

# S10mini.

HITACHI

**S10mini**  
**HARDWARE MANUAL**

**OPTION**  
***RS-232C/422***  
***(LQE160/165)***

SME-1-121 (C)

**S10mini**  
**HARDWARE MANUAL**

**OPTION**  
***RS-232C/422***  
***(LQE160/165)***

First Edition, May 2003, SME-1-121(B) (out of print)  
Second Edition, September 2008, SME-1-121(C)

All Rights Reserved, Copyright © 2003, 2008, Hitachi, Ltd.

The contents of this publication may be revised without prior notice.

No part of this publication may be reproduced in any form or by any means without permission in writing from the publisher.

Printed in Japan.

BI-NR-TI<IC-IC> (FL-MW20, AI8.0)

## SAFETY PRECAUTIONS

Be sure to read this manual and all other attached documents carefully before installing, operating inspecting or conducting maintenance on this unit. Always use this unit properly. Be sure to carefully read the information about the device, the safety information and precautions before using this unit. Be sure that the person(s) responsible for maintenance receives and understands this manual completely.

This manual divides the safety precautions into DANGERS and CAUTIONS.



: Failure to observe these warnings may result in death or serious injury.



: Failure to observe these cautions may result in injury or property damage.

Failure to observe any  may lead to serious consequences.

All of these DANGERS and CAUTIONS provide very important precautions and should always be observed.

Additional safety symbols representing a prohibition or a requirement are as follows:



: Prohibition. For example, “Do not disassemble” is represented by:



: Requirement. For example, if a ground is required, the following will be shown:



## 1. Installation Precautions



### CAUTION

- Use this product under the environmental conditions specified in the catalogs and manual.  
Utilizing this product in a hot, damp, or dusty atmosphere or in an atmosphere of corrosive gas, vibration or impact may lead to a malfunction, shock hazard or fire.
- Install this product according to the procedure outline in the manual.  
Imperfect installation may lead to a part drop, failure or malfunction.
- Do not put any wire chip or other foreign matter into this product.  
This may cause a malfunction, failure or fire.

## 2. Wiring Precautions



### **REQUIREMENT**

Be sure to ground this product with FG.  
Failure to ground this product may lead to a malfunction or shock hazard.



### **CAUTION**

- Connect this product to a power supply with the same ratings.  
Connecting this product to a power supply exceeding its voltage rating may lead to a fire.
- Wiring must be conducted by a qualified technician.  
Miswiring may lead to failure, shock hazard or fire.

### 3. Operating Precautions



#### **DANGER**

- Do not touch any terminal while this product is live, as this may lead to a shock hazard.
- Configure an emergency stop circuit, interlocking circuit and related circuitry outside the programmable controller.  
A programmable controller failure may lead to a general breakdown or an accident.



#### **CAUTION**

- Make sure that everything is safe before changing programs, running or stopping this product while on the fly or producing forced output.  
Mishandling may lead to product breakdown or an accident.
- Turn on the product according to the correct power - on procedure.  
Mishandling may lead to product breakdown or an accident.
- Do not use a transceiver, mobile phone, or the like near this module. Using such a device nearby may lead to a malfunction or a module failure due to electrical noise.



#### **CAUTION**

The sample program is specifically written to promote easier understanding. With a working program, make an error check the return code of the transmission handler and the system register (S).

#### 4. Maintenance Precautions



### **PROHIBITION**

Do not disassemble or remodel this product, as this may lead to a malfunction, failure or fire.



### **CAUTION**

- Before installing the module, discharge any static buildup from your body because static electricity may render the module defective.
- Power off this product before attaching or detaching any module or unit as this may lead to a malfunction, failure or shock hazard.



## WARRANTY AND SERVICING

Unless a special warranty contract has been arranged, the following warranty is applicable to this product.

### 1. Warranty period and scope

#### Warranty period

The warranty period for this product is for one year after the product has been delivered to the specified delivery site.

#### Scope

If a malfunction should occur during the above warranty period while using this product under normal product specification conditions as described in this manual, please deliver the malfunctioning part of the product to the dealer or Hitachi Engineering & Services Co., Ltd. The malfunctioning part will be replaced or repaired free of charge. If the malfunctioning is shipped, however, the shipment charge and packaging expenses must be paid for by the customer.

This warranty is not applicable if any of the following are true.

- The malfunction was caused by handling or use of the product in a manner not specified in the product specifications.
- The malfunction was caused by a unit other than that which was delivered.
- The malfunction was caused by modifications or repairs made by a vendor other than the vendor that delivered the unit.
- The malfunction was caused by a relay or other consumable which has passed the end of its service life.
- The malfunction was caused by a disaster, natural or otherwise, for which the vendor is not responsible.

The warranty mentioned here means the warranty for the individual product that is delivered. Therefore, we cannot be held responsible for any losses or lost profits that result from the operation of this product or from malfunctions of this product. This warranty is valid only in Japan and is not transferable.

### 2. Range of services

The price of the delivered product does not include on-site servicing fees by engineers. Extra fees will be charged for the following:

- Instruction for installation and adjustments, and witnessing trial operations.
- Inspections, maintenance and adjustments.
- Technical instruction, technical training and training schools.
- Examinations and repairs after the warranty period is concluded.
- Even if the warranty is valid, examination of malfunctions that are caused by reasons outside the above warranty scope.

This manual provides information for the following hardware product:

<Hardware product>

RS-232C/422 (LQE160/165)

<Changes added to this manual>

Description of added changes	Page
Supplementary, "Replacing or adding on the module" is newly added.	Z-1

In addition to the above changes, all the unclear descriptions and typographical errors found are also corrected without prior notice.



## PREFACE

Thank you for purchasing the RS-232C/422 module, which is an option for use with the CPU. This manual, named "HARDWARE MANUAL OPTION RS-232C/422," describes how to use the RS-232C/422 module. For proper use of the RS-232C/422 module, it is requested that you thoroughly read this manual.

The S10mini series products are available in two types: standard model and environmentally resistant model. The environmentally resistant model has thicker platings and coatings than those for the standard model.

The model number of the environmentally resistant model is marked by adding the suffix "-Z" to the model number of the standard model.

(Example) Standard model: LQE160

Environmentally resistant model: LQE160-Z

This manual is applicable to both the standard model and environmentally resistant models. Although the descriptions contained in this manual are based on the standard model, follow the instructions set forth in this manual for proper use of the product even if you use the environmentally resistant model.

### <Related manuals>

For information about the External Device Link System For Windows® (Type S-7890-24) that is used in editing the LGB table, refer to the following manual:

Number	Name
SAE-3-143	SOFTWARE MANUAL OPTION EXTERNAL SERIAL LINK For Windows®

For information about the Backup Restore System For Windows® (Type S-7890-09) that is used in backup-restoring system and application programs, refer to the following manual:

Number	Name
SAE-3-127	SOFTWARE MANUAL OPERATION BACKUP RESTORE For Windows®

Not that the following terms are used in special sense this manual:

<u>Term</u>	<u>Meaning</u>
Tool	The personal computer on which the External Device Link System For Windows® is installed.
Backup Restore	Backup Restore System For Windows® (Type S-7890-09)
External Device Link System	External Device Link System For Windows® (Type S-7890-24)

<Trademarks>

Microsoft® Windows® operating system, Microsoft® Windows® 95 operating system, Microsoft® Windows® 98 operating system are registered trademarks of Microsoft Corporation in the United States and/or other countries.

<Note for storage capacity calculations>

- Memory capacities and requirements, file sizes and storage requirements, etc. must be calculated according to the formula  $2^n$ . The following examples show the results of such calculations by  $2^n$  (to the right of the equals signs).
  - 1 KB (kilobyte) = 1,024 bytes
  - 1 MB (megabyte) = 1,048,576 bytes
  - 1 GB (gigabyte) = 1,073,741,824 bytes
- As for disk capacities, they must be calculated using the formula  $10^n$ . Listed below are the results of calculating the above example capacities using  $10^n$  in place of  $2^n$ .
  - 1 KB (kilobyte) = 1,000 bytes
  - 1 MB (megabyte) = 1,000<sup>2</sup> bytes
  - 1 GB (gigabyte) = 1,000<sup>3</sup> bytes

## CONTENTS

1	BEFORE USE.....	1-1
1.1	CPU Mount Base.....	1-2
1.2	Mounting the Module.....	1-2
1.3	Ground Wiring .....	1-4
2	SPECIFICATIONS .....	2-1
2.1	Use.....	2-2
2.2	Specifications .....	2-3
2.2.1	System specifications .....	2-3
2.2.2	Software specifications.....	2-4
3	NAMES AND FUNCTIONS OF EACH PART.....	3-1
3.1	Names and Functions of Each Part.....	3-2
4	OPERATION .....	4-1
4.1	Starting the System.....	4-2
4.2	Editing the LGB Table .....	4-4
4.3	LGB Table Settings .....	4-5
4.4	Host Interrupts.....	4-26
4.4.1	Host interrupt register.....	4-26
5	PROGRAMMING .....	5-1
5.1	Software Configuration .....	5-2
5.2	System Registers .....	5-4
5.2.1	Transmitted information.....	5-4
5.2.2	Received information .....	5-5
5.3	Send/Receive Handlers.....	5-6
5.3.1	Computing functions .....	5-6
5.3.2	Subroutines.....	5-9
5.4	Incorporating Received Data .....	5-16
5.5	Hardware Controlled by Software Implementation.....	5-17
6	SAMPLE PROGRAMS.....	6-1
6.1	Sample RS-232C Wiring with a Printer.....	6-2

6.1.1	Overview .....	6-2
6.1.2	System configuration .....	6-2
6.1.3	Print format .....	6-3
6.1.4	Program configuration .....	6-4
6.1.5	Ladder program linkage table configuration .....	6-5
6.1.6	RS-232C module.....	6-7
6.1.7	Setting the LGB table.....	6-8
6.1.8	C language program flowchart.....	6-10
6.1.9	C language sample program.....	6-11
6.1.10	Ladder program.....	6-13
6.2	PC-based Program Loading .....	6-14
6.2.1	System configuration .....	6-14
6.2.2	Program configuration .....	6-14
6.2.3	Motorola 'S' format (16-bit).....	6-15
6.2.4	LGB table settings.....	6-16
6.2.5	Registering a receive task.....	6-17
6.2.6	Start task.....	6-18
6.2.7	C language sample program.....	6-19
6.2.8	Loading programs .....	6-21
7	MAINTENANCE.....	7-1
7.1	Maintenance and Check .....	7-2
7.2	Backing Up User Setup Items .....	7-3
7.2.1	LGB table, receive task entry table, and user computing function entry table.....	7-3
7.2.2	Replacing modules.....	7-4
7.3	Troubleshooting .....	7-5
7.3.1	CPU module indicator display messages .....	7-5
7.3.2	Hardware errors.....	7-6
7.3.3	Transmit errors.....	7-7
7.3.4	Receive errors.....	7-8
7.3.5	Error freeze.....	7-9
7.3.6	Communications trace information.....	7-11
7.3.7	Handler trace information .....	7-13
7.3.8	H-7338 error trace information .....	7-15
7.3.9	Error Counters.....	7-17

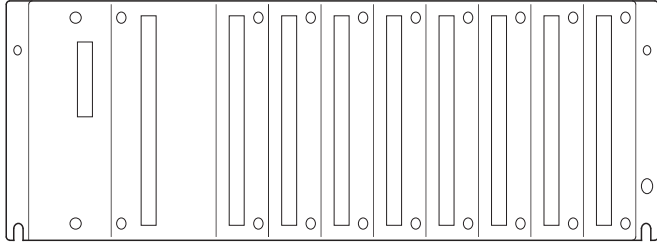
APPENDIX.....	A-1
A.1 Seven-Bit Code Table (JIS X 0201).....	A-2
A.2 Eight-Bit Code Table (JIS X 0201).....	A-3
A.3 Control Code Definitions .....	A-4
A.4 Abbreviations .....	A-5
A.5 Trouble Report .....	A-6
SUPPLEMENTARY .....	Z-1
Supplementary: Replacing or adding on the module .....	Z-2



# 1 BEFORE USE

# 1 BEFORE USE

## 1.1 CPU Mount Base



8-slot mount base

There are three types of CPU mount bases:

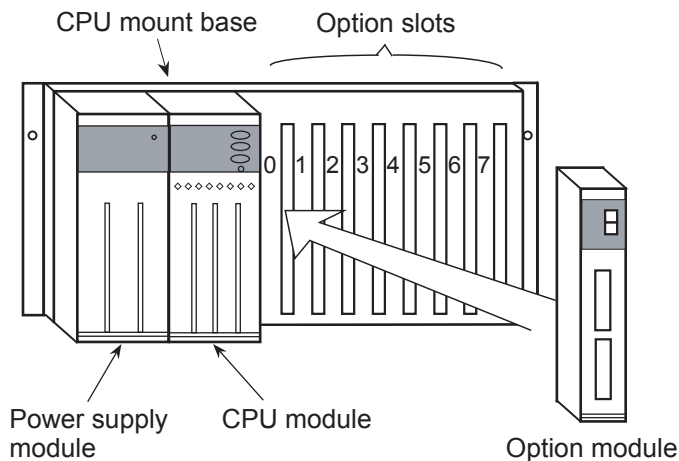
2-slot mount base (model: HSC-1020)

4-slot mount base (model: HSC-1040)

8-slot mount base (model: HSC-1080)

On the 8-slot mount base, for example, in addition to the power supply and CPU modules, up to eight modules can be mounted.

## 1.2 Mounting the Module



CPU mount base: HSC-1080

PS slot: A slot into which the power supply (LQV000, LQV020 or LQV100) module is inserted.

CPU slot: A slot into which the CPU module (LQP000, LQP010, LQP011 or LQP120) is inserted.

Slot 0 to 7: Slot into which option modules or I/O modules are inserted.

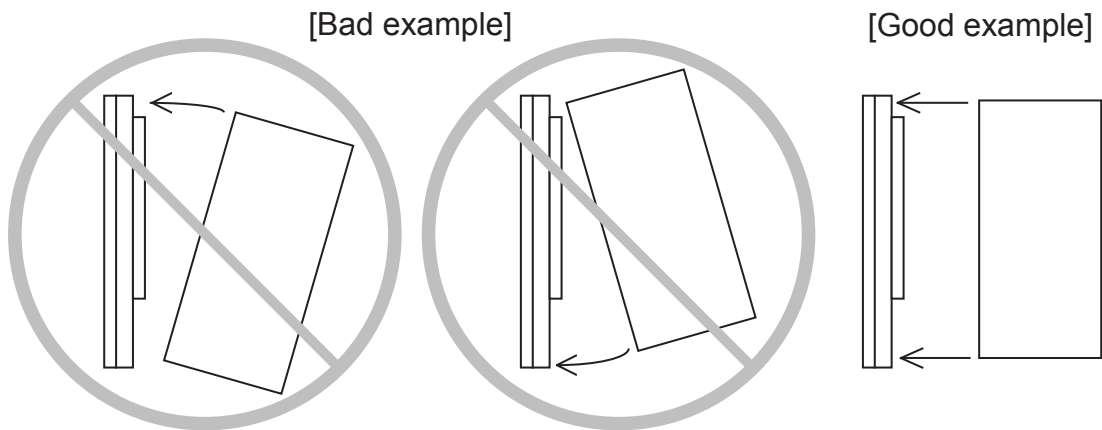


### CAUTION

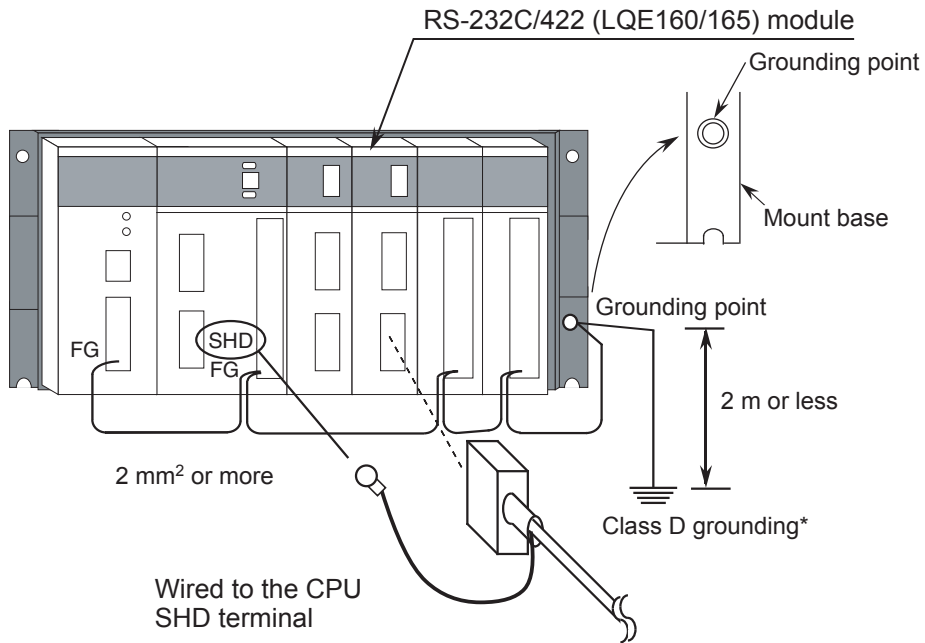
Mount option modules, left-justified, to the left of the CPU module, with no I/O module intervening. An I/O module intervening between the CPU module and I/O modules would disable them from functioning correctly. Vacant slots between the CPU module and option modules would also disable correct functioning.

### Tips on mounting option modules

Mount an option module straight from the front in relation to the CPU mount base as shown below. A slantwise mounted option module like that shown in [Bad] below could damage the connector, causing the module to malfunction.



### 1.3 Ground Wiring



\* Class D grounding is defined in the Technical Standard for Electrical Facilities of Japan. This standard states that the grounding resistance must be 100 ohms or less for equipment operating on 300 VAC or less, and 500 ohms or less for devices that shut down automatically within 0.5 seconds when shorting occurs in low tension lines.



#### REQUIREMENT

- For the ground wiring of the FG (frame ground), connect the FG terminal of each module with an external terminal to the grounding point of the mount base.  
Hold the grounding distance to 2 m or less. Carry out JIS Class D grounding from the grounding point of the mount base.
- Use a ground wire with a wire diameter of 2 mm<sup>2</sup> or more.
- Connect the shielding cable of the communication cable to the SHD terminal of the CPU module.

## **2 SPECIFICATIONS**

## 2 SPECIFICATIONS

---

### 2.1 Use

The RS-232C module (Model: LQE160) and the RS-422 module (Model: LQE165) execute data communication under a free-running protocol or under the H-7338 protocol pursuant to the EIA RS-232C/RS-422 specifications.

## 2.2 Specifications

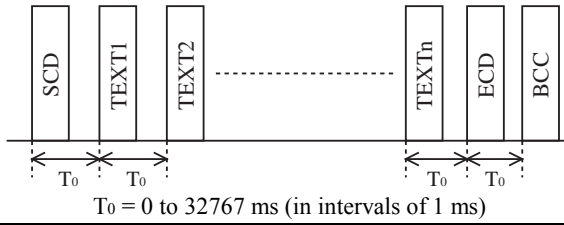
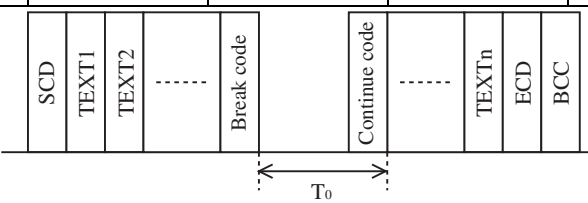
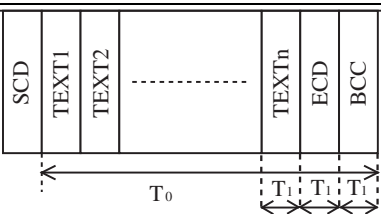
### 2.2.1 System specifications

Item	Specification											
Model	LQE160	LQE165										
Protocol	Free-running/H-7338, switched											
Maximum number of RS-232C/ RS-422 modules mounted	S10mini: Two modules/CPU (mounted left-justified) S10V: Two modules/LPU (no need to mount left-justified)											
Module slot width	One-slot width module											
Mass	220 g	220 g										
Transmission mode	Serial transmission (Bit serial transmission)											
Communication mode	Half-duplex/full-duplex, switched											
Synchronization method	Start-stop synchronization											
Interface	Compliant with EIA RS-232C	Compliant with EIA RS-422										
Data frame structure	<table border="1"> <thead> <tr> <th>ST</th> <th>DATA</th> <th>PT</th> <th>SP</th> </tr> <tr> <th>Start bit</th> <th>Data bit</th> <th>Parity bit</th> <th>Stop bit</th> </tr> </thead> </table>				ST	DATA	PT	SP	Start bit	Data bit	Parity bit	Stop bit
	ST	DATA	PT	SP								
	Start bit	Data bit	Parity bit	Stop bit								
		Start	Data length	Parity	Stop							
	Free-running	1 bit	7 bits	Even	2 bits							
				Odd								
				Even	1 bit							
				Odd								
			8 bits	No	2 bits							
					1 bit							
Even				2 bits								
Odd												
Even	1 bit											
Odd												
H-7338 protocol	1 bit	8 bits	Odd	1 bit								
Baud rate	Free-running	300, 600, 1200, 2400, 4800, 9600, 19200 (bps)										
	H-7338 protocol	19200 (bps)										
Connecting cable	Distance	Up to 15 m	Up to 500 m									
	Wire type	Shielded twisted pair cable										
	Wire diameter	0.08 mm <sup>2</sup> or more	0.3 mm <sup>2</sup> or more									
	Resistance	229 Ω/km or less (20°C)	54.4 Ω/km or less (20°C)									
	Recommended brand	CO-VV-SB(MA)13P × 28AWG(7/0.127) (*) (manufactured by Hitachi Cable, Ltd.)	CO-SPEV-SB-5P 0.3 mm <sup>2</sup> (manufactured by Hitachi Cable, Ltd.)									
Connector	Type	D-sub 9-pin connector										
	Remarks	Cover: HDE-CTH1 (manufactured by HIROSE ELECTRIC CO., LTD.) Connector: HDEB-9S (manufactured by HIROSE ELECTRIC CO., LTD.)										
Cable grounding condition	Double-ended grounding											

(\*) Choose the number of cores from between 5P and 8P to meet the number of signals required.

## 2 SPECIFICATIONS

### 2.2.2 Software specifications

Item		Specification			
Transmission control procedure		Free-running			
Priority level		Local station prioritized (Rejects receive requests while transmitting)			
		Remote station prioritized (Accepts receive requests even while transmitting)			
		No priority level (Full-duplex communication)			
Data change mode		Transmits and receives text data as it is.			
		Transmits text data after ASCII conversion and receives text data after binary conversion.			
Transmitted block structure	Start code	No, 1 to 4 characters			
	Text	No, 1 to 512 bytes			
	End code	No, 1 to 4 characters			
	Block check character	No, horizontal even parity, horizontal odd parity			
Send delay time		 <p style="text-align: center;"><math>T_0 = 0</math> to 32767 ms (in intervals of 1 ms)</p>			
Send break and continue codes	Break code	No	1 character	2 characters	
	Continue code	No	1 character	2 characters	1 character    2 characters
Send break timeout		 <p style="text-align: center;"><math>T_0 = 0</math> to 3276.7 s (in intervals of 100 ms)</p>			
Receive monitoring time		 <p style="text-align: center;"><math>T_0 = 0</math> to 3276.7 s (in intervals of 100 ms)  <math>T_1 = 1</math> to 32767 ms (if "No priority level" is specified)</p>			
Request to Send (RS) output		RS output (RS pin held ON)			
		No RS output			
Equipment Ready (ER) output		Not Ready output			
		Ready output (ER pin held ON)			
Data Set Ready (DR) input		No checking			
		Checking enabled			
Control signal automatic control		Manual control			
		Automatic control			
Send buffer size		512 bytes			
Receive buffer size		512 bytes × 8 buffers			

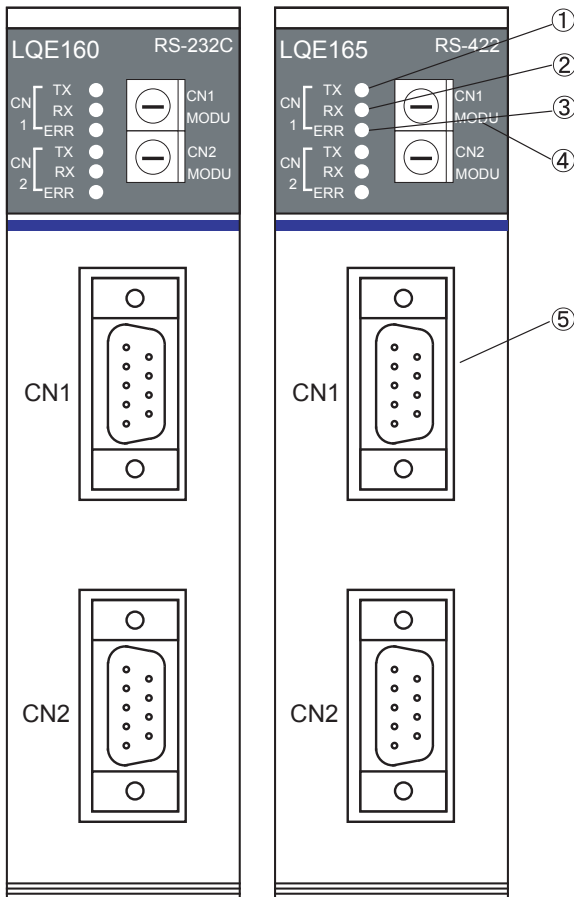


# **3 NAMES AND FUNCTIONS OF EACH PART**

### 3 NAMES AND FUNCTIONS OF EACH PART

#### 3.1 Names and Functions of Each Part

##### (1) Names and functions of each part



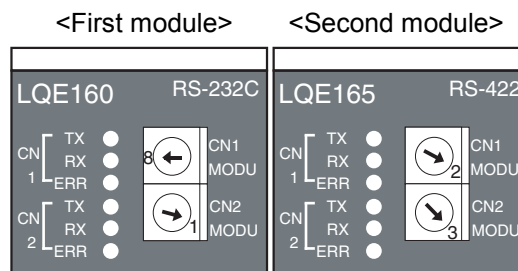
- ① TX LED  
Glow when transmitting data.
- ② RX LED  
Glow when receiving data.
- ③ ERR LED  
Glow when a hardware error has occurred.
- ④ Module switch  
Use to set a communications protocol and a channel number.  
Settings do not take effect until the module restarts.

Module switch	Communications protocol	Channel number
0	Free-running – Computing function	#0
1	Free-running – Computing function	#1
2	Free-running – Computing function	#2
3	Free-running – Computing function	#3
4	Free-running – Task	#0
5	Free-running – Task	#1
6	Free-running – Task	#2
7	Free-running – Task	#3
8	H-7338 protocol	#0
9	H-7338 protocol	#1
A	H-7338 protocol	#2
B	H-7338 protocol	#3
C	Reserved for maintenance use	
D		
E		
F		

- ⑤ RS-232C connector (LQE160), RS-422 connector (LQE165)  
Used to connect the module to a Remote device.  
For the pin configuration, see the next page.

 **PROHIBITION**

- Do not alter the channel number and protocol setup switching while the module is switched on.
- Do not use a computing function and a task system intermixed on the same CPU.
- Do not allow channel numbers to be defined in duplicate on the same CPU.
- Up to two modules (four channels) of both types can be mounted on the same CPU; do not mount more than three modules.



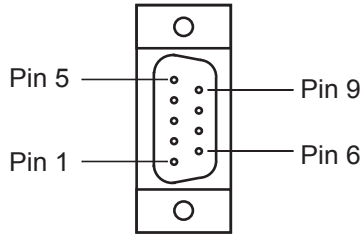
<Setup example>

	Module switch	Communications protocol	Channel number
First module-CN1	8	H-7338 protocol	#0
First module-CN2	1	Free-running – Computing function	#1
Second module-CN1	2	Free-running – Computing function	#2
Second module-CN2	3	Free-running – Computing function	#3

### 3 NAMES AND FUNCTIONS OF EACH PART

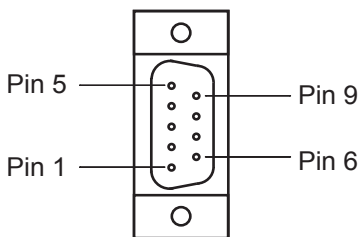
#### (2) Connector pin configuration

##### (a) Pin configuration (RS-232C)



Pin No.	Signal name	I/O	Explanation
1	CD (Data Carrier Detect)	Input	On: Receiver carrier detected Off: Receive carrier not detected
2	RD (Receive Data)		On: Received data space Off: Received data mark
3	SD (Send Data)	Output	On: Received data space Off: Received data mark
4	ER (Equipment Ready)		On: Module ready to transmit and receive Off: Module not ready to transmit and receive
5	SG (Signal Ground)	-	Signal ground
6	DR (Data Set Ready)	Input	On: Remote station ready to operate Off: Remote station not ready to operate
7	RS (Request to Send)	Output	On: Module having a request to send Off: Module having no request to send
8	CS (Clear to Send)	Input	On: Remote station ready to transmit Off: Remote station not ready to transmit
9	Not used	-	

##### (b) Pin configuration (RS-422)



Pin No.	Signal name	I/O	Explanation
1	RD-L (Receive Data Low)	Input	Received data standard
2	RD-H (Receive Data High)		On: Received data space Off: Received data mark
3	SD-H (Send Data High)	Output	On: Transmitted data space Off: Transmitted data mark
4	SD-L (Send Data Low)		Transmitted data standard
5	SG (Signal Ground)	-	Signal ground
6	Not used	-	
7	ATT-H (Attention-int High)	Output	On: No ATT request Off: ATT request existing
8	Not used	-	
9	ATT-L (Attention-int Low)	Output	Standard

## (3) Signal definitions

## ● RS-232C

- SD ..... Transmitted data line from the RS-232C module to the remote station. Enabled when the four control lines of RS, CS, DR and ER are on.
- RD ..... Received data line from the remote station to the RS-232C module. Enabled when CD is on. RD is kept in the mark (off) state while the remote station is not transmitting data (that is, CD is off).
- RS ..... A control signal line reporting that data is available from the RS-232C module for transmission to the remote station. The remote station remains ready to receive data from the RS-232C module while RS is on. Once RS has turned off, it cannot turn on again until CS turns off.
- CS ..... A control signal line reporting that the remote station is ready to transmit over the communication line if it is a modem. The remote station is ready to receive transmitted data from the RS-232C module the CS is on.
- DR ..... A control signal line reporting that the remote station is ready to operate. The remote station is connected to the line if it is a modem, allowing control signals to be transmitted to and from the RS-232C module.
- SG ..... Signal ground. It provides a reference voltage (0 V) for all the signals.
- CD ..... A control signal line reporting that the remote station is receiving a valid signal from the communication line if it is a modem. RD is enabled when CD is on. When CD turns on, the RS-232C module enters data from the remote station. RD is kept in the mark (off) state while CD is off.
- ER ..... A control signal line reporting that the RS-232C module is ready to transmit data to and from the remote station. If the remote station is a modem, it is connected to the line when ER turns on, and is disconnected when ER turns off.
- SHD ..... Cable shield ground. (The CPU SHD terminal is used; no connector is specifically assigned.)

## ● RS-422

- SD ..... Transmitted data line from the RS-422 module to the remote station.
- RD ..... Receive data line from the remote station to the RS-422 module.
- ATT ..... An interrupt signal line from the RS-422 module to the remote station.
- SG ..... Signal ground between stations.

### 3 NAMES AND FUNCTIONS OF EACH PART

---

#### (4) Voltage levels

- RS-232C

	Mark	Space
Interpretation	1/off	0/on
Output condition	-5 V to -15 V	+5 V to + 15 V
Input condition	$\leq -3$ V	+3 V $\leq$

- RS-422

	Mark	Space
Interpretation	1/off	0/on
Output condition	-3 V to -6 V	+3 V to + 6 V
Input condition	$\leq -0.2$ V	+0.2 V $\leq$

(5) Wiring methods

● RS-232C

Connection	Name	Wiring method (Logical connection)			Remote station example
		RS-232C module	Cable	Remote station	
Direct	Full modem support (Standard type)  ( Transmits and receives data while implementing CD-based received management and DR-based transmit management. )				Personal computer
	CD-based received management  ( Reads the RS-232C module to receive on Request to Send (RS) from the remote station. )				
	DR-based received management  ( Transmits data from the RS-232C module on Equipment Ready (ER) from the remote station. )				
	Data only  ( Transmits and receives data without checking the remote station and RS-232C module status. )				
Modem	Modem connection				Modem

Note: In the table, (P) denotes the availability of a Request to Send (RS), or the act of keeping ER in the ready state.

### 3 NAMES AND FUNCTIONS OF EACH PART

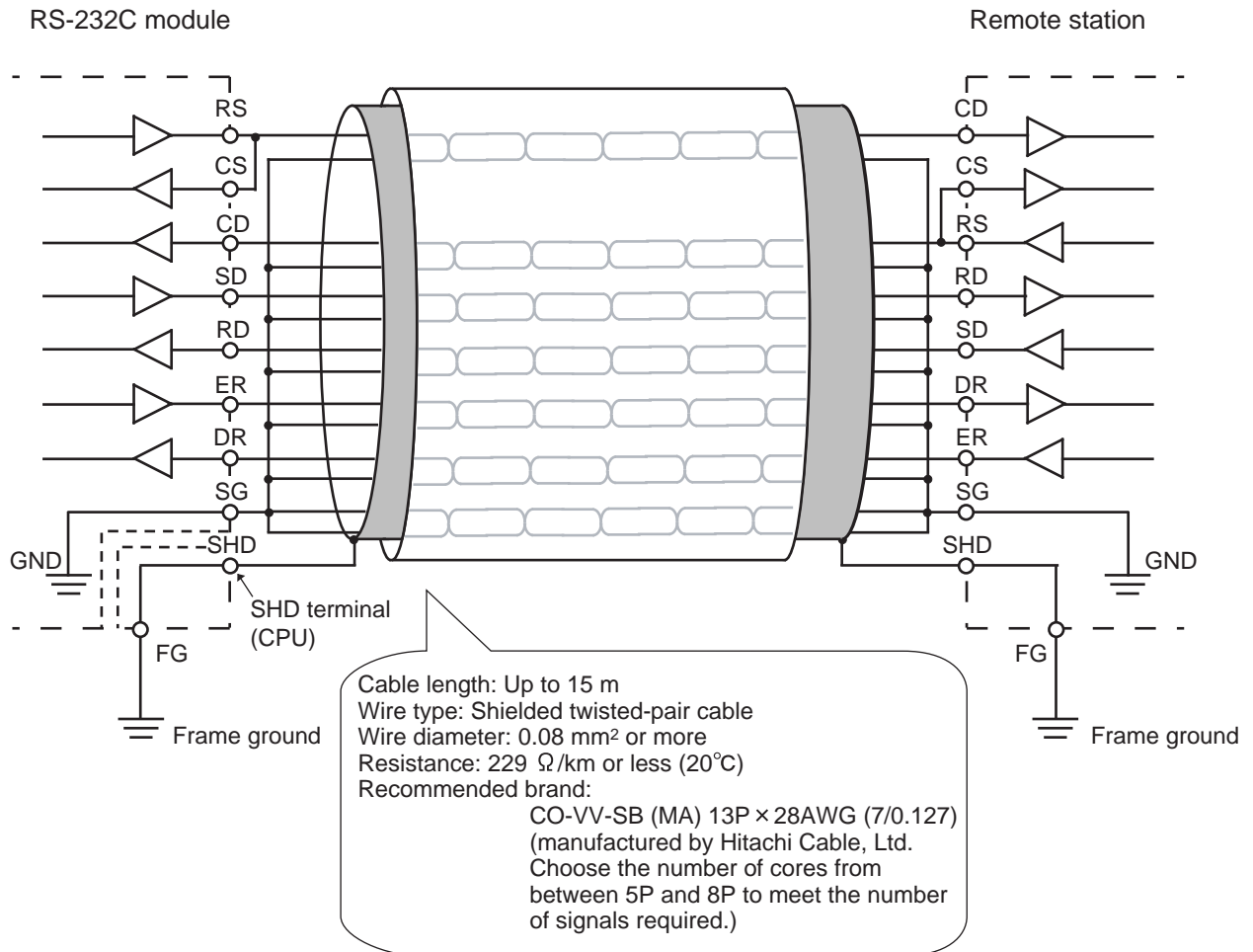
- RS-422

Connection	Name	Wiring method (Logical connection)			Remote station example
		RS-422 module	Cable	Remote station	
Direct	Data only	SD-H SD-L RD-H RD-L ATT-H ATT-L SG		SD-H SD-L RD-H RD-L ATT-H ATT-L SG	Indicator
	Data and attention interrupt	SD-H SD-L RD-H RD-L ATT-H ATT-L SG		SD-H SD-L RD-H RD-L ATT-H ATT-L SG	LINK-PCS



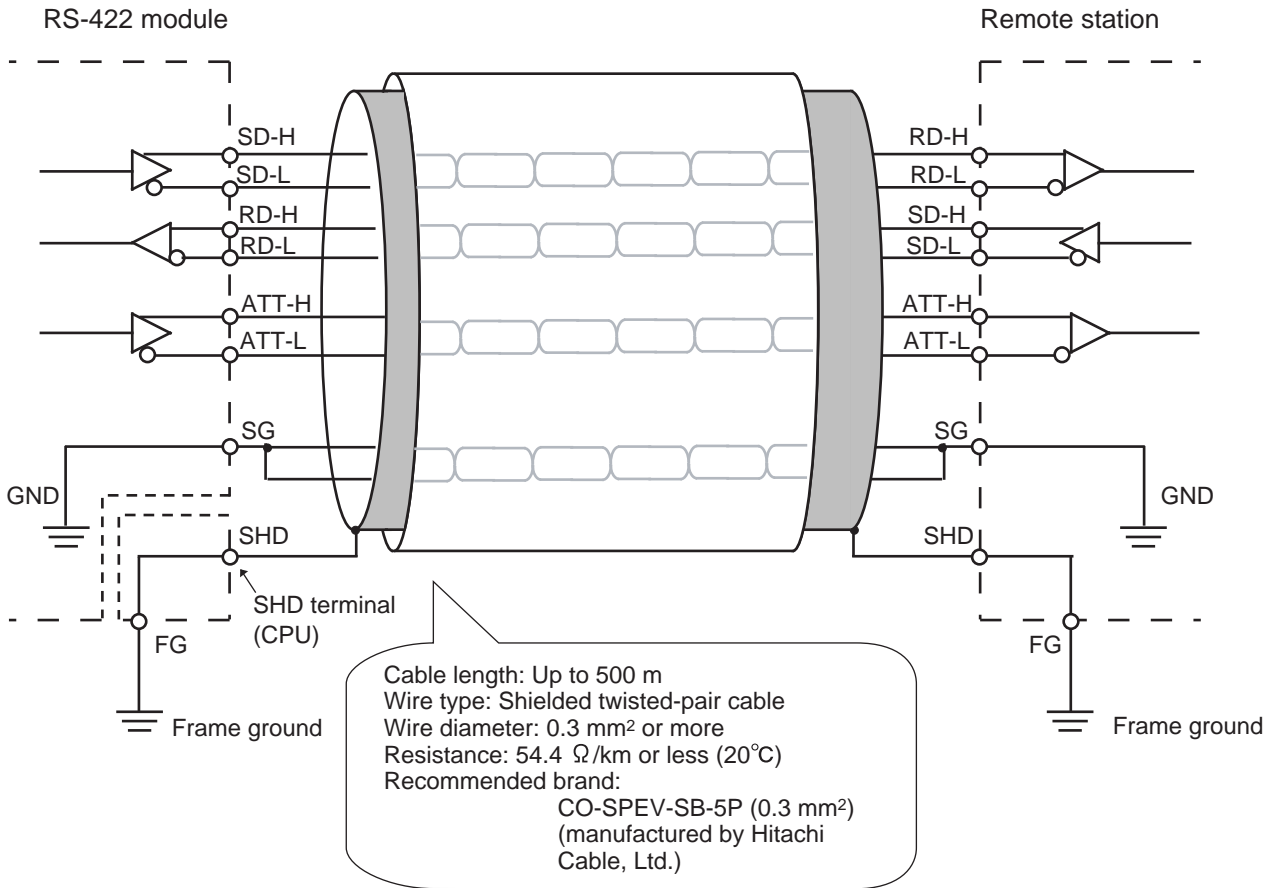
- RS-232C

An example of standard wiring is shown below.



### 3 NAMES AND FUNCTIONS OF EACH PART

● RS-422





#### CAUTION

- Be sure to connect the signal ground (SG) of the RS-232C/422 module and that of the remote station with an interface cable.
- Ground the shielded grounding terminal of the interface cable at both RS-232C/422 module and the remote station to provide added noise immunity.
- Connect the shielded grounding (SHD) terminal of the interface cable (at the RS-232C/422 module) to the shielded grounding (SHD) terminal of the CPU module terminal strip.



# 4 OPERATION

### 4.1 Starting the System

- |     |                                 |  |
|-----|---------------------------------|--|
| (1) | Mount module                    | (1) Switch off the CPU and mount an RS-232C or RS-422 module.  |
| (2) | Set rotary switch               | (2) Set the module switch (communications protocol, channel number) to suit specific applications per connector. For how to use H-7338 protocol host interrupts (ATT signal), see “4.4 Host Interrupts.” |
| (3) | Start up the setup tool         | (3) Start up the setup tool by connecting it to the CPU. Refer to the “SOFTWARE MANUAL OPTION EXTERNAL SERIAL LINK SYSTEM For Windows® (manual number SAE-3-143).”                                       |
| (4) | Edit LGB table                  | (4) Edit the LGB table to meet the specifications of the remote station connected. Refer to the “SOFTWARE MANUAL OPTION EXTERNAL SERIAL LINK SYSTEM For Windows® (manual number SAE-3-143).”             |
| (5) | Register a computing function   | (5) Register a computing function to run an application program. Refer to the “SOFTWARE MANUAL OPTION EXTERNAL SERIAL LINK SYSTEM For Windows® (manual number SAE-3-143).”                               |
| (6) | Register a receive startup task | (6) Register a receive startup task to create an application program in C-mode. Refer to the “SOFTWARE MANUAL OPTION EXTERNAL SERIAL LINK SYSTEM For Windows® (manual number SAE-3-143).”                |

**NOTE**

- Select either of the following settings to suit the application usage.
  - [Free-running – Computing function]
  - [Free-running – Task]  
An extension memory module is needed if the LQP000 is used as a CPU.
  - H-7338 protocol
- The following software packages are needed to edit the LGB table:  
[External serial link system For Windows®] (Model: S-7890-24)
- With the free-running protocol, it is necessary to edit the LGB table to suit the specifications of the remote station.  
With the H-7338 protocol, there is no need to set LGB table (no effect).
- Edits to the LGB table do not take effect until after the module is reset (using the CPU module reset switch). If the power fails and then recovers after or during a reset after the LGB table has been edited, the table will be reset to its previous settings. In this case, edit the LGB table again and reset the module.

## 4.2 Editing the LGB Table

Term LGB (Line Group Block) refers to a set of information that a communications control program transmits and receives over a line. Users decide this information by using the external serial link system that has been started from the tool system.

The way the LGB table is set is so important that it could inhibit successful hardware wiring to the remote station or disable correct transmission/reception due to inconsistent transmission procedures. Edit the LGB table to meet the specifications of the remote station connected to each channel number.

### ■ LGB Table Settings

Item	LQE160 (RS-232C)	LQE165 (RS-422)
Data frame	√	√
Baud rate	√	√
Priority level	√	√
Data change mode	√	√
Text size	√	√
Start code	√	√
End code	√	√
Block check character	√	√
Send delay time	√	√
Send break and continue codes	√	√
Send break timeout	√	√
Receive timeout	√	√
RS-422 gate control	–	–
Request to Send (RS)	√	–
Equipment Ready (ER)	√	–
Data Set Ready (DR)	√	–
Control signal automatic control	√	–
System selection	√	√

√: Setting enabled

–: Setting disable

### 4.3 LGB Table Settings

The individual items of the LGB table that users can edit are described on the pages that follow.

#### ■ Data frame

Decide the structure of one byte of data transmitting over the line.

Tool selection item		Data frame content	Default
No.	Display		
1	ST+7DT+EP+2SP	ST   2 <sup>0</sup>   _____   2 <sup>6</sup>   EP   SP   SP	
2	ST+7DT+OP+2SP	ST   2 <sup>0</sup>   _____   2 <sup>6</sup>   OP   SP   SP	
3	ST+7DT+EP+1SP	ST   2 <sup>0</sup>   _____   2 <sup>6</sup>   EP   SP	
4	ST+7DT+OP+1SP	ST   2 <sup>0</sup>   _____   2 <sup>6</sup>   OP   SP	
5	ST+8DT+2SP	ST   2 <sup>0</sup>   _____   2 <sup>7</sup>   SP   SP	
6	ST+8DT+1SP	ST   2 <sup>0</sup>   _____   2 <sup>7</sup>   SP	
7	ST+8DT+EP+1SP	ST   2 <sup>0</sup>   _____   2 <sup>7</sup>   EP   SP	
8	ST+8DT+OP+1SP	ST   2 <sup>0</sup>   _____   2 <sup>7</sup>   OP   SP	√
9	ST+7DT+2SP	ST   2 <sup>0</sup>   _____   2 <sup>6</sup>   SP   SP	
10	ST+7DT+1SP	ST   2 <sup>0</sup>   _____   2 <sup>6</sup>   SP	
11	ST+8DT+EP+2SP	ST   2 <sup>0</sup>   _____   2 <sup>7</sup>   EP   SP   SP	
12	ST+8DT+OP+2SP	ST   2 <sup>0</sup>   _____   2 <sup>7</sup>   OP   SP   SP	

ST: Start bit  
DT: Data bit  
EP: Even parity bit

OP: Odd parity bit  
SP: Stop bit



## 4 OPERATION

---

### ■ Baud rate

Set the baud rate (bps) of the line (between 300 and 19,200 bps).

Tool selection item		Baud rate content	Default
No.	Display		
1	300 [BPS]	300 [bps]	
2	600 [BPS]	600 [bps]	
3	1200 [BPS]	1200 [bps]	
4	2400 [BPS]	2400 [bps]	
5	4800 [BPS]	4800 [bps]	√
6	9600 [BPS]	9600 [bps]	
7	19200 [BPS]	19200 [bps]	

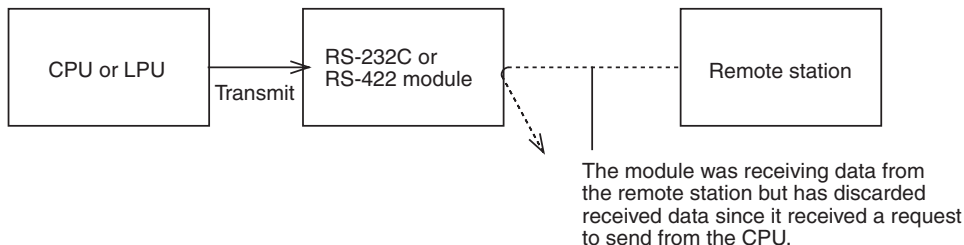
BPS, bps: bits per second

### ■ Priority level

Specify the priority level of the local station (RS-232C or RS-422 module) or the remote station (mating device). The priority level indicates that which the RS-232C or RS-422 module will give priority to, the CPU module or the remote station, when action is initiated from them concurrently.

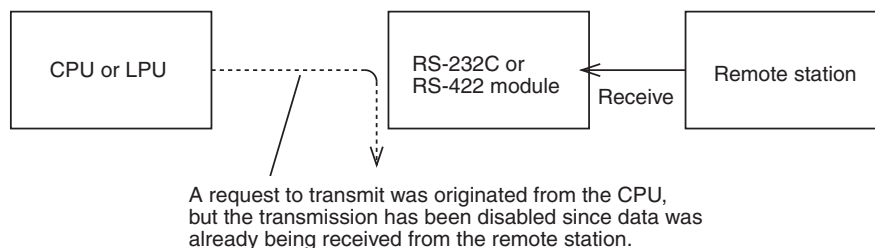
Tool selection item		Priority level content	Default
No.	Display		
1	Local station prioritized	Local station prioritized (Half-duplex communication)	√
2	Remote station prioritized	Remote station prioritized (Half-duplex communication)	
3	No priority level (Full-duplex communication)	No priority level (Full-duplex communication)	

● Local station prioritized



In the example shown above, the application program running on the CPU learns from the S-register that data reception from the remote station has made way for transmission. The remote station, however, is unable to recognize that data received from the remote station has been discarded. It is the CPU's responsibility to notify the remote station.

● Remote station prioritized



In the example shown above, the application program running on the CPU learns from the S-register that data transmission has been disabled.

■ Data change mode

Specify whether to handle text data in circulation as ASCII data or as binary data.

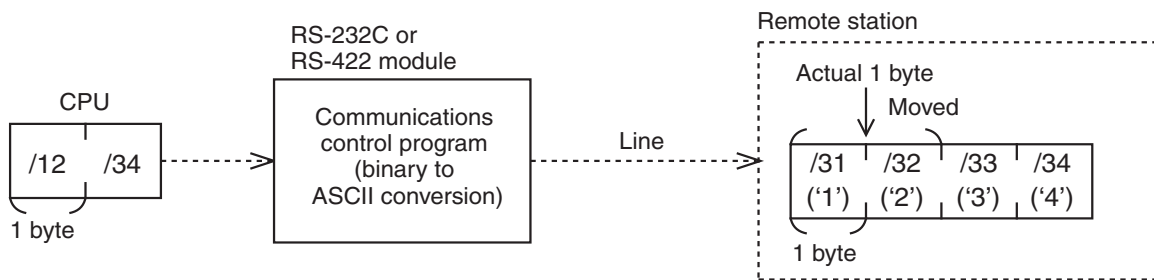
Tool selection item		Data change mode content	Default
No.	Display		
1	ASCII	Handle text data as ASCII	
2	Binary	Handle text data as binary	√

## 4 OPERATION

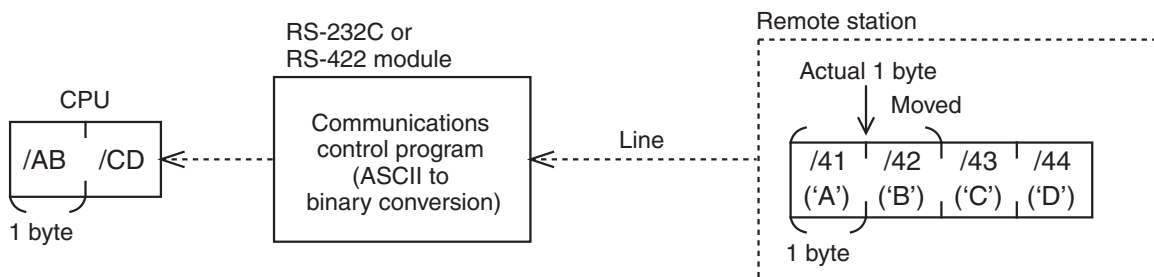
- ASCII specification

With the ASCII specification, a remote station program performs a data conversion between ASCII and binary, so that data traffic over the line doubles.

Transmitting any text data other than '0' to '9' and 'A' to 'F' would invoke an error.



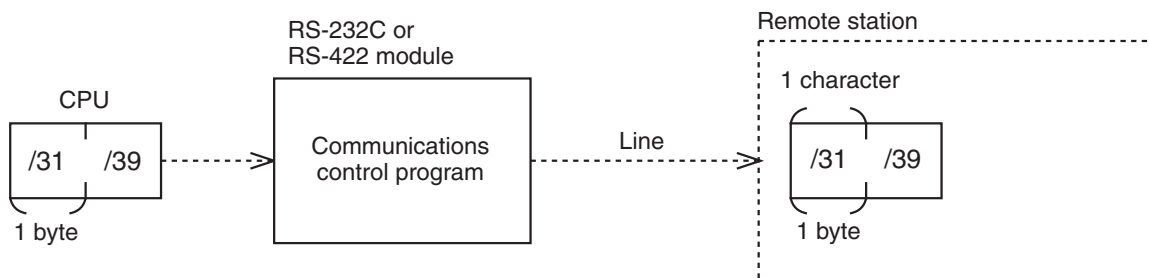
In the example shown above, if data /12 and /34 is transmitted from the CPU, the communications control program will perform a binary to ASCII conversion on the data to transmit data /31 ('1'), /32 ('2'), /33 ('3'), and /34 ('4'). A remote station program must be created to meet this condition.



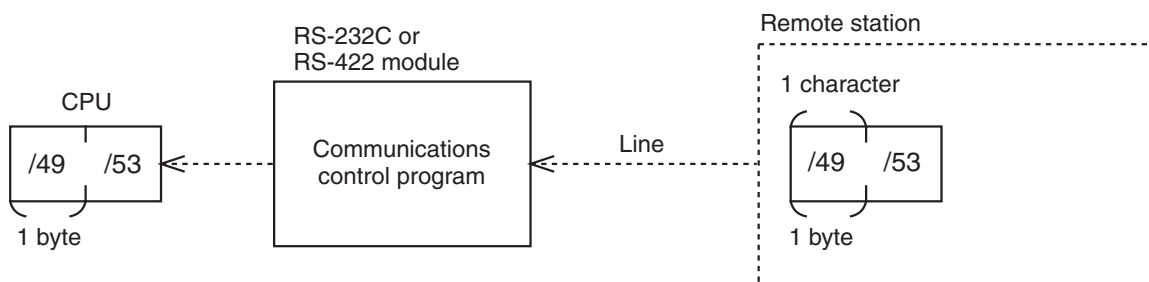
In the example shown above, if data /41 ('A'), /42 ('B'), /43 ('C'), and /44 ('D') is received from the remote station, the communications control program will perform an ASCII to binary conversion on the data to pass data /AB and /CD to the CPU. A remote station program must be created to meet this rule.

● Binary specification

With the binary specification, there is no need for the remote station program to perform a data conversion between ASCII and binary.



In the example shown above, if data /31 and /39 is transmitted from the CPU, the communications control program will pass the data on to the remote station as it is.



In the example shown above, if data /49 and /53 is received from the remote station, the communications control program will pass the data on to the CPU as it is.

■ Text size

Specify the word length of text data between 0 and 512.

Tool selection item		Text size content	Default
Setting	Display		
0	No text	No text	
1 to 512	001 to 512 [BYTE]	1 to 512 (bytes)	256
-	Text variable length	Text variable length (up to 512 bytes)	

## 4 OPERATION

---

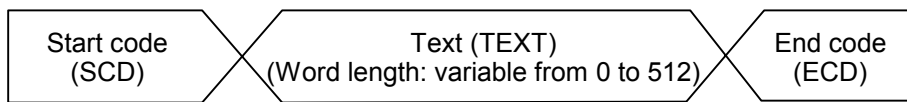
Text data starts with the data next to an SCD received if available and ends when an ECD is encountered or data has been received in a specified word length.

Hence, a text word length, and an SCD and an ECD can be specified to allow various forms of blocks to be transmitted and received.

With the ASCII specification, the communications control program converts transmitted from binary to ASCII and converts received data from ASCII to binary.

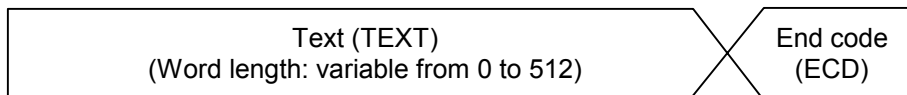
With the text variable length specification, a receive timeout period specification is mandatory.

- If an SCD and an ECD are available



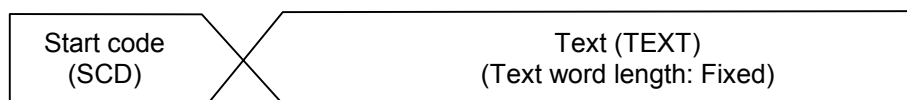
In the case above, if users set an ECD in the text even though the text length is 512, the communications control program will stop transmitting or receiving the text as soon as the ECD is encountered. When an ECD does not exist, the communications control program assume a text length of 512, and an SCD before the text and an ECD after.

- If an ECD is available



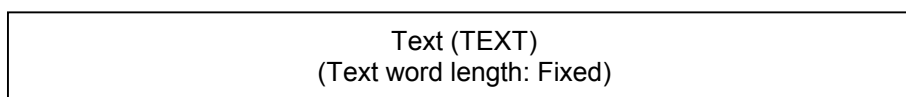
In this case as well, users may set an ECD in the text to let the communications control program to handle the text as being variable-length.

- If an SCD is available

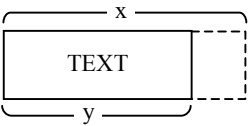
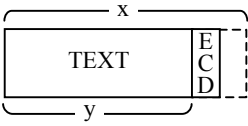
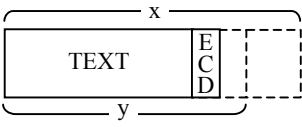
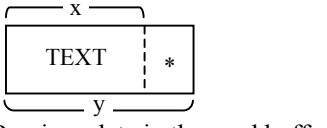
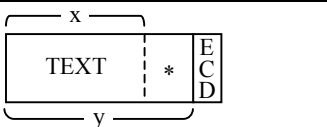
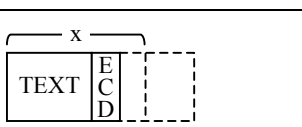


In this case, the text is fixed in the text word length specified.

- If only text is available

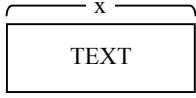
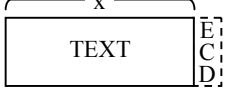
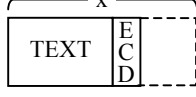


The table gives the relationship between the text word length specified by LGB and the word length transmitted by the send handler. The word length transmitted by the send handler is assumed x bytes, and the text word length specified by LGB is assumed y bytes.

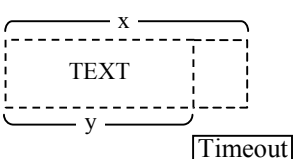
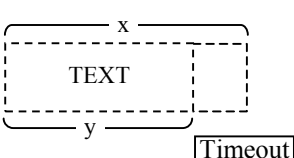
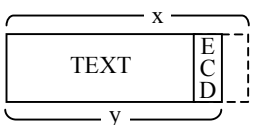
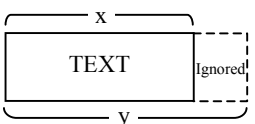
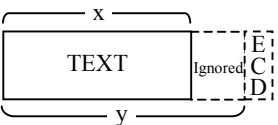
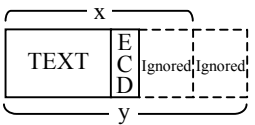
Size relation	LGB-specified end code	End code embedded in the text	Data transmitted over the line	
$x \geq y$	No	–		Text is transmitted in the LGB-specified word length.
	Yes	No		LGB-specified text + ECD (end code) are transmitted.
		Yes		Text is transmitted, from the beginning to the embedded ECD (end code).
$x < y$	No	–		Text + previous data in the send buffer are transmitted.
	Yes	No		Text + previous data in the send buffer + ECD (end code) are transmitted.
		Yes		Text is transmitted, from the beginning to the embedded ECD (end code).

## 4 OPERATION

With the text variable length specification, the following kinds of data are transmitted over the line depending on the word length transmitted by the send handler.

LGB-specified end code	End code embedded in the text	Data transmitted over the line	
No	-		Text is transmitted in the transmitted word length specified by the send handler.
Yes	No		Text is transmitted in the transmitted word length specified by the send handler, + ECD (end code).
	Yes		Text is transmitted, from the beginning to the embedded ECD (end code).

The table gives the relationship between the text word length specified by LGB and the word length incorporated by the receive handler. The text word length specified by LGB is assumed x bytes, and the data word length from the line is assumed y bytes.

Size relation	LGB-specified end code	End code embedded in the text	Data stored in the received data buffer	
$x > y$	No	-		A text timeout occurs while waiting for received data until the text word length specified by LGB is reached. When a text variable length is set, however, the text data that has been received prior to a timeout is stored in the received data buffer.
	Yes	No		
	Yes	Yes		Text is received, from the beginning to the embedded ECD (end code).
$x \leq y$	No	-		Text is received only in the word length specified by LGB, with extra data being ignored.
	Yes	No		
	Yes	Yes		Text is received, from the beginning to the embedded ECD (end code).



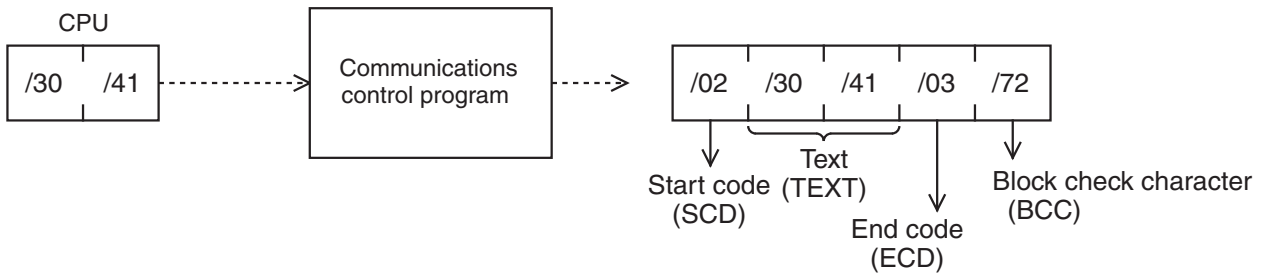
## 4 OPERATION

The table below gives the relationship between the receive buffer for the receive handler and received data.

The word length incorporated into the receive buffer is assumed x bytes, and the word length stored in the receive buffer is assumed y bytes.

Size relation	Data incorporated into a user-specified area
$x \geq y$	<div style="display: flex; align-items: center;"> <div style="margin-right: 20px;"> <p>(Receive buffer)</p> <div style="border: 1px solid black; width: 80px; height: 20px; margin: 5px auto; text-align: center;">Received data</div> </div> <div style="margin-right: 20px;"> <p>Received data area</p> <div style="border: 1px solid black; width: 100px; height: 20px; margin: 5px auto; display: flex; justify-content: space-between; align-items: center;"> <span style="width: 80px;"></span> <span>0—0</span> </div> </div> <div style="margin-left: 20px;"> <p>Cleared to 0</p> </div> </div>
$x < y$	<div style="display: flex; align-items: center;"> <div style="margin-right: 20px;"> <p>(Receive buffer)</p> <div style="border: 1px solid black; width: 80px; height: 20px; margin: 5px auto; text-align: center;">Received data</div> </div> <div style="margin-right: 20px;"> <p>Received data area</p> <div style="border: 1px solid black; width: 60px; height: 20px; margin: 5px auto;"></div> </div> <div style="margin-left: 20px;"> <p>Uncollected data is ignored for the computing function handler. For the ask handler, see “5.3.2 Subroutines.”</p> </div> </div>

The communications control program attaches a start code (SCD), an end code (ECD), and a block check character (BCC) code to the data to be transmitted according to LGB table specifications.



The example shown above assumes:

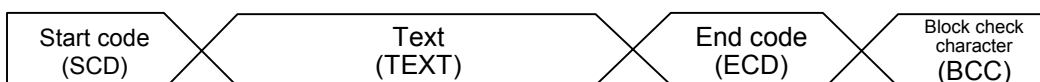
Start code: One character specified /02 (STX: Start of Text)

End code: One character specified /03 (ETX: End of Text)

Block check character: Horizontal even parity

It is necessary, therefore, to create a remote station program to meet these LGB table specifications.

A typical block structure is shown below.



### ■ Start code (SCD)

Signifies the beginning of text. “Available” or “Unavailable,” and, if “Available” is specified, a code length (1 to 4 characters) and code data can be set.

Tool selection item			Start code content	Default	
No.	Menu display	Start code display			
1	No start code	No start code	No start code	√	
2	One start code	CD1	One start code		CD=/02(STX)
3	Two start codes	CD1+CD2	Two start codes		
4	Three start codes	CD1+CD2+CD3	Three start codes		
5	Four start codes	CD1+CD2+CD3+CD4	Four start codes		

CD1 to 4: Hexadecimal values that denote start codes /00 to /FF

- If an SCD is available, the communications control program recognizes an incoming SCD as a sign of starting reception from the remote station, ignoring all data that had been received earlier.

When the communications control program transmits text to the remote station, it prefixes the text with an SCD code.

The SCD is not converted to ASCII even when ASCII is specified.

### ■ End code (ECD)

Signifies the end of text. Available or unavailable, and, if available is specified, a code length (1 to 4 characters) and code data can be set.

Tool selection item			End code content	Default	
No.	Menu display	End code display			
1	No end code	No start code	No end code	√	
2	One end code	CD1	One end code		CD=/03(STX)
3	Two end codes	CD1+CD2	Two end codes		
4	Three end codes	CD1+CD2+CD3	Three end codes		
5	Four end codes	CD1+CD2+CD3+CD4	Four end codes		

CD1 to 4: Hexadecimal values that denote end codes /00 to /FF

## 4 OPERATION

- If an ECD is available, the communications control program recognizes an incoming ECD as a sign of ending reception from the remote station.

When the communications control program transmits text to the remote station, it postfixes the text with the specified ECD code.

The ECD is not converted to ASCII even when ASCII is specified.

### ■ Block check character (BCC)

Used to check a transmitted or received frame for validity. If an ECD is available, a BCC exists next to the ECD; if an ECD is Unavailable, it exists next to the text.

Tool selection item		BCC content	Default
No.	Menu display		
1	No BCC	No BCC	√
2	Even parity check	Horizontal even parity check	
3	Odd parity check	Horizontal odd parity check	

As for the BCC check, “Available” or “Unavailable,” and, if “Available” is specified, horizontal even or horizontal odd parity can be specified.

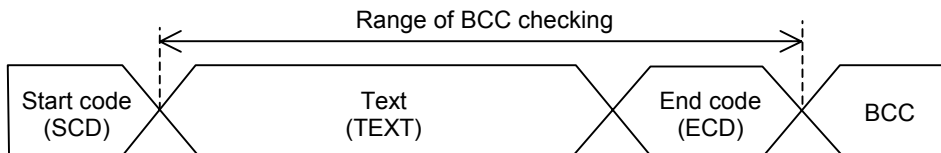
$$\text{Horizontal even parity} \cdots (\text{BCC})_E = (/00) \text{ EOR } (\sum_{i=0}^n \text{EOR } D_i)$$

$$\text{Horizontal odd parity} \cdots (\text{BCC})_{07} = (/7F) \text{ EOR } (\sum_{i=0}^n \text{EOR } D_i) \quad (7 \text{ data bits})$$

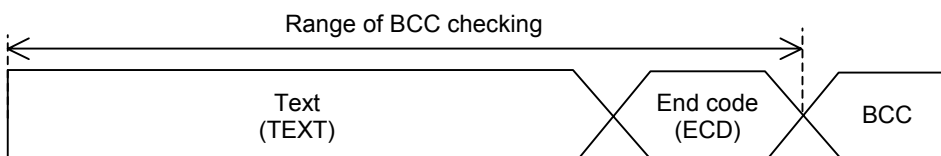
$$(\text{BCC})_{08} = (/FF) \text{ EOR } (\sum_{i=0}^n \text{EOR } D_i) \quad (8 \text{ data bits})$$

The possible ranges of BCC checking are shown below.

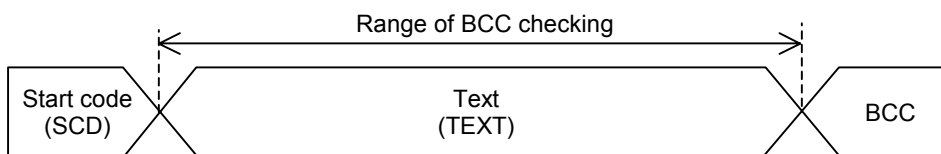
- If an SCD and an ECD are available



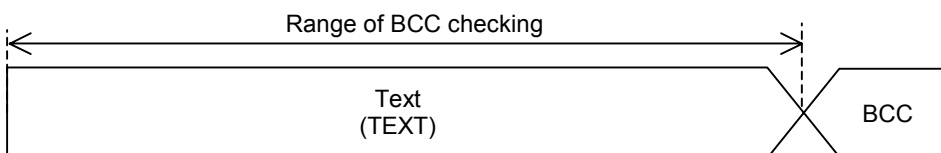
- If an ECD is available



- If an SCD is available



- If only text is available



If an ECD is Unavailable, the text is checked by assuming that it has a fixed length of data as specified.

Create a remote station program to meet these rules if a BCC checking is available.

With the ASCII specification, the text (binary data) before its ASCII conversion and the ECD are checked.

## 4 OPERATION

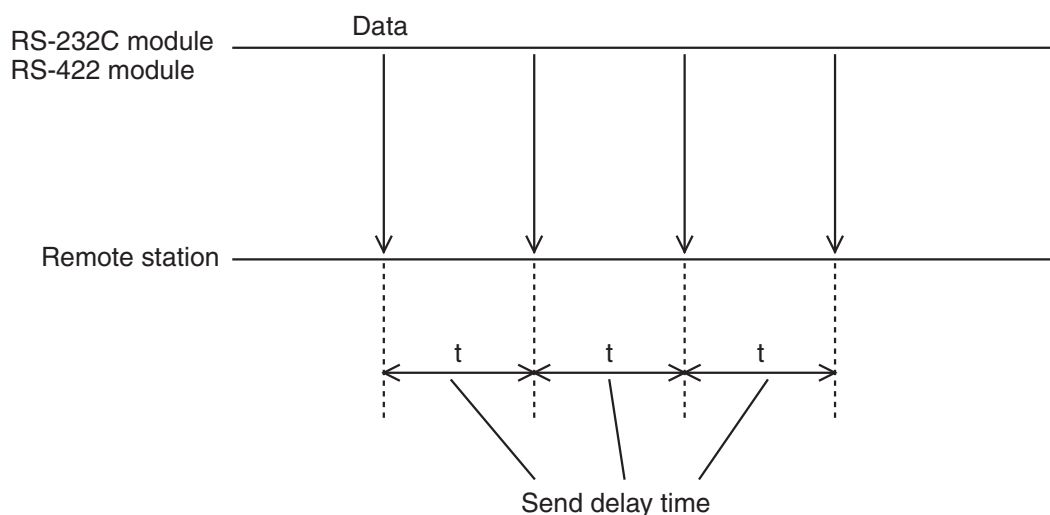
### ■ Send delay time

When the communications control program transmits data to the remote station, it specifies the time interval between two successive instances of data transmission.

Tool setting		Send delay time content	Default
Setting	Display		
0	No data send delay time	No data send delay time	√
1 to 32767	00001 to 32767 [ms]	1 to 32767 [ms]	

### Limitations

Baud rate	Send delay time setting range
300 [BPS]	64 to 32767 [ms]
600 [BPS]	32 to 32767 [ms]
1200 [BPS]	16 to 32767 [ms]
2400 [BPS]	8 to 32767 [ms]
4800 [BPS]	4 to 32767 [ms]
9600 [BPS]	2 to 32767 [ms]
19200 [BPS]	1 to 32767 [ms]



### ■ Send break and continue codes

Use the send break and continue codes to allow the remote station to request the ongoing transmission of text from the communications control program to break or continue (for reasons such as inability to handle the incoming data).

Tool selection item			Break and continue code content	Default
No.	Menu display	Break and continue code display		
1	No break and continue code	No break and continue code	No break and continue code	√
2	One break code, one continue code	BR: CD1      CN: CD2	One break code, one continue code	
3	One break code, two continue codes	BR: CD1      CN: CD2+CD3	One break code, two continue codes	
4	Two break codes, one continue codes	BR: CD1+CD2    CN: CD3	Two break codes, one continue codes	
5	Two break codes, two continue codes	BR: CD1+CD2    CN: CD3+CD4	Two break codes, two continue codes	

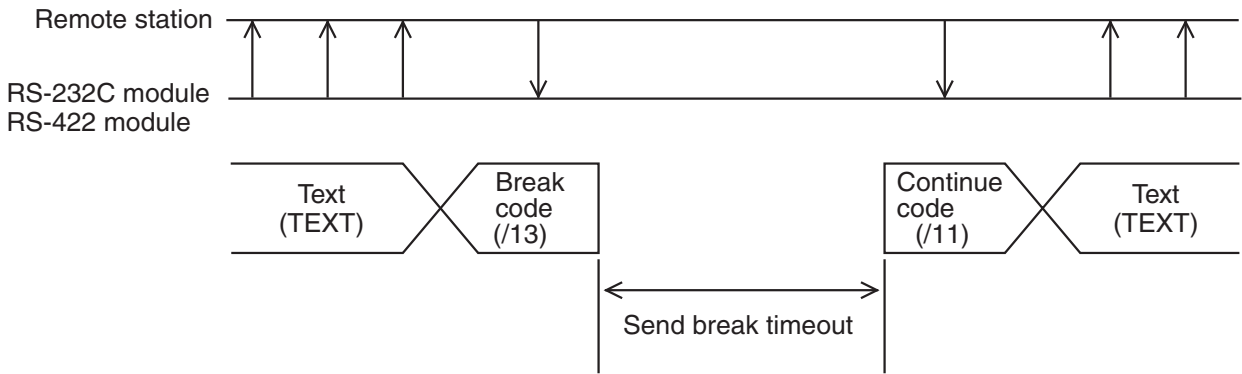
BR: Break code      CN: Continue code

CD1 to 4: Hexadecimal values that denote send break and continue codes /00 to /FF

“Available” or “Unavailable” for send break and continue handling and, if “Available” is specified, break codes (1 to 2 characters) and continue codes (1 to 2 characters) can be specified. Once the communications control program receives a break code, it receives only a continue code and ignores any other incoming code. Both break and continue codes are used without conversion even if they are in ASCII.

A send break timeout refers to the period of time that expires before the communications control program receives a continue code after receiving a break code. An error would occur if this period is exceeded.

## 4 OPERATION



The example shown above assumes:

Break code: One character specified /13 (CD3: Device control 3 [X-OFF])

Continue code: One character specified /11 (CD1: Device control 1 [X-ON])

Because the communications control program monitors independently the send break timeout and the receive timeout defined later, an error would still occur if the receive monitoring time is exceeded even while transmission has been interrupted.

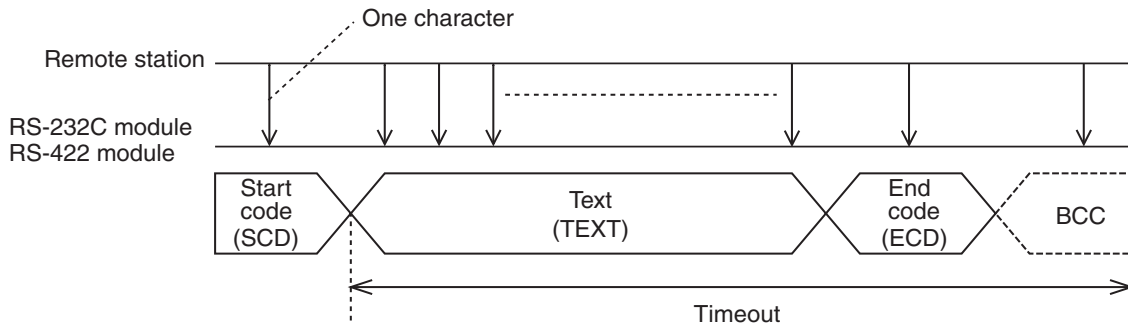
### ■ Send break timeout

Tool setting		Send break timeout content	Default
Setting	Display		
0	No text send break timeout monitoring	No text send break timeout monitoring	32767
1 to 32767	00001 to 32767 [100 ms]	0.1 to 3276.7 [s]	

### ■ Receive timeout

Define the period of time spent by the communications control program receiving an entire set of text data from its beginning.

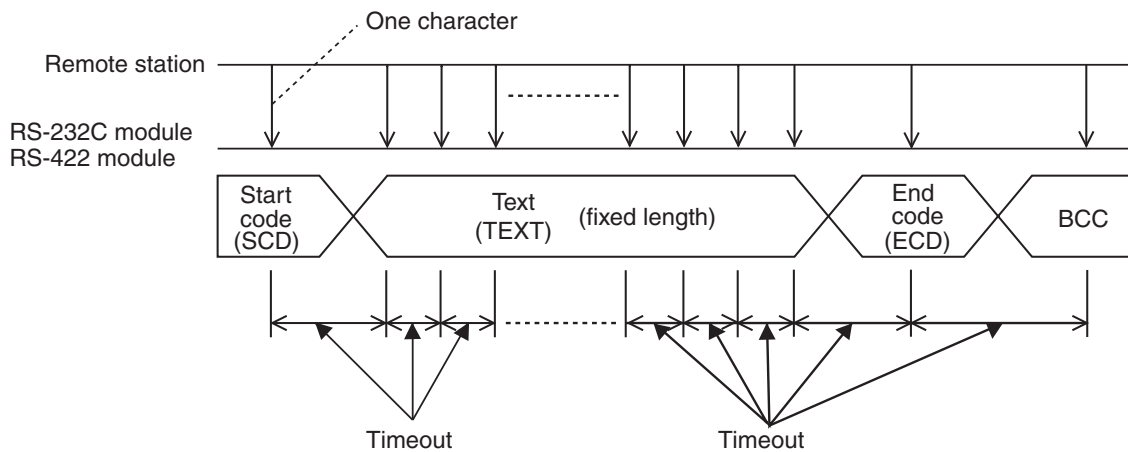
Tool setting		Receive timeout setting	Default
Setting	Display		
0	No text receive timeout monitoring	No text receive timeout monitoring	32767
1 to 32767	00001 to 32767	0.1 to 3276.7 [s]	



If a text variable length has been set

The period of time that expires between two characters is set as a receive timeout.

If the next batch of data is not received during this period, the communications control program assumes the end of data and turns on the receive-complete flag.



■ RS-422 gate control

RS-422 gate control is not used. Its setting would be ignored.



## 4 OPERATION

---

### ■ Request to Send (RS)

Specify Request to Send (RS pin status) available/unavailable output to the remote station.

The RS-232C module can transmit send data only if “RS available” is specified.

This setting is invalid on the RS-422 module.

Tool selection item		RS content	Default
No.	Display		
1	RS available output	RS available output	√
2	RS unavailable output	RS unavailable output	

#### ● RS available

The RS-232C module continues transmitting constant RS available output to the remote station while becoming ready to transmit.

#### ● RS unavailable

The RS-232C module continues transmitting constant RS unavailable output to the remote station while becoming not ready to transmit.

If send data is transmitted when “RS available” is specified, the transmitted data would be kept from being transmitted and the ready-to-send flag in the CPU system registers (see “5.2.1 Transmitted information”) would remain to indicate “Now transmitting.”

○ Set “RS available” when transmitting data to the remote station.

○ Set “RS unavailable” when not transmitting data to the remote station.

If the remote station supports a ready-to-receive/not-ready-to-receive switching function, connect the RS pin of the RS-232C module and the ready-to-receive/not-ready-to-receive detection pin of the remote station (typically, the CD pin) to prevent false reception of invalid data (such as noise) other than transmitted data.

### ■ Equipment Ready (ER)

Specify RS-232C module ready or not ready output to the remote station.

Ready and not ready definitions are determined by the protocol between the RS-232C module and the remote station. Generally, the state of the RS-232C module being ready to receive is defined as being “ready.”

This setting is inbvide on the RS-422 module.

Tool selection item		ER content	Default
No.	Display		
1	Not Ready output	Not Ready output	
2	Ready output	Ready output	√

- If ready

The RS-232C module continues transmitting the Ready state to the remote station via the Equipment Ready pin.

- If not ready

The RS-232C module continues transmitting the Not Ready state to the remote station via the Equipment Ready pin.

- Set “Ready” when receiving data form the remote station.

- Set “Not Ready” when not receiving data from the remote station.

If the remote station supports a ready-to-send/not-ready-to-send switching function, connect the ER pin of the RS-232C module and the ready-to-send/not-ready-to-send detection pint of the remote station (typically, the DR or CS pin) to control the remote station into the ready or not ready to transmit state.

#### ■ Data Set Ready (DR)

Specify whether to check the remote station for its ready state (DR pin status).

This setting is invalid on the RS-422 module.

Tool selection item		Data set ready content	Default
No.	Display		
1	Checking unavailable	Checking unavailable	√
2	Checking available	Checking available	

- If checking is available

The communications control program checks the remote station for its ready state (DR pin status) and transmits data only when it is ready. Transmitting data while the remote station is not ready would invoke an error.

## 4 OPERATION

---

- If checking unavailable

The communications control program transmits data to the remote station without checking it for its ready state (DR pin status).

- If the remote station supports a ready to receive output function

Connect the DR pin of the RS-232C module and the ready-to-receive output pin of the remote station (typically, the ER pin) to enable checking.

- If the remote station does not support a ready to receive output function

Set to disable checking.

- Control signal automatic control

Specify whether to carry out control signal input checking and output control automatically or manually.

This setting is invalid on the RS-422 module.

Tool selection item		Setting content	Default
No.	Display		
1	Manual setting	Control signal manual setting	√
2	Automatic control	Control signal automatic control	

- Manual setting

The communications control program behaves as directed by the settings of Request to Send (RS), Equipment Ready (ER), and Data Set Ready (DR).

- Automatic control

“RS available” is transmitted only at transmitted data transmission.

Equipment Ready (ER) turns to ready after startup and turns to not ready when a hardware error occurs. Equipment Ready (ER), Carrier Detect (CD), and Clear to Send (CS) are all set to enable input checking.

### ■ System selection

Use the module switch to select a computing function system or a task system.

Register a computing function or set a task.

#### ● Computing function

Register a computing function to transmit data to and from the remote station.

Name	Function
SD0	Channel No. 0 send computing function
SD1	Channel No. 1 send computing function
SD2	Channel No. 2 send computing function
SD3	Channel No. 3 send computing function
RV0	Channel No. 0 receive computing function
RV1	Channel No. 1 receive computing function
RV2	Channel No. 2 receive computing function
RV3	Channel No. 3 receive computing function

#### ● Task system

Specify a user task to be started from the OS on the CPU when the RS-323C module has received data.

Item	Setting range	Remarks
Started task number	0 to 127	Integer input
Start cause	0 to 16	Integer input

## 4 OPERATION

---

### 4.4 Host Interrupts

If the H-7338 protocol is set in the RS-422 module (LQE165), interrupts can be raised in the host computer.

#### 4.4.1 Host interrupt register

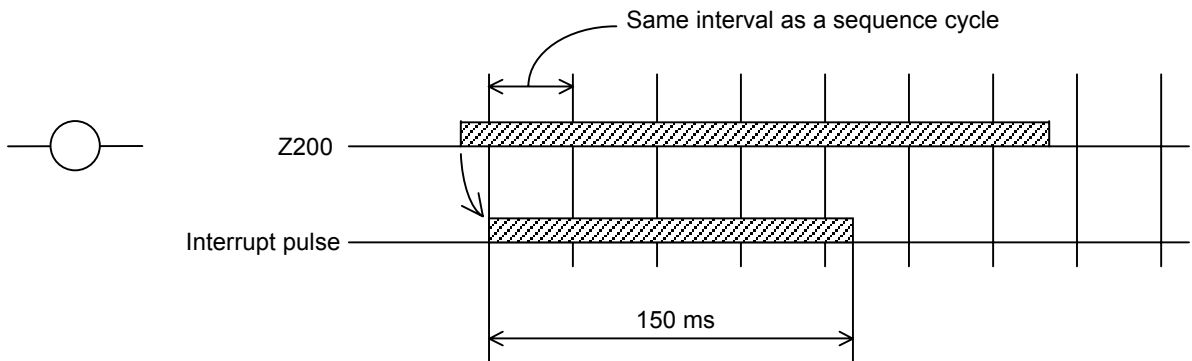
Whenever the start of a Z-coil is detected, a 150 ms interrupt is raised in the host computer. Though this routine is carried out at the same interval as a sequence cycle, it is not synchronized with a sequence cycle. Table below lists the Z-coils associated with channel numbers.

Channel number	Host interrupt register
Channel 0	Z200
Channel 1	Z201
Channel 2	Z202
Channel 3	Z203

An example of raising an interrupt in the host computer where the RS-422 module (LQE565) is used on channel 0 is shown below.

Settling pulse width: Minimum one sequence cycle

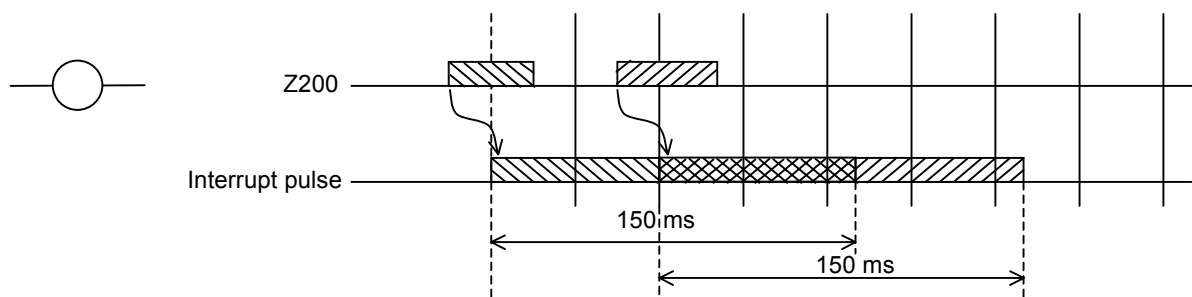
Timing chart



[Stretched interrupt pulse width]

If Z200 makes two times or more on to off transitions within a period of 150 ms, double interrupts would arise, resulting a stretched interrupt pulse width.

Timing chart

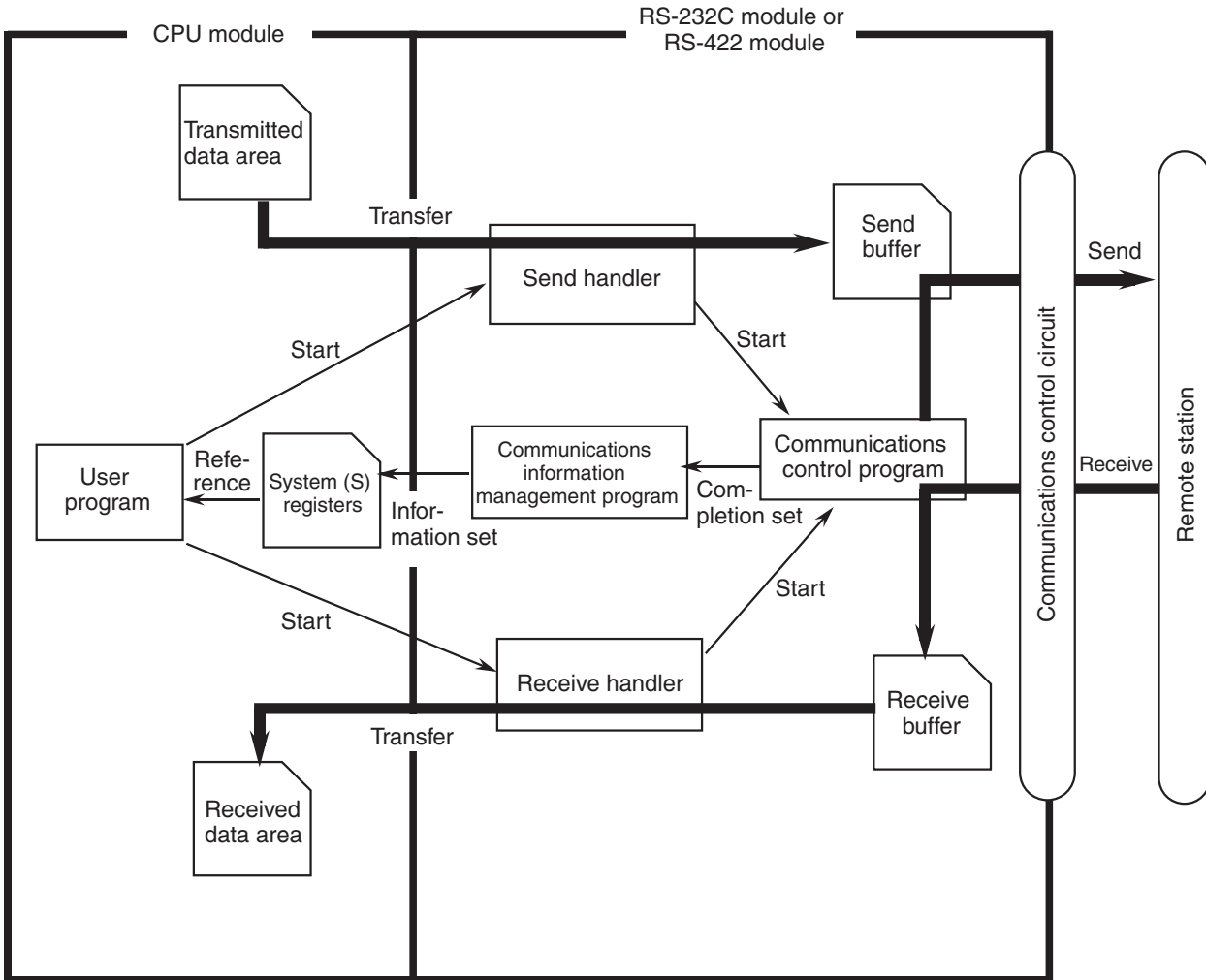


To prevent this, implement an interlock with the host that receives.

# 5 PROGRAMMING

5.1 Software Configuration

Communication with an external station is carried out in the following way:





- Communications control program

On receiving a transmission request from the send handler, the communications control program transmits send data over the line. When it has finished receiving all data from the line, it notifies the communications information control program of the completion of reception.

- Communications information management program

The communications information management program loads information on transmission and reception by the communications control program into the CPU system (S) registers to notify the user program. (See “5.2.1 Transmitted information,” and “5.2.2 Received information.”)

- Send handler

The send handler provides the following functions:

- Transfer transmitted data in a specified word length from the transmitted data area specified by the user program to the send buffer in the RS-232C module.
- Issue a transmissions start request to the communications control program.
- Make various error checks on launching transmission

Use the user program to launch the send handler at the desired timing of transmission.

- Receive handler

The receive handler provides the following functions:

- If yet-to-be-incorporated received data is available in the receive buffer, transfer data in a specified word length to the received data area specified by the user program.
- Notify the communications control program that the receive buffer has now been emptied.
- Make various error checks on launching reception.

The user program must recognize the end of reception before launching the receive handler to incorporate the data. (For information on recognizing the end of reception, see “5.2.2 Received information.”)

- System registers

Contain information, such as whether transmission enabled, reception complete, and errors encountered. Create a user program by referencing this information.

## 5 PROGRAMMING

### 5.2 System Registers

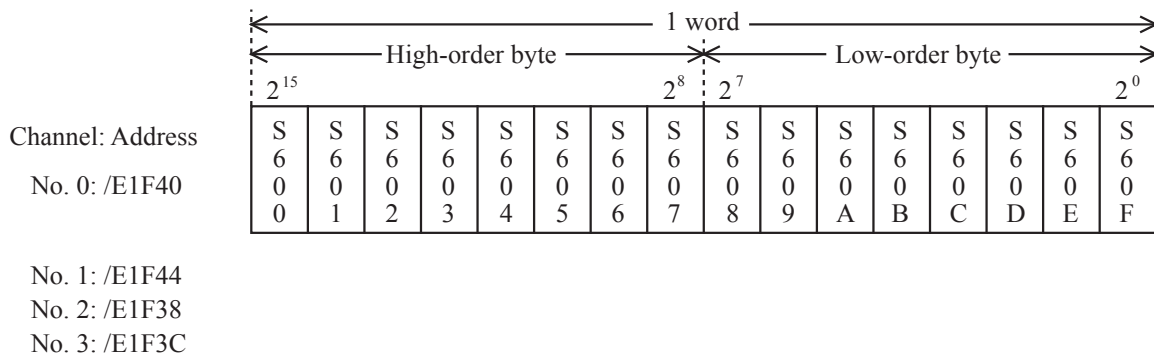
#### 5.2.1 Transmitted information

While the send handler executes transmission from the CPU to a remote station, transmission information is loaded into the CPU system (S) registers.

The user program references the S-registers to determine whether transmission is enabled or disabled, and identify send errors encountered.

Channel				Explanation	Bit definition	
No.0	No.1	No.2	No.3		0	1
S600	S620	S5C0	S5E0	Send-enabled flag	Transmission enabled	Transmitting now
S601	S621	S5C1	S5E1	Handler error flag	No error	Error encountered
S602	S622	S5C2	S5E2	Communications control program error	No error	Error encountered
S603	S623	S5C3	S5E3	Reception canceled transmission	No error	Error encountered
S604	S624	S5C4	S5E4	Not used	Not used	
S605	S625	S5C5	S5E5			
S606	S626	S5C6	S5E6			
S607	S627	S5C7	S5E7			
S608	S628	S5C8	S5E8	Error detail code	Coded representations of the handler and communications control errors See the low-order byte of each error code appearing in "7.3.3 Transmit errors."	
S609	S629	S5C9	S5E9			
S60A	S62A	S5CA	S5EA			
S60B	S62B	S5CB	S5EB			
S60C	S62C	S5CC	S5EC			
S60D	S62D	S5CD	S5ED			
S60E	S62E	S5CE	S5EE			
S60F	S62F	S5CF	S5EF			

The S-registers are initialized to 0 on a CPU reset. The S-registers can also be read from the CPU as word data.

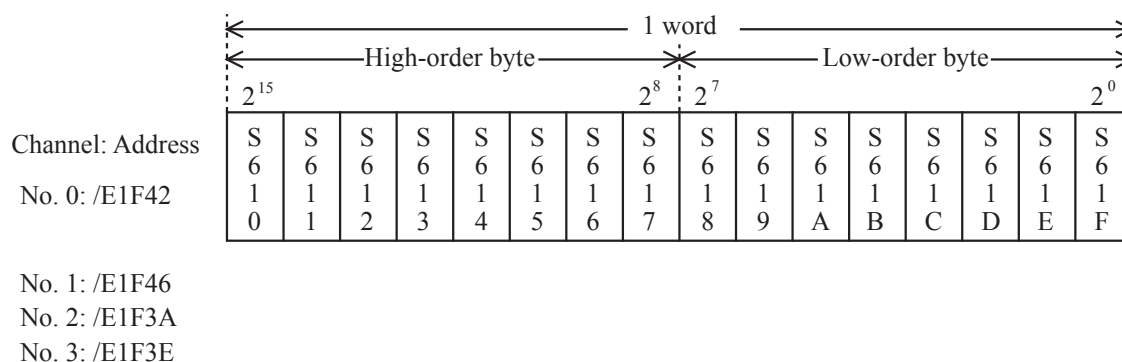


## 5.2.2 Received information

While the communications control program executes reception by a remote station from the CPU, successful or unsuccessful reception information is loaded into the CPU system (S) registers. The user program references the S-registers to determine whether received data is available and identify receive errors encountered.


Channel				Explanation	Bit definition	
No.0	No.1	No.2	No.3		0	1
S610	S630	S5D0	S5F0	Receive-complete flag	Received data unavailable	Received data available
S611	S631	S5D1	S5F1	Handler error flag	No error	Error encountered
S612	S632	S5D2	S5F2	Communications control program error	No error	Error encountered
S613	S633	S5D3	S5F3	System error	No error	Error encountered
S614	S634	S5D4	S5F4	Handler error code	Handler error description See a high-order bit of the low-order byte of each error code appearing in "7.3.4 Receive errors."	
S615	S635	S5D5	S5F5			
S616	S636	S5D6	S5F6			
S617	S637	S5D7	S5F7			
S618	S638	S5D8	S5F8	Error detail code	Coded representations of the handler and communications control errors See a low-order bit of each error code appearing in "7.3.4 Receive errors."	
S619	S639	S5D9	S5F9			
S61A	S63A	S5DA	S5FA			
S61B	S63B	S5DB	S5FB			
S61C	S63C	S5DC	S5FC			
S61D	S63D	S5DD	S5FD			
S61E	S63E	S5DE	S5FE			
S61F	S63F	S5DF	S5FF			

The S-registers are initialized to 0 on a CPU reset. The S-registers can also be read from the CPU as word data.



**5.3 Send/Receive Handlers**

Computing functions are used as send/receive handlers when the user program to be invoked is a ladder, or subroutines are used as such when it is a C-mode task.

 <b>PROHIBITION</b>
<p>It is not permitted to define a computing function on channel No. 0 and a subroutine on channel No. 1 or use subroutines for transmission and computing functions for reception.</p> <p>All send/receive handlers associated with a given CPU unit must be uniformly defined as computing functions or subroutines.</p>

**5.3.1 Computing functions**

Eight transmit/receive computing functions are available as listed below.

Name	Function
SD0	Channel No. 0 send computing function
SD1	Channel No. 1 send computing function
SD2	Channel No. 2 send computing function
SD3	Channel No. 3 send computing function
RV0	Channel No. 0 receive computing function
RV1	Channel No. 1 receive computing function
RV2	Channel No. 2 receive computing function
RV3	Channel No. 3 receive computing function

Note: When using computing functions to build a ladder program, be sure to register them in the LGB table as instructed “4.2 Editing the LGB Table” and then select [Build] – [Receive] from the ladder program edit window in ladder chart system to receive the CPU data with the tool. This will allow the computing functions to be assembled into a ladder program.

SD0, SD1, SD2, SD3

Send computing functions

Function	Transmit data in a specified word length from a parameter-specified area to the remote station.
Parameters	Transfer address: Mnemonic, such as XW000 and FW000 Transfer word length: 1 to 512
Return code	A return code is loaded in a system register. (See “5.2.1 Transmitted information.”)
Sample program	Transmits 32 bytes (hexadecimal: /20) from FW000 to the remote station attached to channel No. 0 when input condition S600 is off.



## 5 PROGRAMMING

RV0, RV1, RV2, RV3

Receive computing functions

Function

Transfer data in a specified word length to a parameter-specified area. This function performs no action when no received data is available.

Parameters

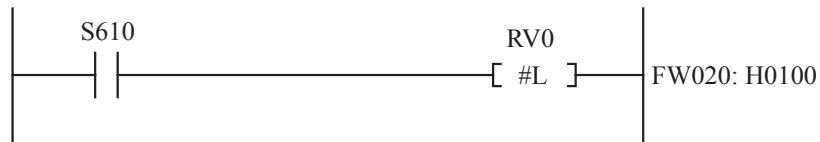
Transfer address: Mnemonic, such as XW000 and FW000  
Transfer word length: 1 to 512

Return code

A return code is loaded in a system register.  
(See “5.2.2 Received information.”)

Sample program

Transfers 256 bytes (hexadecimal: /100) from FW020 to the remote station attached to channel No. 0 when input condition S610 is on.



Remarks

- A receive handler incorporates the oldest received data. It keeps system register S610 from turning off as long as yet-to-be-incorporated received data is available. It turns off system register S610 when yet-to-be-incorporated received data is no longer available.
- A receive computing function can read one block of received data (as stored in one receive buffer) only in a batch (and not in segments).

(Example)

Receive data	Incorporated word length setting on receive handler launch	Incorporated data
<div style="border: 1px solid black; padding: 2px; display: inline-block;">‘ABCDEF0123’</div> 10 characters received	7	<div style="border: 1px solid black; padding: 2px; display: inline-block;">‘ABCDEF0’</div>

The receive handler, when relaunched after incorporating seven characters, ignores yet-to-be-incorporated “123” and proceeds to incorporate the next block of received data (data in the next receive buffer).

- If the incorporated word length is set larger than the actual block, the incorporated data is padded with an end code. If space still remains unfilled, /00 is written to the remaining space.

(Example)

Receive data	End code setting	Incorporated word length setting on receive handler launch	Incorporated data
<div style="border: 1px solid black; padding: 2px; display: inline-block;">‘ABCD’</div> 4 characters received	/03	8	<div style="border: 1px solid black; padding: 2px; display: inline-block;">‘ABCD’ /03000000</div>

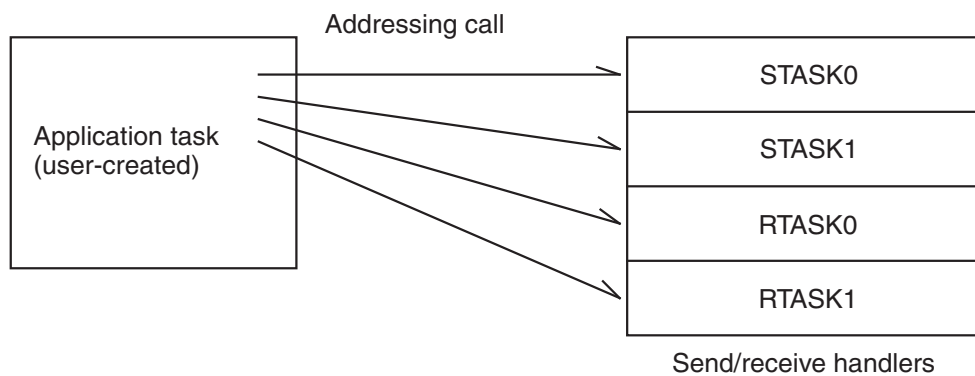
### 5.3.2 Subroutines

Eight send/receive handlers for user-created tasks (application tasks) are available as listed below.

Name	Address	Function
STASK0	/107000	Channel No. 0 send subroutine
STASK1	/107006	Channel No. 1 send subroutine
STASK2	/107018	Channel No. 2 send subroutine
STASK3	/10701E	Channel No. 3 send subroutine
RTASK0	/10700C	Channel No. 0 receive subroutine
RTASK1	/107012	Channel No. 1 receive subroutine
RTASK2	/107024	Channel No. 2 receive subroutine
RTASK3	/10702A	Channel No. 3 receive subroutine

User-created tasks (application tasks) are created in the C language or in a 68000 assembler language.

Because send/receive handlers for application tasks are called by addressing, the application tasks cannot be created (linked) to include send/receive handlers.



## 5 PROGRAMMING

STASK0, STASK1, STASK2, STASK3

Send subroutines

<div style="border: 1px solid black; padding: 2px; text-align: center;">Function</div>	<p>Transmit data in a specified word length from a parameter-specified area to the remote station.</p>				
<div style="border: 1px solid black; padding: 2px; text-align: center;">Link procedure</div> <p>(Example) Channel No. 0</p>	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">C language</th> <th style="width: 50%; text-align: center;">Assembler language</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px; vertical-align: top;"> <pre> long (*stask0)() ; long rtn, sadr, sbyte ;  stask0 = (long(*)()) 0x107000 ;  rtn = (*stask0)(sadr, sbyte) ;                     </pre> </td> <td style="padding: 5px; vertical-align: top;"> <pre> move.l    #sbyte, -(A7) move.l    #sadr, -(A7) lea       \$107000, A0 jsr       (A0) addq.l    #8, A7                     </pre> </td> </tr> </tbody> </table>	C language	Assembler language	<pre> long (*stask0)() ; long rtn, sadr, sbyte ;  stask0 = (long(*)()) 0x107000 ;  rtn = (*stask0)(sadr, sbyte) ;                     </pre>	<pre> move.l    #sbyte, -(A7) move.l    #sadr, -(A7) lea       \$107000, A0 jsr       (A0) addq.l    #8, A7                     </pre>
C language	Assembler language				
<pre> long (*stask0)() ; long rtn, sadr, sbyte ;  stask0 = (long(*)()) 0x107000 ;  rtn = (*stask0)(sadr, sbyte) ;                     </pre>	<pre> move.l    #sbyte, -(A7) move.l    #sadr, -(A7) lea       \$107000, A0 jsr       (A0) addq.l    #8, A7                     </pre>				
	<div style="border: 1px solid black; padding: 5px;"> <p>With an assembler language, the contents of registers other than D0 (return code storage) are guaranteed. (The C language allows users to be unconscious of registers.)</p> </div>				
<div style="border: 1px solid black; padding: 2px; text-align: center;">Parameters</div>	<p>sadr: Transmitted data storage address  sbyte: Transmitted byte length  rtn: Return code</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p>With an assembler language, the return code is loaded in the D0 register.</p> </div>				
<div style="border: 1px solid black; padding: 2px; text-align: center;">Return code</div>	<p>=0: Normal end  =/FFFFFFFF: Send handler launch error  Error information is loaded in a system register. (See “5.2.1 Transmitted information.”)</p>				
<div style="border: 1px solid black; padding: 2px; text-align: center;">Remarks</div>	<p>long (f) ( ); Declares function f that is returned as a function of a pointer to a double-precision integer.</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p>Calling a send subroutine while the module is not available would invoke an error.</p> </div>				



C language example
-----------------------

Check the most significant bit (send-enabled flag) of transmit system register S600 (address: /E1F40) and, if transmission is enabled, transmit 32 bytes (hexadecimal: /20) from the transmitted data area at address /140000.

```

:
register long (*stask0)( ) ;
register long rtn ;
:
if ( (*(short*)0xE1F40 & 0x8000) == 0 )
{
    stask0 = (long(*) ( )) 0x107000 ;
    rtn = (*stask0) (0x140000, 0x20) ;
    if ( rtn != 0 )
        goto errb ;
}
else
{
    :
}

```

Assembler language example
----------------------------------

Check the most significant bit (send-enabled flag) of transmit system register S600 (address: /E1F40) and, if transmission is enabled, transmit 256 bytes (hexadecimal: /100) from the transmitted data area at address /150000.

```

:
btst    #7, $E1F40
bne     LB1 → Go to LB1 if transmission is disabled
move.l  #$100, -(A7) → 256 bytes of data transferred
move.l  #$150000, -(A7) → Transmitted data area address /150000
lea     $107000, A0
jsr     (A0)
addq.l  #8, A7
tst.l   D0
bne     ERRB → Go to ERRB if an error occurs
:

```

## 5 PROGRAMMING

RTASK0, RTASK1, RTASK2, RTASK3

Receive subroutines

### Function

Transfer data in a specified word length from a parameter-specified area to the remote station. These subroutines perform no action when no received data is available. Receiver handlers (subroutines) incorporate the oldest received data.

### Link procedure

(Example)  
Channel No. 0

C language	Assembler language
long (*rtask0)();	
long rtn, radr, rbyte ;	move.l #rbyte, -(A7)
	move.l #radr, -(A7)
	lea \$10700C, A0
rtask0 = (long(*) ( ) ) 0x10700C ;	jsr (A0)
	addq.l #8, A7
rtn = (*rtask0)(radr, rbyte) ;	

With an assembler language, the contents of registers other than D0 (return code storage) are guaranteed. (The C language allows users to be unconscious of registers.)

### Parameters

radr: Transmitted data storage address  
rbyte: Transmitted byte length  
rtn: Return code

With an assembler language, the return code is loaded in the D0 register.

Return code
-------------

=0: Normal end

If there still is yet-to-be-incorporated data, the received data available bit in the system remains set to indicate received data available.

=1: No received data is available in the receive buffer.

=/001A0000: The last data of text is found in the buffer while incorporating received data from it, an end code has been encountered, or data has been incorporated in the text word length that has been specified by LGB.

=/001A00xx: Data with a receive error is found in the buffer while incorporating received data from it. The received data area is cleared to 0 from the data in error to the received byte length. xx in the return code denotes the error code in the low-order byte of the receive error code. (See “7.3.4 Receive errors.”) Error information is loaded in a system register. (See “5.2.2 Received information.”)

=/FFFFFFF: Receive handler start error

Error information is loaded in a system register. (See “5.2.2 Received information.”)

Calling a send subroutine while the module is not available would invoke an error.
--

## 5 PROGRAMMING

C language example

- Check the most significant bit (receive-complete flag) of receive system register S610 (address: /E1F42) and, if received data is available, transmit 20 bytes (hexadecimal: /14) from the received data buffer to the received data storage area at address /140000.

```

:
register long (*rtask0) ( );
register long rtn ;
:
if ( (*(short*)0xE1F42 & 0x8000) != 0 )
{
    rtask0 = (long(*) ( )) 0x10700C ;
    rtn = (*rtask0) (0x140000, 0x14) ;
    if ( rtn != 0 )
        goto errb ;
}
else
{
    :
}

```

- According to an application program, a receive C mode subroutine can read one block of received data (as stored in one receive buffer) in segments. (A receive computing function, on the other hand, can only read one block of received data in a batch.)

An example of the following data received from a remote station is shown below.

“1234567890” 10 characters received

Incorporated word length setting on receive handler launch	Return code	Incorporated data
3	0 (normal)	“123”
4	0 (normal)	“4567”
4	/001A0000 (normal, block end)	“890” 0

An EOF code (/001A) in the high-order word of the return code (long length) signifies the end of the block. If the low-order word is equal to 0, that block has been normally received; if not, it has been abnormally received. (Data before that had been normally received, though.) The code that designates abnormal reception is the same as in the low-order byte of the receive error code.

If the incorporated word length is set larger than the received word length, the incorporated data is padded with an end code. If space still remains unfilled, 0 is written to the remaining space.

An example of receiving the following data from a remote station is shown below.

“12345” 5 characters received, end code /030001

Incorporated word length setting on receive handler launch	Return code	Incorporated data
7	/001A0000	“12345” /0300
8	/001A0000	“12345” /030001
9	/001A0000	“12345” /03000100

Assembler  
language  
example

Check the most significant bit (receive-complete flag) of receive system register S610 (address: /E1F42) and, if received data is available, transmit 256 bytes (hexadecimal: /100) from the received data buffer to received data storage area at address /150000.

```

:
btst      #7, $E1F42
beq       LB1      → Go to LB1 if received data is available
move.l    #100, -(A7)
           → 256 bytes of data transferred
move.l    #150000, -(A7)
           → From data area address /150000
lea       $10700C, A0
jsr       (A0)
addq.l   #8, A7
tst.l    D0
bne      ERRB     → Go to ERRB if an error occurs
:

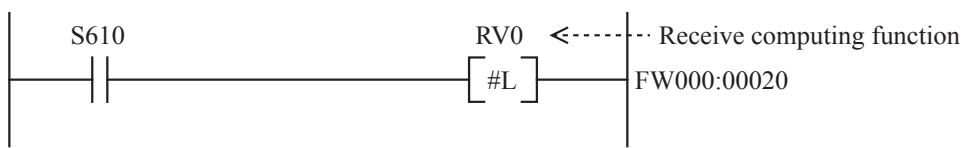
```

5.4 Incorporating Received Data

When the communications control program finished receiving data, relevant information is set in the system (S) registers. Create a user program by referencing this information.

● Creating as a ladder program

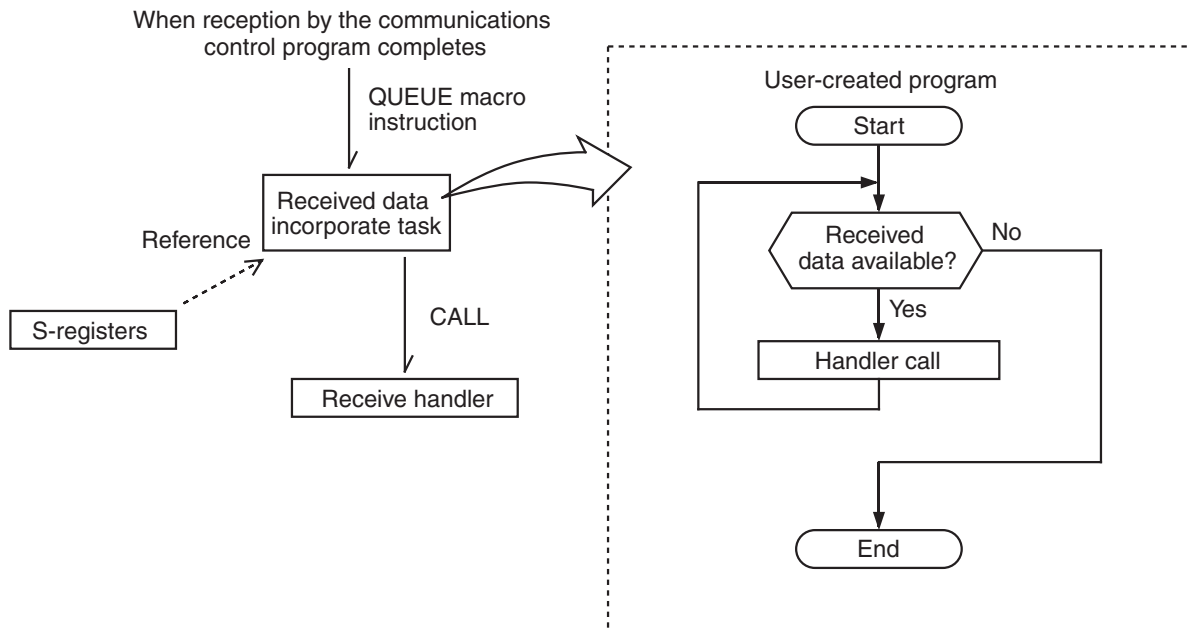
When a receive handler (computing function) is launched to meet the rules of the S-registers, the received data incorporate delay would be confined to within the sequence cycle (standard 30 ms).



● C mode program

Create and register the user task that the communications control program launches at the completion of reception.

This removes the need for the user program to monitor the completion of reception. Received data can be incorporated by simply making a subroutine call to a receive handler from a task invoked from the communications control program.



## 5.5 Hardware Controlled by Software Implementation

The following parameters can be set as send handler parameters to control the hardware of the RS-232C and RS-422 modules:

- Using computing functions  
Send handler name: SD? (? denotes a channel number)
- Using subroutines  
Send handler name: STASK? (? denotes a channel number)

Transfer address	Transferred word length	Content	Return information														
<ul style="list-style-type: none"> <li>• Computing function Data register DWFFF</li> <li>• C-mode subroutine Address /62FFE</li> </ul>	/8080	Software reset As the module in question is reset, other channels on the same module are reset as well (similar to a reset of the RS-232C/RS-422 module implemented by using the CPU reset switch)	A normal ending is evidenced by the display of a successful operation message on the CPU LED.  (See “7.3.1 CPU module indicator display messages.”)														
	/0000 or /0001	Latest hardware status incorporate request	<ul style="list-style-type: none"> <li>• Channel No. 0 High-order byte of DWFFF (/62FFE)</li> <li>• Channel No. 1 Low-order byte of DWFFF (/62FFF)</li> <li>• Channel No. 2 High-order byte of DWFFF (/62FFC)</li> <li>• Channel No. 3 Low-order byte of DWFFF (/62FFD)</li> </ul> <div style="text-align: center; margin: 10px 0;"> <math>2^7</math> <span style="float: right;"><math>2^0</math></span>  <table border="1" style="border-collapse: collapse; width: 100%; text-align: center;"> <tr> <td style="width: 15%;">①</td> <td style="width: 15%;">②</td> <td style="width: 15%;">③</td> <td style="width: 15%;">④</td> <td style="width: 15%;"></td> <td style="width: 15%;"></td> <td style="width: 15%;">⑤</td> </tr> </table> </div> <ul style="list-style-type: none"> <li>① RS output state</li> <li>② CS input state</li> <li>③ CD input state</li> <li>④ ER output state</li> <li>⑤ DR input state</li> </ul> <div style="margin-left: 150px;"> <table style="border: none;"> <tr> <td style="font-size: 2em;">}</td> <td style="padding-left: 10px;">0: ON(*)</td> </tr> <tr> <td style="font-size: 2em;">}</td> <td style="padding-left: 10px;">1: OFF(*)</td> </tr> <tr> <td style="font-size: 2em;">}</td> <td style="padding-left: 10px;">0: OFF(*)</td> </tr> <tr> <td style="font-size: 2em;">}</td> <td style="padding-left: 10px;">1: ON(*)</td> </tr> </table> </div>	①	②	③	④			⑤	}	0: ON(*)	}	1: OFF(*)	}	0: OFF(*)	}
①	②	③	④			⑤											
}	0: ON(*)																
}	1: OFF(*)																
}	0: OFF(*)																
}	1: ON(*)																

(\*) ON: Designates a high on the line.  
OFF: Designates a low on the line.

## 5 PROGRAMMING

Transfer address	Transferred word length	Content	Return information
<ul style="list-style-type: none"> <li>• Computing function Data register DWFFF</li> <li>• C-mode subroutine Address /62FFE</li> </ul>	/0200	DR OFF request (*)	<ul style="list-style-type: none"> <li>• Channel No. 0 High-order byte of DWFFF (/62FFE)</li> <li>• Channel No. 1 Low-order byte of DWFFF (/62FFF)</li> <li>• Channel No. 2 High-order byte of DWFFF (/62FFC)</li> <li>• Channel No. 3 Low-order byte of DWFFF (/62FFD)</li> <li>/00: OFF report</li> <li>/01: ON report</li> </ul>
	/0201	DR ON request (*)	
	/0300	RS OFF request (*)	<ul style="list-style-type: none"> <li>• Channel No. 0 High-order byte of DWFFF (/62FFE)</li> <li>• Channel No. 1 Low-order byte of DWFFF (/62FFF)</li> <li>• Channel No. 2 High-order byte of DWFFF (/62FFC)</li> <li>• Channel No. 3 Low-order byte of DWFFF (/62FFD)</li> <li>/00: ON report</li> <li>/01: OFF report</li> </ul>
	/0301	RS ON request (*)	

(\*) ON: Designates a high on the line.  
 OFF: Designates a low on the line.

With a send parameter address setting of DWFFF, /62FFE, functions or subroutines would return by directing return information to /FF if the parameter word length is other than those listed in Tables.



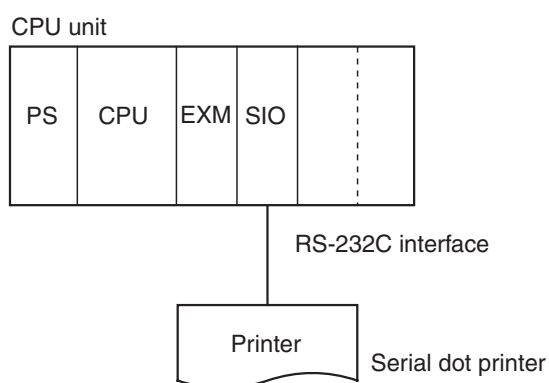
# 6 SAMPLE PROGRAMS

## 6.1 Sample RS-232C Wiring with a Printer

### 6.1.1 Overview

An RS-232C interface connecting the CPU unit to a serial dot printer allows memory data to be printed out in a specified word length, beginning with a specified address.

### 6.1.2 System configuration

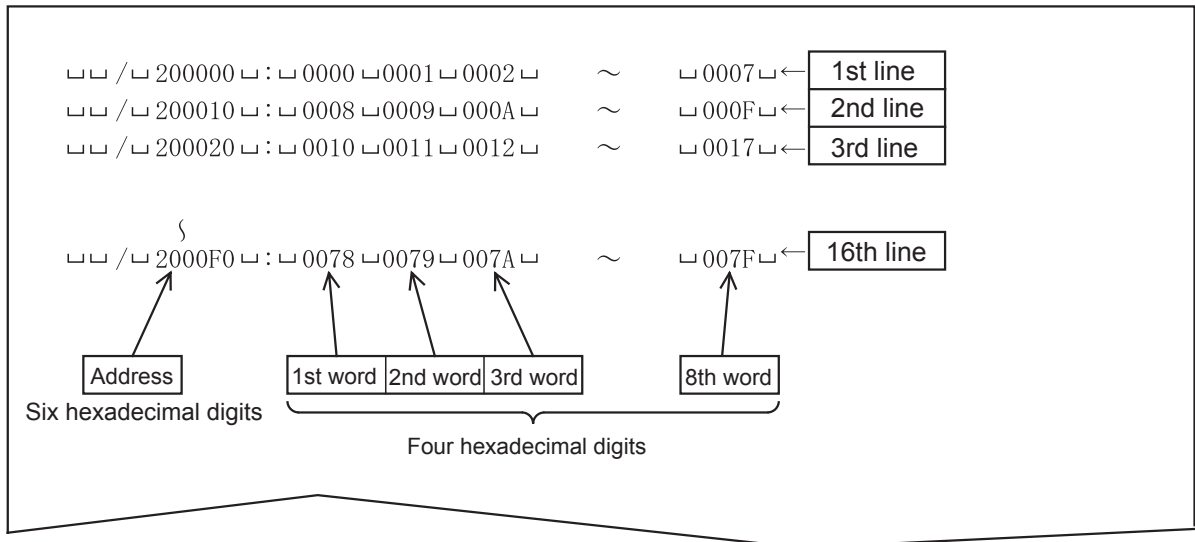


Equipment configuration of CPU unit

Abbreviation	Name	Type	Quantity
PS	Power supply module	LQV000	1
CPU	CPU module	LQP000	1
EXM	Extension memory module	LQM000	1
SIO	RS-232C module	LQE160	1
	CPU mount base	HSC-1080	1

## 6.1.3 Print format

The print format is shown below.

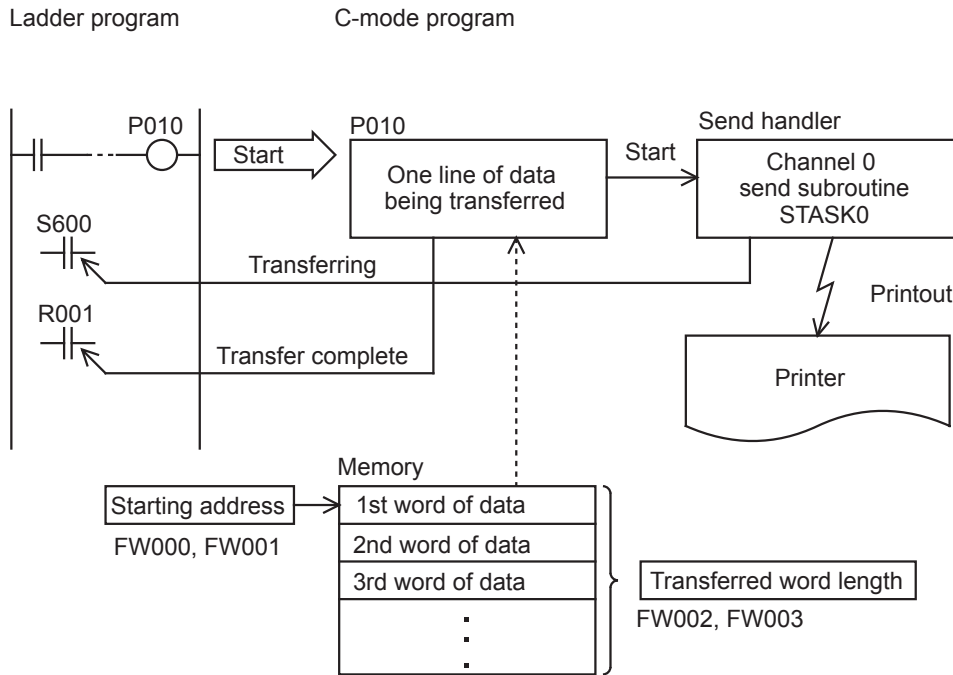


␣	Space (blank)	[/20]	
/	Slash	[/2F]	Address mark
:	Colon	[/3A]	Data delimiter

## 6 SAMPLE PROGRAMS

### 6.1.4 Program configuration

The printer output control program runs as a C-mode program created in the C language.



The C-mode program is programmed as a subroutine that prints CPU memory contents line by line and is assigned to P010.

The C-mode program starts when coil P010 is turned on in the ladder program.

Before starting the C-mode program, set the starting address (FW000 to FW001) of the control information table and the transferred word length (FW002 to FW003).

## 6.1.5 Ladder program linkage table configuration

## (1) Printout control information table

Symbol	Address	$2^{15}$	$2^0$
FW000	/0E2000	Starting address	
FW001	/0E2002		
FW002	/0E2004	Transferred word length	
FW003	/0E2006		

} 32 bits long  
} 32 bits long  
(binary data)

Setting example:

Starting address = /120000

Transferred word length = 16 (/10) words

Symbol	Address	Data
FW000	/0E2000	/0012
FW001	/0E2002	/0000
FW002	/0E2004	/0000
FW003	/0E2006	/0010

## (2) Print-complete flag

The print-complete flag turns on when printout of a specified word length of data is completed, and turns off when printout of one line of data is initiated for the first time.

Symbol	Address	$2^{15}$	$2^1$	$2^0$
R001	/0AC002			

Because only the LSB (bit  $2^0$ : the least significant bit) of this memory area is valid, the on/off data is set as follows:

When on = /0001

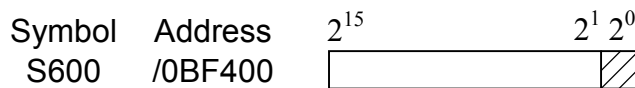
When off = /0000

## 6 SAMPLE PROGRAMS

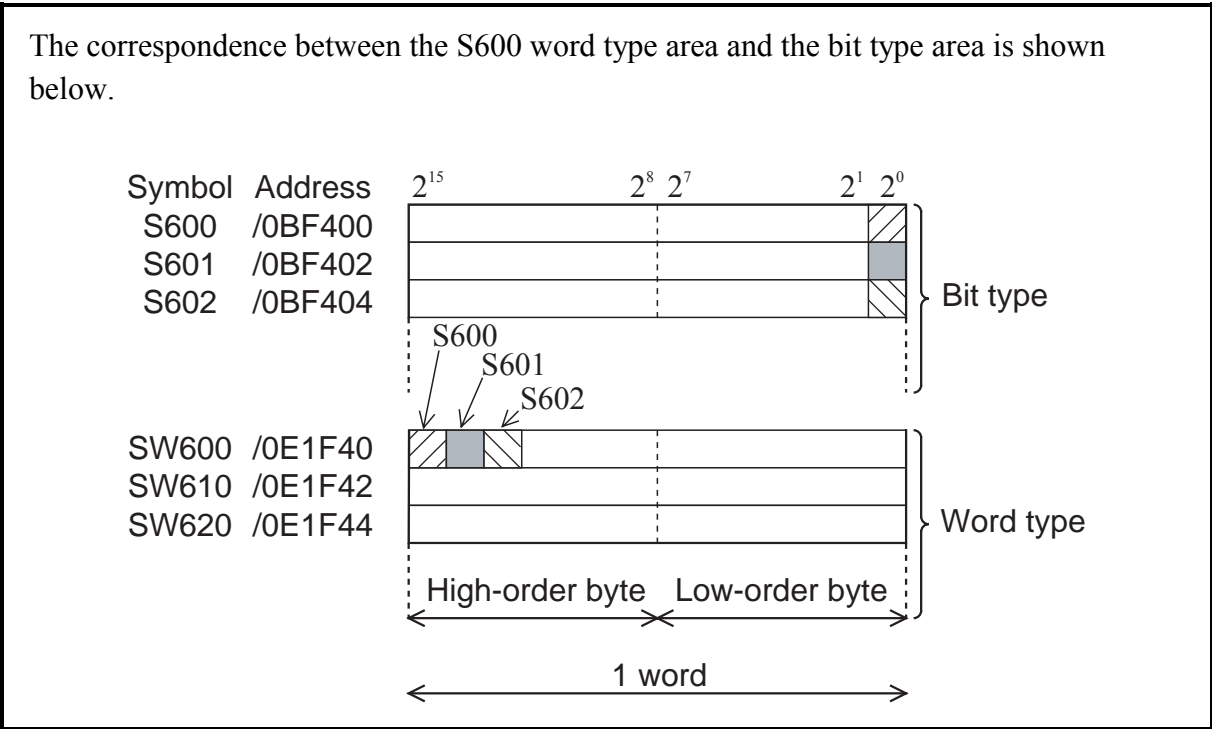
### (3) Transfer-in-progress flag

System register S600 of the transfer-in-progress flag of the send handler (STASK0) that is launched on one-line data printout processing is used.

Here, access is made to a bit type area to ease software processing.



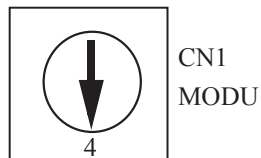
Because only the LSB (bit  $2^0$ : the least significant bit) of this memory area is valid, the on/off data is set as follows:  
 When on = /0001  
 When off = /0000



## 6.1.6 RS-232C module

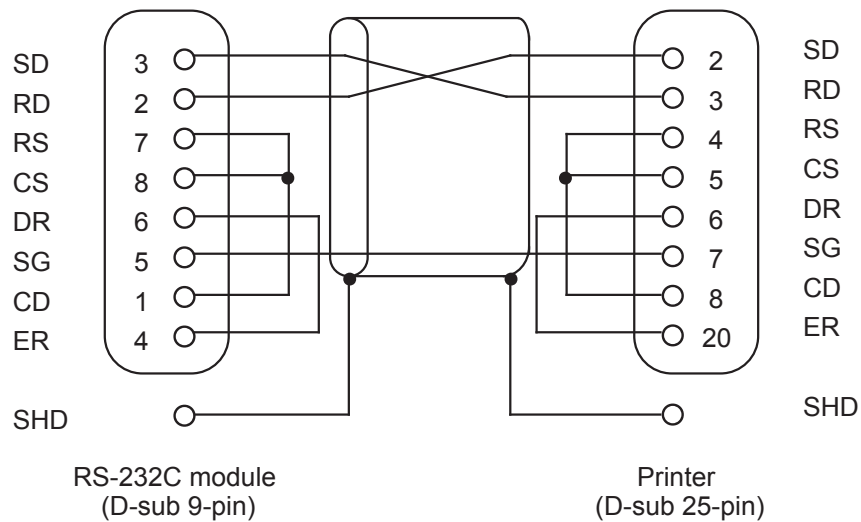
## (1) Module switch setting

To use CN1 for “Free-running – Task,” set the module switch to 4.



## (2) RS-232C signal lines

Only data signal lines are used as RS-232C signal connections. Other control lines are out of use. The RS-232C signal lines are wired as shown below.



## 6 SAMPLE PROGRAMS

---

### 6.1.7 Setting the LGB table

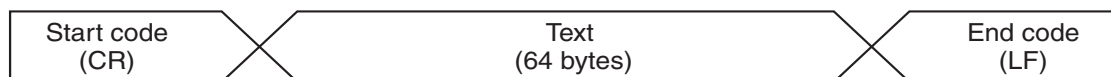
The specifications of the printer serial interface as shown below.

Item	Description
Data frame	Start bit: 1 bit Data bits: 8 bits Parity bit: Yes, even parity Stop bit: 1 bit
Baud rate	4800 bps
Print control	A received data buffer (1 KB) is available. The printer prints the contents of the received data buffer when it receives a line feed code (LF: /0A), executing a line feed automatically.

#### <Data block structure>

The data block is assumed to have a text length of 64 bytes.

The start code (SCD) and the end code (ECD) of the data block are used in the following format:



CR: Carriage return (/0D)

LF: Line feed (/0A)

No block check character (BCC)

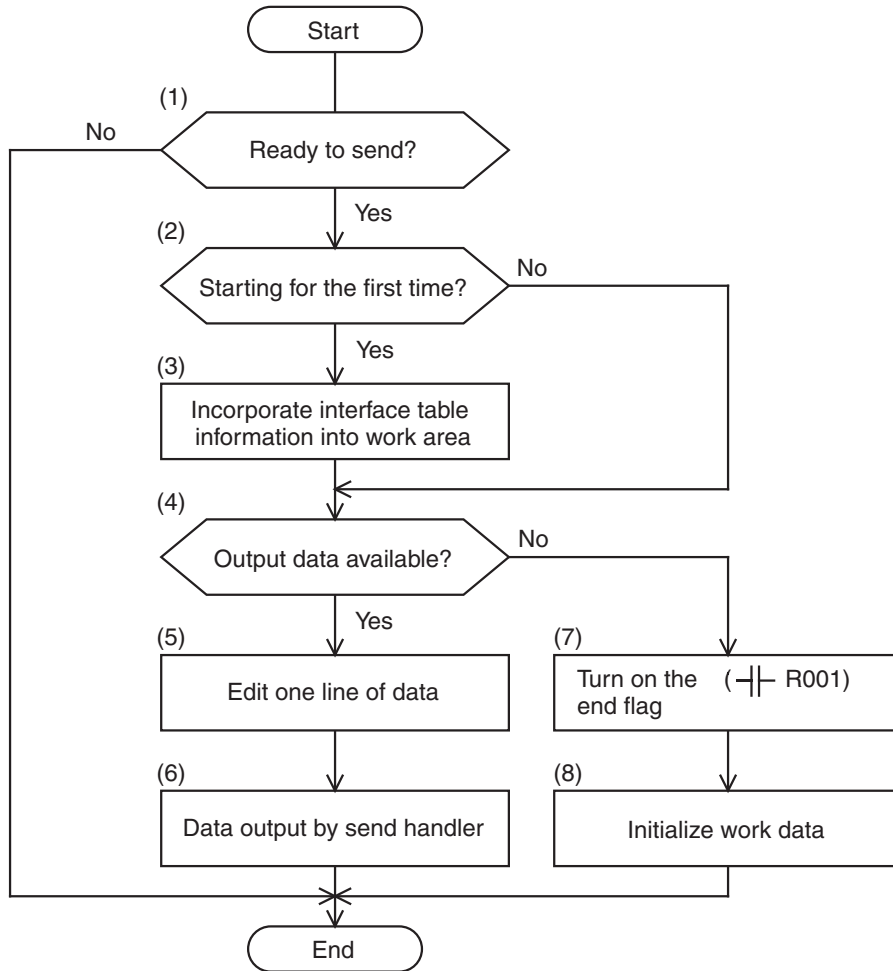


## &lt;LGB Table Setting Items&gt;

Item	Description
Data frame	ST+8DT+EP+1SP (*)
Baud rate	4800[bps] (*)
Priority level	Local station prioritized
Data change mode	Binary
Text size	64 bytes
Start code	/0D (*)
End code	/0A (*)
Block check character	No BCC
Send delay time	No send delay
Send break and continue codes	No break and continue codes
Send break timeout	3276.7[s]
Receive timeout	3276.7[s]
RS-422 gate control	–
Request to Send (RS)	RS available
Equipment Ready (ER)	Ready
Data Set Ready (DR)	No check
Control signal automatic control	Manual control
System selection	Task system

When using any other printer, edit the items marked by an asterisk (\*) to meet the specifications of that printer as needed.

6.1.8 C language program flowchart



- (1) Reference the status of system register S600 to check for readiness to transmit.
- (2), (3) Check to see if this is the first instance of startup and, if so, incorporate interface table information into a task work area.
- (4) Check the next task work area for the output word length remaining.
- (5) to (6): If output data is available, edit one line of data and direct the print data to the printer with the send handler.
- (7) to (8): When all the output has completed, turn on the end flag (-| R001) and turn off the initial start flag.

## 6.1.9 C language sample program

## (1) Program body

```

1:  /*****
2:  /*      Sample No.1 :: Memory dump task                                */
3:  /*****
4:
5:  #define TXSUB0      0x107000      Send handler address
6:  #define IFTB       0xE2000      F000 control information table address
7:  #define R001       0xAC002      R001 print-complete flag address
8:  #define S600       0xBF400      S600 transfer-in-progress flag address
9:  #define MASK       0x0001      Mask data '1'
10:
11: static struct WORK {      short      flag ;      Processing-in-progress flag
12:                          long      addr ;      Transfer address in process
13:                          long      word ;      Transferred word length in process
14:                          } work ;
15:
16: static char linebf[64] ;      One-line print buffer
17:
18:
19: p010()
20: {
21: register long (*txsub)() ;
22: register long *lpt ;
23: register char *cpt ;
24: register short wk ;
25: register short ct ;
26: register long retncd ;
27:
28: if( ( *(short *)S600 & MASK ) == 0 )      Check for readiness to transmit
29: {
30:     if( work.flag == 0 )      Incorporate control information table
31:     {
32:         lpt = (long *)IFTB ;
33:         work.addr = *lpt++ ;
34:         work.word = *lpt ;
35:         work.flag = 1 ;
36:     }
37:     if( word.word > 0 )      Printout processing
38:     {      Initialize the line buffer
39:         ct = 64 ;
40:         cpt = &linebf[0] ;
41:         while( --ct >= 0 )
42:             *cpt++ = ' ' ;
43:
44:         (long)cpt = &(work.addr) ;      Set address data
45:         btoas( &linebf[3] , cpt[1] ) ;
46:         btoas( &linebf[5] , cpt[2] ) ;
47:         btoas( &linebf[7] , cpt[3] ) ;
48:
49:         (long)cpt = word.addr ;      Set memory data
50:         ct = 12 ;
51:         while( ( work.word > 0 ) && ( ct < 50 ) )

```

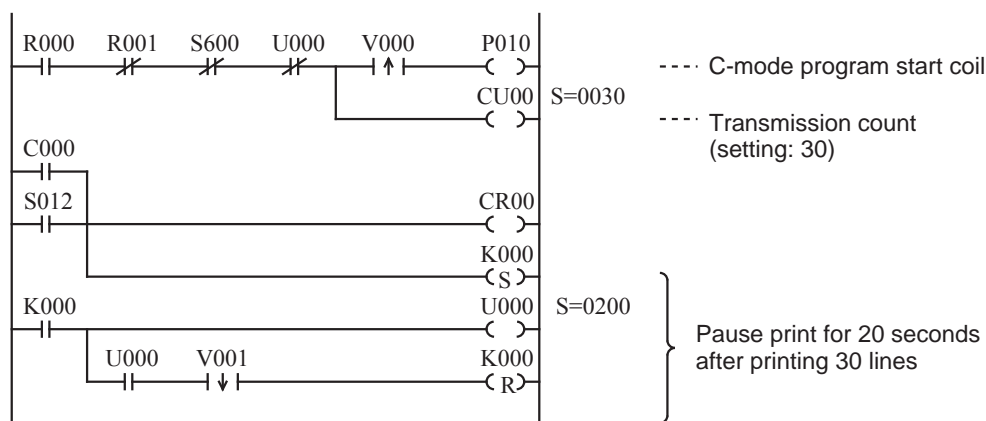
## 6 SAMPLE PROGRAMS

### (2) Binary to ASCII conversion subroutine

```
52:      {
53:      btoas( &linebf[ct] , *cpt++ ) ;           High-order byte data
54:      btoas( &linebf[ct+2] , *cpt++ ) ;       Low-order byte data
55:      ct += 5 ;                               Set SP (space)
56:      work.word -= 1 ;
57:      }
58:      work.addr += 0x000010 ;
59:
60:      linebf[2] = '/' ;                         Address mark "/"
61:      linebf[10] = ':' ;                       Data delimiter ":"
62:
63:      txsub = (long(*)())TXSUB0 ;              Data transfer
64:      retncd = (*txsub)( &linebf[0] , 64 ) ;
65:      }
66:  else{
67:      work.flag = 0 ;
68:      *(short *)R001 = 1 ;                     Set the print-complete flag
69:      }
70:  }
71:  return ;
72:  }
73:
74:  /*****
75:  /*      BINARY → ASCII function ( byte size )      */
76:  /*****
77:  btoas( stp , data )
78:  register char *stp ;                          Character set pointer
79:  register char data ;                          Binary data
80:  {
81:  register char wk ;                            Work register
82:
83:  wk = data ;                                  Set the high-order digit
84:  wk >>= 4 ;
85:  wk &= (char)0x0F ;
86:  if ( wk <= (char)0x09 )
87:      wk += (char)0x30 ;
88:  else wk += (char)0x37 ;
89:  *stp++ = wk ;
90:
91:  data &= (char)0x0F ;                          Set the low-order digit
92:  if( data <= (char)0x09 )
93:      data += (char)0x30 ;
94:  else data += (char)0x37 ;
95:  *stp = data ;
96:
97:  return ;
98:  }
99:  /*****
```

## 6.1.10 Ladder program

A ladder program that starts the C-mode program assigned to P010 is needed to carry out printout. A sample ladder program is shown below.



- R000 ..... Print request  
 R001 ..... Print complete  
 S600 ..... Transferring a remote station link  
 C000 ..... 30-line print counter  
 U000 ..... Timer to pause print after printing 30 lines  
 S012 ..... STOP→RUN signal  
 K000 ..... Save 30-line print counter on power failure

The printer used this time prints so slow when compared with the speed of data transfer that the continuous transmission of print data to it would cause the printer to malfunction with an received data buffer overflowing. The ladder program pauses transmission for 20 seconds after printing 30 lines to prevent this.

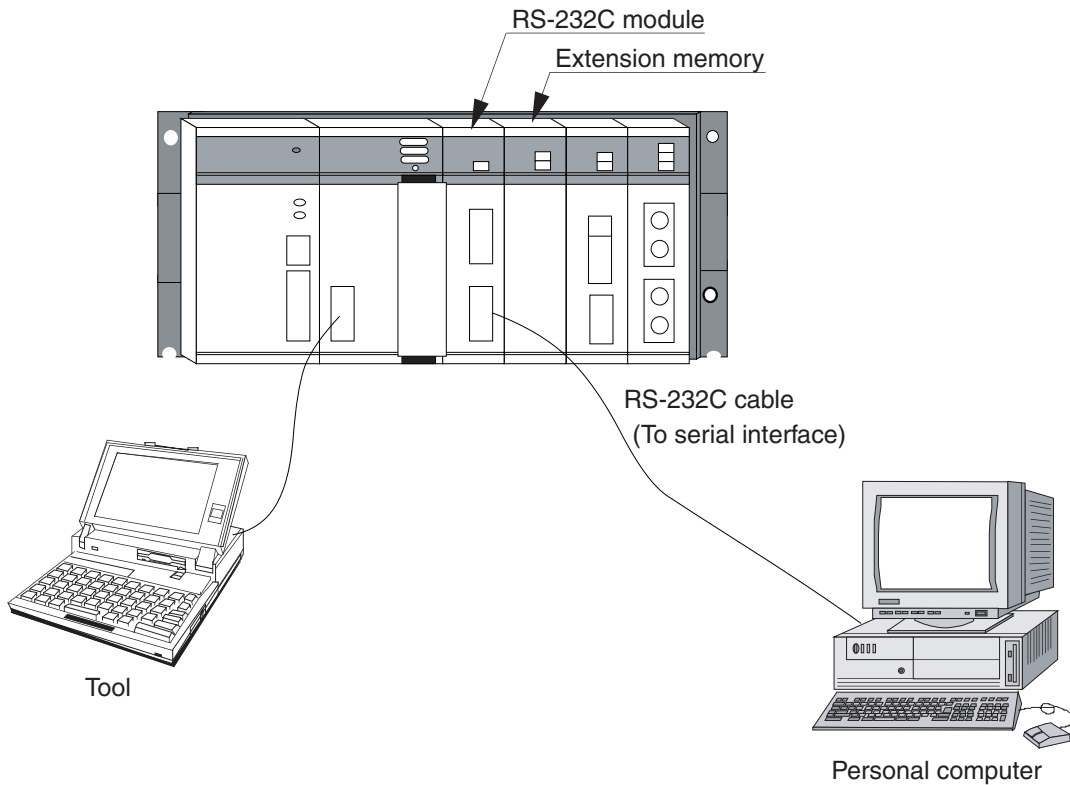
**NOTE**

The sample program is designed to promote understanding. The working program should make error checks on the send handler return code and system registers (S).

## 6.2 PC-based Program Loading

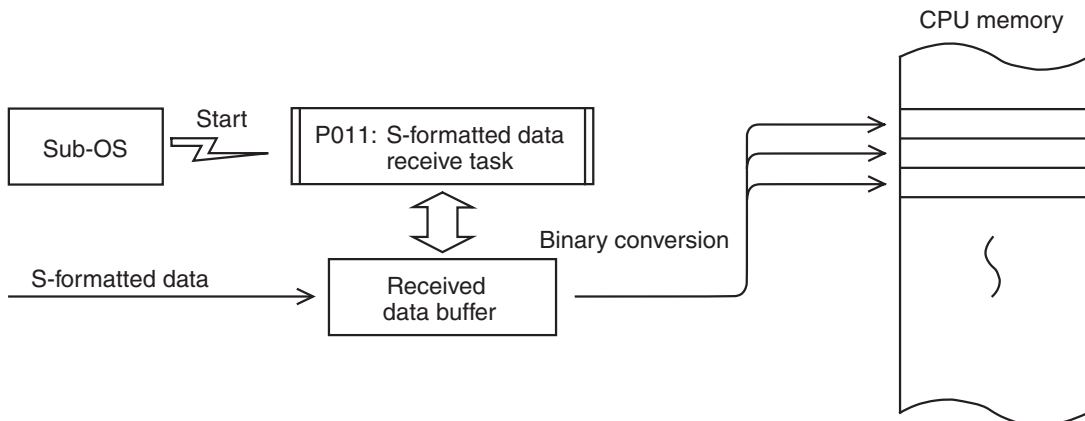
### 6.2.1 System configuration

Connect a PC to the CPU unit by way of an RS-232C interface to load programs written in the C or other languages into CPU memory directly.



### 6.2.2 Program configuration

The S-formatted data receive task starts on receiving S-formatted data from the PC, setting the incoming data at a memory address.





## 6 SAMPLE PROGRAMS

---

### 6.2.4 LGB table settings

An example of settings in the LGB table is shown below.

Item	Description
Data frame	ST+8DT+OP+1SP
Baud rate	1200[bps]
Priority level	Local station prioritized
Data change mode	Binary
Text size	256 bytes
Start code	/53
End code	/0D+0A
Block check character	No BCC
Send delay time	No send delay
Send break and continue codes	No break and continue codes
Send break timeout	3276.7[s]
Receive timeout	3276.7[s]
RS-422 gate control	–
Request to Send (RS)	RS available
Equipment Ready (ER)	Ready
Data Set Ready (DR)	No check
Control signal automatic control	Manual control
System selection	Task system

**Baud rate:** Too high a baud rate could result in an increased CPU load, resulting in occasional inability to receive data. To avoid this, set the baud rate rather lower.

**Data change mode:** The receive task converts text data to binary.

**Text size:** Used 256 bytes as a standard size.

**Start code:** An S-formatted record begins with ‘S’ so that ‘S’ is used as a standard code.

**End code:** A CR and an LF follow the checksum field of the S-formatted record. These codes are used as an end code.

**Others:** Set to meet the specifications of the PC.



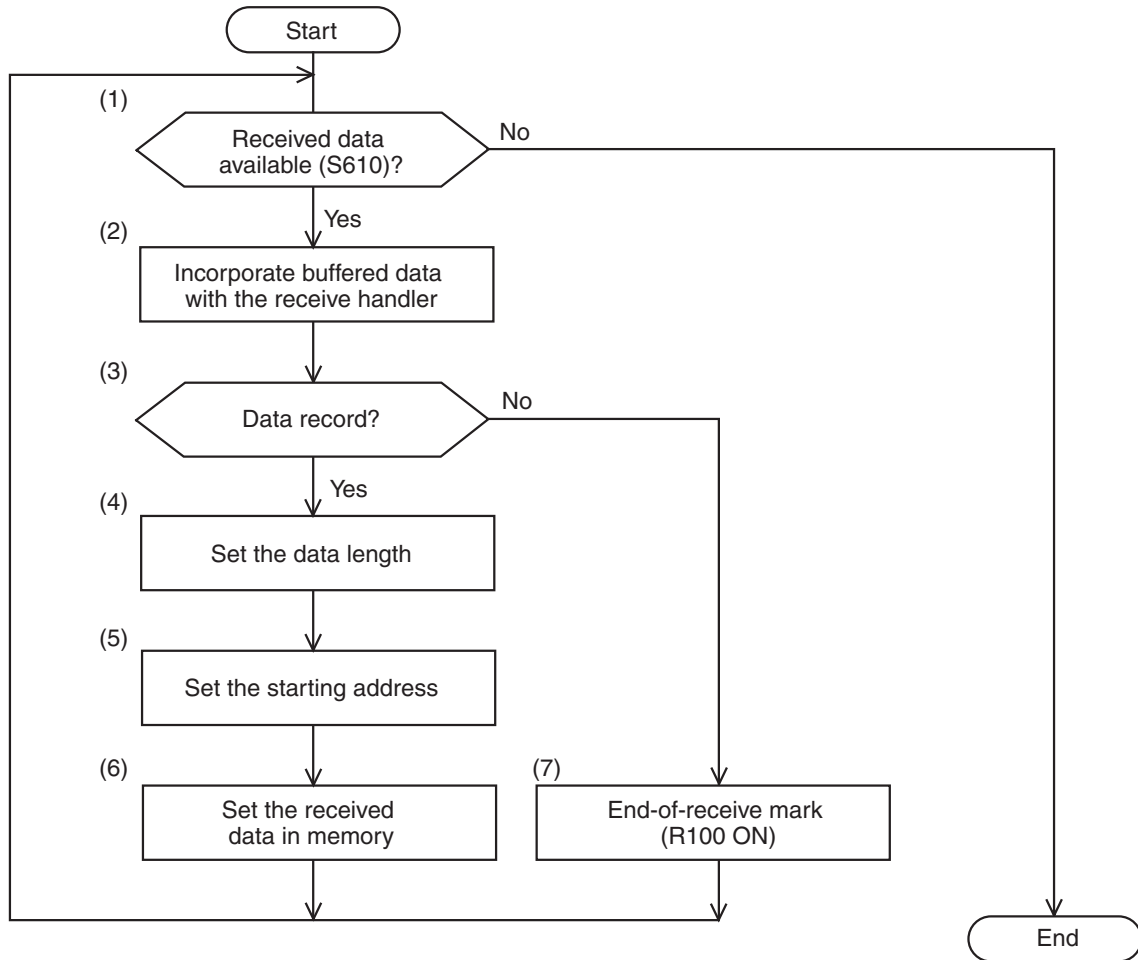
### 6.2.5 Registering a receive task

Register a PC mode program as a receive task and assign it to P011 (task number /11). The task is started by start cause /01.

Example of receive task registration

Item	Description
Receive task registration	Task number: /01
	Start cause: /01

6.2.6 Start task



- (1) Check that the receive flag (S610) is on.
- (2) Incorporate the received data with the receive handler.
- (3) Check that the record type is '2' (/32).
- (4) to (6) If the record is a data record, incorporate its data length and starting address and set the received data in memory as specified by such information.
- (7) If the record is not a data record, it is assumed the last record and the receive-complete flag (R100) is set on. This data read sequence is iterated until the flag turns off.

## 6.2.7 C language sample program

```

1:  /*****
2:  /*      Sample No.2 :: Program Loading task      */
3:  /*****
4:
5:  #define RXSUB0      0x10700C      Receive handler address
6:  #define S610      0xBF420      S610 transfer-in-progress flag
7:  #define R100      0xAC200      Receive-complete flag address
8:  #define MASK      0x0001      Mask data '1'
9:
10: static char buff[512];      Received data buffer (512 bytes)
11:
12: p011()
13: {
14: extern char atob();
15: register long (*sub)();      Receive handler
16: register char *addr;      Address pointer
17: register short *dpt;      Data pointer
18:
19: register long retncd;      Return code
20: register short ct;      Loop counter
21:
22: union { long lad;
23:         char cad[4];
24:         } adwk;
25:
26: sub = (long(*)())RXSUB0;      Incorporate received data
27: while( ( *(short *)S610 & MASK ) != 0 )
28:     {
29:         retncd = (*sub)( &buff[0], 80 );
30:         if( buff[0] == '2' )
31:             {      Set the data number
32:                 ct = (short)atob( &buff[1] );
33:                 ct &= 0x003F;      Set the address number
34:                 adwk.cad[1] = atob( &buff[3] );
35:                 adwk.cad[2] = atob( &buff[5] );
36:                 adwk.cad[3] = atob( &buff[7] );
37:                 adwk.cad[0] = (char)0;
38:                 (long)addr = adwk.lad;      Write the data to memory
39:                 (char *)dpt = &buff[9];
40:                 ct -= 3;
41:                 while( --ct > 0 )
42:                     *addr++ = atob( dpt++ );
43:             }
44:         else *(short *)R100 = 1;      Set the receive-complete flag
45:     }
46: return;
47: }
48: /*****
49: /*      ASCII → BINARY function      */
50: /*****
51: char atob( pt )
52: register char *pt;

```

## 6 SAMPLE PROGRAMS

---

```
53: {
54: register char wkh, wkl ;
55:
56: wkh = *pt++ ;
57: wkh -= '0' ;
58: if( wkh > (char)9 )
59:     wkh -= 7 ;
60: wkh <<= 4 ;
61: wkh &= (char)0xF0 ;
62:
63: wkl = *pt ;
64: wkl -= (char)0x30 ;
65: if( wkl > (char)9 )
66:     wkl -= 7 ;
67: wkl &= (char)0x0F ;
68:
69: wkh |= wkl ;
70: return( wkh );
71: }
72: /*****
```

### 6.2.8 Loading programs

Load the receive task (P011) into CPU memory before making an entry in the PRET, configuring the LGB, and then registering the receive task.

Next, set the PC's baud rate, data frame and so forth and then connect the RS-232C cable.

This will allow S-formatted data to be serially transmitted from the PC to the CPU.

(Supplement)

- A baud rate of 1200 bps has been set (to allow for the CPU load).
- The sample program does not make error checks at receive. The working program should make error checks to launch error handling as needed.

# 7 MAINTENANCE

7.1 Maintenance and Check

To keep the module running in optimal condition, it requires checks. Make checks daily or periodically (twice a year or more often).

Maintenance and Inspection Items

Item	Point to check
Module appearance	Check the module case for cracks, flaws and other defects. Such defects may be signs of breakage in the internal circuitry, leading to system malfunctions.
LED	Check to see if the RS-232C or RS-422 module ERR LED has not glowed.
Loose mounting screws	Check the mounting screws for tightness. Retighten them if found loose. Loose screws could lead to system malfunctions and eventually result in a burnout under heat.
Cable covering conditions	Check the cable coverings for defects. Coverings out of position could lead to system malfunctioning, electrical shock hazards, and a burnout after shorts.
Dust	Check the module to see if it has not caught dust. Remove dust with a vacuum cleaner if found. Dust could cause internal circuitry to short, resulting in a burnout.
Module replacement	Hot replacement could lead to hardware and software failures. Be sure to switch off the module before replacing it.
Connector status	Dust or foreign matter on the connector contacts could degrade connector characteristics. Be sure to attach the dust cap supplied to connectors when they are out of use.

 **DANGER**

- Static electricity could cause damage to the module. Before handling the module, discharge static electricity on the human body.
- Before replacing the module, switch it off to avoid electrical shock hazards and also to prevent it from being damaged or malfunctioning.

## 7.2 Backing Up User Setup Items

### 7.2.1 LGB table, receive task entry table, and user computing function entry table

(1) Backup to flash memory in the module

Edits made to the LGB table, receive task entry table, and user computing function entry table using are saved to flash memory in the module when CPU module is reset after the editing. When power recovers, the LGB table, receive task entry table, and user computing function entry table stored in flash memory in the module are loaded to start operation.

(2) Backup in a batch save

Table below lists the areas of the LGB table, receive task entry table, and user computing function entry table that can be batch-saved by the batch save/load system.

Name	Channel No.	Address		Batch save
		S10mini	S10V	
LGB table	1	/F48100 to /F481FE		Batch-saved
	2	/F58100 to /F581FE		Batch-saved
	3	/F68100 to /F681FE		Batch-saved
	4	/F78100 to /F781FE		Batch-saved
Receive task entry table	1	/1070CA to /1070D0	/F481C0 to /F481D0	Not batch-saved (user-specified)*
	2	/10714A to /107150	/F581C0 to /F581D0	Not batch-saved (user-specified)*
	3	/1071CA to /1071D0	/F681C0 to /F681D0	Not batch-saved (user-specified)*
	4	/10724A to /107250	/F781C0 to /F781D0	Not batch-saved (user-specified)*
User computing function entry table	1 to 4	/FAB40 to /FAD3E	No	Batch-saved

\* (Note pertaining to the S10mini Series only)

When executing a batch save, back up extension memory as well to cover the areas of interest. With Batch Save/Load System Version 08-00 and higher, whole extension memory is backed up by default. With earlier versions, be sure to expressly specify extension memory addresses.



### NOTE

- The receive task entry table is not automatically batch-saved. Users should specify its address when saving it. The LGB table and the user computing function entry table are automatically batch-saved.
- If a power failure occurs before or during the reset that is carried out after the end of editing from a tool or batch loading, the data written to flash memory in the module would take effect, rather than the edits or settings entered by the batch load. In this case, reset the module again after the end of editing or batch loading.

#### 7.2.2 Replacing modules

- When the RS-232C or RS-422 module has been replaced  
When the RS-232C or RS-422 module has been replaced after failures or other adverse conditions, there are two ways to set the LGB table again as follows:
  - Set the LGB table and the receive task entry table from the tool and activate the CPU module reset switch.
  - If the tables have been batch-saved, carry out a reset after loading from the tool, and the tables would be written back to flash memory in the module and take effect from that moment on.
- When the CPU module has been replaced  
When the CPU module has been replaced after failures or other adverse conditions, there is no need to set the LGB table and the receive task entry table again (since they have been saved to flash memory in the module). Perform a reset after a batch load for the tables to take effect from that moment on.

## 7.3 Troubleshooting

### 7.3.1 CPU module indicator display messages

The messages listed below is displayed in the CPU module indicator when a certain event or error occurs in the RS-232C or RS-422 module.

Message	Explanation	User action
R2△ □. □	The module has started up successfully.	
R2△● * * * *	An error has occurred in the RS-232C or RS-422 module.	Take action as instructed in Subsection 7.3.2 to Subsection 7.3.4.
EXA● PTY	A parity error occurred when the CPU read from the RS-232C or RS-422 module's internal memory.	Reset the CPU using the reset switch. If the display persists, replace the module.

- △ shows "M" for channel number 0 or 1, or "S" for channel number 2 or 3.
- □.□ denotes the version and revision of the module.
- ● denotes a channel number.
- \* \* \* \* denotes an error message. For more information, see Subsection 7.3.2 to Subsection 7.3.4.

## 7 MAINTENANCE

---

### 7.3.2 Hardware errors

When the RS-232C or RS-422 module detects a hardware error, the CPU displays in the CPU module indicator as shown in the table below.

The ERR LED on the RS-232C or RS-422 module glows and error freeze information is collected at the same time. The RS-232C or RS-422 module shuts down its operation.

Error message	Explanation	User action
R2△ BUS	Bus error	Reset the CPU. If the same error message recurs, the RS-232C or RS-422 module may have failed. Replace the module.
R2△ ADDR	Address error	
R2△ ILLG	Illegal instruction	
R2△ ZERO	Division by zero	
R2△ PRIV	Privilege violation	
R2△ WDT	WDT error	
R2△ EXCP	Unused exception	
R2△ PTY	RAM parity error	
R2△ ROM	ROM checksum error	
R2△ RAM	RAM check error	
R2△ MDSW	Invalid module switch setting	Review the module switch setting.
R2△● LGB	LGB setup error	Reset the LGB table again.

- △ shows “M” for channel number 0 or 1, or “S” for channel number 2 or 3.
- ● denotes a channel number.

### 7.3.3 Transmit errors

The table below lists the error messages that relate to transmission, and the error codes that are loaded in the system register (S-register).

Handler errors are not displayed in the CPU indicator.

Error message	Error code	Explanation	User action
Handler error not shown	/4002	The send handler was launched while transmission had been broken.	Review the application program.
	/4003	Transmission launched while transmission was already in progress.	
	/4004	Send handler parameter error	
	/4005	The channel to be launched for transmission shut down.	Reset the CPU. If the same error message recurs, replace the module.
R2△● SPRV	/2080	Unable to transmit, because data is being received with the remote station prioritized specification.	Transmission starts after the reception completes.
R2△● NOCS	/2081	Unable to transmit because CS (Clear to Send) input is set to CS Unavailable.	<ul style="list-style-type: none"> <li>• Verify the correct settings of the control signals of the remote station.</li> <li>• Verify the correct cable connections.</li> </ul>
R2△● NODR	/2082	Unable to transmit because DR (Data Set Ready) input with the DR check specified is set to Not Ready.	
R2△● BRTO	/2083	Transmission in progress was broken by a break code but no continue code was issued to allow the transmission to resume within the send timeout out period.	<ul style="list-style-type: none"> <li>• Review the remote station settings and the communications program.</li> <li>• Review the LGB settings.</li> </ul>
R2△● CSTO	/2084	Transmission in progress was broken as CS (Clear to Send) input was set to CS Unavailable. But CS would not set to CS Available during the send timeout period, keeping the transmission from resuming.	<ul style="list-style-type: none"> <li>• Verify the remote station settings.</li> <li>• Check the cable for breakage.</li> </ul>
R2△● DRTO	/2085	Transmission in progress was broken as DR (Data Set Ready) input was set to Not Ready. But DR would not set to Ready during the send timeout period, keeping the transmission from resuming.	
R2△● SRBR	/1000	Transmission was launched with the local station prioritized specification while reception was in progress. Therefore, the data reception was canceled and the transmission resumed.	Set the LGB priority level to No Priority Level (full-duplex communication).

- △ shows “M” for channel number 0 or 1, or “S” for channel number 2 or 3.
- ● denotes a channel number.

## 7 MAINTENANCE

### 7.3.4 Receive errors

The table below lists the error messages that relate to reception, and the error codes that are loaded in the system register (S-register).

Handler errors are not displayed in the CPU indicator.

Error message	Error code	Explanation	User action
Handler error not shown	/4200	Receive handler parameter error	Review the application program.
	/4400	The receive handler was launched while the channel shut down.	Reset the CPU. If the same error message recurs, replace the module.
R2△● RPTY	/2080	A parity error occurred in the receive data.	<ul style="list-style-type: none"> <li>• Verify the LGB settings to meet the communications settings of the remote station.</li> <li>• Check the cable route for sources of noise interference.</li> </ul>
R2△● ROVR	/2081	An overrun error occurred in the receive data.	
R2△● PFRM	/2082	A framing error occurred in the receive data.	
R2△● RVTO	/2083	Not all data could be received during the specified period of time.	Review the receive timeout period setting in the LGB table.
R2△● NOAS	/2084	Data other than “0” to “9” and “A” to “T” was received during ASCII conversion.	Review the application running on the remote station.
R2△● NOEC	/2085	Data other than “0” to “9” and “A” to “T” or data other than an end code was received during ASCII conversion.	
R2△● BCCE	/2086	The received BCC did not match during BCC checking.	
R2△● CDTO	/2087	Reception in progress was broken as CD (Carrier Detect) input was set to Off. But CD would not set to On during the receive timeout period, keeping the reception from resuming.	<ul style="list-style-type: none"> <li>• Verify the remote station settings.</li> <li>• Check the cable for breakage.</li> </ul>
R2△● RVOV	/2088	Received data was discarded because all the eight cases of receive buffers were full.	Launch the receive handler to incorporate the received data.
R2△● RVNZ	/2089	Noise was detected in the received data.	Check the cable route for sources of interference.
R2△● RRBR	/2002	Some buffer has been filled with data halfway since transmission was launched while receiving data with the local station prioritized option.	Set the priority level in the LGB table to no priority level (full-duplex communication).

- △ shows “M” for channel number 0 or 1, or “S” for channel number 2 or 3.
- ● denotes a channel number.

## 7.3.5 Error freeze

When the RS-232C or RS-422 module detects a hardware error, the ERR LED glows and error freeze information is collected at the same time. The RS-232C or RS-422 module shuts down its operation.

The Table below lists the formats in which error information is presented. For the error codes and stack frames in these formats, see the next page.

Channel 0	Channel 1	Channel 2	Channel 3	2 <sup>31</sup> _____2 <sup>16</sup>	2 <sup>15</sup> _____2 <sup>0</sup>
/F48800	/F58800	/F68800	/F78800	Error code	—
/F48804	/F58804	/F68804	/F78804	Time from reset clear (ms)	
/F48808 	/F58808 	/F68808 	/F78808 	Not used	
/F4880C	/F5880C	/F6880C	/F7880C		
/F48810	/F58810	/F68810	/F78810	D0	
/F48814	/F58814	/F68814	/F78814	D1	
/F48818	/F58818	/F68818	/F78818	D2	
/F4881C	/F5881C	/F6881C	/F7881C	D3	
/F48820	/F58820	/F68820	/F78820	D4	
/F48824	/F58824	/F68824	/F78824	D5	
/F48828	/F58828	/F68828	/F78828	D6	
/F4882C	/F5882C	/F6882C	/F7882C	D7	
/F48830	/F58830	/F68830	/F78830	A0	
/F48834	/F58834	/F68834	/F78834	A1	
/F48838	/F58838	/F68838	/F78838	A2	
/F4883C	/F5883C	/F6883C	/F7883C	A3	
/F48840	/F58840	/F68840	/F78840	A4	
/F48844	/F58844	/F68844	/F78844	A5	
/F48848	/F58848	/F68848	/F78848	A6	
/F4884C	/F5884C	/F6884C	/F7884C	A7	
/F48850 	/F58850 	/F68850 	/F78850 	Stack frame	
/F488FC	/F588FC	/F688FC	/F788FC		

## 7 MAINTENANCE

< Error code >

Code	Explanation	User action
/0010	Bus error	The RS-232C or RS-422 module may have failed. Replace the module.
/0011	Address error	
/0012	Illegal instruction	
/0013	Division by zero	
/0014	Privilege violation	
/0015	WDT error	
/0018	Unused exception	
/0019	RAM parity error	
/0102	ROM checksum error	
/0103	RAM check error	
/0100	Invalid module switch setting	Review the module switch setting.
/0112	LGB setup error	Set the LGB table again.

<Stack frame formats>

Address	Stack frames other than bus errors/address errors	Stack frames for bus errors/ address errors			
	$2^{15}$ — $2^0$	$2^{15}$ — $2^5$	$2^4$	$2^3$	$2^2$ — $2^0$
/50	Status register		R/W	I/N	FC
/52	Program counter	Access address			
/54		Instruction register			
/56		Status register			
/58		Program counter			
/5A					
/5C					

R/W (read/write): Write = 0, read = 1  
 I/N (instruction/non-instruction): Instruction = 0, non-instruction = 1  
 FC: Function code

### 7.3.6 Communications trace information

The RS-232C or RS-422 module can trace communications information and events. Using this function, trace data can be created to aid in problem determination and corrective action.

<Trace buffer structure>

Channel 0	Channel 1	Channel 2	Channel 3	2 <sup>15</sup> -----2 <sup>0</sup>
/F4E000	/F5E000	/F6E000	/F7E000	Pointer
/F4E002	/F5E002	/F6E002	/F7E002	Run/Stop
/F4E004	/F5E004	/F6E004	/F7E004	Stop condition type
/F4E006	/F5E006	/F6E006	/F7E006	Trace mode
/F4E008 to /F4E01E	/F5E008 to /F5E01E	/F6E008 to /F6E01E	/F7E008 to /F7E01E	Not used
/F4E020 to /F4E03E	/F5E020 to /F5E03E	/F6E020 to /F6E03E	/F7E020 to /F7E03E	Trace data #0
/F4E040 to /F4E05E	/F5E040 to /F5E05E	/F6E040 to /F6E05E	/F7E040 to /F7E05E	Trace data #1
/F4FFE0 to /F4FFFE	/F5FFE0 to /F5FFFE	/F6FFE0 to /F6FFFE	/F7FFE0 to /F7FFFE	Trace data #254

- **Pointer**  
Points to the buffer case in which the next batch of trace data is stored. The pointer is initialized to /20. Valid from /20 to /1FE0.
- **Run/Stop**  
Run or stop tracing (= 0: stop, ≠ 0: run (default = 1)).
- **Stop condition type**  
Specify the type of the starting word of the trace data, so that tracing will stop on tracing data of the same type.
- **Trace mode**  
Specify the trace mode in which tracing executes.  
= 0: Trace stopped  
= 1: Infinite tracing  
= 2: Stop on error (default)  
= 3: Stop on handler trace shutdown
- **Trace data**  
Trace buffers, which trace data is cyclically stored from #254 back to #0. (See the next page for more details.)



## 7 MAINTENANCE

---

### <Trace Data Details>

/00	Type
/02	Control signal state
/04 /1A	Transmitted/received data (24 bytes)
/1C /1E	Time from reset clear (ms)

- Type

Denotes the distinction between transmission and reception, and an error.

/1000: Normal transmission

/2000: Normal reception

/30\*\*: Transmission error

/40\*\*: Reception error

\*\* denotes the low-order byte of the error code.

- Control signal state

Stores the control signal I/O state.

Similar to the information that is incorporated by a latest hardware state incorporate request as described in “5.5 Hardware Controlled by Software Implementation.”

### 7.3.7 Handler trace information

The RS-232C or RS-422 module can trace handler startups from applications and responses.

< Trace buffer structure >

Channel 0	Channel 1	Channel 2	Channel 3	2 <sup>15</sup> -----2 <sup>0</sup>
/F4D000	/F5D000	/F6D000	/F7D000	Pointer
/F4D002	/F5D002	/F6D002	/F7D002	Run/Stop
/F4D004	/F5D004	/F6D004	/F7D004	Stop condition type
/F4D006	/F5D006	/F6D006	/F7D006	Trace mode
/F4D008 to /F4D00E	/F5D008 to /F5D00E	/F6D008 to /F6D00E	/F7D008 to /F7D00E	Not used
/F4D010 to /F4D01E	/F5D010 to /F5D01E	/F6D010 to /F6D01E	/F7D010 to /F7D01E	Trace data #0
/F4D020 to /F4D02E	/F5D020 to /F5D02E	/F6D020 to /F6D02E	/F7D020 to /F7D02E	Trace data #1
/F4DFF0 to /F4DFFE	/F5DFF0 to /F5DFFE	/F6DFF0 to /F6DFFE	/F7DFF0 to /F7DFFE	Trace data #254

- **Pointer**

Points to the buffer case in which the next batch of trace data is stored. The pointer is initialized to /10. Valid from /10 to /FF0.

- **Run/Stop**

Run or stop tracing (= 0: stop, ≠ 0: run (default = 1)).

- **Stop condition type**

Specify the type of the starting word of the trace data, so that tracing will stop on tracing data of the same type.

- **Trace mode**

Specify the trace mode in which tracing executes.

= 0: Trace stopped

= 1: Infinite tracing

= 2: Stop on error (default)

= 3: Stopped on communications trace shutdown

- **Trace data**

Trace buffers, which trace data is cyclically stored from #254 back to #0. (See the next page for more details.)

## 7 MAINTENANCE

---

<Trace data details>

/00	Type
/02	Error code
/04	Parameter 1
/06	
/08	Parameter 2
/0A	
/0C	Time from reset clear (ms)
/0E	

- Type

Denotes the distinction between transmission and reception, and an error.

/8000: Normal send handler startup

/9000: Normal receive handler startup

/8800: Send handler error

/9800: Receive handler error

- Error code

Stores a handler error code. For more information, see “7.3.3 Transmit errors,” and “7.3.4 Receive errors.”

- Parameters 1 and 2

Parameters that are passed on to the handler from the application.

## 7.3.8 H-7338 error trace information

The RS-232C or RS-422 module can trace errors as they occur during sessions of H-7338 communication, along with the communications data.

<Trace buffer structure>

Channel 0	Channel 1	Channel 2	Channel 3	2 <sup>31</sup> _____ 2 <sup>0</sup>
/F48920	/F58920	/F68920	/F78920	Error trace case number
/F48924	/F58924	/F68924	/F78924	Not used
/F48930 to /F4894E	/F58930 to /F5894E	/F68930 to /F6894E	/F78930 to /F7894E	Trace data #0
/F48950 to /F4896E	/F58950 to /F5896E	/F68950 to /F6896E	/F78950 to /F7896E	Trace data #1
/F48AD0 to /F48AEE	/F58AD0 to /F58AEE	/F68AD0 to /F68AEE	/F78AD0 to /F78AEE	Trace data #13

- Error trace case number

Points to the buffer case in which the next batch of trace data is stored. The pointer is initialized to /0. Valid from /0 to /0D.

- Trace data

Trace buffers, which trace data is cyclically stored from #13 back to #0. (See the next page for more details.)

## 7 MAINTENANCE

### <Trace Data Details>

/00	Error code
/04	H-7338 command code
/18	
/1C	

- Error code

Stores a command and line error code.

Error code	Explanation	Corrective action
/0000001	No space was found between parameters.	Verify the remote station settings. Check the cable route for sources of noise interference.
/0000002	A valid parameter range was exceeded.	
/00000101	Receive parity error	Verify the remote station settings. Check also the cable route for breakage and sources of noise interference.
/00000102	Receive overrun error	
/00000103	Receive framing error	
/00000104	Receive noise error	

- Command code

Stores an H-7338 communications command.

## 7.3.9 Error Counters

The RS-232C or RS-422 module has error counters to count communications errors as they occur. The error counters are initialized on a reset.

Channel 0	Channel 1	Channel 2	Channel 3	2 <sup>15</sup> _____ 2 <sup>0</sup>
/F48900	/F58900	/F68900	/F78900	Normal transmission
/F48902	/F58902	/F68902	/F78902	CS lost while transmitting
/F48904	/F58904	/F68904	/F78904	Send timeout
/F48906	/F58906	/F68906	/F78906	Normal reception
/F48908	/F58908	/F68908	/F78908	Receive overrun error
/F4890A	/F5890A	/F6890A	/F7890A	CD lost while receiving
/F4890C	/F5890C	/F6890C	/F7890C	Receive framing error
/F4890E	/F5890E	/F6890E	/F7890E	Receive parity error
/F48910	/F58910	/F68910	/F78910	Receive noise error
/F48912	/F58912	/F68912	/F78912	Break sequence received
/F48914	/F58914	/F68914	/F78914	Receive timeout
/F48916	/F58916	/F68916	/F78916	Received data discarded counter
/F48918 to /F4891E	/F58918 to /F5891E	/F68918 to /F6891E	/F78918 to /F7891E	Not used

# APPENDIX

A.1 Seven-Bit Code Table (JIS X 0201)

								b7	0	0	0	0	1	1	1	1
								b6	0	0	1	1	0	0	1	1
								b5	0	1	0	1	0	1	0	1
b7	b6	b5	b4	b3	b2	b1			0	1	2	3	4	5	6	7
0	0	0	0	0	0	0			NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	1	1			SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	2	2			STX	DC2	“①	2	B	R	b	r
0	0	1	1	3	3	3			ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	4	4			EOT	DC4	\$	4	D	T	d	t
0	1	0	1	5	5	5			ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	6	6			ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	7	7			BEL	ETB	' ②	7	G	W	g	w
1	0	0	0	8	8	8			BS	CAN	(	8	H	X	h	x
1	0	0	1	9	9	9			HT	EM	)	9	I	Y	i	y
1	0	1	0	10	10	10			LF	SUB	* : ⑥	10	J	Z	j	z
1	0	1	1	11	11	11			VT	ESC	+ ; ⑦	11	K	[	k	{
1	1	0	0	12	12	12			FF	IS4	, ③	12	L	¥	l	
1	1	0	1	13	13	13			CR	IS3	- ④	13	M	]	m	}
1	1	1	0	14	14	14			SO	IS2	. ⑤	14	N	^	n	~
1	1	1	1	15	15	15			SI	IS1	/ ?	15	O	_ ⑧	o	DEL

- ① Quote
- ② Apostrophe
- ③ Comma
- ④ Minus
- ⑤ Period
- ⑥ Colon
- ⑦ Semicolon
- ⑧ Underline



A.2 Eight-Bit Code Table (JIS X 0201)

b8	b7	b6	b5	b4	b3	b2	b1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15				
0	0	0	0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p	Free space						Undefined					
0	0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q	。⑨	ア											チ	ム
0	0	1	0	2	STX	DC2	“①	2	B	R	b	r	「	イ	ツ											メ	
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s	」	ウ	テ											モ	
0	1	0	0	4	EOT	DC4	\$	4	D	T	d	t	、⑩	エ	ト											ヤ	
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u	・⑪	オ	ナ											ユ	
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v	ヲ	カ	ニ											ヨ	
0	1	1	1	7	BEL	ETB	' ②	7	G	W	g	w	ア	キ	ヌ											ラ	
1	0	0	0	8	BS	CAN	(	8	H	X	h	x	イ	ク	ネ											リ	
1	0	0	1	9	HT	EM	)	9	I	Y	i	y	ウ	ケ	ノ											ル	
1	0	1	0	10	LF	SUB	* :⑥	J	Z	j	z	エ	コ	ハ	レ												
1	0	1	1	11	VT	ESC	+ ;⑦	K	[	k	{	オ	サ	ヒ	ロ												
1	1	0	0	12	FF	IS4	, ③	<	L	¥	l		ヤ	シ	フ											ワ	
1	1	0	1	13	CR	IS3	-④	=	M	]	m	}	ユ	ス	ヘ											ン	
1	1	1	0	14	SO	IS2	. ⑤	>	N	^	n	—	ヨ	セ	ホ											゜⑬	
1	1	1	1	15	SI	IS1	/ ?	O	_⑧	o	DEL	ッ	ソ	マ	°⑭											⑮	

- ① Quote
- ② Apostrophe
- ③ Comma
- ④ Minus
- ⑤ Period
- ⑥ Colon
- ⑦ Semicolon
- ⑧ Underline
- ⑨ Japanese period
- ⑩ Japanese comma
- ⑪ Middle point
- ⑫ Prolonged sound
- ⑬ Voiced consonant
- ⑭ P-sound sign attached to kana
- ⑮ Undefined

## A.3 Control Code Definitions

Control code	Code	Control code name	Definition
NUL	/00	Null	Blank
SOH	/01	Start of Heading	Start of heading
STX	/02	Start of Text	Start of text
ETX	/03	End of Text	End of text
EOT	/04	End of Transmission	End of transmission
ENQ	/05	Enquiry	Enquiry
ACK	/06	Acknowledge	Acknowledge
BEL	/07	Bell	Bell
BS	/08	Backspace	Backspace
HT	/09	Horizontal Tabulation	Horizontal tabulation
LF	/0A	Line Feed	Line feed
VT	/0B	Vertical Tabulation	Vertical tabulation
FF	/0C	Form Feed	Form feed
CR	/0D	Carriage Return	Carriage return
SO	/0E	Shift Out	Shift out
SI	/0F	Shift In	Shift in
DLE	/10	Data Link Escape	Data link escape
DC1	/11	Device Control 1 (X-ON)	Device control 1 (used before initiating transmission)
DC2	/12	Device Control 2	Device control 2
DC3	/13	Device Control 3 (X-OFF)	Device control 3 (used before terminating transmission)
DC4	/14	Device Control 4	Device control 4
NAC	/15	Negative Acknowledge	Negative acknowledge
SYN	/16	Synchronous Idle	Synchronous idle
ETB	/17	End of Transmission Block	End of data block
CAN	/18	Cancel	Cancel
EM	/19	End of Medium	End of medium
SUB	/1A	Substitute Character	Substitute character
ESC	/1B	Escape	Escape (used for escaping control codes, as for displays and graphics)
FS	/1C	File Separator	File separator
GS	/1D	Group Separator	Group separator
RS	/1E	Record Separator	Record separator
US	/1F	Unit Separator	Unit separator
SP	/20	Space	Space
DEL	/7F	Delete	Delete

## A.4 Abbreviations

Abbreviation	Explanation
ACIA	Asynchronous Communications Interface Adapter
ASCII	American Standard Code for Information Interchange
BCC	Block Check Character
BPS	Bits Per Second
CD	data Carrier Detect
CPMS	Compact Process Monitor System
CPU	Central Processing Unit
CRT	Cathode Ray Tube
CS	Clear to Send
DR	Data set Ready
ECD	End Code
EIA	Electronic Industries Association
EOR	Exclusive OR
ER	Equipment Ready
FE	Framing Error
FG	Frame Ground
GR	General Reset
IRQ	Interrupt Request
LED	Light Emitting Diode
LGB	Line Group Block
MCS	Man-machine Communication System
OVRN	Overflow error
PCs	Programmable Controllers
PE	Parity Error
RD	Receive Data
RS	Request to Send
SCD	Start Code
SD	Send Data
SG	Signal Ground
SHD	Shield
TERM	Terminating Resistor
UFET	User Function Edition Table
WDT	Watchdog Timer

## A.5 Trouble Report

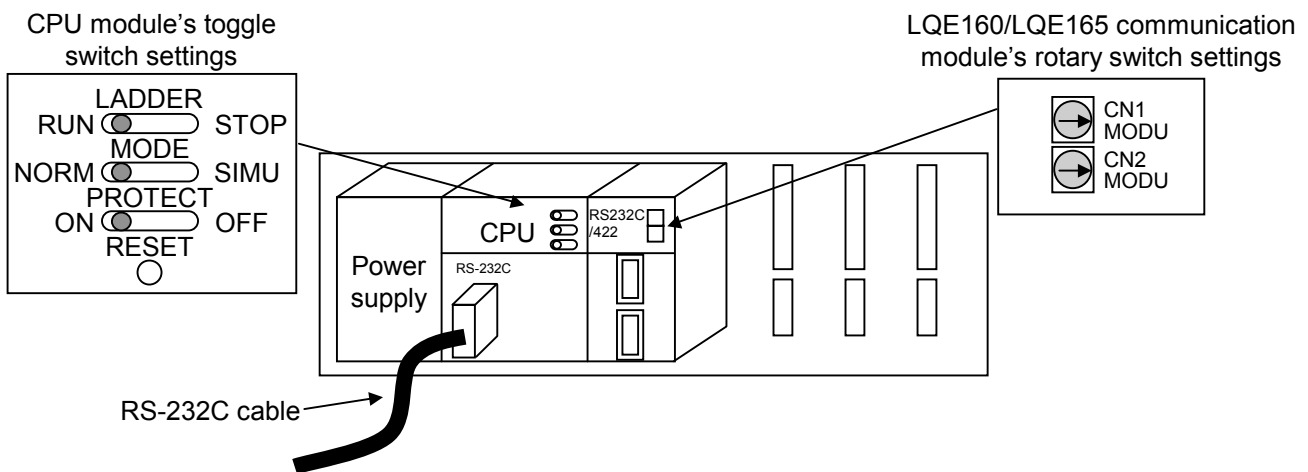
Fill out this form and submit it to local source.

Your company name			Person in charge		
Data and time of occurrence			(year / month / day / hour / minute)		
Where to make contact	Address				
	Telephone				
	FAX				
	E-mail				
Model of defective module			CPU model		
OS	Ver.	Rev.	Program name:		Ver. Rev.
Support program			Program name:		Ver. Rev.
Symptom of defect					
Connection load	Type				
	Model				
	Wiring state				
System configuration and switch setting					
Space for correspondence					

# SUPPLEMENTARY

Supplementary: Replacing or adding on the module

- What you should get in preparation
  - ① Personal computer (with Hitachi's S10 EXTERNAL SERIAL LINK SYSTEM installed in it)
  - ② RS-232C cable (or 10BASE-T cable if the communication module used is an ET.NET module)
  - ③ New or add-on RS-232C or RS-422 module (either the model LQE160 or LQE165)
  - ④ Copies of the parameter values for the module to be replaced. (These copies are prepared for use in cases where the parameters are not accessible for some reason.)
  - ⑤ The above-mentioned ET.NET module is an optional module and, if it is mounted in place, may be selected as the type of communication module to be used. For more information, refer to Section 1.2, "Mounting Optional Modules," and Section 3.1, "Names and Functions of Each Part," in the S10mini HARDWARE MANUAL, OPTION ET.NET (manual number SME-1-103).
- Replacement procedure
  - ① Write down, on a piece of paper, the current settings of the rotary switches (CN1 MODU and CN2 MODU) that are, as shown below, accessible at the front side of the existing RS-232C/RS-422 module to be replaced.
  - ② Write down also the current settings of the toggle switches (LADDER, MODE, and PROTECT) that are, as shown below, accessible at the front side of the CPU module.
  - ③ Connect the personal computer and the CPU module together with the RS-232C cable.



- ④ Start the S10 EXTERNAL SERIAL LINK SYSTEM and make a record of the set values of all the existing parameters for each channel:
  - Display the "Enter LGB" window for each channel on screen and make a hand-written copy of the set values of the existing parameters for those channels.
  - Note: The S10 EXTERNAL SERIAL LINK SYSTEM does not allow you to use the following functions: "Transmit system program", "Compare system program", and "Delete all channels of task system", all of which are provided for use only on S10/2 $\alpha$  systems.
  - If the existing parameters are not accessible for some reason, use the copies of their set values [item ④] that were obtained in preparation.

- ⑤ Set the CPU module's LADDER switch in STOP position and turn off the power supply of the controller unit.
  - ⑥ Remove the connecting cable from the RS-232C/RS-422 module.
  - ⑦ Replace the existing RS-232C/RS-422 module with the new one and set the new module's rotary switches in the same way as you wrote down in Step ①.
  - ⑧ Turn on the power supply of the controller unit. Then, from the S10 EXTERNAL SERIAL LINK SYSTEM, set the parameters for the new RS-232C/RS-422 module in the same way as you wrote down in Step ④.
  - ⑨ Check that all the set parameter values are identical to those that were recorded in Step ④.
  - ⑩ Reset the CPU module by pressing the RESET switch at its front.
  - ⑪ Turn off the power supply of the controller unit.
  - ⑫ Remove the RS-232C cable from both the personal computer and CPU module, which were connected together in Step ③.
  - ⑬ Connect the connecting cable back to the RS-232C/RS-422 module, the cable that was removed from it in Step ⑥.
  - ⑭ Set the CPU module's toggle switches in the same way as you wrote down in Step ②.
  - ⑮ Turn on the power supply of the controller unit and check that the RS-232C/RS-422 module is running normally.
- Add-on procedure
    - ① Write down, on a piece of paper, the current settings of the toggle switches (LADDER, MODE, and PROTECT) that are accessible at the front side of the CPU module.
    - ② Ensure that your application system has been shut down. Then, set the CPU module's LADDER switch in STOP position and turn off the power supply of the controller unit.
    - ③ Mount the add-on RS-232C/RS-422 module in place according to the instructions given under "1.2 Mounting the Module."
    - ④ Set the add-on RS-232C/RS-422 module's CN1 MODU and CN2 MODU No. setting rotary switches according to the instructions given under "3.1 Names and Functions of Each Part."
    - ⑤ Connect the personal computer and the CPU module together with the RS-232C cable. Then, turn on the power supply of the controller unit and set the parameters for the add-on RS-232C/RS-422 module from the S10 EXTERNAL SERIAL LINK SYSTEM.  
Note: The S10 EXTERNAL SERIAL LINK SYSTEM does not allow you to use the following functions: "Transmit system program", "Compare system program", and "Delete all channels of task system", all of which are provided for use only on S10/2 $\alpha$  systems.
    - ⑥ Reset the CPU module by pressing the RESET switch at its front.
    - ⑦ Turn off the power supply of the controller unit and connect the connecting cable to the add-on RS-232C/RS-422 module.
    - ⑧ Set the CPU module's toggle switches in the same way as you wrote down in Step ①.
    - ⑨ Turn on the power supply of the controller unit and check that the RS-232C/RS-422 module is running normally.