TABLE OF CONTENTS

INTERBUS INTERFACE MODULE

INTERBUS INTERFACE MODULE

Introduction

This manual describes INTERBUS interface module FC3A-SX5SS1 used with the OpenNet Controller™ to interface with the INTERBUS network, and provides brief description on the INTERBUS network system setup and the INTERBUS interface module specifications.

For general information about safety precautions, installation, wiring, and dimensions, see the OpenNet Controller user's manual EM333.

OpenNet Interface Modules

The OpenNet Controller can be linked to three major open networks; INTERBUS, DeviceNet™, and LONWORKS®. For communication through these networks, OpenNet interface modules are available. Mounting the INTERBUS interface module beside the OpenNet Controller CPU module makes a slave station used as an I/O terminal in an INTERBUS network. The slave station can transfer I/O data to and from the master station just as an ordinary I/O module in a distributed network.

INTERBUS Interface Module Features

Since the INTERBUS interface module conforms to the INTERBUS specifications, the OpenNet Controller can be linked to INTERBUS networks consisting of INTERBUS compliant products manufactured by many different vendors, such as I/O terminals, sensors, drives, operator interfaces, and barcode readers.

The transmit/receive data quantity can be selected from 2, 4, 6, or 8 bytes (64 bits). One INTERBUS interface module enables the OpenNet Controller CPU module to transmit 64 bits and receive 64 bits at the maximum to and from the INTERBUS master station.

About INTERBUS

INTERBUS is a network originally developed for controlling sensors and actuators by Phoenix Contact, Germany, and the specifications were opened in 1987. Today, many major automobile manufacturers in the world use the INTERBUS network.

For detailed information about INTERBUS, read documents published by the INTERBUS CLUB or access the INTERBUS CLUB web site at www.interbusclub.com.

INTERBUS Features

The INTERBUS system is a data ring with a central master-slave access method. It has the structure of a spatially distributed shift register. Every module forms with its registers a part of this shift register ring through which the data is shifted serially from the host controller board. The use of the ring topology in this way offers the possibility of sending and receiving data simultaneously (full duplex) and leads to better diagnostic possibilities when compared to a bus structure.

To simplify system installation, the ring is implemented within one cable line (go and return line within one cable). The system therefor appears as a bus system with branching lines (tree structure).

Since a cyclic scan system is used for data transmission, the overhead except transmission data can be reduced. Consequently, unlike a packet communication system, data transmission efficiency is not reduced when the network contains a large number of nodes.

Remote I/O Stations Require No Address

Remote I/O stations do not require addresses. Since no configuration is needed for installing remote I/O stations, troubles will not be caused by incorrect configuration during maintenance.

Powerful Error Detection Function

The powerful error detection function of INTERBUS makes it possible to detect cable disconnection, remote I/O module failures and to locate the errors, so the system downtime can be minimized.

Transmission Distance and Nodes

The maximum transmission distance is 400 meters between nodes, and the total transmission distance is 12.8 km at the maximum.

Monitoring and Diagnostic Functions

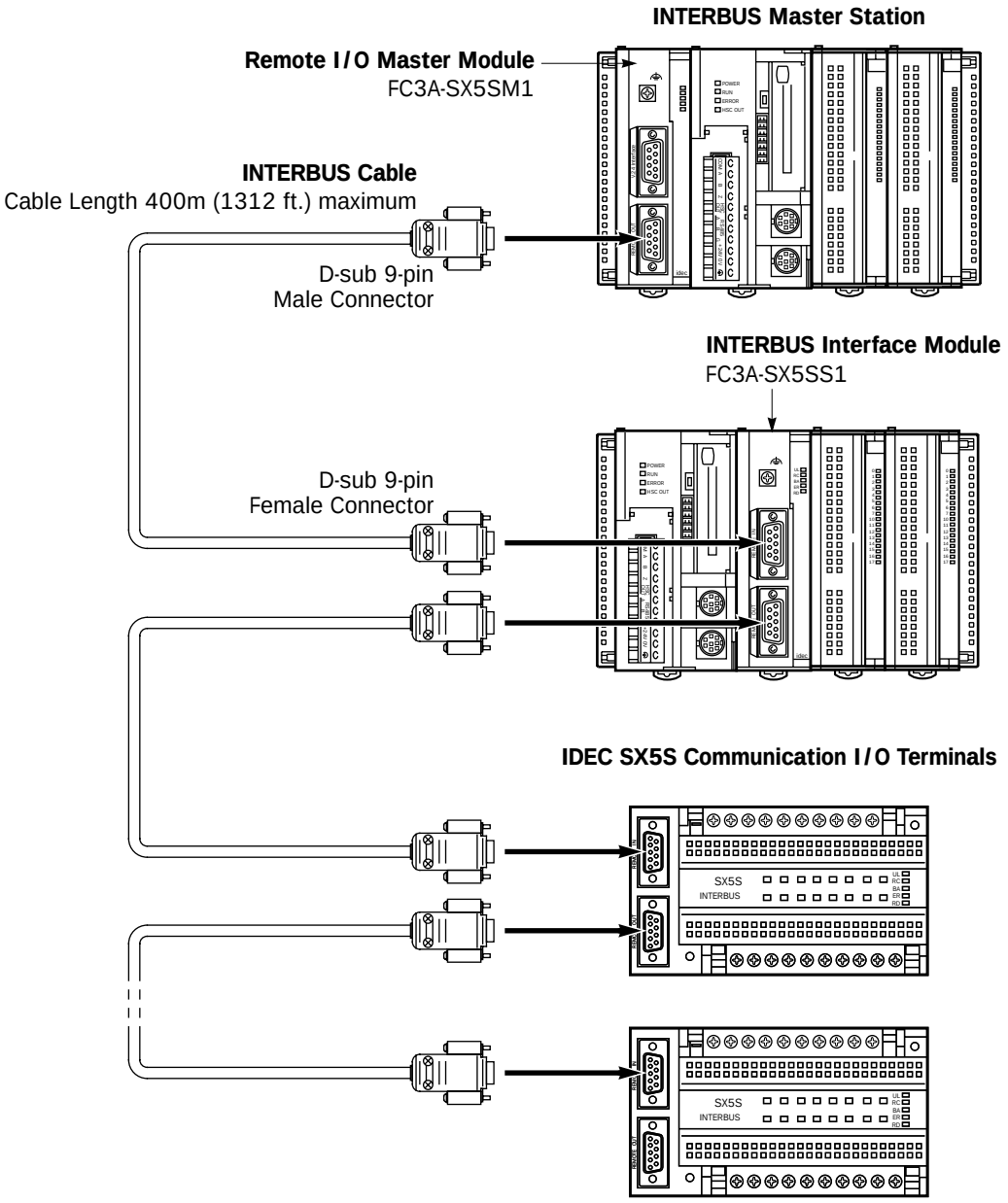
A software tool can be used to monitor the network statuses on a Windows computer, such as configuration, system start-up and diagnostic software CMD available from Phoenix Contact. When an error occurs in the network, cause of the error can be immediately determined from the indicated system error data, status transition number, and error code.

INTERBUS Network System Setup

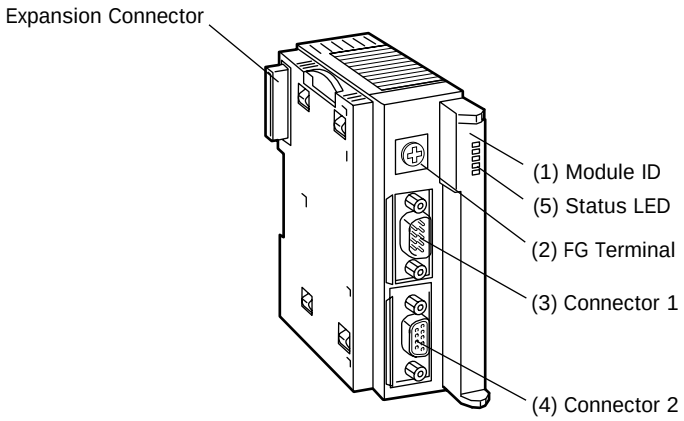
Various INTERBUS compliant devices, such as the INTERBUS interface module and IDEC SX5S communication I/O terminals, can be connected to the INTERBUS network.

The INTERBUS network requires remote I/O master module FC3A-SX5SM1 mounted on the left of the OpenNet Controller CPU module or an INTERBUS master module available from other manufacturers. The OpenNet Controller can be used as a slave station by adding the INTERBUS interface module to the right of the OpenNet Controller CPU module.

A maximum of seven OpenNet interface modules and analog I/O modules can be mounted with one OpenNet Controller CPU module.



INTERBUS Interface Module Parts Description



OpenNet Interface Module for INTERBUS

Module Name	INTERBUS Interface Module
Type No.	FC3A-SX5SS1

(1) Module ID FC3A-SX5SS1 indicates the INTERBUS interface module ID.

(2) FG Terminal Frame ground
Connect the FG terminal to a proper ground using a UL1015 AWG22 or UL1007 AWG18 wire (grounding resistance 100Ω maximum).

(3) Connector 1 REMOTE IN for connecting an input communication cable

(4) Connector 2 REMOTE OUT for connecting an output communication cable

(5) Status LED Indicates operating status

UL	POWER_ON Green ON: Power is on
RC	REMOTE_BUS_CHECK Green ON: Remote IN cable is connected correctly
BA	BUS_ACTIVE Green Flash: Ready for transmitting data Green ON: Transmitting data
ER	MODULE_ERROR Red ON: Module error
RD	REMOTE_BUS_DISABLE Red ON: Network error

INTERBUS Interface Module Specifications

Normal Operating Conditions

Operating Ambient Temperature	0 to +55°C (no freezing)
Storage Temperature	–25 to +70°C (no freezing)
Operating Humidity	Level RH1 30 to 95% (no condensation)
Pollution Degree	2 (IEC 60664)
Corrosion Immunity	Free from corrosive gases
Altitude	Operation: 0 to 2000m Transportation: 0 to 3000m
Vibration Resistance	10 to 57 Hz, amplitude 0.075 mm; 57 to 150 Hz, acceleration 9.8 m/s^2 (1G); 10 sweep cycles each in 3 axes (total 80 minutes) (IEC 61131)
Shock Resistance	147 m/s^2 (15G), 11 ms, 3 shocks each in 3 axes (IEC 61131)

Power Supply (supplied from the OpenNet Controller CPU module)

Dielectric Strength	Between power terminal on CPU module and FG: 500V AC, 1 minute
Insulation Resistance	Between power terminal on CPU module and FG: 10 MΩ (500V DC megger)
Power Consumption	11.8W (24V): CPU module + INTERBUS interface module + 48 I/Os (32-DC input module + 16-relay output module) 21.4W (24V): CPU module + INTERBUS interface module + 128 I/Os (32-DC input module × 2 + 16-DC input module + 16-relay output module × 3)

Grounding

FG Terminal	M3 sems (tightening torque: 0.6 to 1.0 N·m)
Grounding Resistance	100Ω maximum
Grounding Wire	UL1015 AWG22, UL1007 AWG18

Weight

Weight	Approx. 200g

Communication Specifications

Network Protocol	INTERBUS
Transmission Speed	500 kbps
Transmission Distance	Between master and remote bus station: 400m maximum Between remote bus stations: 400m maximum Remote bus total length: 12.8 km maximum
Quantity of Nodes	512 nodes (remote I/O stations) maximum according to INTERBUS specifications
I/O Points per Node	512 points maximum (256 inputs and 256 outputs) according to INTERBUS specifications
Branch Levels	16 maximum (INTERBUS device levels 0 through 15)
Remote I/O Connector (REMOTE IN)	D-sub 9-pin male connector on the INTERBUS interface module
Remote I/O Connector (REMOTE OUT)	D-sub 9-pin female connector on the INTERBUS interface module
Network Cable	INTERBUS cable
Electrostatic Discharge Severity Level	ESD-3 (network interface)

Link Registers for INTERBUS Network Communication

INTERBUS network communication data is stored to link registers in the OpenNet Controller CPU module and the data is communicated through the INTERBUS interface module.

Since seven functional modules including the INTERBUS interface module can be mounted with one OpenNet Controller CPU module, link registers are allocated depending on the position where the INTERBUS interface module is mounted.

Link Register Allocation Numbers

Allocation Number	Area	Function	Description	R/W
L*00	Data area	Receive data	Stores received data from the network	Read
L*01	Data area	Receive data	Stores received data from the network	Read
L*02	Data area	Receive data	Stores received data from the network	Read
L*03	Data area	Receive data	Stores received data from the network	Read
L*04	Data area	Transmit data	Stores transmit data for the network	Write
L*05	Data area	Transmit data	Stores transmit data for the network	Write
L*06	Data area	Transmit data	Stores transmit data for the network	Write
L*07	Data area	Transmit data	Stores transmit data for the network	Write
L*12	Status area	Error data	Stores various error codes	Read
L*13	Status area	I/O counts	Stores the byte counts of transmit/receive data	Read
L*14	Status area	Connection status	Stores the allocation choice byte	Read
L*24	Reserved area	Software version	Stores the system software version	Read

Note: A number 1 through 7 comes in place of * depending on the position where the functional module is mounted, such as OpenNet interface module or analog I/O module. Consequently, operand numbers are automatically allocated to each functional module in the order of increasing distance from the CPU module, starting with L100, L200, L300, through L700.

Error Data (Status Area) L*12

L*12	b15	b14	b13	b12-b0: unused

When a hardware error occurs, the ER LED on the INTERBUS interface module goes on, and a corresponding bit in the link register goes on. The ER LED goes off when the cause of the error is removed. The error data bit remains on until the CPU is powered up again or reset by pressing the communication enable button on the CPU module.

b15 (initialization error)
This bit goes on when the CPU module fails to acknowledge the completion of initialization for communication with the INTERBUS interface module.

b14 (I/O quantity error)
This bit goes on when the slave station transmit/receive quantity is set to an invalid or different values (see page 7).

b13 (I/O error)
This bit goes on when an error occurs during communication through the CPU bus.

I/O Counts (Status Area) L*13

L*13	b15-b12: transmit bytes	b11-b8: receive bytes	b7-b0: unused

This link register stores the transmit and receive byte counts selected in the Function Area Setting > Open Bus in WindLDR™.

Connection Status (Status Area) L*14

L*14	b15-b8: allocation choice	b7-b0: unused

This link register stores the data of the allocation choice byte.

Software Version (Reserved Area) L*24

L*24	b15-b12: major revision	b11-b8: minor revision	b7-b0: unused

This link register stores the system software version number. [Example] Version 1.3 — 1: major revision, 3: minor revision

Function Area Setting for INTERBUS Slave Station

The quantity of transmit/receive data for INTERBUS network communication is specified using the Function Area Setting in WindLDR. The OpenNet Controller CPU module recognizes all functional modules, such as OpenNet interface modules and analog I/O modules, automatically at power-up and exchanges data with the INTERBUS master station through the link registers allocated to each slave station (node).

The quantity of transmit/receive data for INTERBUS network communication can be 2, 4, 6, or 8 bytes each and the same quantity must be selected for transmit and receive data.

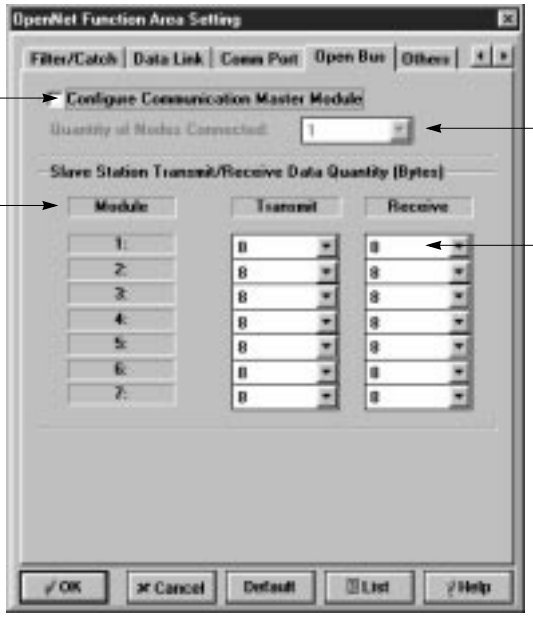
Since these settings relate to the user program, the user program must be downloaded to the OpenNet Controller after changing any of these settings.

Programming WindLDR

1. From the WindLDR menu bar, select **Configure** > **Function Area Settings**. The Function Area Setting dialog box appears.
2. Select the **Open Bus** tab.

Configure Communication Master Module Check Box
Check this box only when the remote I/O master module is used.

Slave Station Transmit / Receive Data Quantity (Bytes)
When using OpenNet interface modules for INTERBUS, DeviceNet, or LONWORKS, specify the data bytes to communicate through each OpenNet interface module.



Quantity of Nodes Connected
When using the remote I/O master module, specify the quantity of nodes from 1 through 32.

Transmit / Receive Bytes 2, 4, 6, or 8 bytes (default: 8 bytes)
This value determines the data quantity 1 through 8 bytes (64 bits) to communicate with the INTERBUS master module. Select the same quantity for transmit and receive bytes. For the example on the next page, select 8 transmit bytes and 8 receive bytes for Module 1.

3. Select transmit and receive data bytes for module position 1 through 7 where the INTERBUS interface module is mounted.
4. Click the **OK** button and download the user program to the OpenNet Controller.

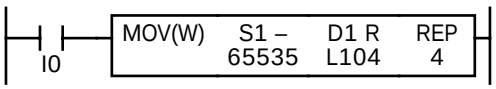
Programming Transmit / Receive Data Using WindLDR

The OpenNet interface module exchanges data between the open network and the link registers in the CPU module allocated to the OpenNet interface module, depending on the slot where the OpenNet interface module is mounted.

To create a communication program for an OpenNet interface module, first determine the slot number where the OpenNet interface module is mounted, and make a program to write data to link registers allocated to transmit data and to read data from link registers allocated to receive data.

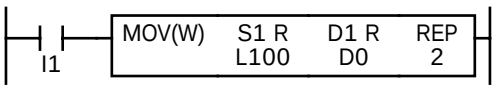
Example: When an INTERBUS interface module is mounted in the first slot of all functional modules

- **Transmit Data**



65535 → L104 through L107
When input I0 is on, constant 65535 (FFFFh) designated by source operand S1 is moved to four link registers L104 through L107 designated by destination operand D1. All 64 bits (8 bytes) in link registers L104 through L107 are turned on. Since link registers L104 through L107 transmit data, the data is transmitted to the network.

- **Receive Data**



L100·L101 → D0·D1
When input I1 is on, 32-bit (4-byte) data in two link registers L100 and L101 designated by source operand S1 is moved to data registers D0 and D1 designated by destination operand D1. Since link registers L100 and L101 receive data, communication data read to L100 and L101 is moved to data registers D0 and D1.

Note: In the example above, the INTERBUS slave module transmits and receives a different quantity of data. When the quantities of transmit and receive data are different, select the larger bytes for the transmit and receive data quantities in the Function Area Settings (see page 7).

Starting Operation

1. Set up the OpenNet Controller CPU and INTERBUS interface modules, and connect the INTERBUS interface module to the INTERBUS network using INTERBUS cables.
2. Power up the CPU module and download the user program to the CPU module using WindLDR.
3. Start the CPU module to run, then INTERBUS communication starts.

The delay until the communication starts after power-up depends on the size of the user program and the system setup.

While the CPU is stopped, data exchange between the CPU and INTERBUS interface modules is halted, but communication with the INTERBUS network continues.

Communication between the INTERBUS interface module and the INTERBUS network is asynchronous with the data exchange between the CPU module and the INTERBUS interface module. Data exchange between the CPU module and the INTERBUS interface module occurs in every user program scanning in the CPU module.

Calculation of the INTERBUS Cycle Time

The I/O data is refreshed continuously through the INTERBUS network. The cycle time of the INTERBUS system depends on few factors and increases almost linearly with an increasing number of I/O points. Due to the high effectiveness of the protocol, the greater part of the cycle time is determined by the number of I/O points. However, it is also required to consider quantities such as the number of installed remote bus devices, the duration of the check sequence, the firmware runtime, and the signal runtime on the transmission medium.

The cycle time (refresh time) can be calculated according to:

$$t_{cycle} = \{K \times 13 \times (6 + n) + 4 \times m\} \times t_{Bit} + t_{SW} + t_{PH} + r \times t_W$$

whereby

t_{cycle}	INTERBUS cycle time (1 scan time)
K	1.15
n	Number of user data bytes
m	Number of installed remote bus devices
t_{Bit}	Bit duration (0.002 ms)
t_{SW}	Firmware run time (1 ms)
t_{PH}	Signal run time on the transmission medium (0.016 ms/km)
r	1
t_W	Conversion time (13×2 µs)

Example: Calculating the INTERBUS Cycle Time

When an INTERBUS network consists of four INTERBUS slave modules (transmit/receive data: 8 bytes each) and the distance between slave stations is 100m each, the cycle time is calculated as follows:

$$t_{cycle} = \{1.15 \times 13 \times (6 + 32) + 4 \times 4\} \times 0.002 + 1 + 0.016 \times 0.4 + 13 \times 0.002$$
$$= 2.2 \text{ ms}$$

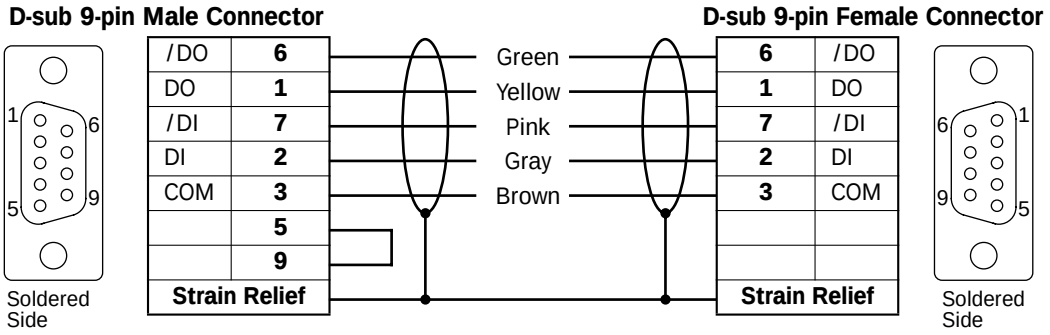
Precautions for Wiring INTERBUS Cable

For wiring the remote I/O master and slave modules, use the INTERBUS cable made of the remote bus cable with D-sub 9-position male and female connectors. The remote bus cable is available from Phoenix Contact. When ordering the remote bus cable from Phoenix Contact, specify the Order No. and cable length in meters.

Remote Bus Cable Type No.

Phoenix Type	Order No.	Specification	Used for
IBS RBC METER-T	28 06 28 6	Standard, 3 x 2 x 0.22 mm^2	Fixed routing
IBS RBC METER/F-T	27 23 12 3	Highly flexible, 3 x 2 x 0.25 mm^2	Flexible power conduits and machinery components which are frequently in motion
IBS RBC METER/E-T	27 23 14 9	Underground, 3 x 2 x 0.22 mm^2	Fixed routing indoors, outdoors or underground

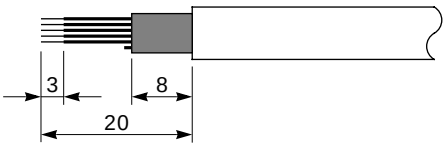
Cable Connector Pinouts



Bridge pins 5 and 9 inside the housing of the male connector.

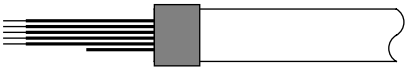
Use inch-sized screws (UNC4-40) to fasten the cable connectors to INTERBUS ports.

Stripping and Clamping Cable Ends



First, strip the cable sheath 20 mm from both ends of the cable and shorten the braided shield by 12 mm.

Bare the wire ends 3 mm. Trim the unused white wire.



Next, place the braided shield back over the cable sheath.

Clamp the shield under the strain relief in the connector housing for conductive connection with the housing.

- Do not install the INTERBUS cable in parallel with or close to motor lines. Keep the INTERBUS cable away from noise sources.
- Turn power off before wiring the INTERBUS cable. Make sure of correct wiring before turning power on.
- Use a special INTERBUS cable and connect the cable as shown above. Use D-sub connectors with metal or metal-coated housing. Connect the cable shield with the connector housing electrically.
- Leave open the remote out connector at the last station in the network.
- Supply power to each slave station or to each group of stations separately.
- Master and slave stations may be powered up in any order. But, if a slave station is not powered up while the master is in preparation for transmission, a network error will result.
- Causes of network errors include disconnection or short-circuit of the network cable, strong external noise, invalid command sent to the master station, momentary power voltage drop below the minimum power voltage, faulty transmission line, incorrect cable, and transmission longer than the rated distance.

INTERBUS Network Troubleshooting

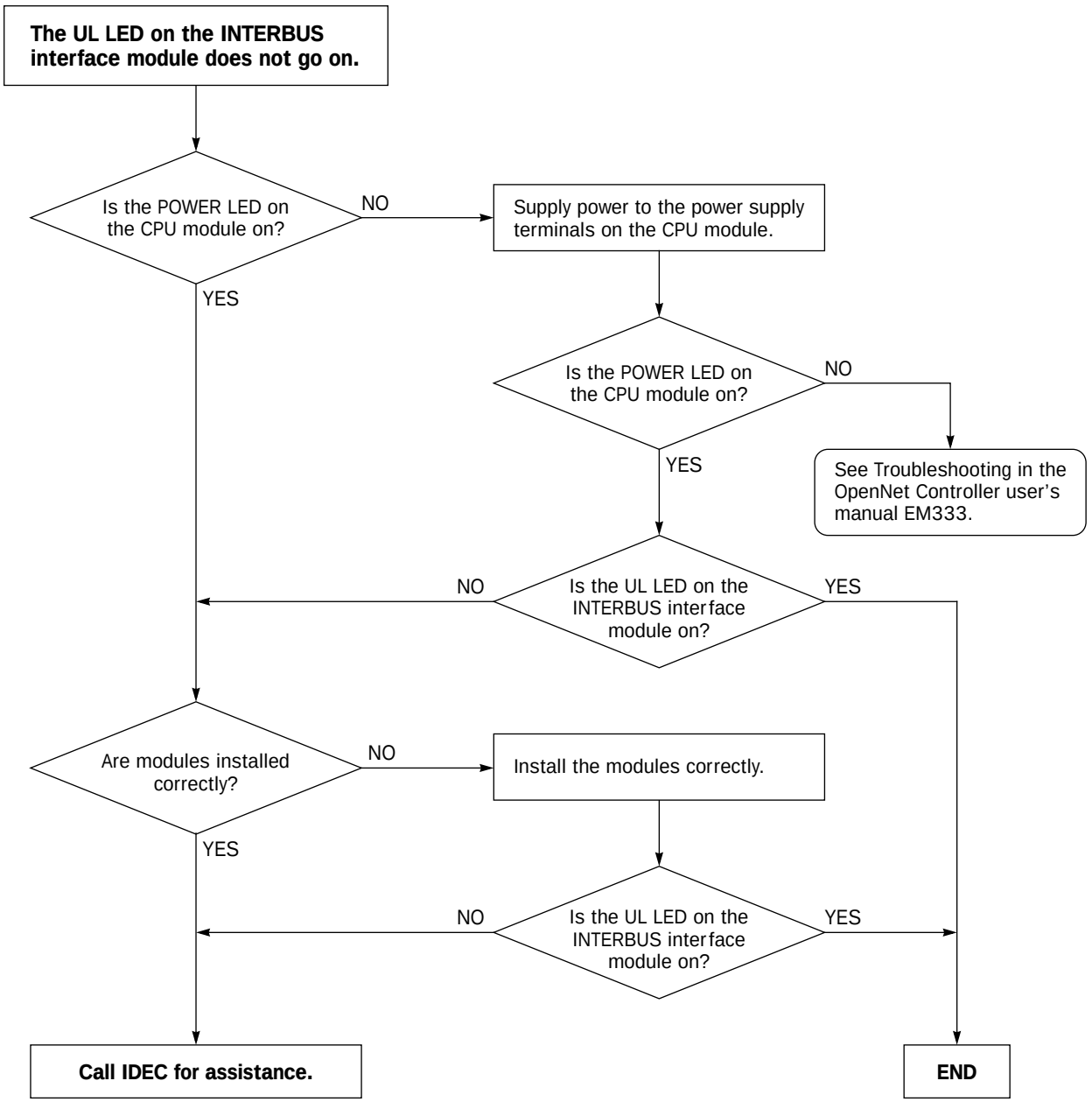
Five LED indicators are provided on the INTERBUS interface module to indicate operating status. When a trouble occurs during INTERBUS communication, these status LEDs go on or off depending on the error. When the INTERBUS interface module does not operate normally, locate the error referring to the troubleshooting diagrams below.

Probable Causes for Network Errors

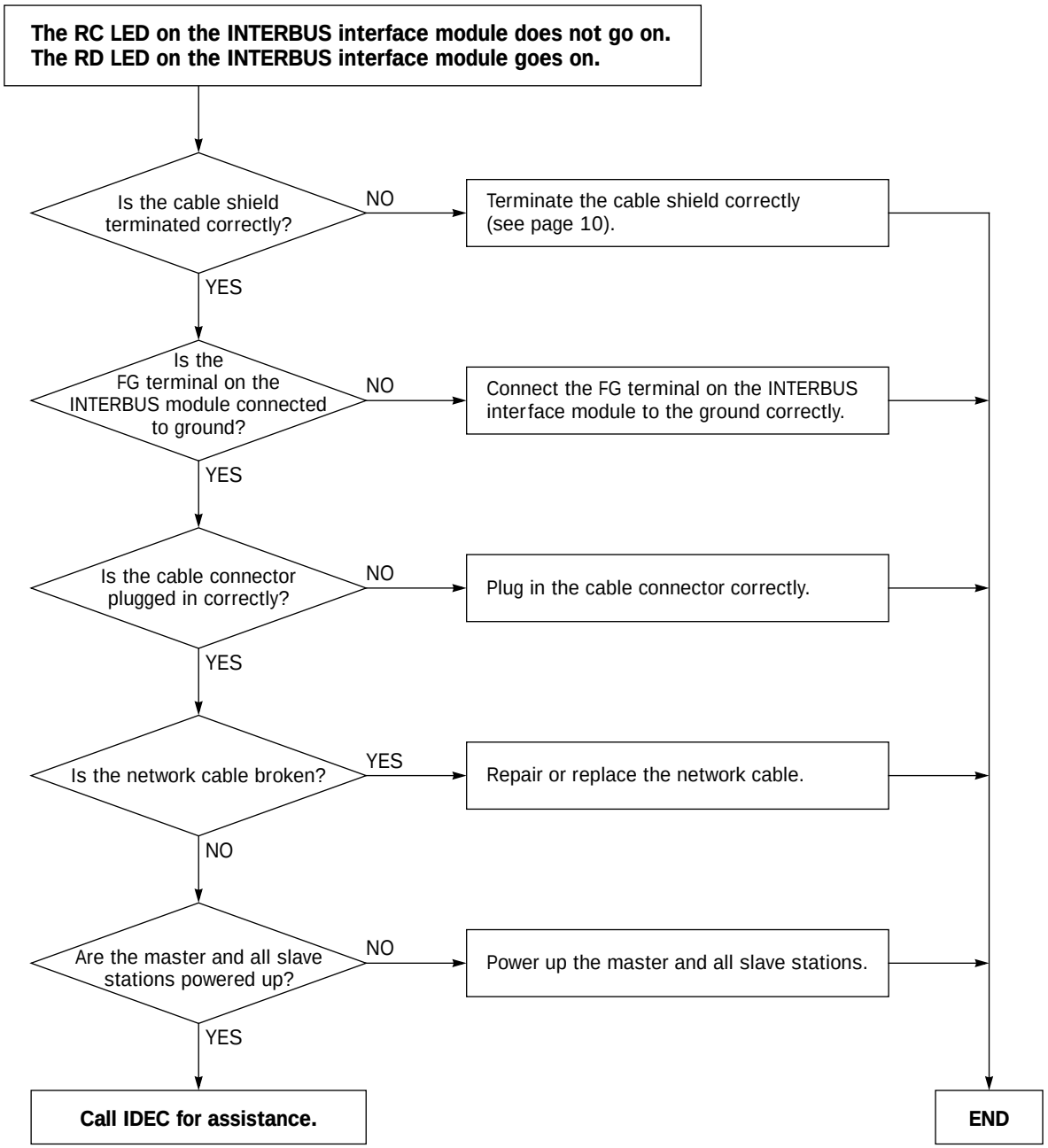
When a trouble occurs during INTERBUS communication, the following causes are suspected.

- Strong external noise
- The power voltage to the INTERBUS interface module has dropped below the minimum operating voltage (at least momentarily).
- Use of a faulty communication line, incorrect cable, or transmission beyond the rated distance

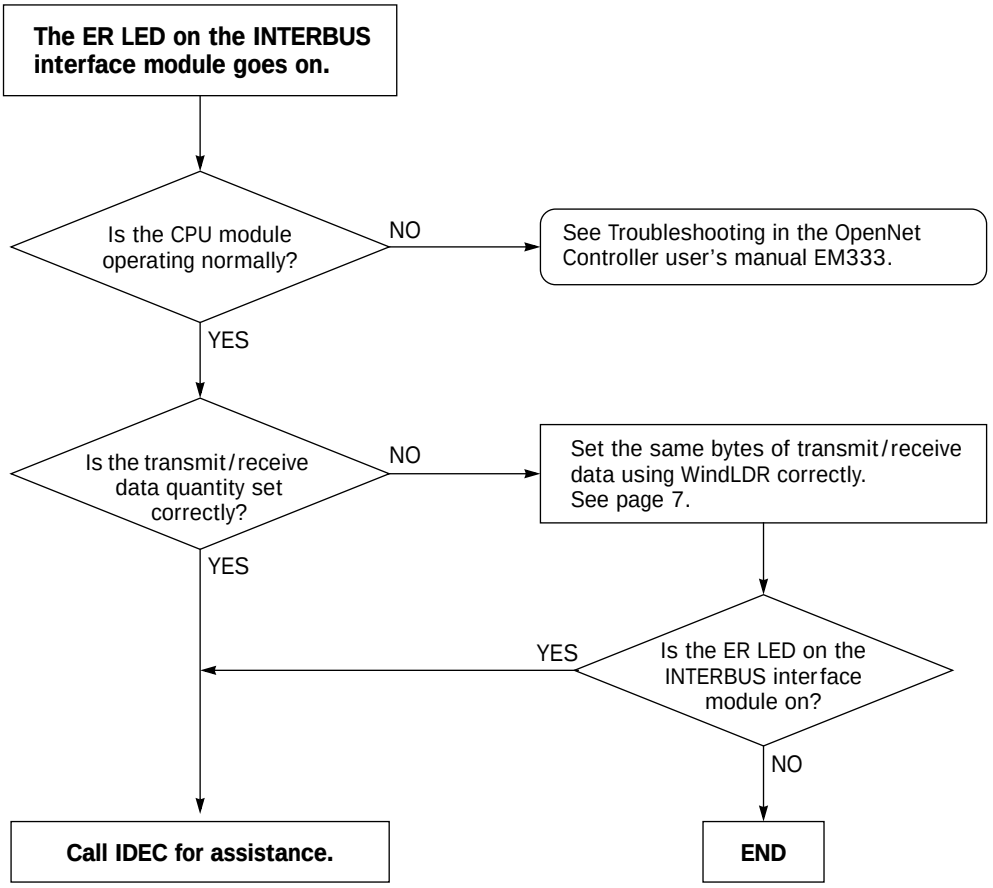
Troubleshooting Diagram 1



Troubleshooting Diagram 2



Troubleshooting Diagram 3



INDEX