078 must be mounted vertically in this configuration as shown in *Figure 7.4*.

Contact your nearest Yaskawa representative directly for more information about this Mounting Bracket (model: DF8305869).

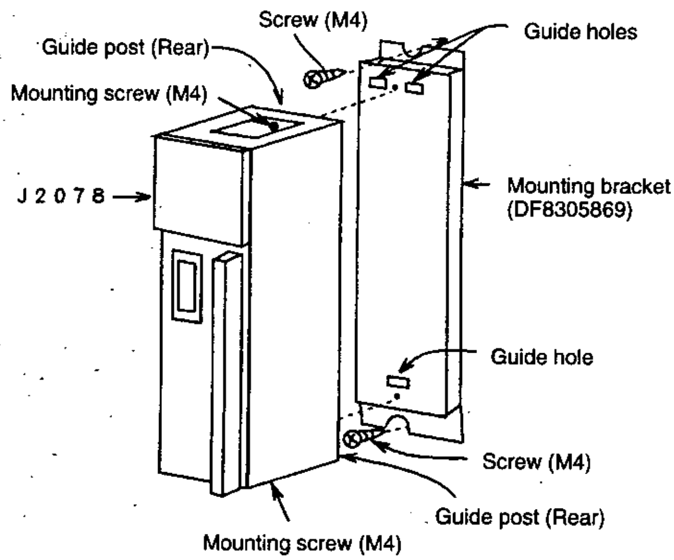



**Figure 7.4 Mounting the J2078 with a Mounting Bracket**

3) **Connecting Procedure**

A 9-pin D-sub connector and a 20-pin connector board (terminal board) are located on the front of the J2078 for external signal connections.

The D-sub connector is used for the RS-232C cable connecting the master to slaves. When connecting the RS-232C cable, make sure that the cable is securely attached with the M3 screw on the cable side of the D-sub connector (male).

Connect 2-core twisted-pair cable and external power supplies to the terminal board. *Figure 7.5* shows the connecting diagram for the terminal board. The diagram also shows the internal J2078 connections on the terminal board.

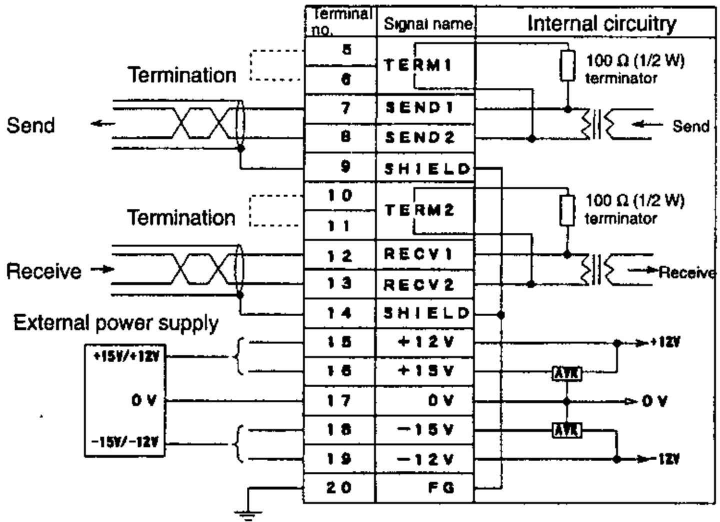



**Figure 7.5 J2078 Terminal Board Connections**

On the terminal board, terminals number 7 and 8 are transmission output terminals, while terminals number 12 and 13 are receive input terminals. Connect 2-core twisted-pair cable to these terminals. Since the 2-core twisted-pair cable is not polarized, there is no need to worry about polarity.

Terminals number 5 and 6 as well as 10 and 11 are used to connect the 2-core twisted-pair cable terminators. Terminators are required for the J2078 Modems at each end of a single run of the MEMOBUS system in order to terminate the 2-core twisted-pair cables. Here, use jumper wires to connect terminals number 5 and 6 as well as terminals number 10 and 11 on the J2078 terminal boards located at both ends. This will connect the terminator (100 Ω, 1/2 W) built into the J2078 to the 2-core twisted-pair cable.

Do not connect anything between terminals number 5 and 6 or terminals number 10 and 11 on J2078 terminal boards anywhere between the two ends. Terminals number 9 and 14 are for grounding the shielded covering of the 2-core twisted-pair cable. The shielded covering of the 2-core twisted-pair cable must be grounded in order to block external noise interference. Therefore, connect terminal no. 9 or 14 to the shielded covering on one end of the 2-core twisted-pair cable. Also connect terminal no. 20 to a ground with a resistance no greater than 100 Ω. Terminals number 9, 14 and 20 are connected internally, so any one of these can be used to ground one end of the shielded covering. The basic idea for grounding the shielded covering is explained later in chapter *8 Wiring*.

7

Terminals number 15 to 19 are used to connect external power supplies. The J2078 requires a ±15 VDC or a ±12 VDC external power supply. With a ±15 VDC power supply, connect the plus side to terminal no. 16, and the minus side to terminal no. 18. With a ±12 VDC power supply, connect the plus side to terminal no. 15, and the minus side to terminal no. 19. In either case, connect 0 V to terminal no. 17. Power supply requirements are given below:

+15 V, +12 V: 200 mA
–15 V, –12 V: 100 mA

Use an M3 screw to secure the crimp terminal to the terminal board for all wiring as shown in *Figure 7.6.* Where two, 2-core twisted-pair cables are connected to J2078 Modems, (i.e., other than at the ones at the ends) overlap the crimp terminals for the two pairs and secure the overlapped terminals with the screw.

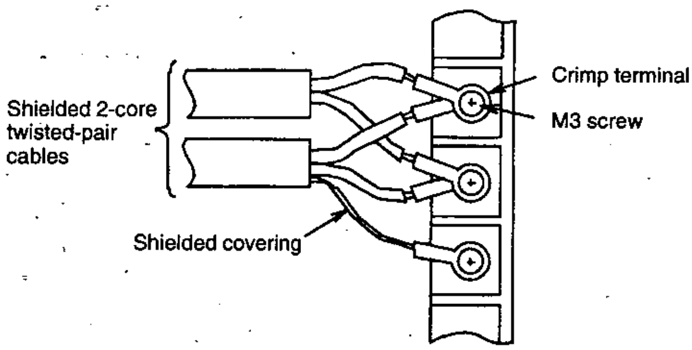



**Figure 7.6 Wiring to the J2078 Terminal Board**

4) **Indicators**

The front panel of the J2078 has four indicators, POWER, RTS, TXD and RXD. The meaning of each indicator is explained in *Table 7.1.*

**Table 7.1 J2078 Indicators**

| Indicator name | Meaning |
|---|---|
| POWER | Lit when external power supply (±15 V or ±12 V) is on. |
| RTS | Request to Send<br>Lit when the RTS signal from a master or slave is a logic 1. |
| TXD | Sending<br>Indicates that data is being sent from the send output terminal. Lights with a logic 1. |
| RXD | Receiving<br>Indicates that data is being received from the receive input terminal. Lights with a logic 1. |

## 7.1.2 CPU20 and CPU30

### 1) Description

The CPU20 and CPU30 CPU Modules for the GL120 and GL130 are each equipped with an RS-232C port so that the units can be used as MEMOBUS slaves. *Figure 7.7* shows the location of the MEMOBUS port on CPU20.

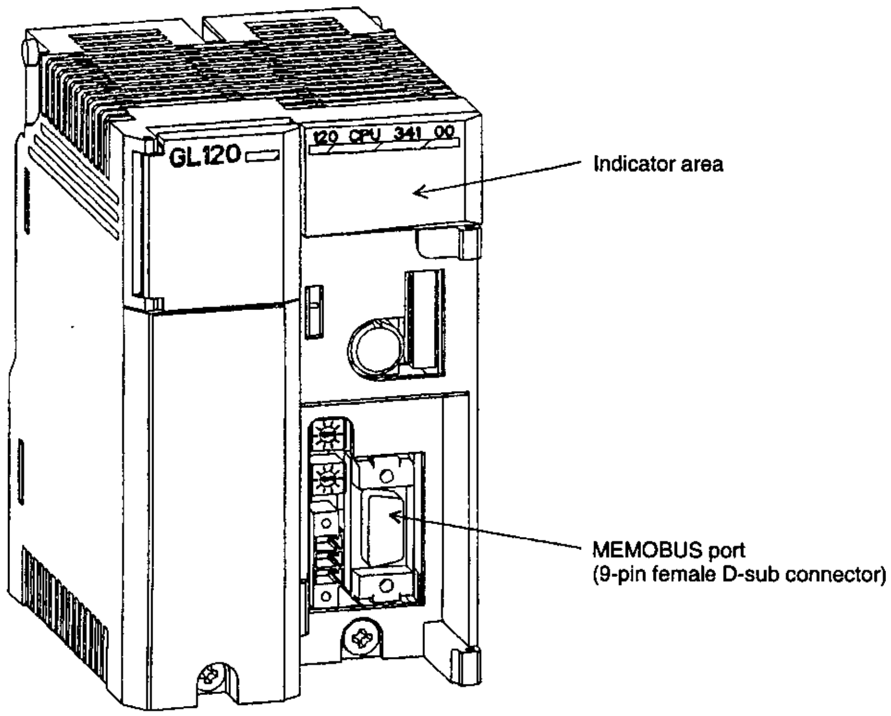


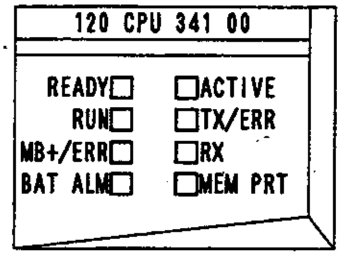

**Figure 7.7 MEMOBUS Port on CPU20**

### 2) Connecting Procedure

The MEMOBUS port connectors on CPU20 and CPU30 are D-sub connectors (9-pin female). When connecting an RS-232C cable to the MEMOBUS port, make sure that the D-sub connector (9-pin male) on the cable is pushed securely into the D-sub connector on the MEMOBUS port side. Then use a Philips screwdriver to tighten the two screws on the back of the D-sub connector (9-pin male) on the cable.

4) **Indicators**

The indicator area on the front panel of the CPU20 and CPU30 has two MEMOBUS indicators. *Figure 7.8* lists the indicators and their meanings.




| Indicator Name | Color | Indicator Meaning |
|---|---|---|
| READY | Green | CPU Module is normal. |
| RUN | Green | CPU Module is running. |
| MB+/ERR | Green | The MEMOBUS PLUS port is sending and receiving data normally. |
| | Red | An error was generated while the MEMOBUS PLUS port was sending or receiving. |
| BAT ALM | Red | Low battery in the CPU Module. |
| ACTIVE | Green | The CPU Module is ready to access another Module. |
| TX/ERR | Green | The MEMOBUS port is sending data normally. |
| | Red | An error was generated while the MEMOBUS port was sending. |
| RX | Green | The MEMOBUS port is receiving data normally. |
| MEM PRT | Green | Memory protect ON is selected by the key switch. |

**Figure 7.8 Indicator Area and Indicator Meanings for CPU20**

## 7.1.3 Micro

1) **Description**

Micros are available in models with one or two RS-232C ports that can be used for MEMOBUS.

| Model | No. of RS-232C ports |
|---|---|
| CPU311, CPU411 | 1 |
| CPU512, CPU612 | 2 |

*Figure 7.9* shows the MEMOBUS ports on the Micro.

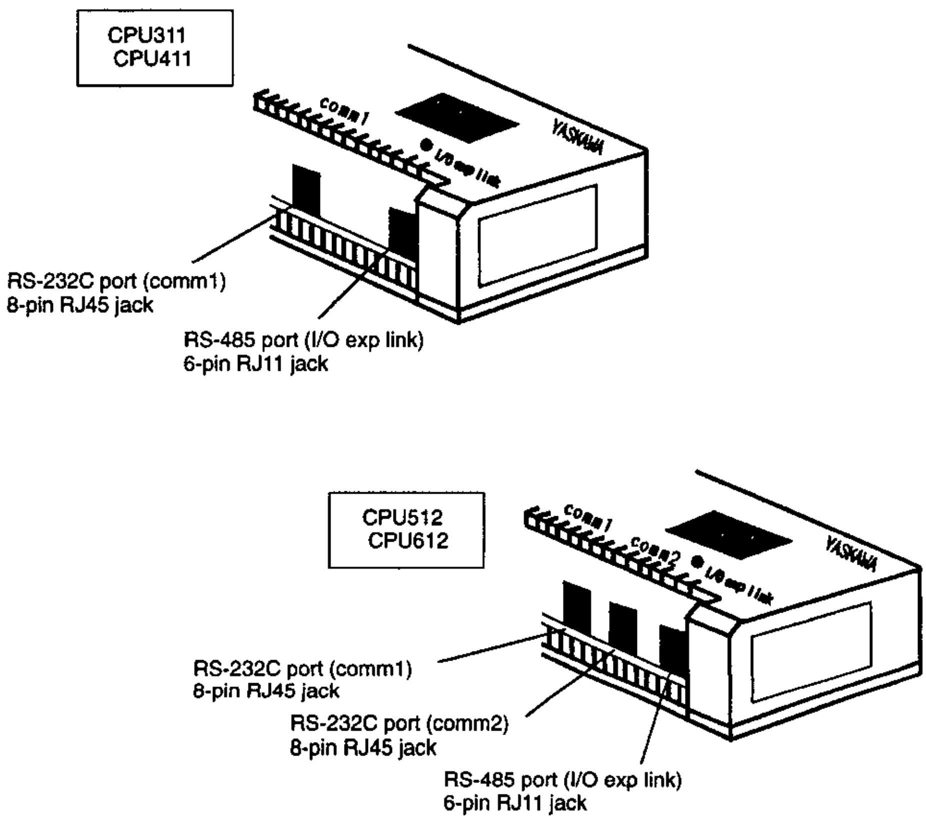



**Figure 7.9 MEMOBUS Ports on the Micro**

2) **Connecting Procedure**

The RS-232C port connector on the Micro is an 8-pin RJ45 jack. When connecting an RS-232C cable to the Micro, be sure to grip the RJ45 plug on the cable securely, and push the plug until you hear a click. To remove the cable, push down on the tab on the top of the RJ45 plug to release the stopper, and pull the plug out toward you.

### 4) Indicators

The status indicator panel on the front of the Micro has two MEMOBUS indicators. *Figure 7.10* shows the status indicator panel on the Micro.

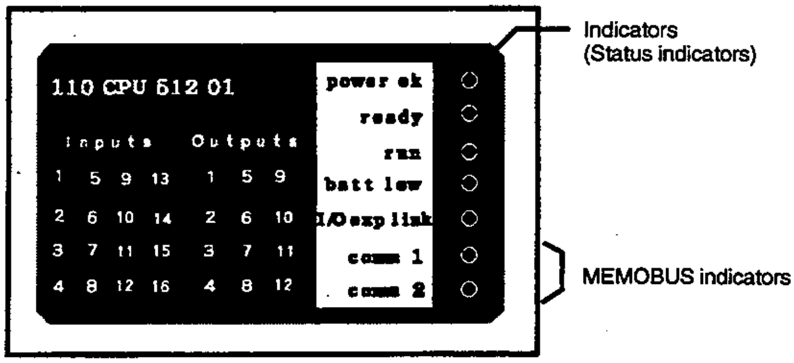



**Figure 7.10 Status Indicator Panel on the Micro**

The comm1 indicator flashes when port 1 is communicating, and the comm2 indicator flashes when port 2 is communicating. CPU311 and CPU411 do not have a comm2 indicator on the status indicator panel.

# 7.2 RS-232C Cable

1) **Description**

RS-232C cables are used to connect masters to slaves, masters to Modems, and slaves to Modems. *Table 7.2* summarizes the RS-232C cables available from Yaskawa.

**Table 7.2 Summary of RS-232C Cables**

| Model | Length | Application |
|---|---|---|
| JZMSZ-120W0200-03<br>JZMSZ-120W0200-05 | 2.5 m<br>15 m | Connects the GL120 and GL130 to personal computer made by NEC (PC-98) (25-pin D-sub connector) |
| JZMSZ-120W0201-03<br>JZMSZ-120W0201-05 | 2.5 m<br>15 m | Connects the GL120 and GL130 to personal computer made by NEC (PC-98) (Half-pitch connector) |
| JZMSZ-120W0202-03<br>JZMSZ-120W0202-05 | 2.5 m<br>15 m | Connects the GL120 and GL130 to DOS personal computers (9-pin D-sub connector) |
| JZMSZ-120W0203-03<br>JZMSZ-120W0203-15 | 2.5 m<br>15 m | Connects the GL120 and GL130 to the P120 programming panel |
| JZMSZ-120W0204-05<br>JZMSZ-120W0204-10<br>JZMSZ-120W0204-15 | 5 m<br>10 m<br>15 m | Connects the GL120 and GL130 to the ACGC4000 Series |
| JZMSZ-120W0205-01<br>JZMSZ-120W0205-03<br>JZMSZ-120W0205-05 | 1 m<br>3 m<br>5 m | Connects the GL120 and GL130 to the J2078 Modem |
| JZMSZ-120W0206-01<br>JZMSZ-120W0206-03<br>JZMSZ-120W0206-05 | 1 m<br>3 m<br>5 m | Connects the GL120 and GL130 to the AX/2400C Modem |
| JZMSZ-120W0207-03<br>JZMSZ-120W0207-15 | 2.5 m<br>15 m | Connects the GL120 and GL130 to the ACGC4200 Series or MEMOCON (GL60 Series) |
| JZMSZ-120W0208-03<br>JZMSZ-120W0208-15 | 2.5 m<br>5 m | Connects between GL120 and GL130 |
| JZMSZ-120W0240-01<br>JZMSZ-120W0240-03<br>JZMSZ-120W0240-05 | 1 m<br>3 m<br>5 m | Connects the J2078 Modem to personal computer made by NEC (PC-98) (25-pin D-sub connector) |
| JZMSZ-120W0241-01<br>JZMSZ-120W0241-03<br>JZMSZ-120W0241-05 | 1 m<br>3 m<br>5 m | Connects the J2078 Modem to DOS personal computers (9-pin D-sub connector) |
| JZMSZ-110XCA21001<br>JZMSZ-110XCA21002<br>JZMSZ-110XCA21003 | 1 m<br>3 m<br>6 m | Connects the Micro to the inverter |
| JZMSZ-110XCA21301<br>JZMSZ-110XCA21302<br>JZMSZ-110XCA21303 | 1 m<br>3 m<br>6 m | Connects the Micro to the MEMOCON (GL70H, GL60H, GL60S, GL40S Series) |
| JZMSZ-110XCA21311<br>JZMSZ-110XCA21312<br>JZMSZ-110XCA21313 | 1 m<br>3 m<br>6 m | Connects the Micro to DOS personal computers (9-pin D-sub connector) |
| JZMSZ-110XCA21321<br>JZMSZ-110XCA21322<br>JZMSZ-110XCA21323 | 1 m<br>3 m<br>6 m | Connects the Micro to the J2078 Modem |
| JZMSZ-110XCA21341<br>JZMSZ-110XCA21342<br>JZMSZ-110XCA21343 | 1 m<br>3 m<br>6 m | Connects the Micro to the GL120 and GL130 |
| JZMSZ-110XCA21401<br>JZMSZ-110XCA21402<br>JZMSZ-110XCA21403 | 1 m<br>3 m<br>6 m | Connects the Micro to personal computer made by NEC (PC-98) (25-pin D-sub connector) |

| Model | Length | Application |
|---|---|---|
| JZMSZ-110XCA21411<br>JZMSZ-110XCA21412<br>JZMSZ-110XCA21413 | 1 m<br>3 m<br>6 m | Connects the Micro to the ACGC4000 Series |
| JZMSZ-110XCA21421<br>JZMSZ-110XCA21422<br>JZMSZ-110XCA21423 | 1 m<br>3 m<br>6 m | Connects the Micro to the J1078 Modem |
| JZMSZ-110XCA21601<br>JZMSZ-110XCA21602<br>JZMSZ-110XCA21603 | 1 m<br>3 m<br>6 m | Connects the Micro to personal computer made by NEC (PC-98) (Half-pitch connector) |

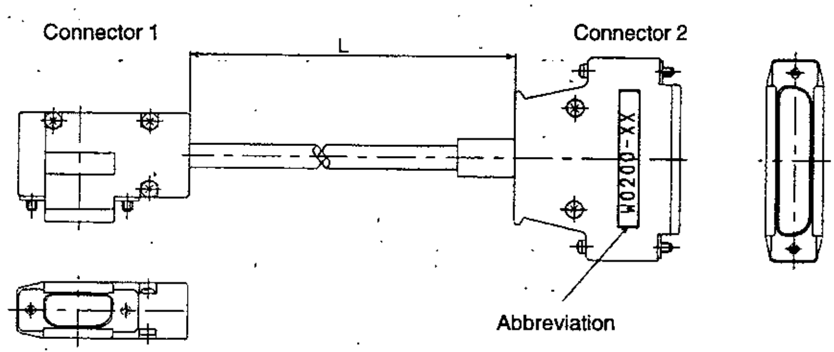
2) Specification, appearance, and connection diagram of the MEMOBUS cables are shown below.

a) W0200 Cables

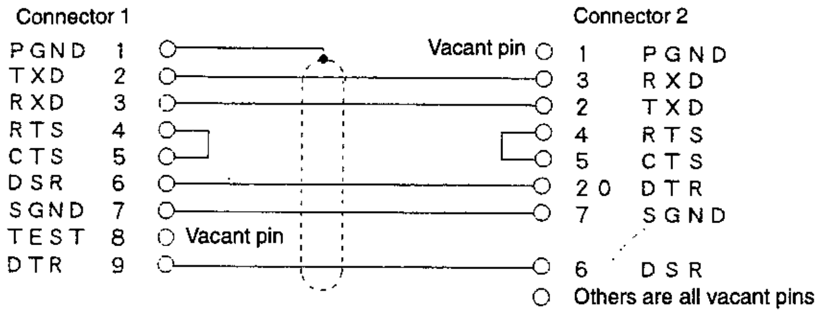
(1) Specifications

| Item | Specifications | |
|---|---|---|
| Abbreviation | W0200-03 | W0200-15 |
| Type | JZMSZ-120W0200-03 | JZMSZ-120W0200-15 |
| Length (L) | 2.5 m | 15.0 m |
| Cable Specifications | Shielded cable, 9 cores, equivalent to UL2921, AWG26 (0.13 mm$^2$) | |
| Connector Specifications | Connector 1 : 17JE-23090-02 (D90B)<br>Connector 2 : 17JE-43250-02 (D8A) (DDK) | |

(2) Appearance



(3) Connection Diagram



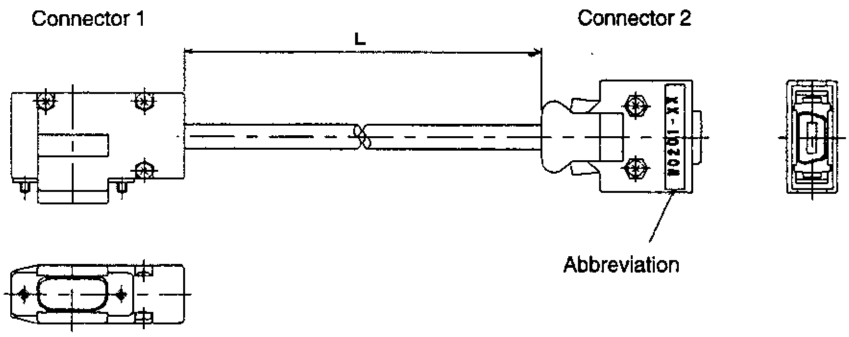

Cable size : AWG26 (0.13 mm$^2$)

b) W0201 Cables

(1) Specifications

| Item | Specifications | |
|---|---|---|
| Abbreviation | W0201-03 | W0201-15 |
| Type | JZMSZ-120W0201-03 | JZMSZ-120W0201-15 |
| Length (L) | 2.5 m | 15.0 m |
| Cable Specifications | Shielded cable, 9 cores, equivalent to UL2921, AWG26 (0.13 mm$^2$) | |
| Connector Specifications | 17JE-23090-02 (D90B) (DDK) + Plug : 10114-3000VE (3M), Connector shell : 10314-42F0-008 (3M) | |

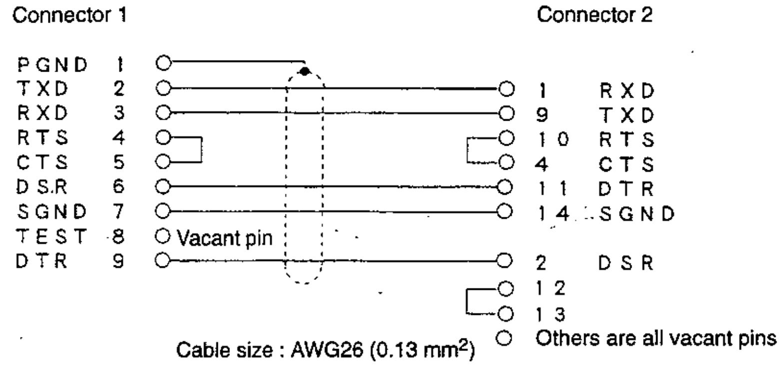
(2) Appearance






7

(3) Connection Diagram

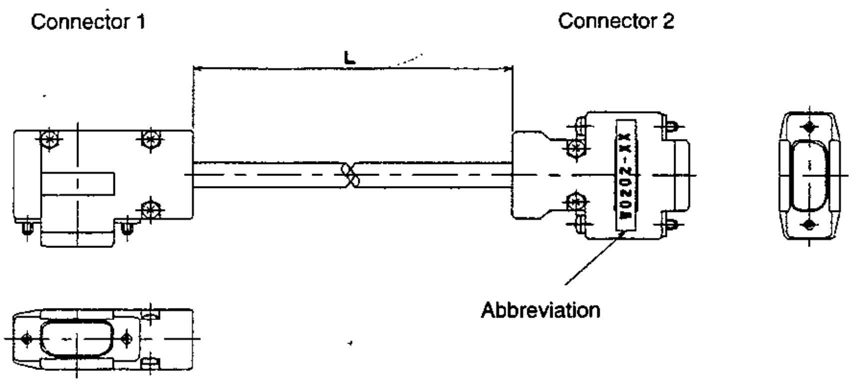



c) W0202 Cables

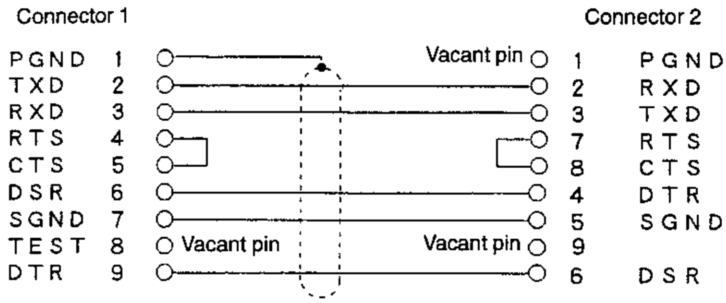
(1) Specifications

| Item | Specifications | |
|---|---|---|
| Abbreviation | W0202-03 | W0202-15 |
| Type | JZMSZ-120W0202-03 | JZMSZ-120W0202-15 |
| Length (L) | 2.5 m | 15.0 m |
| Cable Specifications | Shielded cable, 9 cores, equivalent to UL2921, AWG26 (0.13 mm$^2$) | |
| Connector Specifications | Connector 1 : 17JE-23090-02 (D90B)<br>Connector 2 : 17JE-13090-02 (D8C) (DDK) | |

(2) Appearance



(3) Connection Diagram



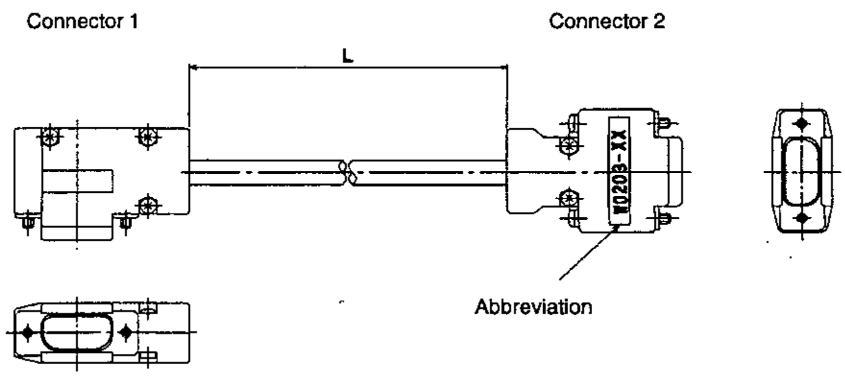

Cable size : AWG26 (0.13 mm$^2$)

d) W0203 Cables

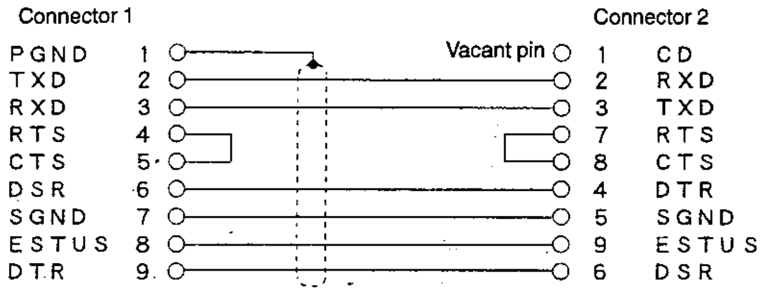
(1) Specifications

| Item | Specifications | |
|---|---|---|
| Abbreviation | W0203-03 | W0203-15 |
| Type | JZMSZ-120W0203-03 | JZMSZ-120W0203-15 |
| Length (L) | 2.5 m | 15.0 m |
| Cable Specifications | Shielded cable, 9 cores, equivalent to UL2921, AWG26 (0.13 mm$^2$) | |
| Connector Specifications | Connector 1 : 17JE-23090-02 (D90B)<br>Connector 2 : 17JE-13090-02 (D8C) (DDK) | |

(2) Appearance



(3) Connection Diagram



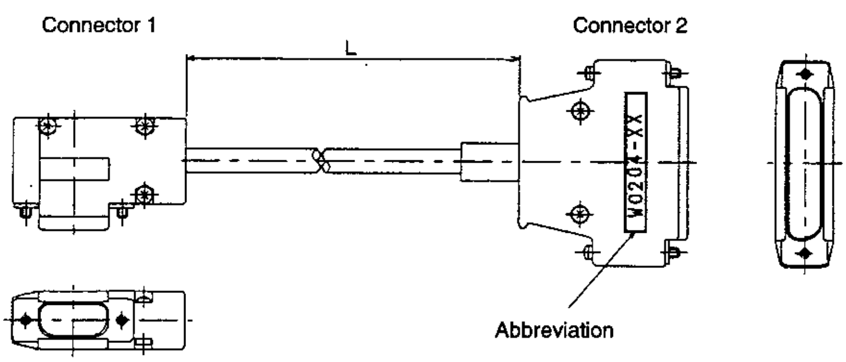

Cable size : AWG26 (0.13 mm²)

e) W0204 Cables

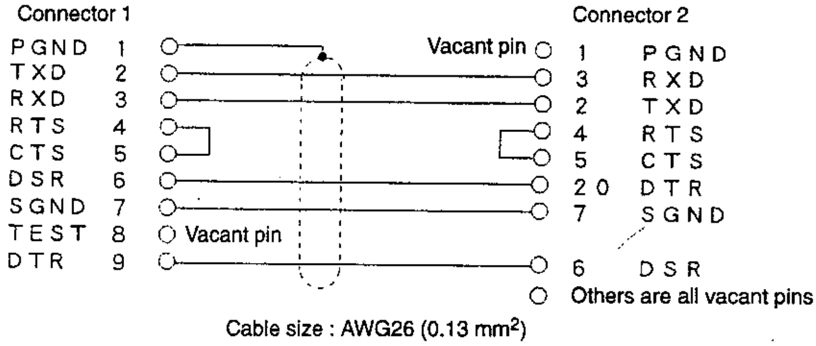
(1) Specifications

| Item | Specifications | | |
|---|---|---|---|
| Abbreviation | W0204-05 | W0204-10 | W0204-15 |
| Type | JZMSZ-120W0204-05 | JZMSZ-120W0204-10 | JZMSZ-120W0204-15 |
| Length (L) | 5.0 m | 10.0 m | 15.0 m |
| Cable Specifications | Shielded cable, 9 cores, equivalent to UL2921, AWG26 (0.13 mm²) | | |
| Connector Specifications | Connector 1 : 17JE-23090-02 (D90B)<br>Connector 2 : 17JE-33250-02 (D8B) (DDK) | | |

(2) Appearance



(3) Connection Diagram

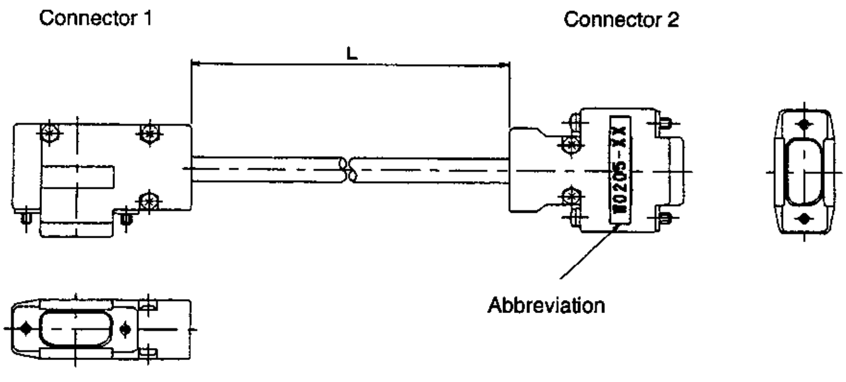



f) W0205 Cables

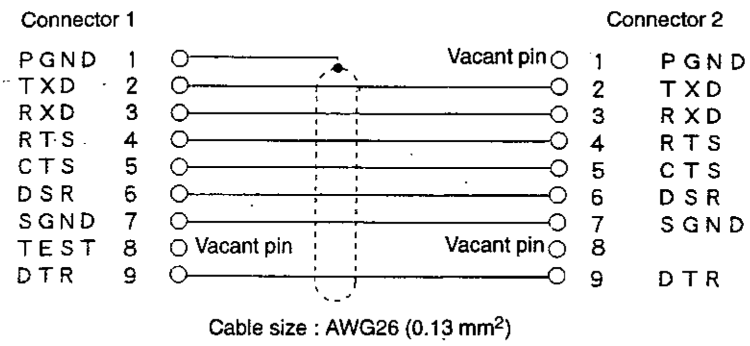
(1) Specifications

| Item | Specifications | | |
|---|---|---|---|
| Abbreviation | W0205-01 | W0205-03 | W0205-05 |
| Type | JZMSZ-120W0205-01 | JZMSZ-120W0205-03 | JZMSZ-120W0205-05 |
| Length (L) | 1.0 m | 3.0 m | 5.0 m |
| Cable Specifications | Shielded cable, 9 cores, equivalent to UL2921, AWG26 (0.13 $mm^2$) | | |
| Connector Specifications | Connector 1 : 17JE-23090-02 (D90B)<br>Connector 2 : 17JE-23090-02 (D8B) (DDK) | | |

(2) Appearance



(3) Connection Diagram

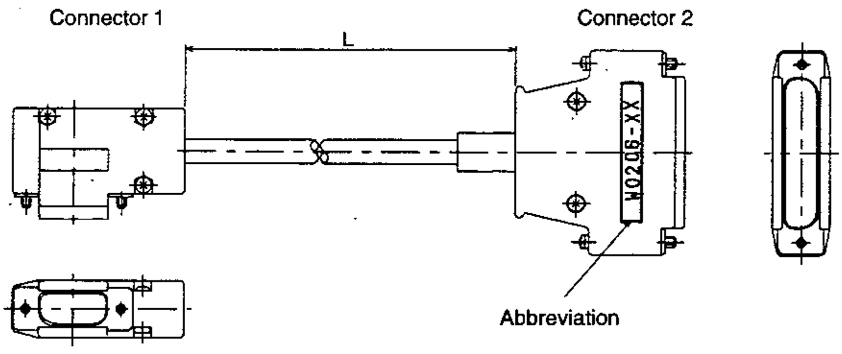



g) W0206 Cables

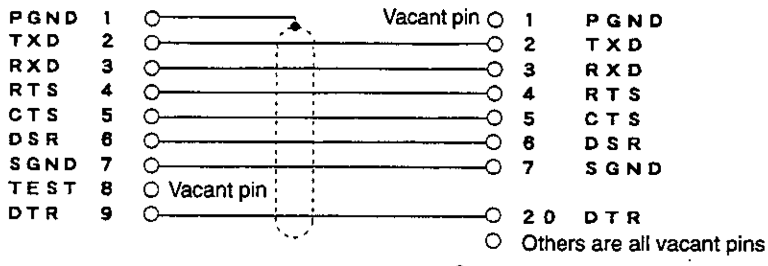
(1) Specifications

| Item | Specifications | | |
|---|---|---|---|
| Abbreviation | W0206-01 | W0206-03 | W0206-05 |
| Type | JZMSZ-120W0206-01 | JZMSZ-120W0206-03 | JZMSZ-120W0206-05 |
| Length (L) | 1.0 m | 3.0 m | 5.0 m |
| Cable Specifications | Shielded cable, 9 cores, equivalent to UL2921, AWG26 (0.13 mm$^2$) | | |
| Connector Specifications | Connector 1 : 17JE-23090-02 (D90B)<br>Connector 2 : 17JE-43250-02 (D8C) (DDK) | | |

(2) Appearance



(3) Connection Diagram



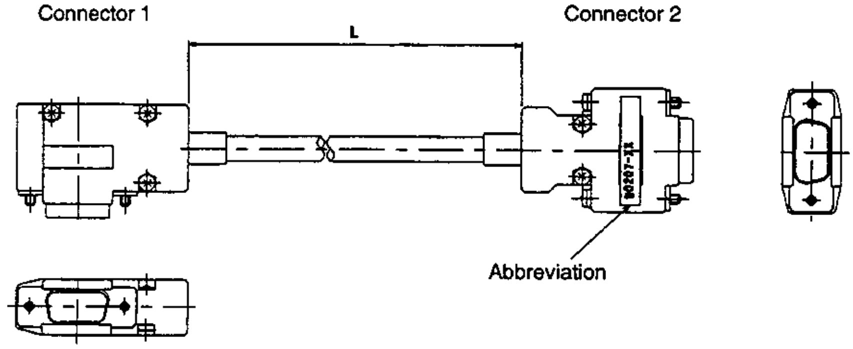

Cable size : AWG26 (0.13 mm²)

h) W0207 Cables

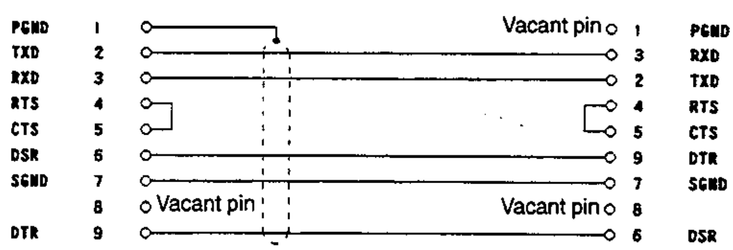
(1) Specifications

| Item | Specifications | |
|---|---|---|
| Abbreviation | W0207-03 | W0207-15 |
| Type | JZMSZ-120W0207-03 | JZMSZ-120W0207-15 |
| Length (L) | 2.5 m | 15.0 m |
| Cable Specifications | Shielded cable, 9 cores, equivalent to UL2921, AWG26 (0.13 mm²) | |
| Connector Specifications | Connector 1 : 17JE-23090-02 (D90B)<br>Connector 2 : 17JE-23090-02 (D8B) (DDK) | |

(2) Appearance




(3) Connection Diagram



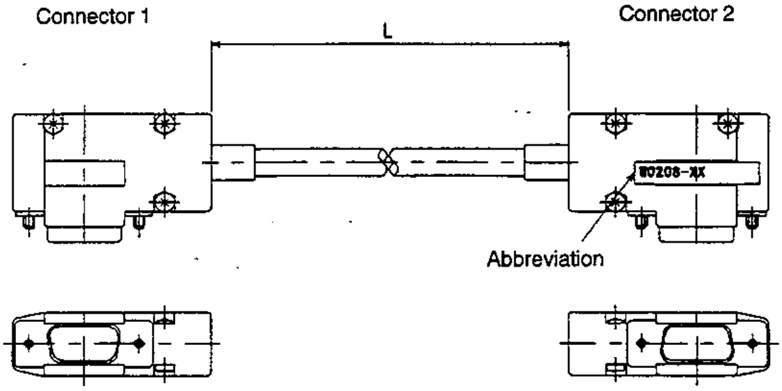

Cable size : AWG26 (0.13 mm²)

i) W0208 Cables

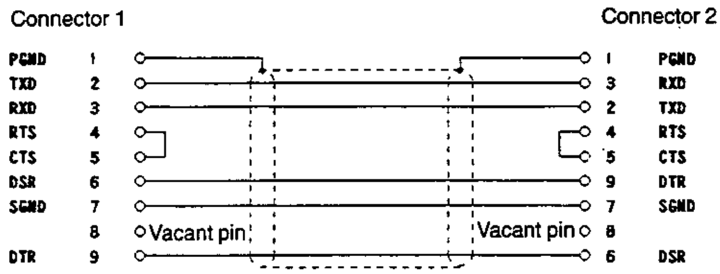
(1) Specifications

| Item | Specifications | |
|---|---|---|
| Abbreviation | W0208-03 | W0208-15 |
| Type | JZMSZ-120W0208-03 | JZMSZ-120W0208-15 |
| Length (L) | 2.5 m | 15.0 m |
| Cable Specifications | Shielded cable, 9 cores, equivalent to UL2921, AWG26 (0.13 mm$^2$) | |
| Connector Specifications | Connector 1 : 17JE-23090-02 (D90B)<br>Connector 2 : 17JE-23090-02 (D90B) (DDK) | |

(2) Appearance




(3) Connection Diagram



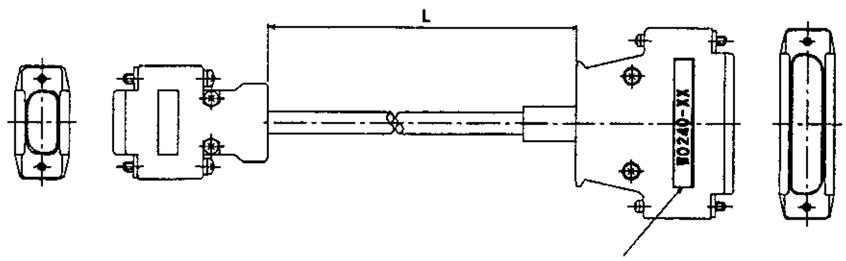

Cable size : AWG26 (0.13 mm$^2$)

j) W0240 Cables

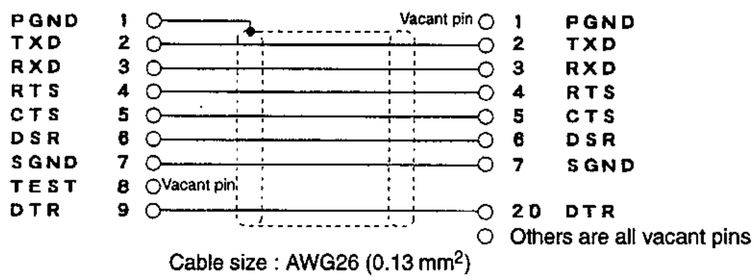
(1) Specifications

| Item | Specifications | | |
|---|---|---|---|
| Abbreviation | W0240-01 | W0240-03 | W0240-05 |
| Type | JZMSZ-120W0240-01 | JZMSZ-120W0240-03 | JZMSZ-120W0240-05 |
| Length (L) | 1.0 m | 3.0 m | 5.0 m |
| Cable Specifications | Shielded cable, 9 cores, equivalent to UL2921, AWG26 (0.13 mm$^2$) | | |
| Connector Specifications | Connector 1 : 17JE-23090-02 (D8B)<br>Connector 2 : 17JE-43250-02 (D8A) (DDK) | | |

(2) Appearance




(3) Connection Diagram

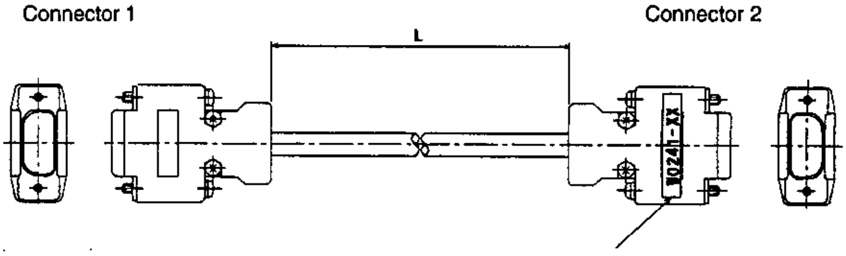



k) W0241 Cables

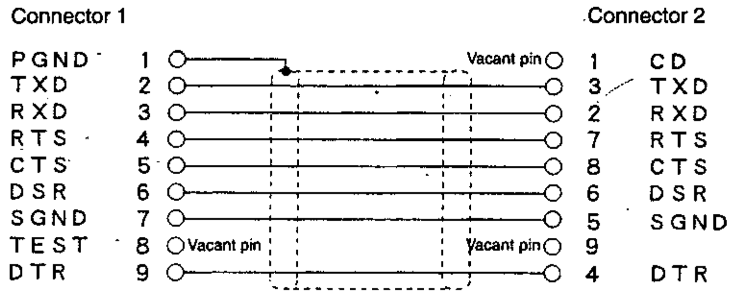
(1) Specifications

| Item | Specifications | | |
|---|---|---|---|
| Abbreviation | W0241-01 | W0241-03 | W0241-05 |
| Type | JZMSZ-120W0241-01 | JZMSZ-120W0241-03 | JZMSZ-120W0241-05 |
| Length (L) | 1.0 m | 3.0 m | 5.0 m |
| Cable Specifications | Shielded cable, 9 cores, equivalent to UL2921, AWG26 (0.13 $mm^2$) | | |
| Connector Specifications | Connector 1 : 17JE-23090-02 (D8B)<br>Connector 2 : 17JE-13090-02 (D8C) (DDK) | | |

(2) Appearance



(3) Connection Diagram

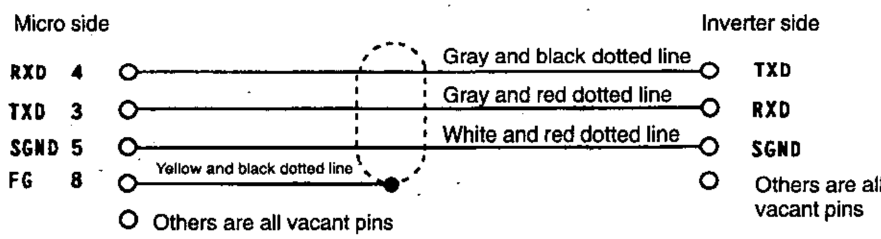



Cable size : AWG26 (0.13 $mm^2$)

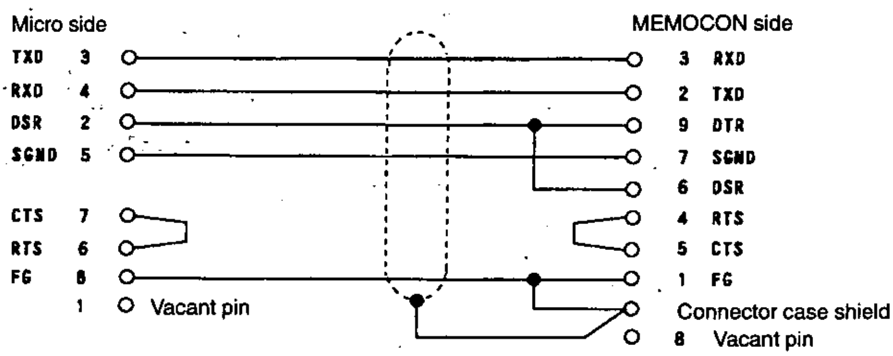
## 3) Connecting Procedure

The internal connections for the Micro related RS-232C cables listed in *Table 7.2* are given below.

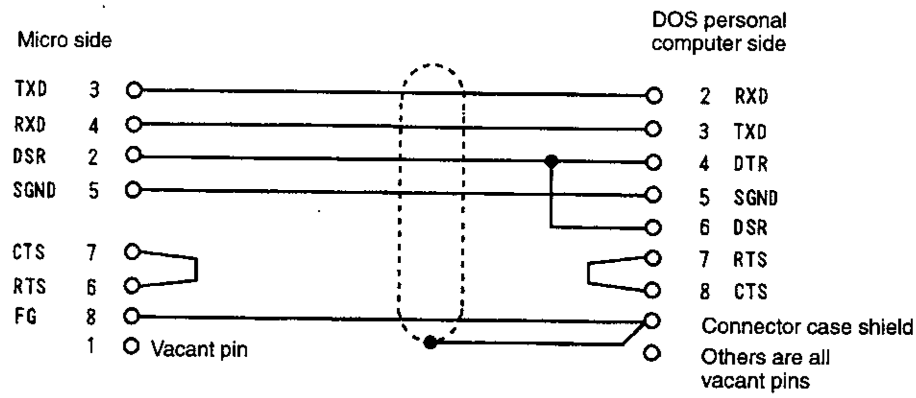
- **JZMSZ-110XCA2100□□**



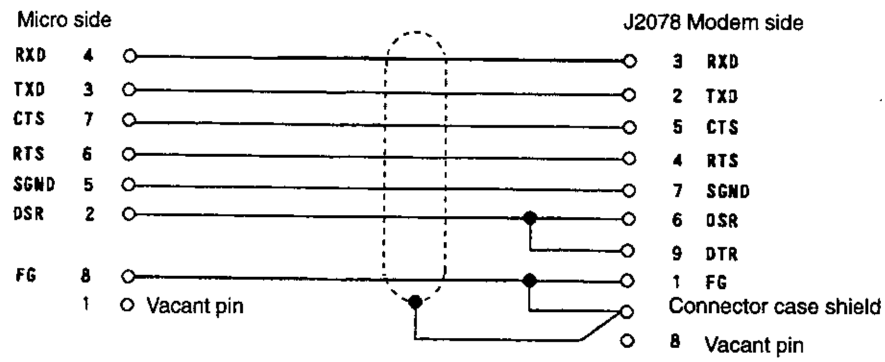

- **JZMSZ-110XCA2130□**



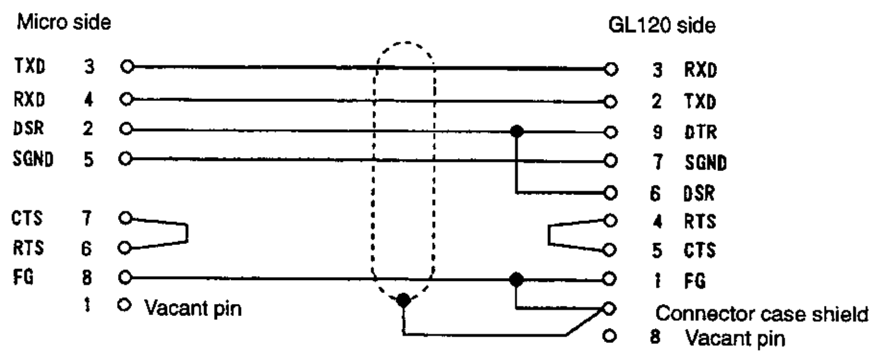
• **JZMSZ-110XCA2131□**




• **JZMSZ-110XCA2132□**

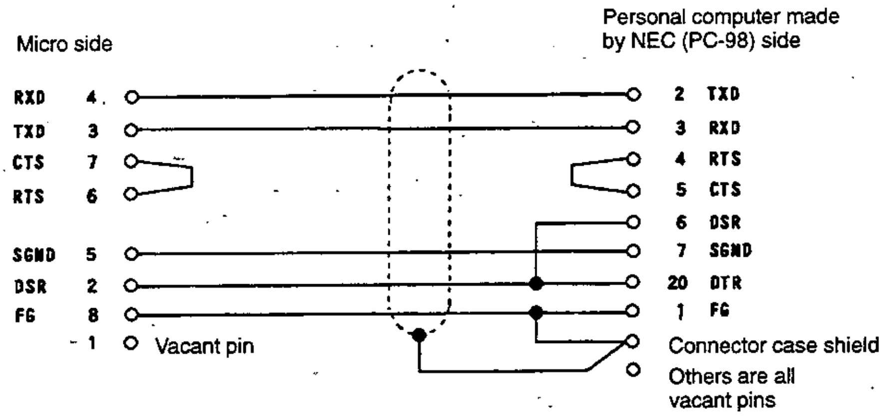



• **JZMSZ-110XCA2134□**

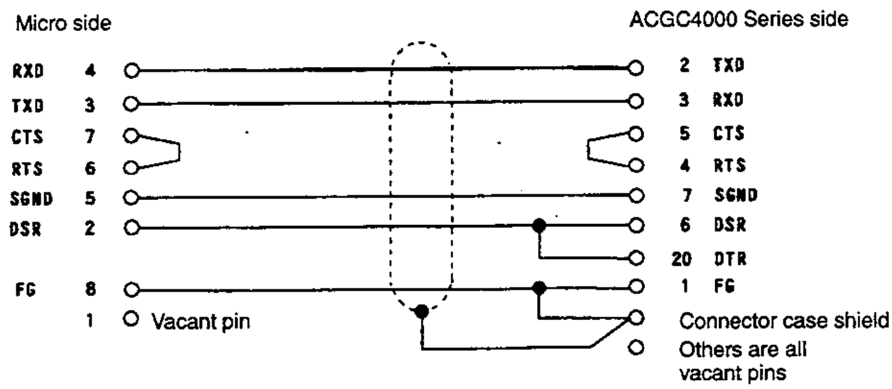



7

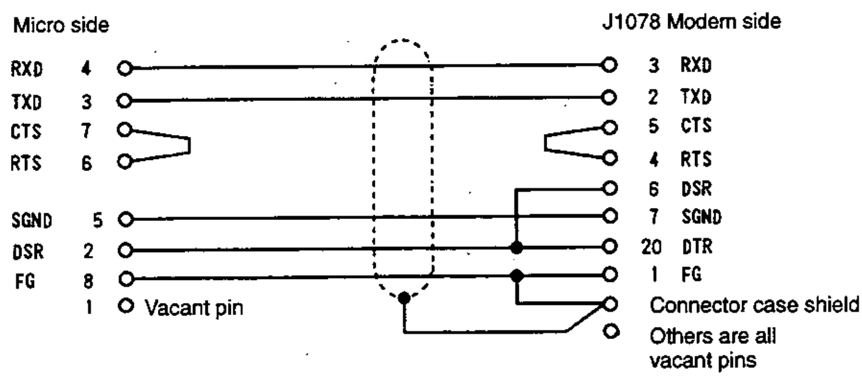
- **JZMSZ-110XCA2140□**



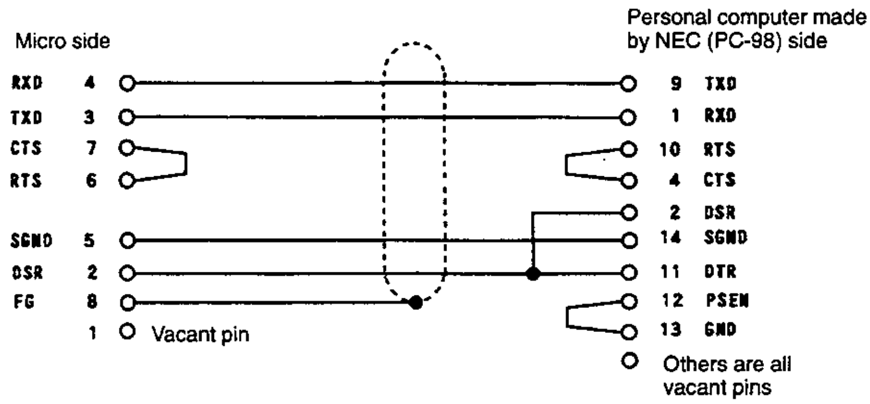

- **JZMSZ-110XCA2141□**




- **JZMSZ-110XCA2142□**



• **JZMSZ-110XCA2160□**



# Wiring

# 8

This chapter describes wiring procedures for RS-232C cables and 2-core twisted cables used to connect individual equipment in the MEMOBUS system.


**8.1 Wiring 2-core Twisted Cable on a Panel ..... 8-2**

**8.2 Wiring 2-core Twisted Cable off a Panel ..... 8-2**

**8.3 Grounding 2-core Twisted Cable ........... 8-3**

**8.4 RS-232C Cable .......................... 8-3**

## 8.1 Wiring 2-core Twisted Cable on a Panel

Do not lay the 2-core twisted cable in the same duct as power or control lines in in-panel wiring. Either use a separate duct or run the 2-core twisted cable alone. The 2-core twisted cable can be laid in the same duct as analog signal or other transmission-related cables.

## 8.2 Wiring 2-core Twisted Cable off a Panel

1) Lay 2-core twisted cable in a pit or the like for off-panel wiring. Be sure to lay the 2-core twisted cable in a separate pit from power and control lines when using pits. If these lines have to be laid in the same pit, then be sure to place the 2-core twisted cable in a separate conduit. In this case, make sure that both ends of the conduit are grounded.

2) If the conduit housing the 2-core twisted cable must be laid in the same pit with power lines, use a tray or some other means to keep the conduit as far away from the power lines as possible.

3) The 2-core twisted cable can be laid on the same tray in the same pit as analog signal and other transmission-related cables.

4) Off-panel wiring for the 2-core twisted cable is summarized in *Figure 8.1*.

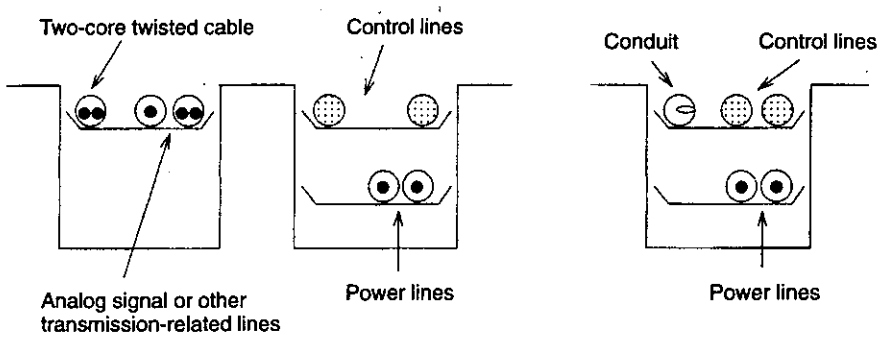



**Figure 8.1 Off-panel Wiring for 2-core Twisted Cable**

## 8.3 Grounding 2-core Twisted Cable

The shielded covering of 2-core twisted cable must be grounded in order to block external noise interference. As a general rule, ground the end of the shielded covering on the receive side. However, the transmission end or both ends may be grounded in some cases due to the nature of certain types of external noise. See *Figure 8.2.*

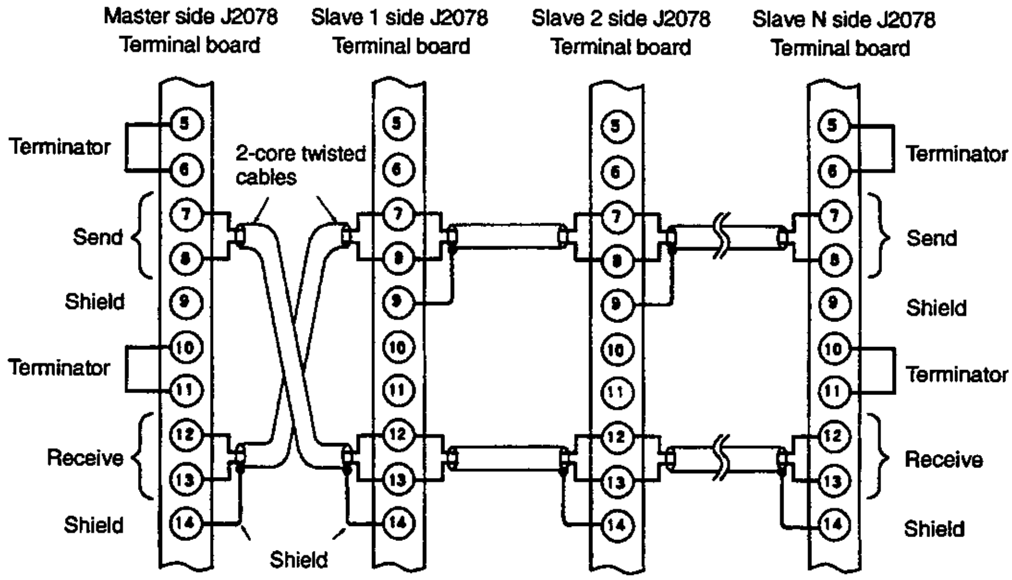



**Figure 8.2 Grounding 2-core Twisted Cable**

## 8.4 RS-232C Cable

1) The maximum length is 15 meters for connecting masters to slaves, masters to modems, and slaves to modems, but use the shortest possible cable for your connections.

2) Do not lay RS-232C cables in the same ducts as power or control lines. Either use a separate duct, or lay the wire alone. RS-232C cable may be laid in the same duct as analog signal and other transmission-related cables.

8

# MEMOCON Micro, GL120, GL130
# MEMOBUS
# USER'S MANUAL


**TOKYO OFFICE**
New Pier Takeshiba South Tower, 1-16-1, Kaigan, Minatoku, Tokyo 105-6891 Japan
Phone 81-3-5402-4511 Fax 81-3-5402-4580

**YASKAWA ELECTRIC AMERICA, INC.**
2121 Norman Drive South, Waukegan, IL 60085, U.S.A.
Phone 1-847-887-7000 Fax 1-847-887-7370

**MOTOMAN INC. HEADQUARTERS**
805 Liberty Lane West Carrollton, OH 45449, U.S.A.
Phone 1-937-847-6200 Fax 1-937-847-6277

**YASKAWA ELÉTRICO DO BRASIL COMÉRCIO LTDA.**
Avenida Fagundes Filho, 620 Bairro Saude-Sao Pàulo-SP, Brazil CEP: 04304-000
Phone 55-11-5071-2552 Fax 55-11-5581-8795

**YASKAWA ELECTRIC EUROPE GmbH**
Am Kronberger Hang 2, 65824 Schwalbach, Germany
Phone 49-6196-569-300 Fax 49-6196-888-301

**Motoman Robotics Europe AB**
Box 504 S38525 Torsås, Sweden
Phone 46-486-48800 Fax 46-486-41410

**Motoman Robotec GmbH**
Kammerfeldstraβe1, 85391 Allershausen, Germany
Phone 49-8166-900 Fax 49-8166-9039

**YASKAWA ELECTRIC UK LTD.**
1 Hunt Hill Orchardton Woods Cumbernauld, G68 9LF, United Kingdom
Phone 44-1236-735000 Fax 44-1236-458182

**YASKAWA ELECTRIC KOREA CORPORATION**
Kfpa Bldg #1201, 35-4 Youido-dong, Yeongdungpo-Ku, Seoul 150-010, Korea
Phone 82-2-784-7844 Fax 82-2-784-8495

**YASKAWA ELECTRIC (SINGAPORE) PTE. LTD.**
151 Lorong Chuan, #04-01, New Tech Park Singapore 556741, Singapore
Phone 65-282-3003 Fax 65-289-3003

**YASKAWA ELECTRIC (SHANGHAI) CO., LTD.**
4F No.18 Aona Road, Waigaoqiao Free Trade Zone, Pudong New Area, Shanghai 200131, China
Phone 86-21-5866-3470 Fax 86-21-5866-3869

**YATEC ENGINEERING CORPORATION**
Shen Hsiang Tang Sung Chiang Building 10F 146 Sung Chiang Road, Taipei, Taiwan
Phone 886-2-2563-0010 Fax 886-2-2567-4677

**YASKAWA ELECTRIC (HK) COMPANY LIMITED**
Rm. 2909-10, Hong Kong Plaza, 186-191 Connaught Road West, Hong Kong
Phone 852-2803-2385 Fax 852-2547-5773

**BEIJING OFFICE**
Room No. 301 Office Building of Beijing International Club, 21
Jianguomenwai Avenue, Beijing 100020, China
Phone 86-10-6532-1850 Fax 86-10-6532-1851

**TAIPEI OFFICE**
Shen Hsiang Tang Sung Chiang Building 10F 146 Sung Chiang Road, Taipei, Taiwan
Phone 886-2-2563-0010 Fax 886-2-2567-4677

**SHANGHAI YASKAWA-TONGJI M & E CO., LTD.**
27 Hui He Road Shanghai China 200437
Phone 86-21-6531-4242 Fax 86-21-6553-6060

**BEIJING YASKAWA BEIKE AUTOMATION ENGINEERING CO., LTD.**
30 Xue Yuan Road, Haidian, Beijing P.R. China Post Code: 100083
Phone 86-10-6233-2782 Fax 86-10-6232-1536

**SHOUGANG MOTOMAN ROBOT CO., LTD.**
7, Yongchang-North Street, Beijing Economic Technological Investment & Development Area, Beijing 100076, P.R. China
Phone 86-10-6788-0551 Fax 86-10-6788-2878



YASKAWA ELECTRIC CORPORATION

YASKAWA





MANUAL NO. SIEZ-C825-70.13

© Printed in Japan January 2001 96-2 ④
99-7③
95-81192, 95-C82-031