To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

H8S Family

Master Transmission/Slave Reception Example Using IEBus Controller

Introduction

Unit A performs master transmission and slave reception between other units using the IEBus controller included in the H8S/2258F.

Target Device

H8S/2258F

Contents

1. Specifications

1.1 Overview

The configuration of the IEBus connections is shown in figure 1. This application note describes a software example for master transmission and slave reception in which unit A uses the IEBus controller of the H8S/2258F and units B and C are virtual units. Note that the IEBus interface of virtual units B and C is controlled using functions equivalent to those of the IEBus controller of the H8S/2258F. Furthermore, unit A uses the HA12187FP as a driver and transceiver for the IEBus interface.

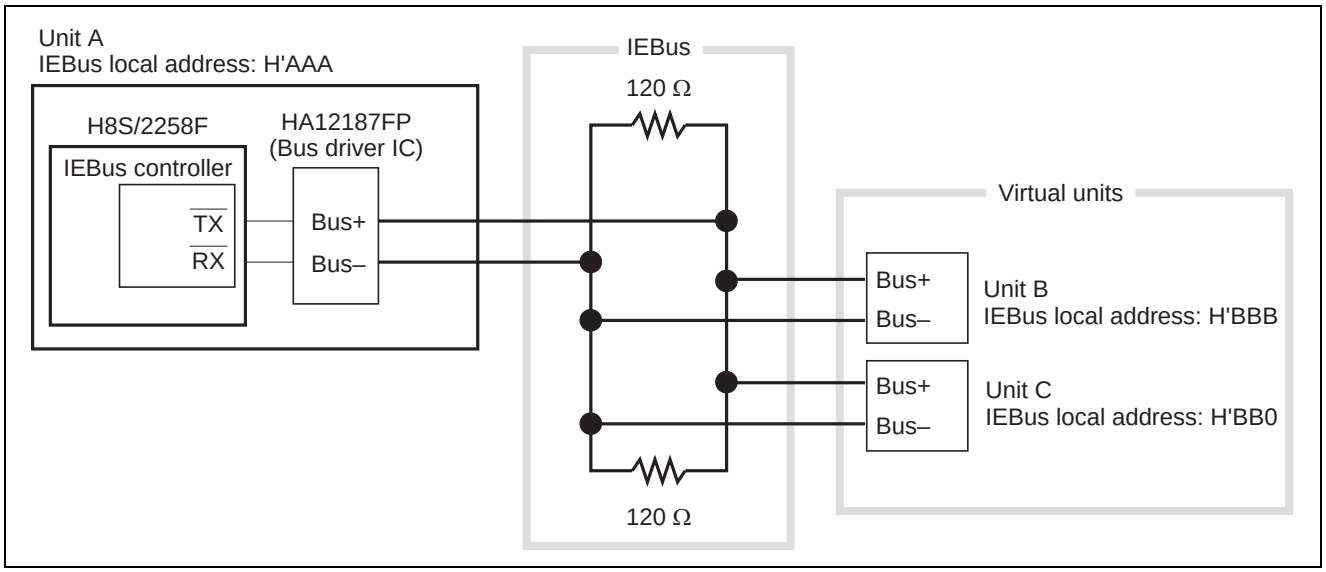


Figure 1 Configuration of IEBus Connections

1.2 Usage Conditions of IEBus

The IEBus controller for the units operates under the following conditions.

(1) System clock: 12.58 MHz
(2) Communications mode: Mode 1
(3) Control bits in IEMCR: Always fixed at H'0F (used for master transmission and slave reception only)
(4) Number of retransmissions in case of arbitration loss: 3
(5) Local addresses: Unit A = H'AAA
Unit B = H'BBB
Unit C = H'BB0

1.3 Master Transmission/Slave Reception Operation

The IEBus controller of the H8S/2258F uses the data transfer controller (DTC) to transmit and receive data during master transmission and slave reception. However, during master transmission the first byte of data is written to the transmit buffer by software processing, without using the DTC. For details, see section 2.3, Error Handling During Master Transmission.

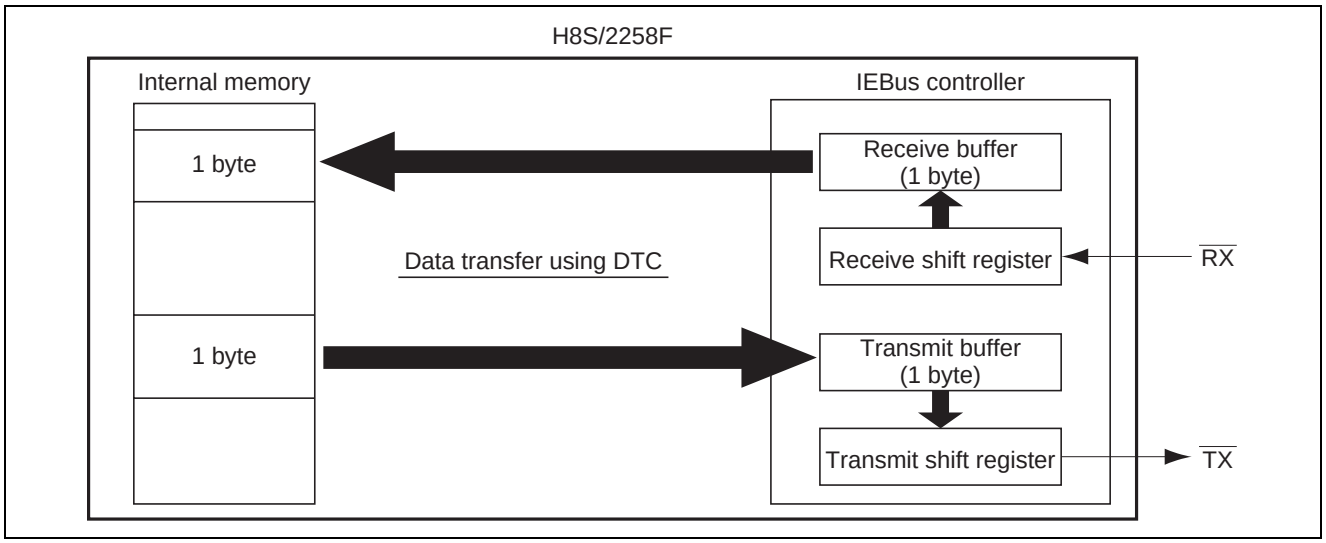


Figure 2 Data Transmission and Reception Using DTC

1.4 Specifications of Communication Operations

Unit A uses master transmission and slave reception to perform the communication operations described below. Note that each unit may begin master transmission arbitrarily. Consequently, CSMA/CD access control is used as the basis for determining which unit has exclusive control of the bus in cases where more than one unit attempts to initiate master transmission simultaneously.

1.4.1 Master Transmission

Three frames of data are transmitted to each unit.

(1) Normal communication (broadcast bit = 1, slave address = H'BBB) is used to transmit 32 bytes of data to unit B. The 32 bytes of data transmitted consists of H'11.
(2) Simultaneous broadcast communication (broadcast bit = 0, slave address = H'FFF) is used to transmit 32 bytes of data to units B and C. The 32 bytes of data transmitted consists of H'22.
(3) Group broadcast communication (broadcast bit = 0, slave address = H'BBB) is used to transmit 32 bytes of data to units B and C. The 32 bytes of data transmitted consists of H'33.

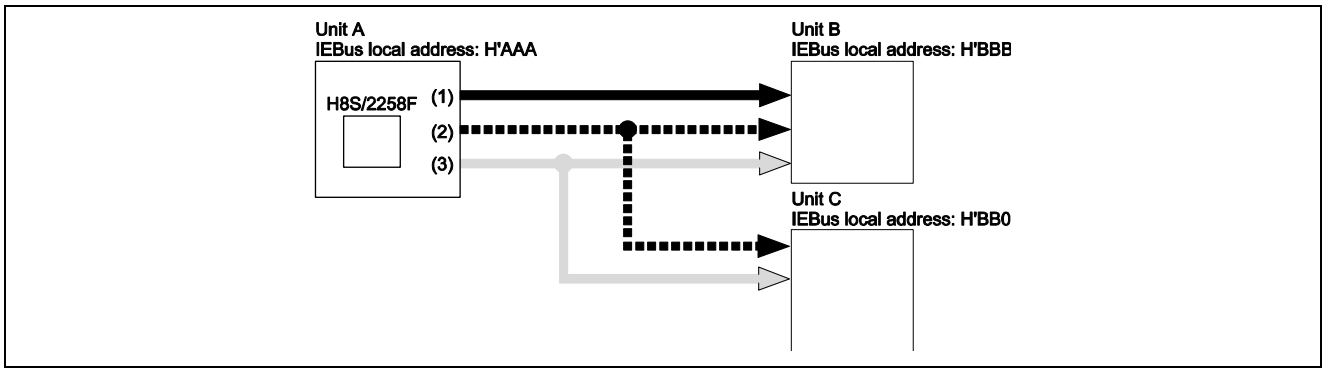


Figure 3 Master Transmission Operation of Unit A

1.4.2 Slave Reception

Three frames of data to be received are transmitted from each unit.

(1) 32 bytes of data transmitted from unit B using normal communication (broadcast bit = 1, slave address = H'AAA) is received. The data values are random.
(2) 32 bytes of data transmitted from unit B using simultaneous broadcast communication (broadcast bit = 0, slave address = H'FFF) is received. The data values are random.
(3) 32 bytes of data transmitted from unit C using group broadcast communication (broadcast bit = 0, slave address = H'AAA) is received. The data values are random.

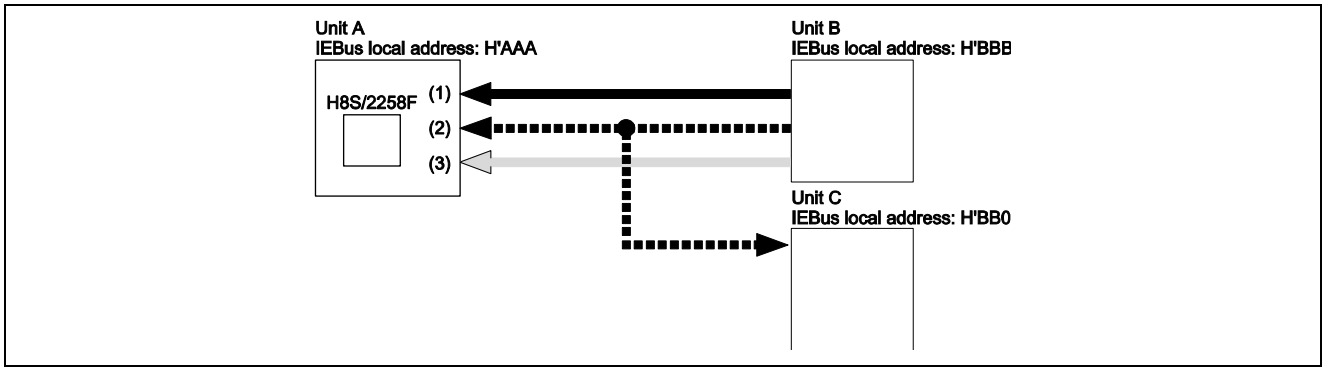


Figure 4 Slave Reception Operation of Unit A

1.5 Specifications of Communication Error Handling

Unit A performs the following error handling operations.

(1) Errors during Master Transmission

The IEBus controller of the H8S/2258F detects the errors listed below. When an error occurs, the frame that was being transmitted is discarded and retransmitted. Retransmission is repeated until transmission completes normally.

- Arbitration Loss (AL)
  Occurs when arbitration is lost three times.
- Underrun Error (UE)
  Occurs during data transmission when no data is waiting in the transmit buffer.
- Timing Error (TTME)
  Occurs when data is not transferred according to the timing specified by the IEBus protocol.
- Transmit Frame Over Maximum Number of Transfer Bytes (RO)
  Occurs when data equivalent to the maximum byte length defined by the communications mode has been transmitted but the transmission has not completed.
- Acknowledge Error (ACK)
  Condition 1: A NAK was received before transmission of the data field.
  Condition 2: During transmission, data equivalent to the maximum byte length defined by the communications mode has been transmitted but no ACK has been received.

(2) Errors during Slave Reception

The IEBus controller of the H8S/2258F detects the errors listed below. When an error occurs, the frame that was being received is discarded. After discarding the frame, the controller returns to ready-to-receive state for the next frame.

- Overrun Error (OVE)
  Occurs when the next data is received before the data in the receive buffer has been read.
- Timing Error (RTME)
  Occurs when data is not transferred according to the timing specified by the IEBus protocol.
- Receive Frame Over Maximum Number of Transfer Bytes (DLE)
  Occurs when data equivalent to the maximum byte length defined by the communications mode has been received but reception has not completed.
- Parity Error (PE)
  Occurs when a parity error cannot be cleared even after reception of a data field has proceeded up to the maximum byte length defined by the communications mode.

2. Principles of Operation

The principles of operation for master transmission and slave reception are illustrated below.

2.1 Master Transmission Operation

The principles of operation of the IEBus controller during master transmission are illustrated in figure 5, and a flowchart of master transmission operation is shown in figure 6. The DTC is used to transmit data fields. However, during master transmission the first byte of data is written to the transmit buffer by software processing, without using the DTC. For an explanation of the reasons for this, see section 2.3, Error Handling during Master Transmission.

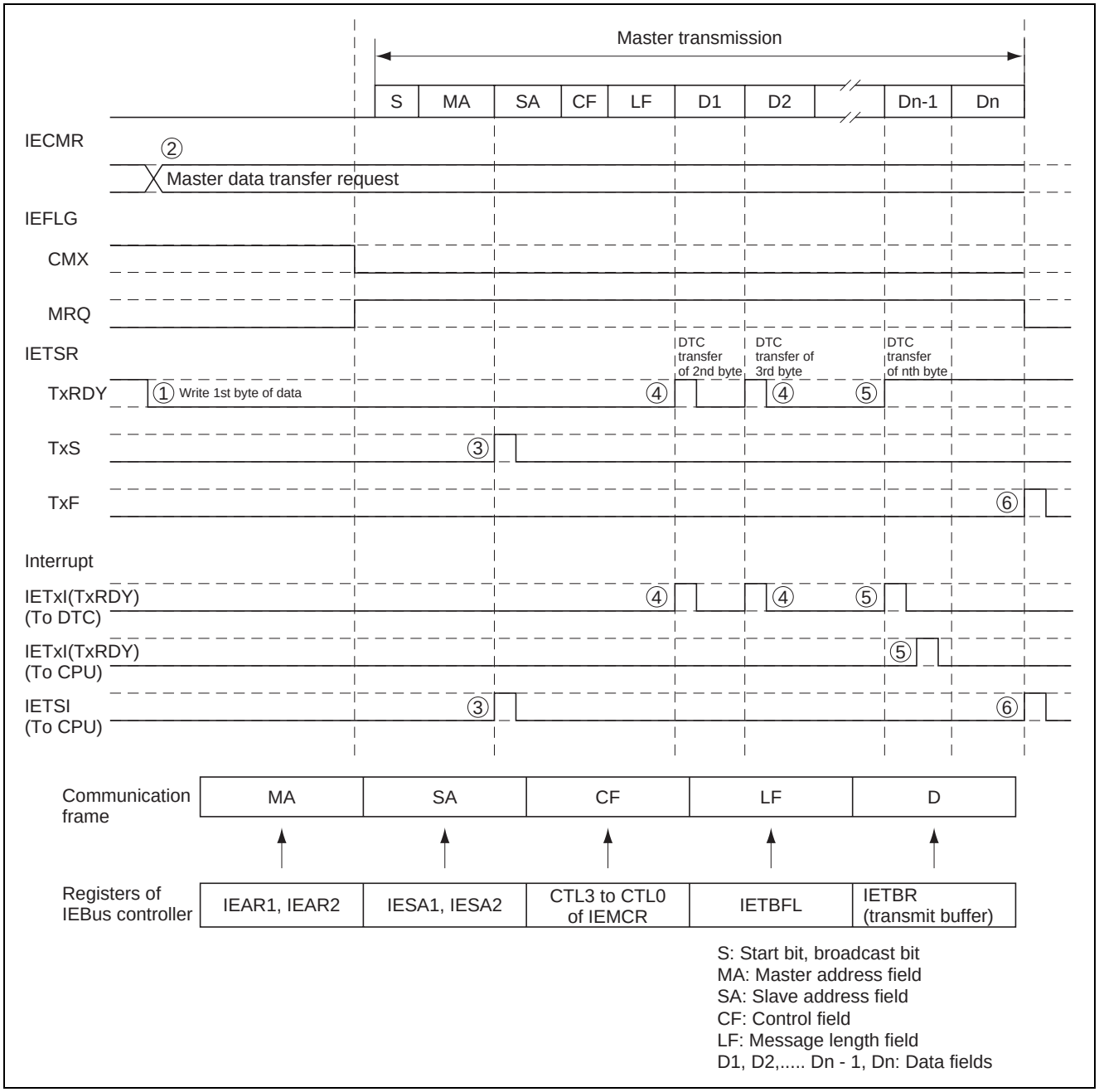


Figure 5 Principles of Operation during Master Transmission

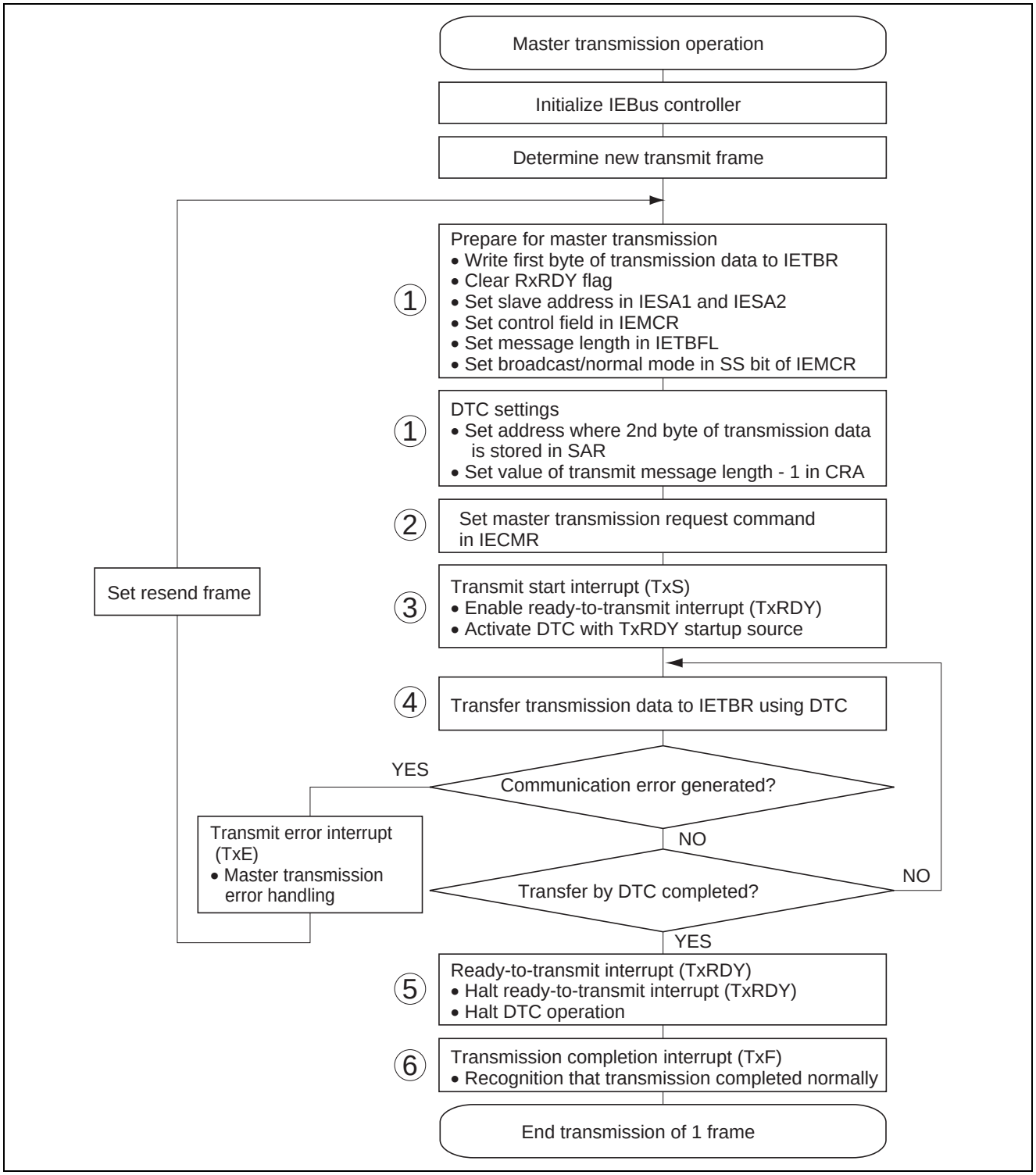


Figure 6 Flowchart of Master Transmission Operation

2.2 Slave Reception Operation

The principles of operation of the IEBus controller during slave reception are illustrated in figure 7, and a flowchart of slave reception operation is shown in figure 8. The DTC is used to receive data fields.

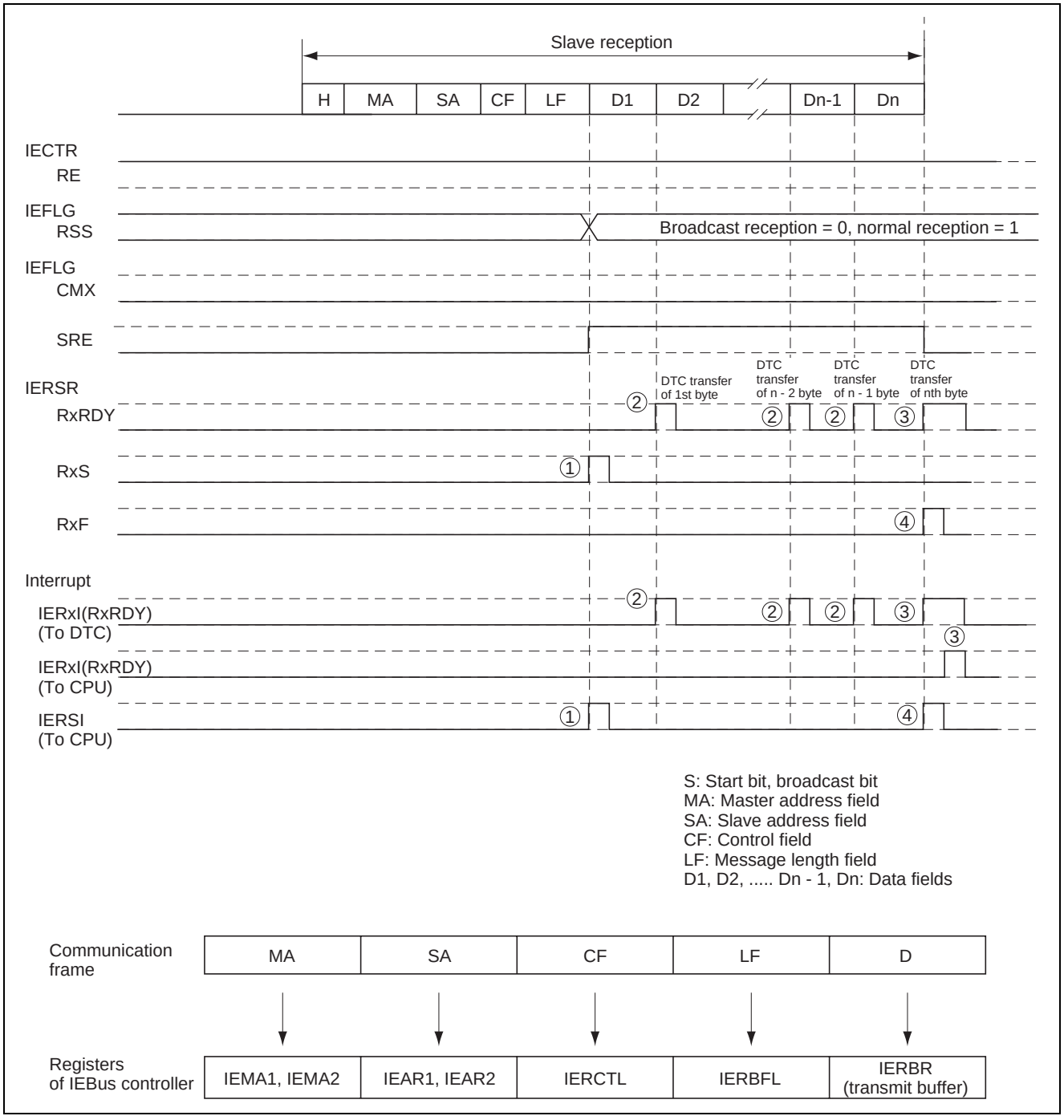


Figure 7 Principles of Operation during Slave Reception

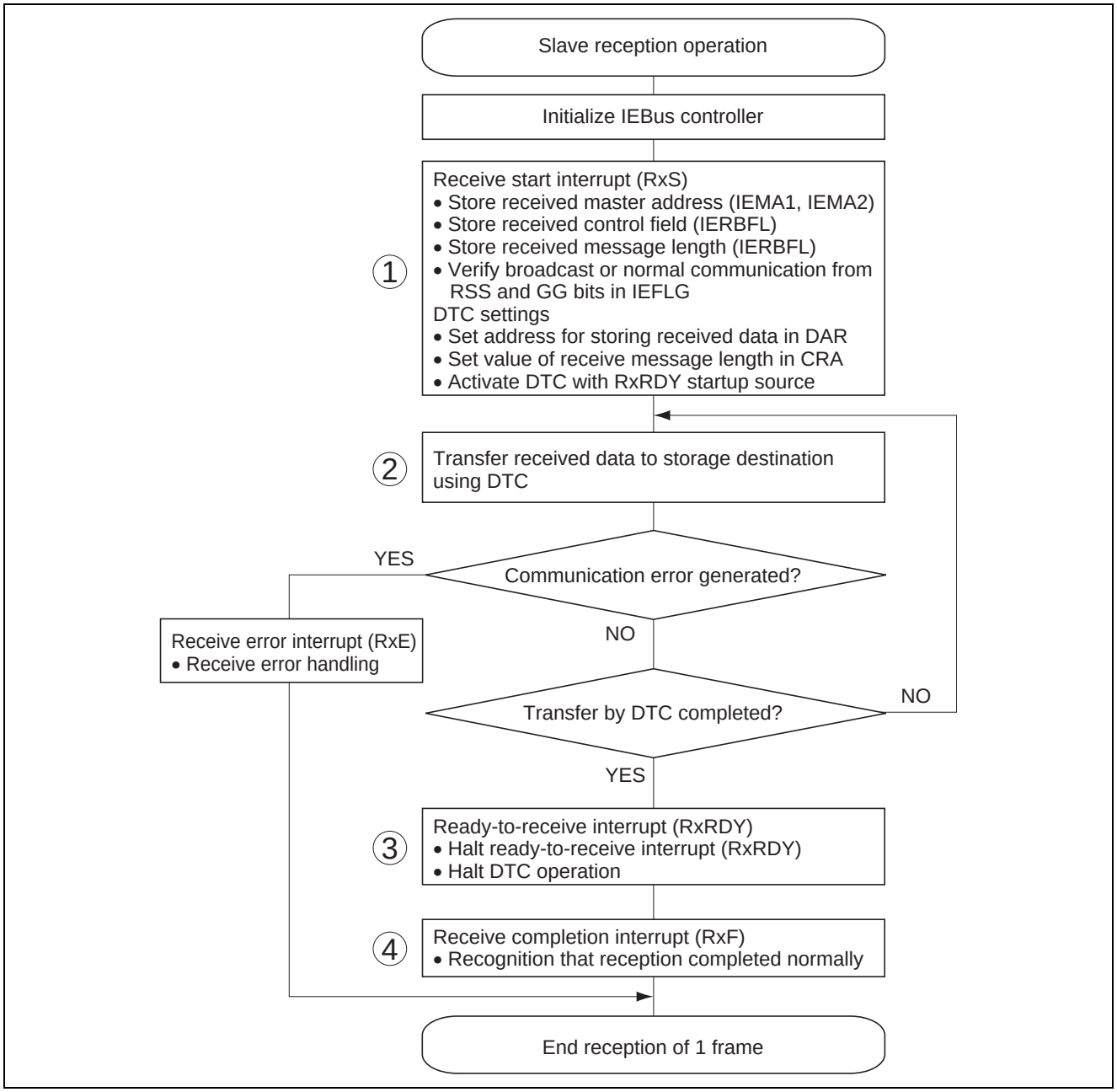


Figure 8 Flowchart of Slave Reception Operation

2.3 Error Handling during Master Transmission

Operation when a timing error is generated is illustrated in figure 9. When a timing error or other error occurs during data transmission ([1]), in some cases the DTC will already have sent the next transmission data to the transmit buffer and the TxRDY flag, which is the startup source of the DTC, will already have been cleared ([2]).

Performing a retransmission in this state would cause the data remaining in the transmit buffer (the data from the previous frame) to be transmitted as the first byte of the data field ([3]).

In this application note, during master transmission the first byte of data in the data field is written to the transmit buffer by software processing, without using the DTC, to avoid the problem described above. The second and subsequent bytes are then transferred using the DTC.

In this sample task the settings for the DTC's SAR (DTC transfer source address register) and CRA (DTC transfer count register A) are as follows:

- Address in internal memory for storing second and subsequent bytes of data: SAR
- Number of data bytes specified in messages minus 1: CRA

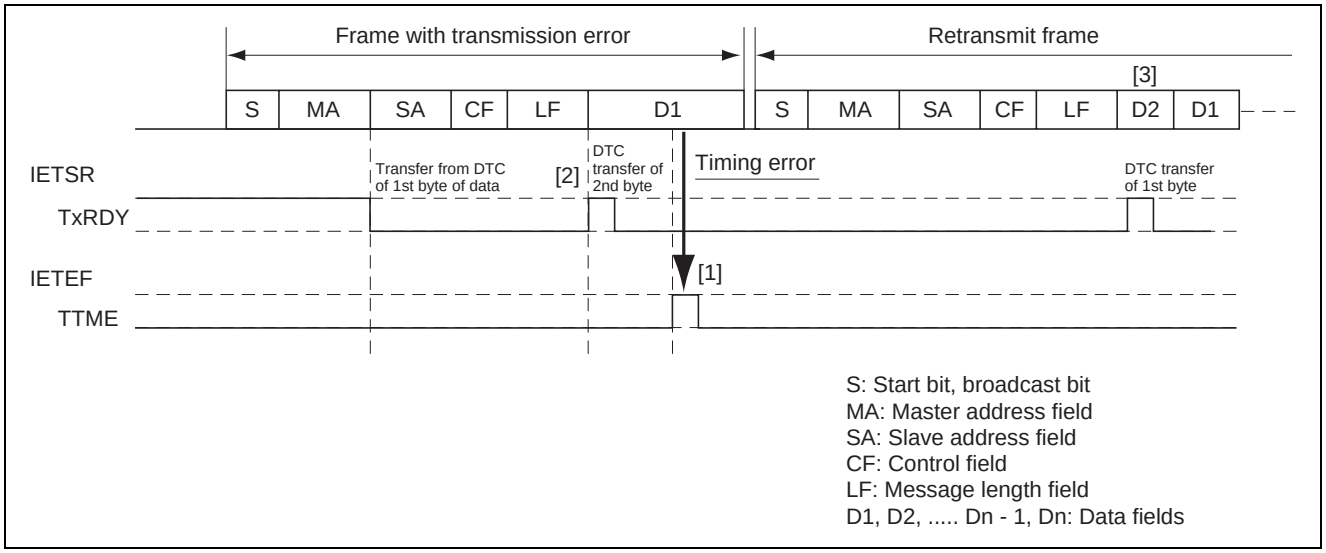


Figure 9 Operation at Generation of Timing Error

If an error occurs during data transmission, the IEBus controller returns to ready-to-transmit state from standby state. Furthermore, DTC transfer processing is halted when the following errors occur.

(1) Handling of Arbitration Loss (AL) Errors

The handling of arbitration loss errors is illustrated in figure 10.

[1] When arbitration loss occurs three times the IEBus controller sets the AL flag and transitions to the wait state.

[2] A TxE interrupt is generated. The arbitration loss error is detected within this interrupt.

[3] The AL flag is cleared within the interrupt.

[4] Retransmission takes place.

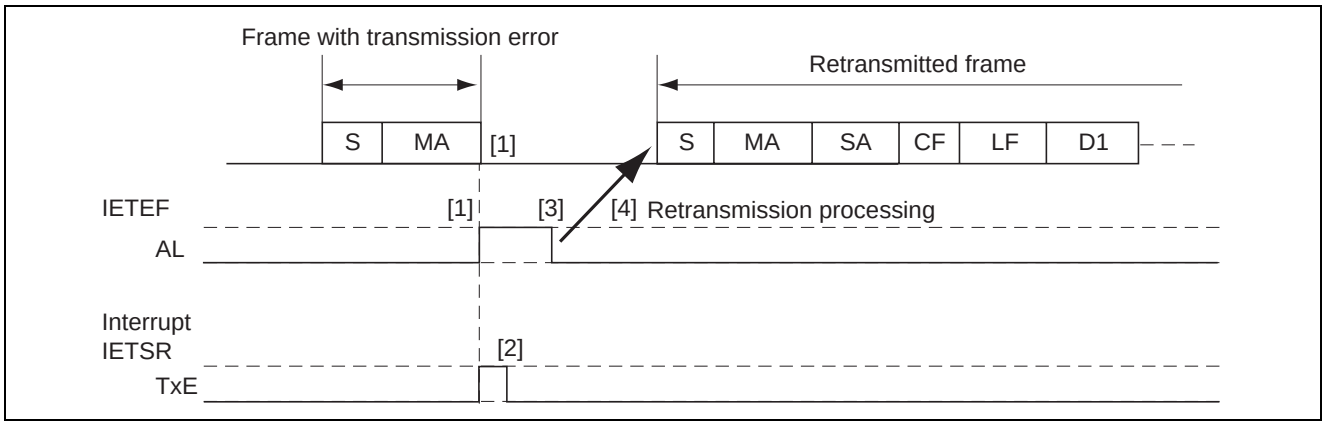


Figure 10 Handling of Arbitration Loss Errors

(2) Handling of Underrun Errors (UE)

The handling of underrun errors is illustrated in figure 11.

[1] When an underrun error occurs the IEBus controller sets the UE flag and transitions to the wait state.

[2] A TxE interrupt is generated. The underrun error is detected within this interrupt.

[3] The UE flag is cleared within the interrupt.

[4] Retransmission takes place.

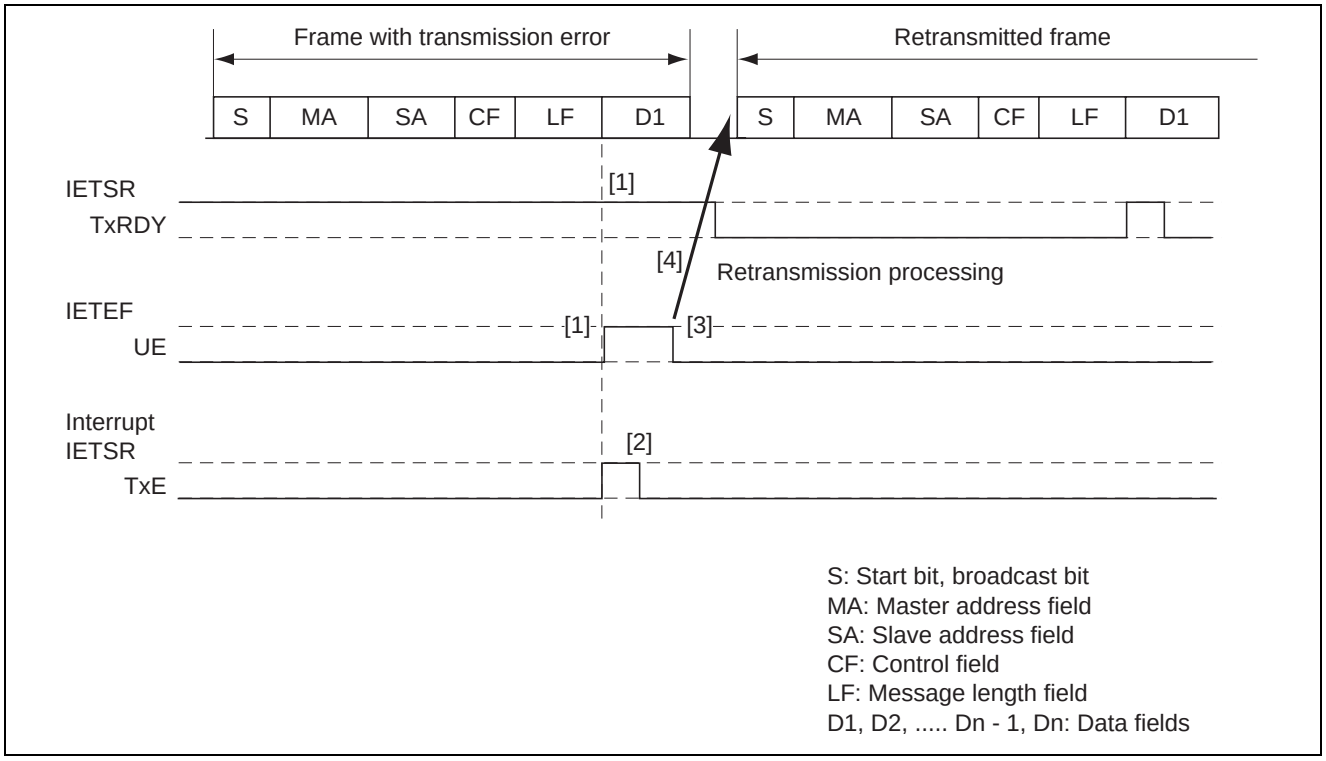


Figure 11 Handling of Underrun Errors

(3) Handling of Timing Errors (TTME)

The handling of timing errors is illustrated in figure 12.

[1] When a timing error occurs the IEBus controller sets the TTME flag and transitions to the wait state.

[2] A TxE interrupt is generated. The timing error is detected within this interrupt.

[3] The TTME flag is cleared within the interrupt.

[4] Retransmission takes place.

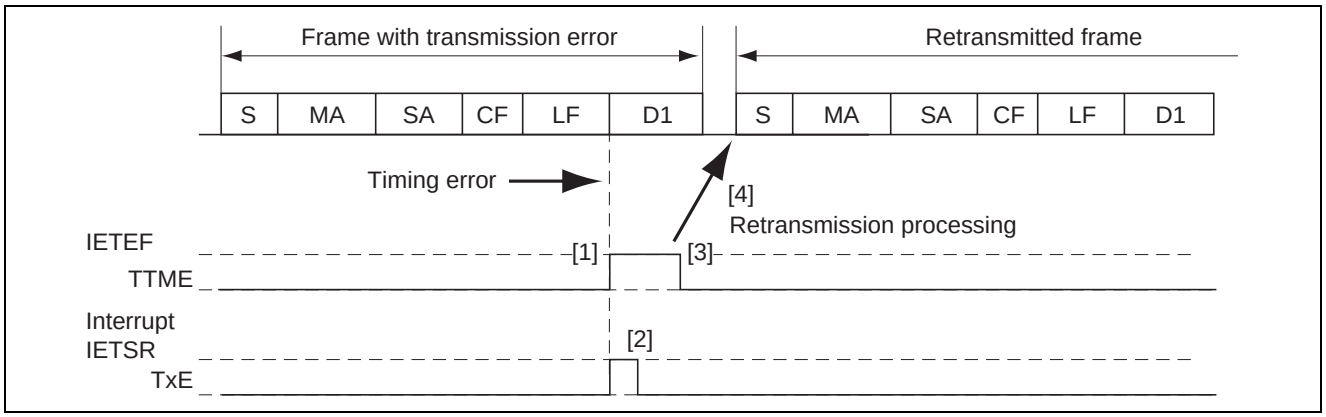


Figure 12 Handling of Timing Errors

(4) Handling of Transmit Frame Over Maximum Number of Transfer Bytes (RO) and Acknowledge (ACK) Errors

The handling of transmit frame over maximum number of transfer bytes and acknowledge errors is illustrated in figure 13. Note that no acknowledge errors are generated during broadcast communication.

[1] When a transmit frame over maximum number of transfer bytes error or an acknowledge error occurs the IEBus controller sets the RO or ACK flag, respectively, and transitions to the wait state.

[2] A TxE interrupt is generated. Within this interrupt error is detected by monitoring both the RO and ACK flags, taking into account the possibility of a transmit frame over maximum number of transfer bytes error or an acknowledge error occurring independently (either an acknowledge error preceding transmission of the data field, or a transmit frame over maximum number of transfer bytes error occurring alone).

[3] The RO and ACK flags are cleared within the interrupt.

[4] Retransmission takes place.

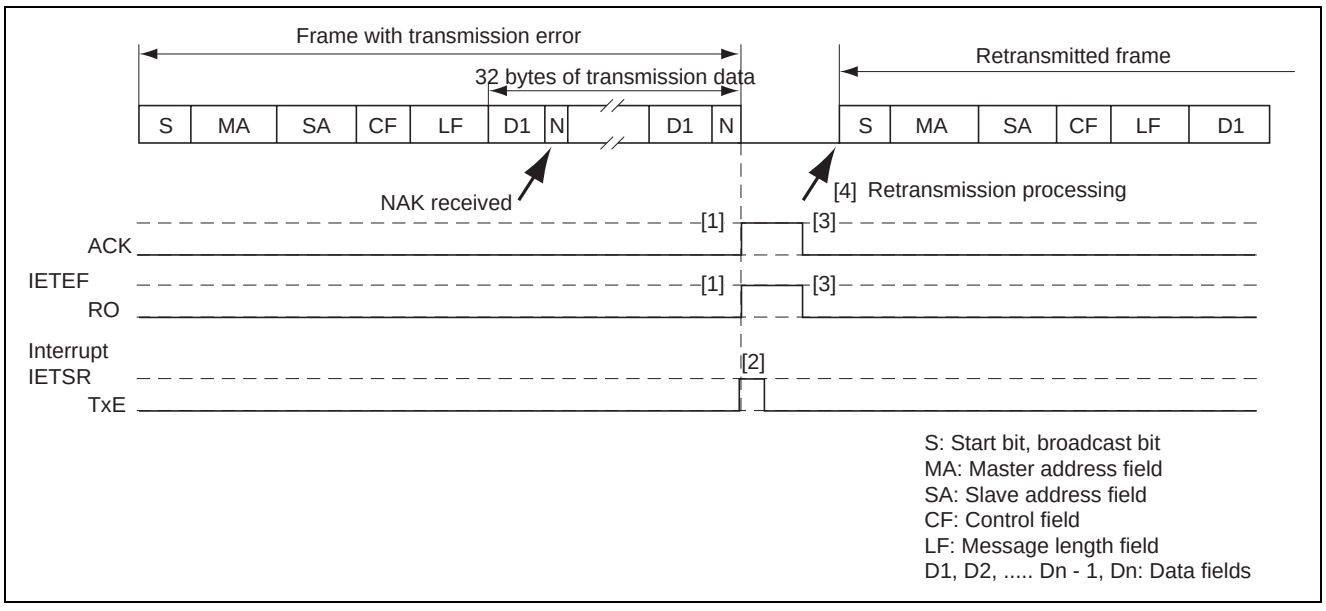


Figure 13 Handling of Transmit Frame over Maximum Number of Transfer Bytes and Acknowledge Errors

2.4 Error Handling During Slave Reception

When an error occurs during slave reception, the frame that was being transmitted is discarded and the IEBus controller returns to ready-to-transmit/receive state from standby state. Furthermore, DTC transfer processing is halted when the following errors occur.

(1) Handling of Overrun Errors (OVE)

The handling of overrun errors is illustrated in figure 14.

- Condition 1: Normal Communication [1] to [5]

  [1] An overrun error occurs. The IEBus controller sets the OVE flag.

  [2] An RxE interrupt is generated. Within this interrupt it is determined whether the communications mode is normal or broadcast, and, if an overrun error is detected and the communications mode is normal, the OVE flag is not cleared until a receive frame over maximum number of transfer bytes error (DLE) has been generated. The IEBus controller transmits a NAK. (The received frame is discarded.)

  [3] When the receive frame over maximum number of transfer bytes error is generated the IEBus controller sets the DLE flag and transitions to the wait state.

  [4] An RxE interrupt is generated. The error state is detected within the interrupt and the DLE, OVE, and RxRDY flags are cleared.

  [5] The IEBus controller returns to ready-to-transmit/receive state.

- Condition 2: Broadcast Communication [6] to [8]

  [6] When an overrun error occurs the IEBus controller sets the OVE flag and transitions to the wait state.

  [7] An RxE interrupt is generated. Within this interrupt it is determined whether the communications mode is normal or broadcast, and, if an overrun error is detected and the communications mode is broadcast, the OVE flag is cleared. Furthermore, the RxRDY flag is cleared.

  [8] The IEBus controller returns to ready-to-transmit/receive state.

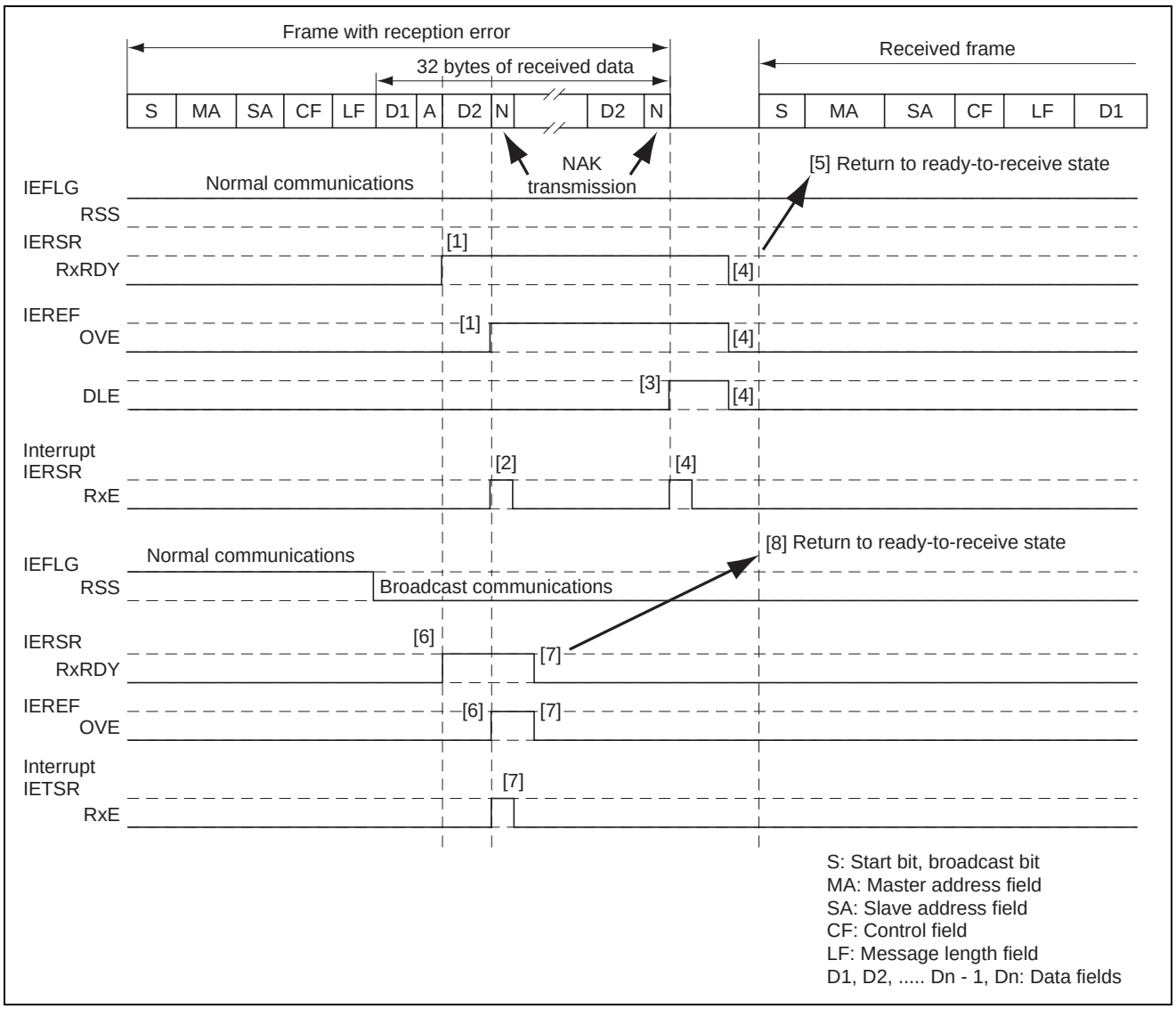


Figure 14 Handling of Overrun Errors

(2) Handling of Timing Errors (RTME)

The handling of timing errors is illustrated in figure 15.

[1] When a timing error occurs the IEBus controller sets the RTME flag and transitions to the wait state.

[2] An RxE interrupt is generated. The timing error is detected within this interrupt. Furthermore, the RTME flag is cleared.

[3] The IEBus controller returns to ready-to-transmit/receive state.

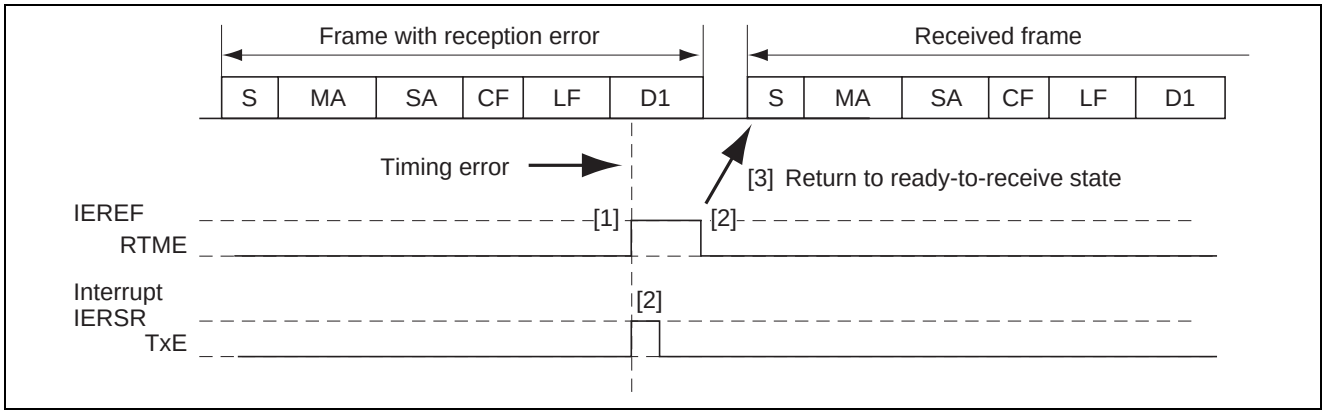


Figure 15 Handling of Timing Errors

(3) Handling of Parity Errors (PE) and Receive Frame Over Maximum Number of Transfer Bytes Errors (DLE)

The handling of parity errors and receive frame over maximum number of transfer bytes errors is illustrated in figure 16.

[1] When a parity error or a receive frame over maximum number of transfer bytes error occurs the IEBus controller sets the PE or the DLE flag, respectively, and transitions to the wait state.

[2] An RxE interrupt is generated. The parity error or a receive frame over maximum number of transfer bytes error is detected within this interrupt. Furthermore, the PE or the DLE flag is cleared. Within the interrupt the error is detected by monitoring both the PE and the DLE flags, taking into account the possibility of a parity error or a receive frame over maximum number of transfer bytes error occurring independently (a parity error when in broadcast communications mode, for example).

[3] The IEBus controller returns to ready-to-transmit/receive state.

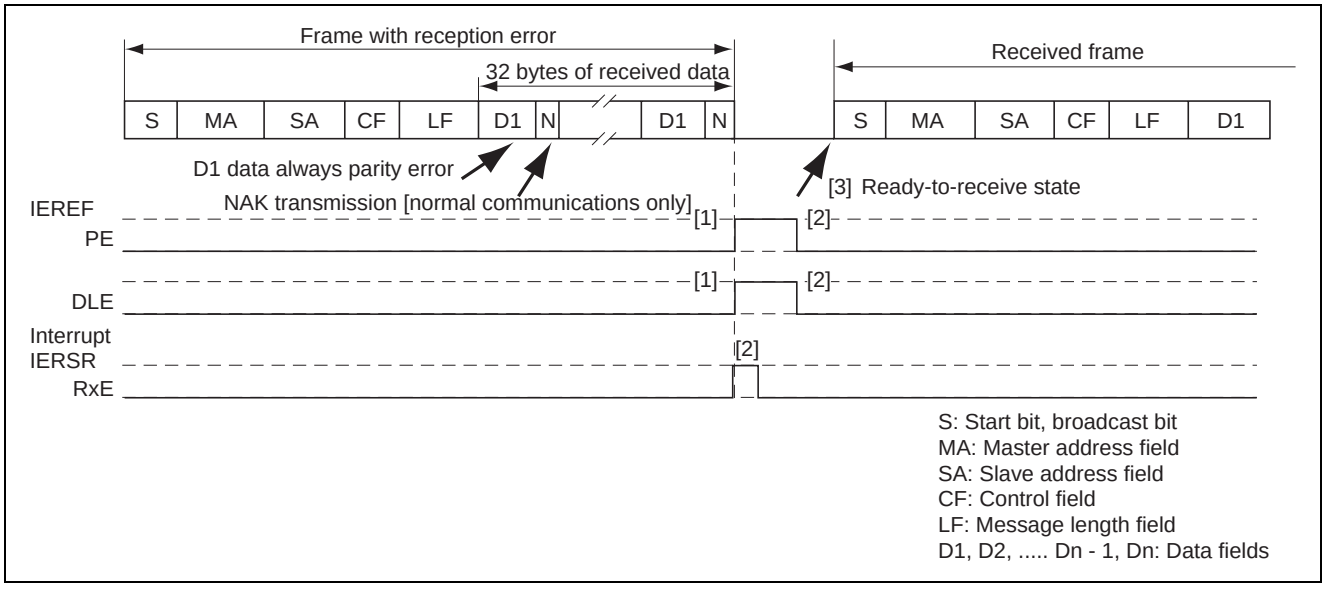


Figure 16 Handling of Parity Errors and Receive Frame over Maximum Number of Transfer Bytes Errors

2.5 Processing of IEBus Runaway State (IRA)

The processing of IEBus control runaway state is illustrated in figure 17. When IEBus control runaway occurs the processing shown in figure 17 is performed within the IRA interrupt.

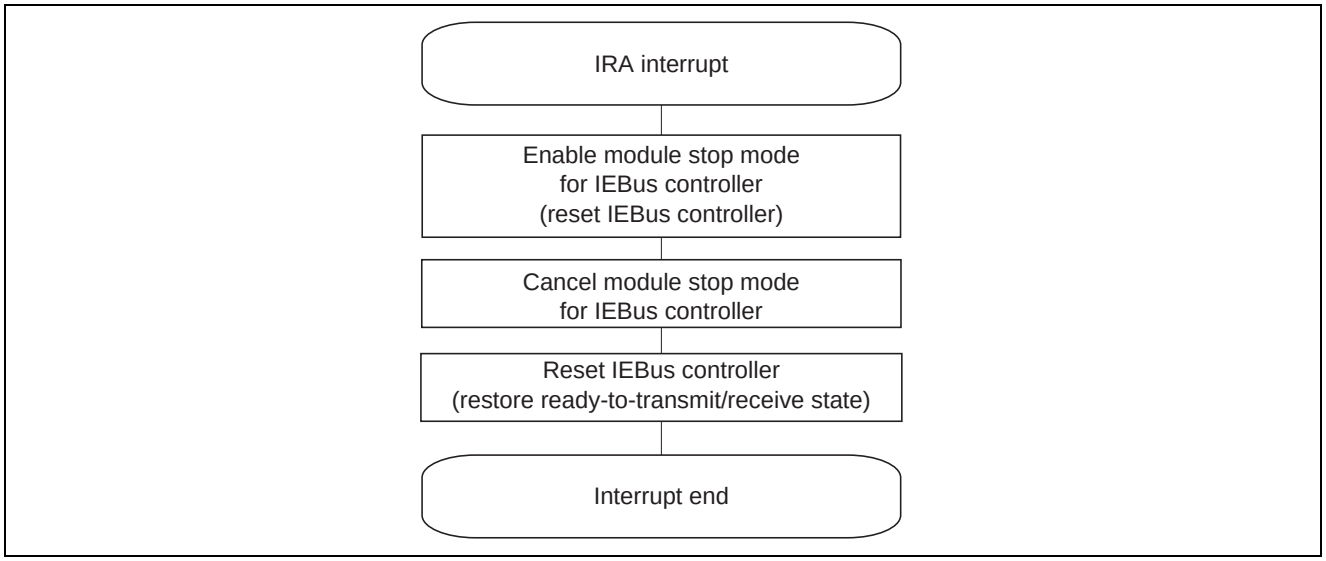


Figure 17 Handling of IEBus Runaway Status

3. Hardware Configuration

3.1 Example of Connections with IEBus Driver/Transceiver

An example of connections between the H8S/2258F and the HA12187FP IEBus driver/transceiver is shown in figure 18.

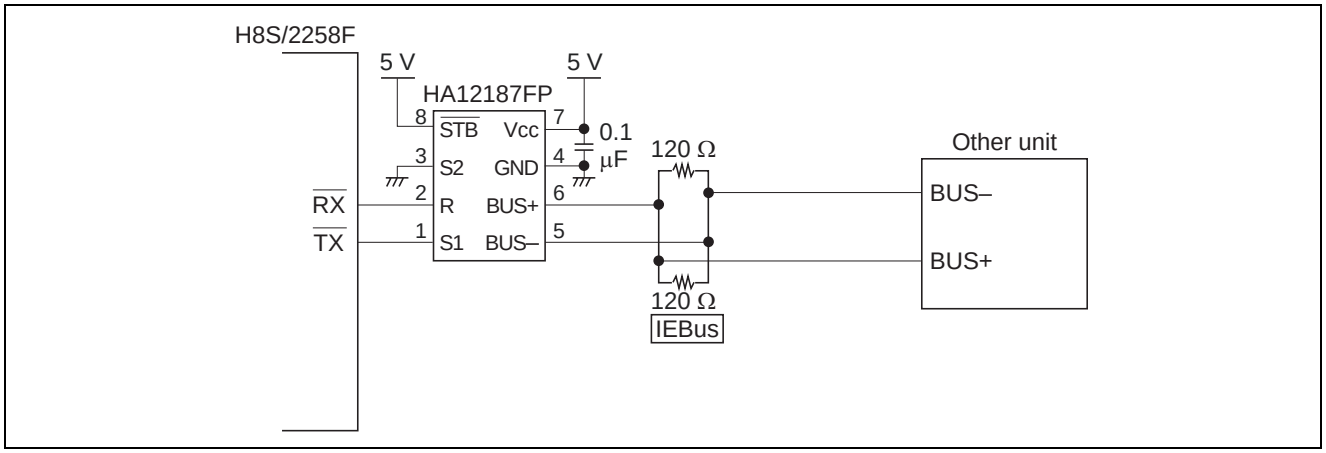


Figure 18 Example of Connections with HA12187FP IEBus Driver/Transceiver

4. Description of Software

The software used for master transmission/slave reception is described in this section.

4.1 DTC Settings

(1) DTC Register Information During Master Transmission

The DTC vector table and memory mapping used during master transmission are shown in figure 19. DTC register information, in the order MRA, SAR, MRB, DAR, CRA, and CRB, is set in internal RAM beginning at address H'FFEC00. The DTC register settings mapped to memory during master transmission are listed in table 1.

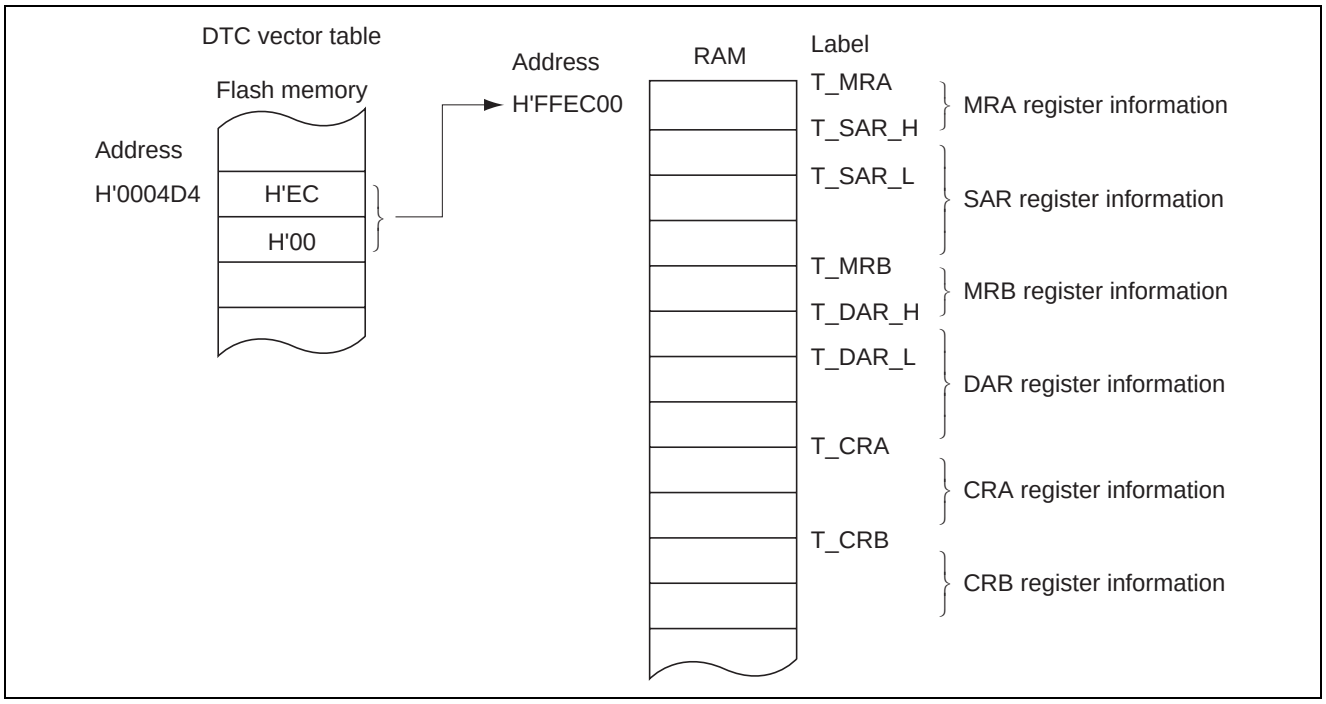


Figure 19 DTC Vector Table and Memory Mapping During Master Transmission

Table 1 DTC Register Settings During Master Transmission

DTC Register	Settings
MRA	• DAR is fixed • SAR is incremented following transfers • Normal mode • Byte transfer mode
SAR	• Address in memory where 2nd byte of transmission data is stored
MRB	• No DTC chain transfer • Interrupts to CPU disabled unless transfer counter value is 0
DAR	• Address of the IEBus transmit buffer register (IETBR)
CRA	• Value of number of transfer bytes set in the IEBus transmit message length register (IETBFL) minus 1
CRB	• Not used

(2) DTC Register Information during Slave Reception

The DTC vector table and memory mapping used during slave reception are shown in figure 20. DTC register information, in the order MRA, SAR, MRB, DAR, CRA, and CRB, is set in the internal RAM beginning at address H'FFEC0C. The DTC register settings mapped to memory during slave reception are listed in table 2.

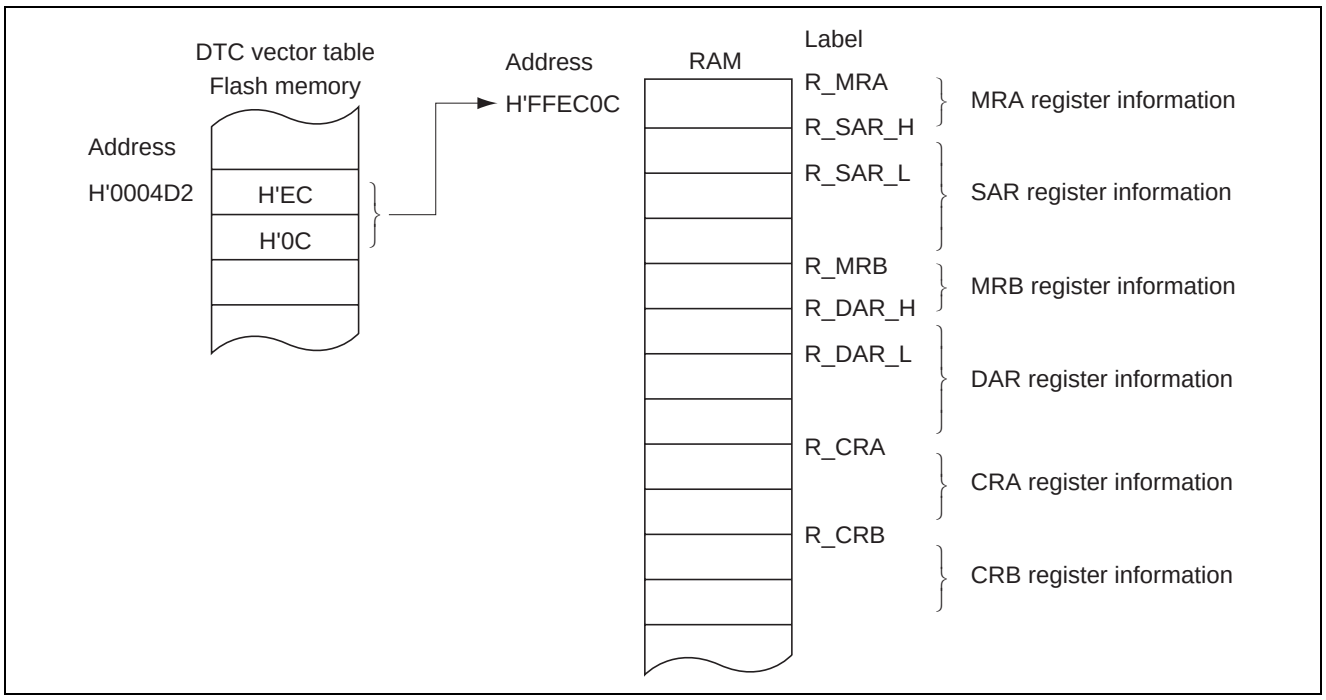


Figure 20 DTC Vector Table and Memory Mapping During Slave Reception

Table 2 DTC Register Settings During Slave Reception

DTC Register	Settings
MRA	• SAR is fixed • DAR is incremented following transfers • Normal mode • Byte transfer mode
SAR	• Address of the IEBus receive buffer register (IERBR)
MRB	• No DTC chain transfer • Interrupts to CPU disabled unless transfer counter value is 0
DAR	• Address in memory where received data is stored
CRA	• Value of the IEBus receive message length register (IERBFL)
CRB	• Not used

4.2 Send/Receive Data Settings

(1) Transmit Data during Master Transmission

Figure 21 illustrates the transmit frames used for master transmission. During master transmission the frames shown in figure 21 are sent in sequence, [1] to [3]. The transmit frames are mapped in software by transdata (label).

[1] : Normal Communications

B	MA	SA	CF	LF	D (32 bytes)
1	H'AAA	H'BBB	H'F	H'20	H'11 • • •

[2] : General Broadcast Communications

B	MA	SA	CF	LF	D (32 bytes)
0	H'AAA	H'FFF	H'F	H'20	H'22 • • •

[3] : Group Broadcast Communications

B	MA	SA	CF	LF	D (32 bytes)
0	H'AAA	H'BBB	H'F	H'20	H'33 • • •

B: Broadcast bit
MA: Master address field
SA: Slave address field
CF: Control field
LF: Message length field
D: Data field

Figure 21 Transmit Frames for Master Transmission

(2) Storing Received Data during Slave Reception

Figure 22 illustrates the way received data is stored during slave reception. During slave reception the communications mode is recognized as normal, general broadcast, or group broadcast, and the received data is stored in the appropriate destinations (in the internal RAM).

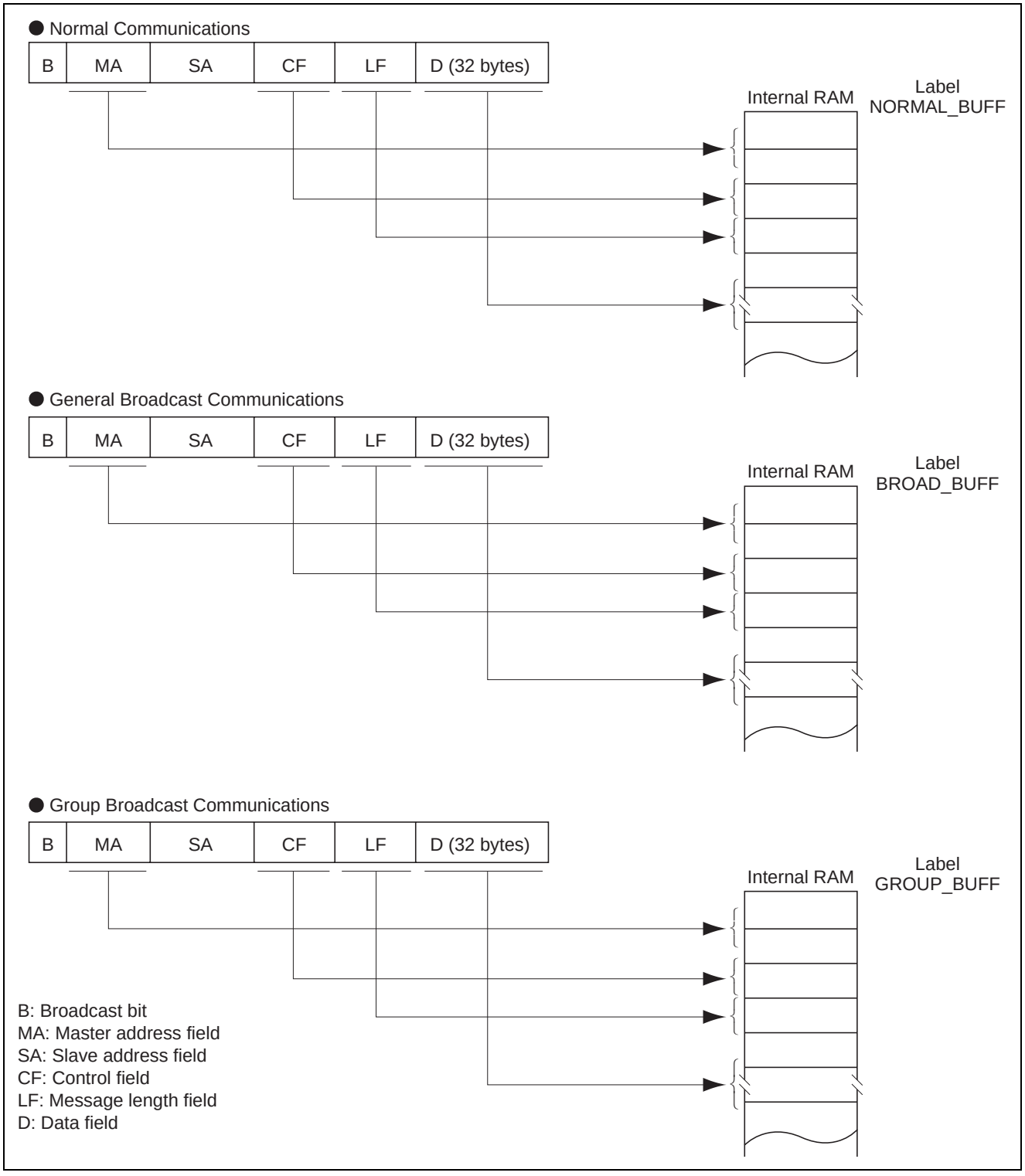


Figure 22 Storing Received Data during Slave Reception

4.3 Software Configuration

(1) Modules

Table 3 Modules

Module	Label	Description
Power-on reset routine	preset	Performs power-on reset processing
Internal I/O initialization routine	io_int	Initializes I/O functions, the IEBus controller, and the DTC
DTC initialization routine	dtc_int	Initializes DTC related registers in internal RAM
IEBus controller initialization routine	iebus_int	Initializes the IEBus controller
Transmit/receive main routine	iebus_main	Manages the number of transmit and receive frames. Controls retransmission when transmit errors occur
Start master transmission routine	master_trans	Controls the start of master transmission
Set up master transmission routine	mtrans_setup	Sets up transmit frames for master transmission
Input command routine	iebuscmd_input	Inputs commands to the IEBus controller
Ready-to-transmit interrupt handler	txrdyi	Recognizes the end of transmission data transfer by the DTC
txsti interrupt handler	txsti	Controls the following interrupts • IEBus controller runaway interrupt • Transmit start interrupt • Transmit error interrupt • Transmit end interrupt
IEBus controller runaway interrupt handler	ira_sub	Performs error processing when IEBus controller runaway occurs
Transmit start interrupt handler	txs_sub	Enables transfer of transmission data by the DTC
Transmit error interrupt handler	txe_sub	Detects transmit errors and performs error handling
Transmit end interrupt handler	txf_sub	Counts the number of frames transmitted
Ready-to-receive interrupt handler	rxrdyi	Recognizes the end of received data transfer by the DTC
rxsti interrupt handler	rxsti	Controls the following interrupts • Receive start interrupt • Receive error interrupt • Receive end interrupt
Receive start interrupt handler	rxs_sub	Performs setup for reception and enables transmission of data by the DTC
Set up slave reception routine	sreceiv_setup	Performs setup for slave reception
Receive end interrupt handler	rxf_sub	Counts the number of frames received
Receive error interrupt handler	rxe_sub	Detects receive errors and performs error handling

(2)Module Configuration

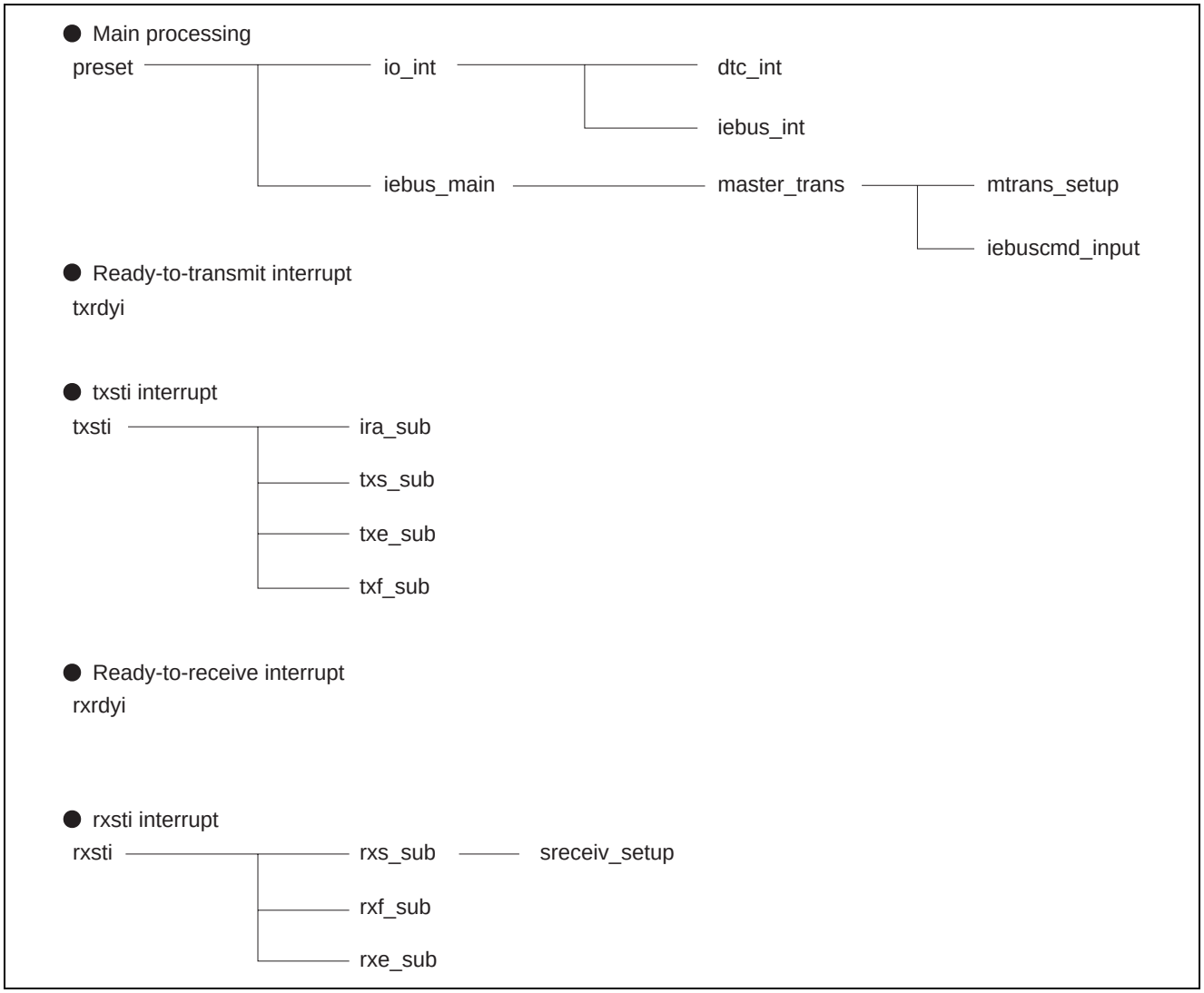


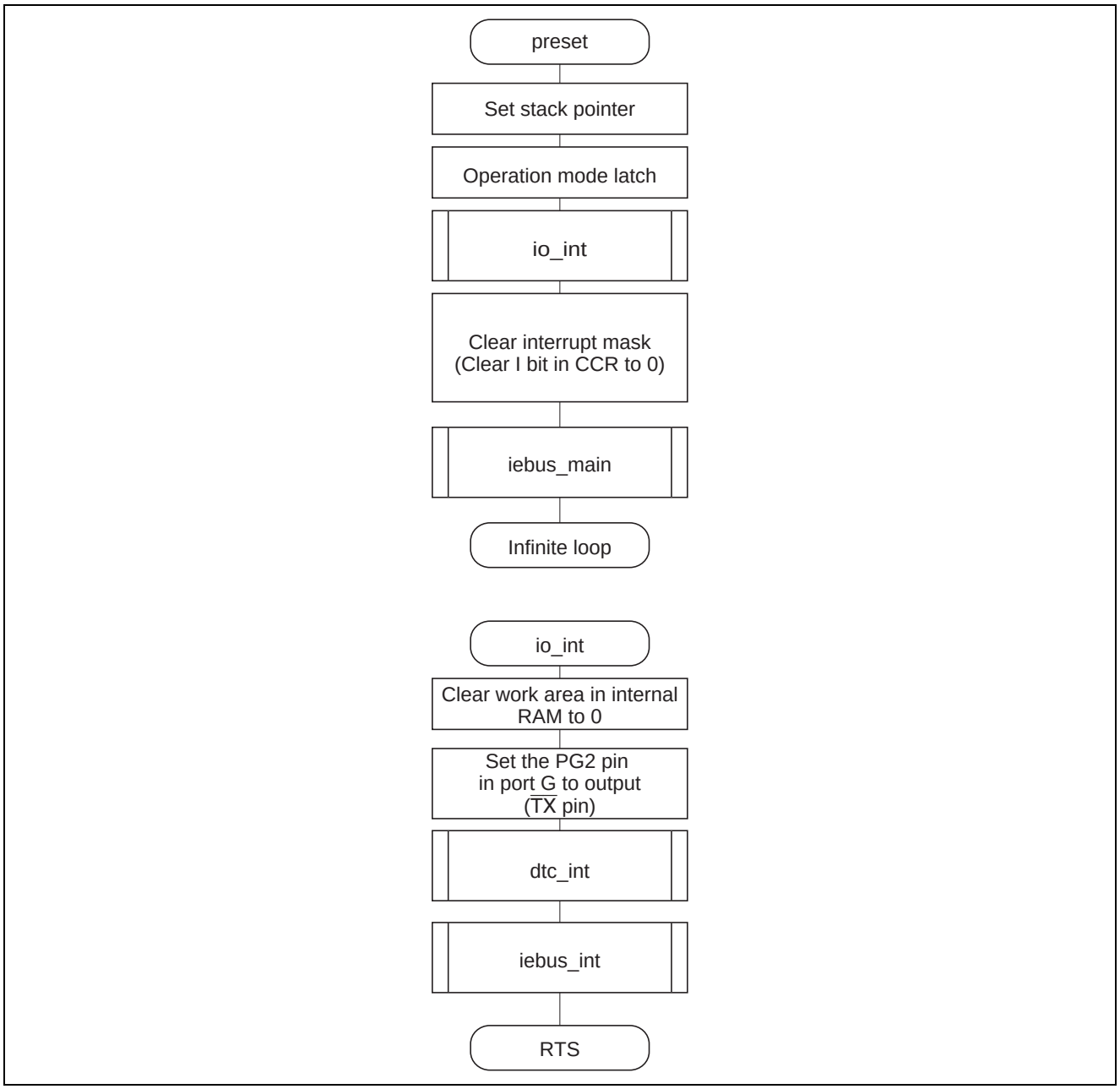
Figure 23 Module Configuration

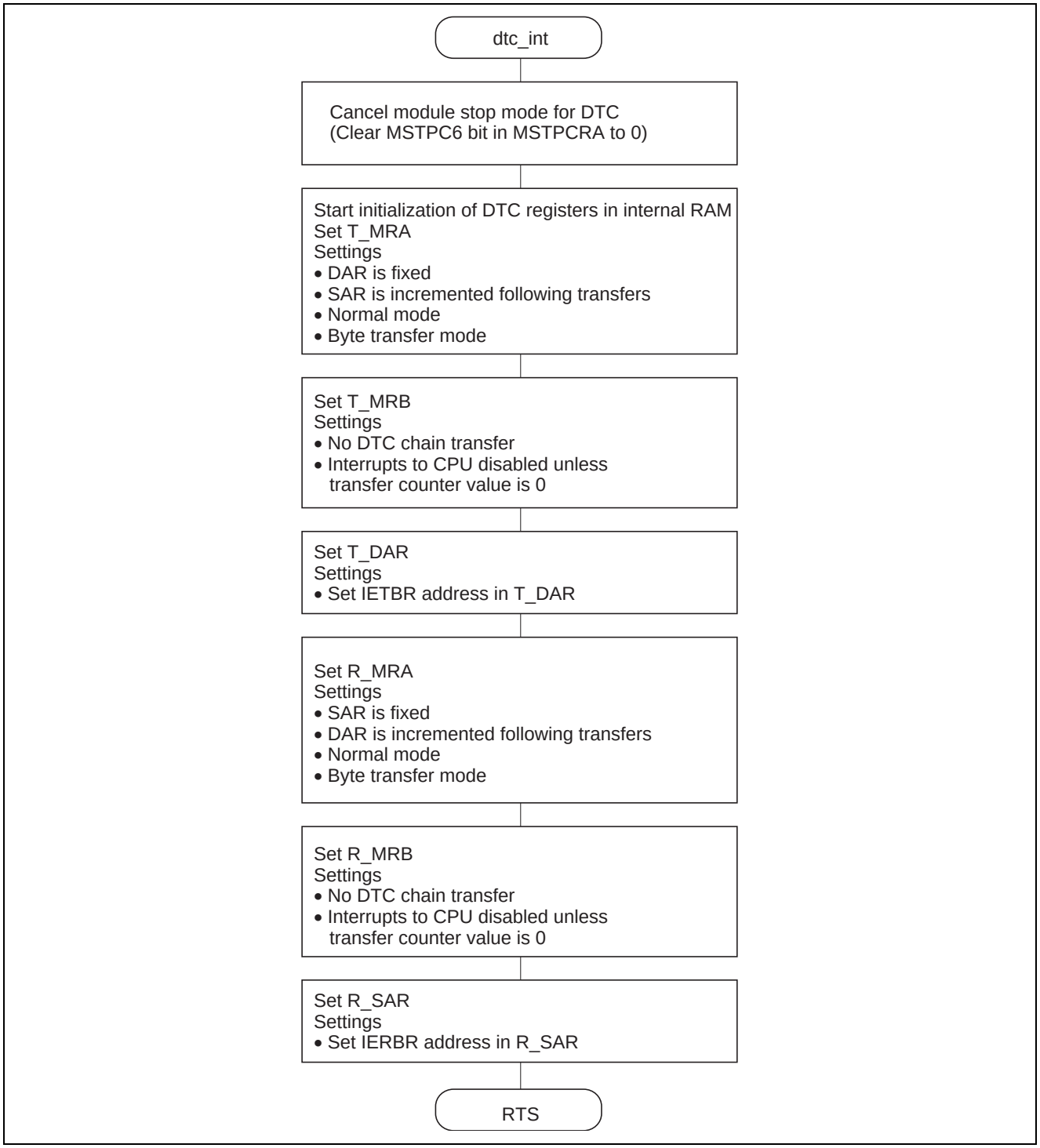
(3) Arguments

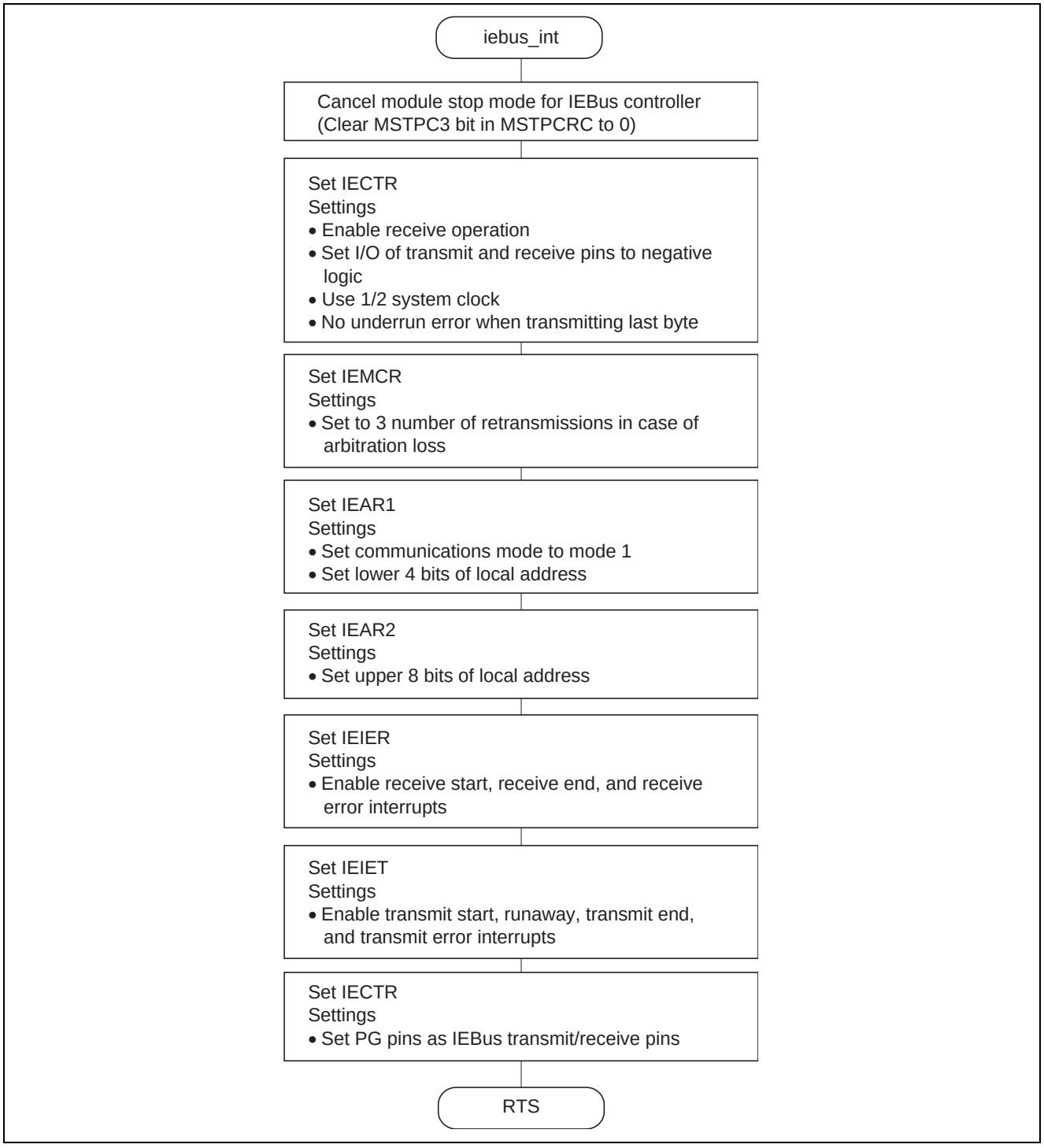
Table 4 Arguments

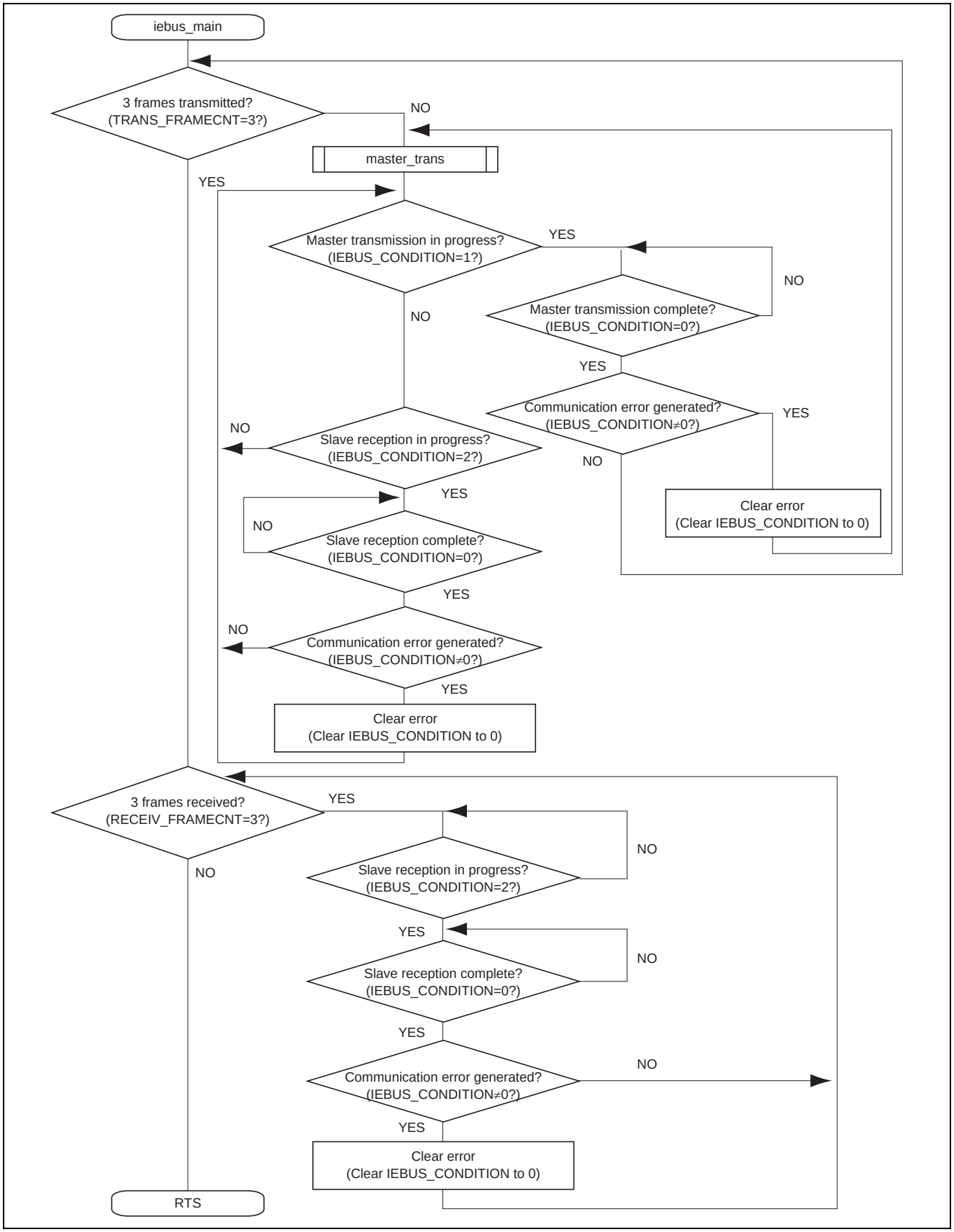
Label	Function	Data Length	Modules Used by	I/O
IEBUS_CMD	Stores IEBus controller commands	Byte	master_trans	Output
			iebuscmd_input	Input
TRANS_FRAMECNT	Stores the number of frames transmitted	Byte	txf_sub	Output
			mtrans_setup	Input
			iebus_main	Input
RECEIV_FRAMECNT	Stores the number of frames received	Byte	rxf_sub	Output
			iebus_main	Input
IEBUS_CONDITION	Indicates the transmit/receive state of the IEBus controller 0: Master transmission, slave reception complete 1: Master transmission in progress 2: Slave reception in progress	Byte	rxe_sub	Output
			rxf_sub	Output
			rxs_sub	Output
			txe_sub	Output
			txf_sub	Output
			txs_sub	Output
			ira_sub	Output
			iebus_main	Input
IEBUS_ERROR	Indicates IEBus controller transmission errors Slave reception 0: No error 1: Timing error 2: Transfer byte count error 3: Parity error 4: Overrun error Master transmission 5: Arbitration error 6: Underrun error 7: Timing error 8: Transfer byte over 9: ACK error	Byte	rxe_sub	Output
			ira_sub	Output
			txe_sub	Output
			iebus_main	Input

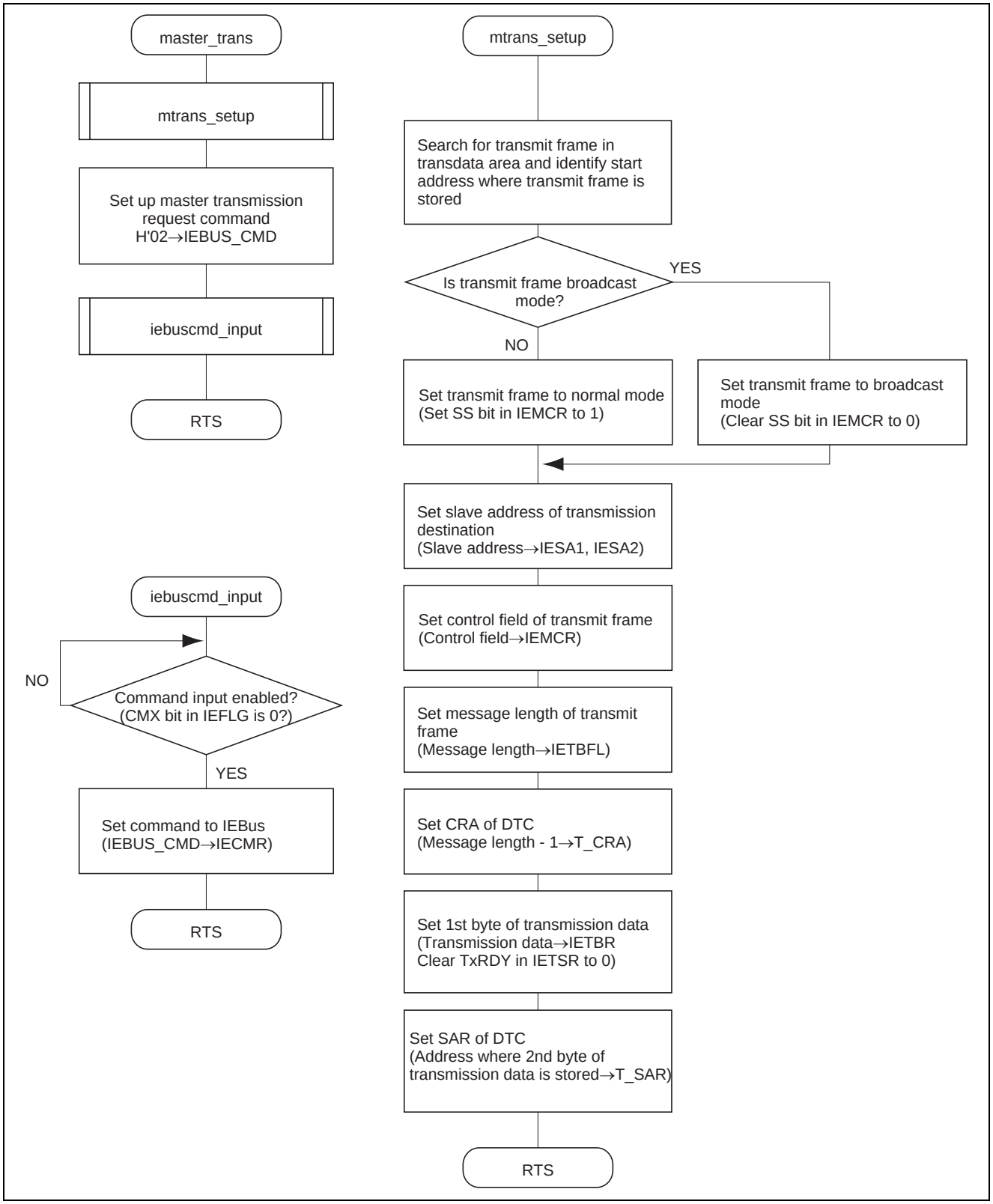
5. Flowcharts

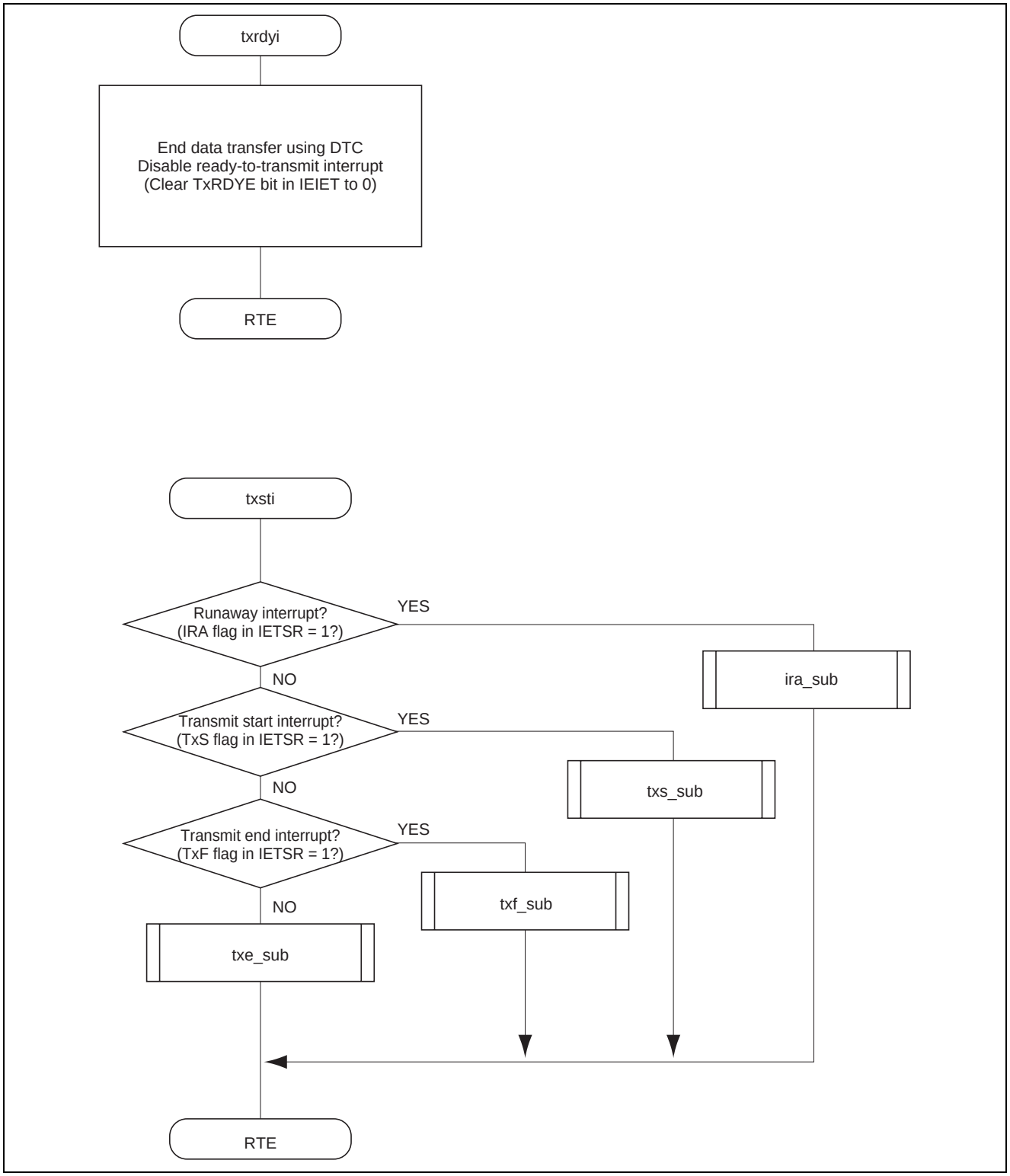


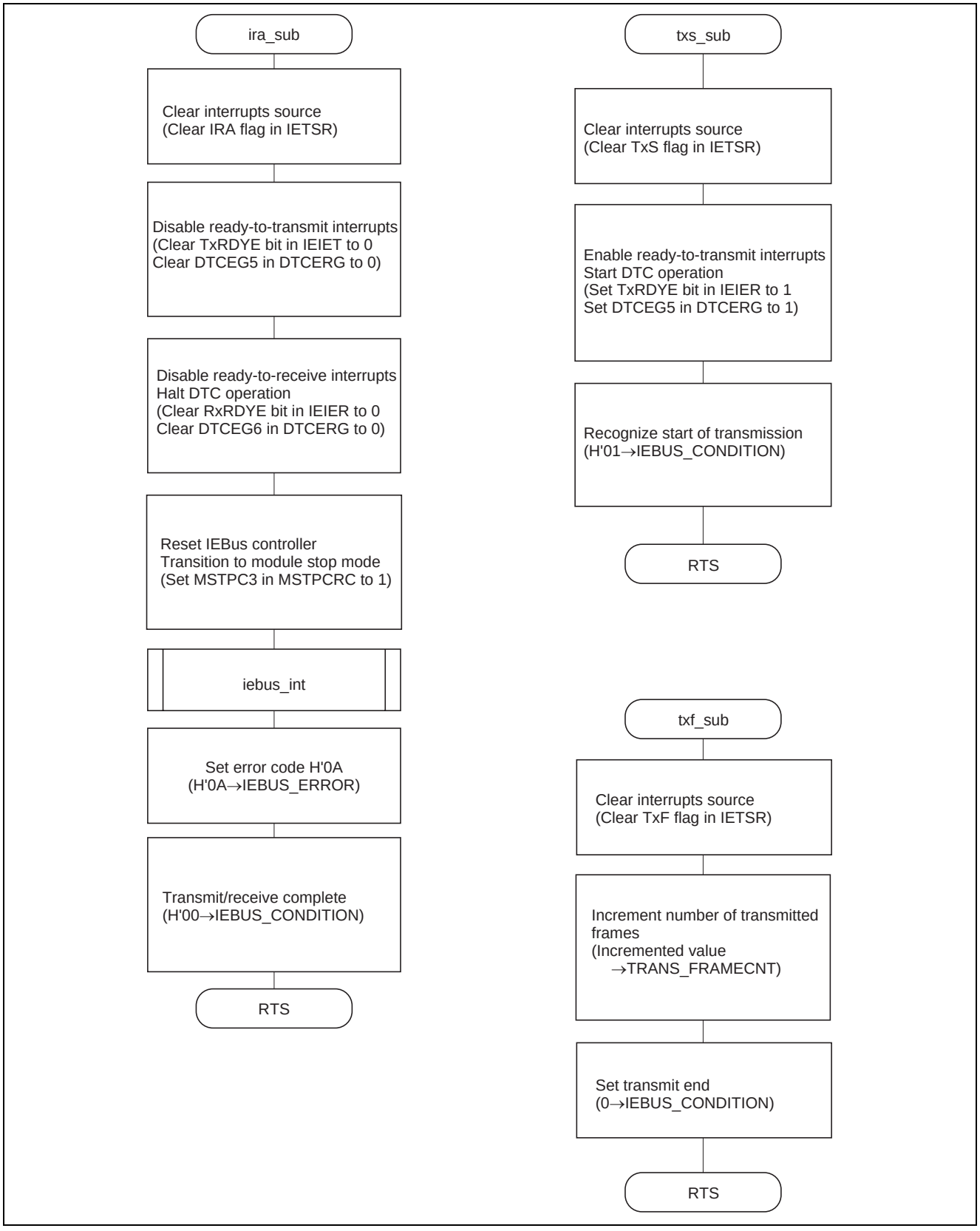

dtc_int
Cancel module stop mode for DTC
(Clear MSTPC6 bit in MSTPCRA to 0)
Start initialization of DTC registers in internal RAM
Set T_MRA
Settings
• DAR is fixed
• SAR is incremented following transfers
• Normal mode
• Byte transfer mode
Set T_MRB
Settings
• No DTC chain transfer
• Interrupts to CPU disabled unless transfer counter value is 0
Set T_DAR
Settings
• Set IETBR address in T_DAR
Set R_MRA
Settings
• SAR is fixed
• DAR is incremented following transfers
• Normal mode
• Byte transfer mode
Set R_MRB
Settings
• No DTC chain transfer
• Interrupts to CPU disabled unless transfer counter value is 0
Set R_SAR
Settings
• Set IERBR address in R_SAR
RTS

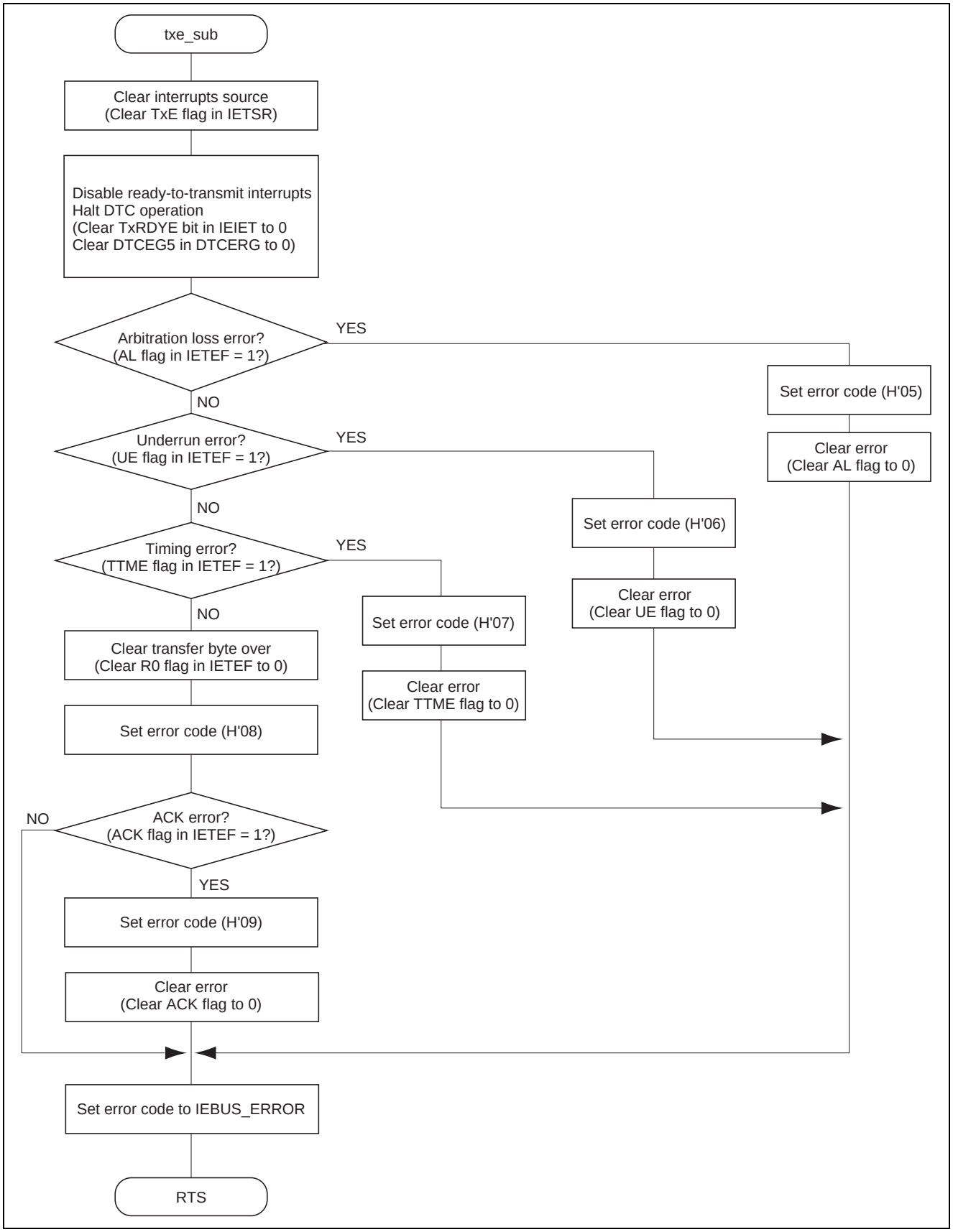

iebus_int
Cancel module stop mode for IEBus controller
(Clear MSTPC3 bit in MSTPCRC to 0)
Set IECTR
Settings
• Enable receive operation
• Set I/O of transmit and receive pins to negative logic
• Use 1/2 system clock
• No underrun error when transmitting last byte
Set IEMCR
Settings
• Set to 3 number of retransmissions in case of arbitration loss
Set IEAR1
Settings
• Set communications mode to mode 1
• Set lower 4 bits of local address
Set IEAR2
Settings
• Set upper 8 bits of local address
Set IEIER
Settings
• Enable receive start, receive end, and receive error interrupts
Set IEIET
Settings
• Enable transmit start, runaway, transmit end, and transmit error interrupts
Set IECTR
Settings
• Set PG pins as IEBus transmit/receive pins
RTS

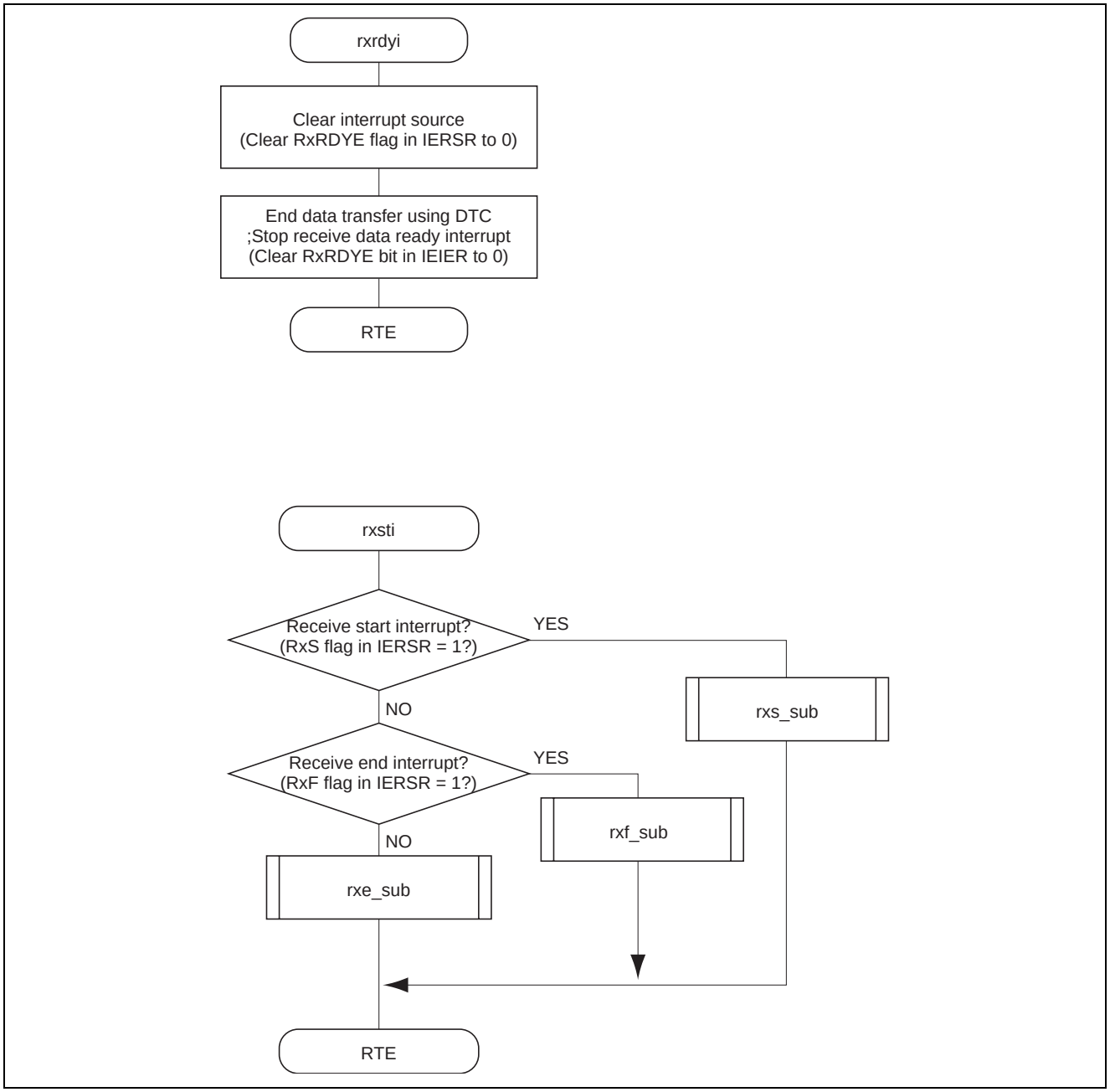

iebus_main
3 frames transmitted?
(TRANS_FRAMECNT=3?)
NO
master_trans
YES
Master transmission in progress?
(IEBUS_CONDITION=1?)
YES
NO
Master transmission complete?
(IEBUS_CONDITION=0?)
NO
YES
Communication error generated?
(IEBUS_CONDITION≠0?)
YES
NO
Clear error
(Clear IEBUS_CONDITION to 0)
NO
Slave reception in progress?
(IEBUS_CONDITION=2?)
YES
NO
Slave reception complete?
(IEBUS_CONDITION=0?)
YES
NO
Communication error generated?
(IEBUS_CONDITION≠0?)
YES
Clear error
(Clear IEBUS_CONDITION to 0)
3 frames received?
(RECEIV_FRAMECNT=3?)
YES
NO
Slave reception in progress?
(IEBUS_CONDITION=2?)
NO
YES
Slave reception complete?
(IEBUS_CONDITION=0?)
NO
YES
Communication error generated?
(IEBUS_CONDITION≠0?)
NO
YES
Clear error
(Clear IEBUS_CONDITION to 0)
RTS

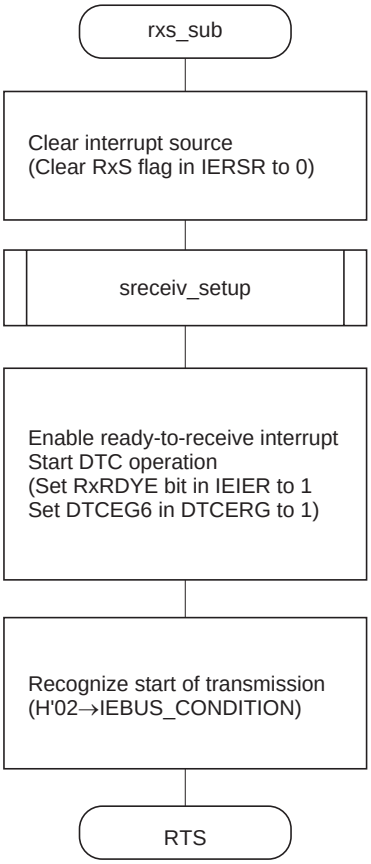

master_trans
mtrans_setup
Set up master transmission request command H'02→IEBUS_CMD
iebuscmd_input
RTS
iebuscmd_input
NO
Command input enabled? (CMX bit in IEFLG is 0?)
YES
Set command to IEBus (IEBUS_CMD→IECMR)
RTS
mtrans_setup
Search for transmit frame in transdata area and identify start address where transmit frame is stored
Is transmit frame broadcast mode?
YES
NO
Set transmit frame to normal mode (Set SS bit in IEMCR to 1)
Set transmit frame to broadcast mode (Clear SS bit in IEMCR to 0)
Set slave address of transmission destination (Slave address→IESA1, IESA2)
Set control field of transmit frame (Control field→IEMCR)
Set message length of transmit frame (Message length→IETBFL)
Set CRA of DTC (Message length - 1→T_CRA)
Set 1st byte of transmission data (Transmission data→IETBR Clear TxRDY in IETSR to 0)
Set SAR of DTC (Address where 2nd byte of transmission data is stored→T_SAR)
RTS

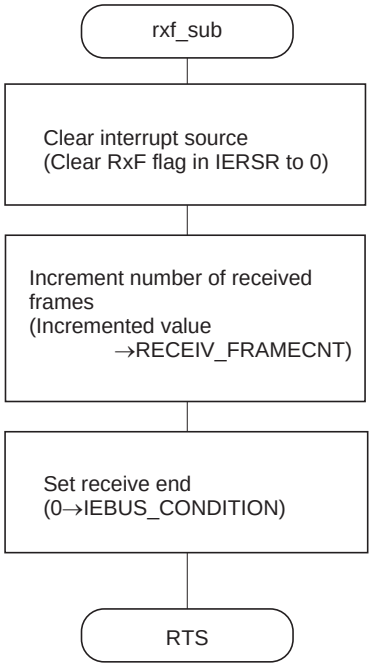

txrdyi
End data transfer using DTC
Disable ready-to-transmit interrupt
(Clear TxRDYE bit in IEIET to 0)
RTE
txsti
Runaway interrupt?
(IRA flag in IETSR = 1?)
YES
NO
ira_sub
Transmit start interrupt?
(TxS flag in IETSR = 1?)
YES
NO
txs_sub
Transmit end interrupt?
(TxF flag in IETSR = 1?)
YES
NO
txf_sub
txe_sub
RTE

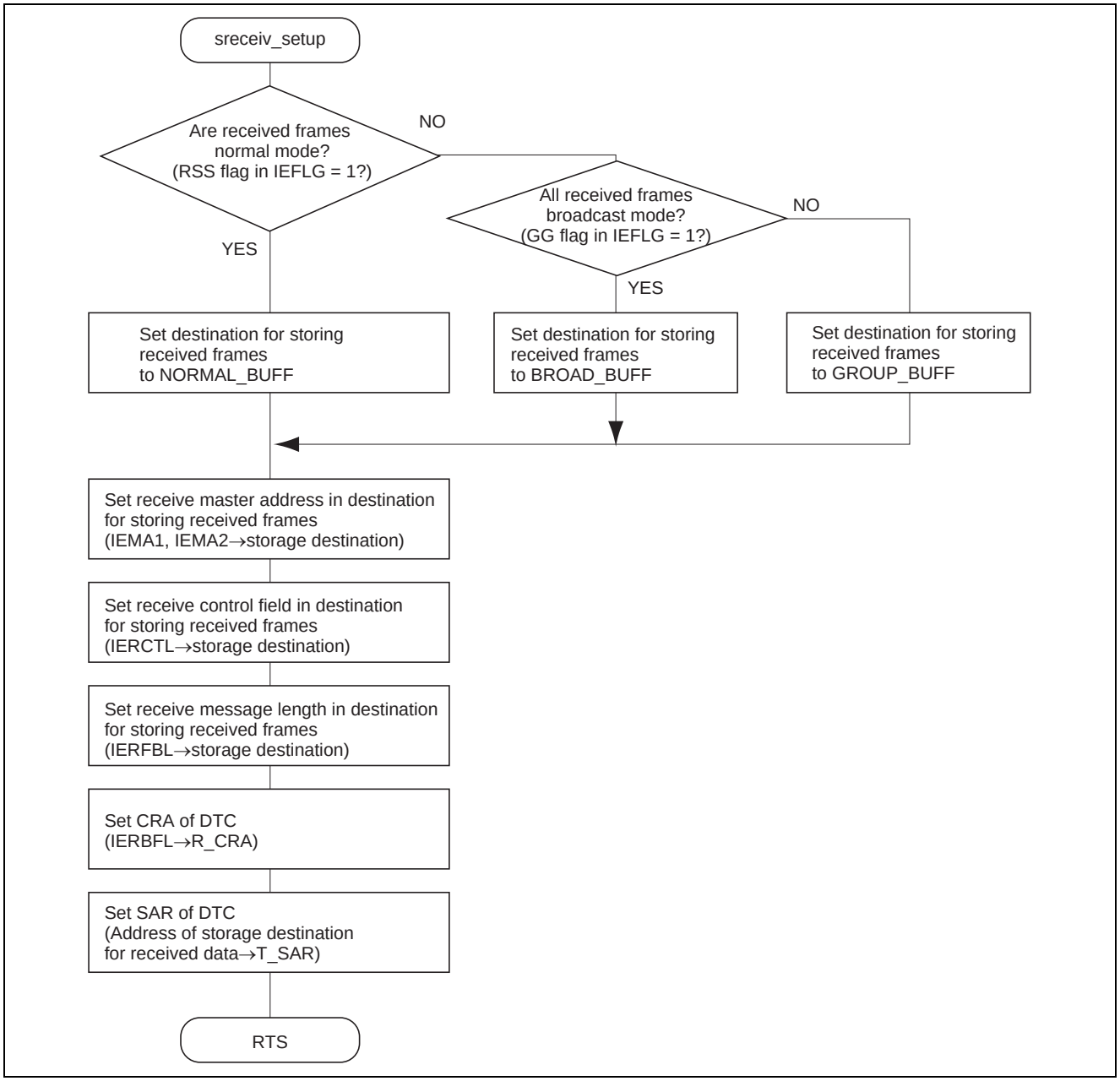

ira_sub
Clear interrupts source
(Clear IRA flag in IETSR)
Disable ready-to-transmit interrupts
(Clear TxRDYE bit in IEIET to 0
Clear DTCEG5 in DTCERG to 0)
Disable ready-to-receive interrupts
Halt DTC operation
(Clear RxRDYE bit in IEIER to 0
Clear DTCEG6 in DTCERG to 0)
Reset IEBus controller
Transition to module stop mode
(Set MSTPC3 in MSTPCRC to 1)
iebus_int
Set error code H'0A
(H'0A→IEBUS_ERROR)
Transmit/receive complete
(H'00→IEBUS_CONDITION)
RTS
txs_sub
Clear interrupts source
(Clear TxS flag in IETSR)
Enable ready-to-transmit interrupts
Start DTC operation
(Set TxRDYE bit in IEIER to 1
Set DTCEG5 in DTCERG to 1)
Recognize start of transmission
(H'01→IEBUS_CONDITION)
RTS
txf_sub
Clear interrupts source
(Clear TxF flag in IETSR)
Increment number of transmitted frames
(Incremented value
→TRANS_FRAMECNT)
Set transmit end
(0→IEBUS_CONDITION)
RTS

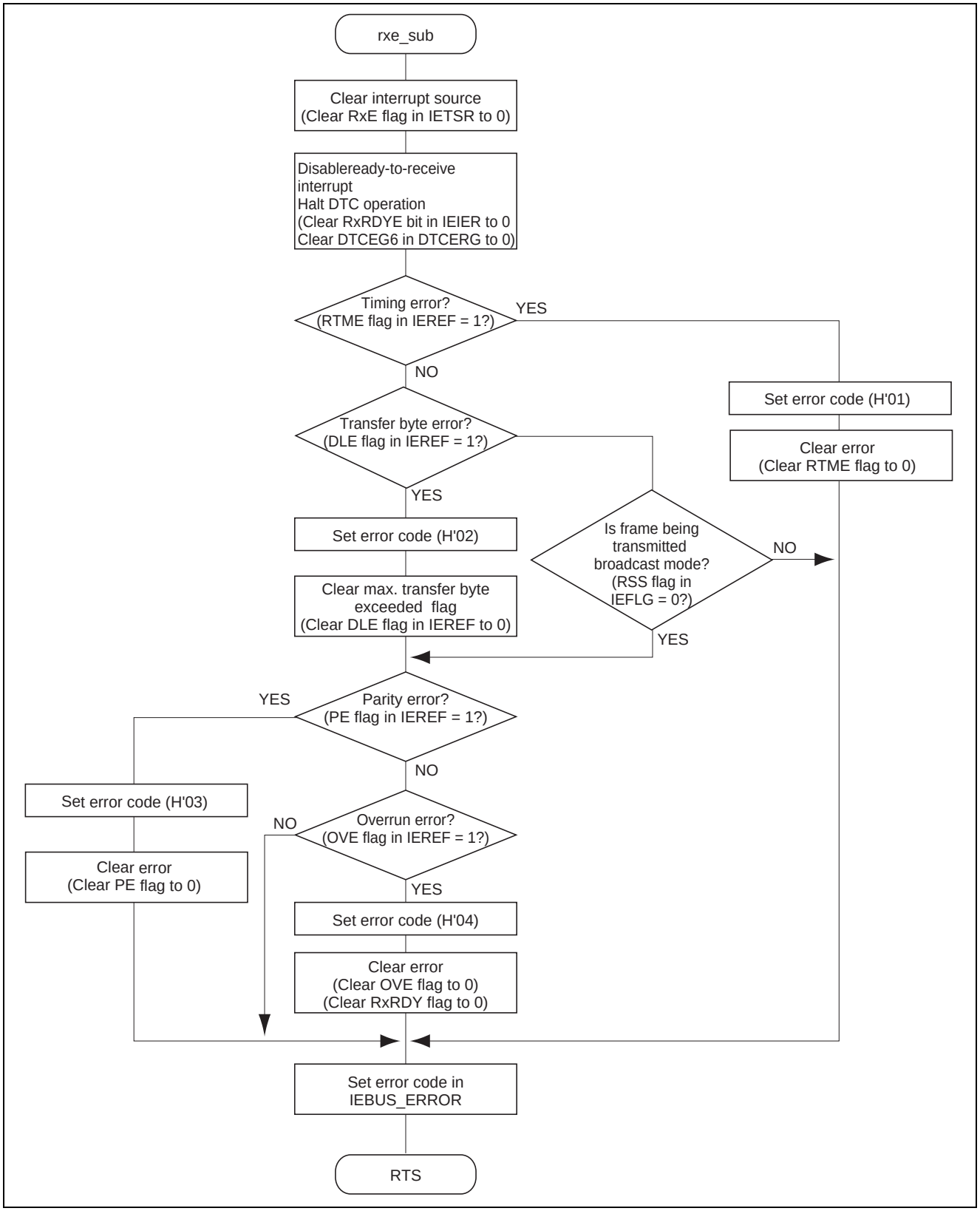

txe_sub
Clear interrupts source
(Clear TxE flag in IETSR)
Disable ready-to-transmit interrupts
Halt DTC operation
(Clear TxRDYE bit in IEIET to 0
Clear DTCEG5 in DTCERG to 0)
Arbitration loss error?
(AL flag in IETEF = 1?)
YES
NO
Set error code (H'05)
Clear error
(Clear AL flag to 0)
Underrun error?
(UE flag in IETEF = 1?)
YES
NO
Set error code (H'06)
Clear error
(Clear UE flag to 0)
Timing error?
(TTME flag in IETEF = 1?)
YES
NO
Set error code (H'07)
Clear error
(Clear TTME flag to 0)
Clear transfer byte over
(Clear R0 flag in IETEF to 0)
Set error code (H'08)
ACK error?
(ACK flag in IETEF = 1?)
NO
YES
Set error code (H'09)
Clear error
(Clear ACK flag to 0)
Set error code to IEBUS_ERROR
RTS


rxrdyi
Clear interrupt source
(Clear RxRDYE flag in IERSR to 0)
End data transfer using DTC
;Stop receive data ready interrupt
(Clear RxRDYE bit in IEIER to 0)
RTE
rxsti
Receive start interrupt?
(RxS flag in IERSR = 1?)
YES
NO
rxs_sub
Receive end interrupt?
(RxF flag in IERSR = 1?)
YES
NO
rxf_sub
rxe_sub
RTE


rxs_sub
Clear interrupt source
(Clear RxS flag in IERSR to 0)
sreceiv_setup
Enable ready-to-receive interrupt
Start DTC operation
(Set RxRDYE bit in IEIER to 1
Set DTCEG6 in DTCERG to 1)
Recognize start of transmission
(H'02→IEBUS_CONDITION)
RTS


rxf_sub
Clear interrupt source
(Clear RxF flag in IERSR to 0)
Increment number of received
frames
(Incremented value
→RECEIV_FRAMECNT)
Set receive end
(0→IEBUS_CONDITION)
RTS


sreceiv_setup
Are received frames normal mode? (RSS flag in IEFLG = 1?)
NO
YES
All received frames broadcast mode? (GG flag in IEFLG = 1?)
NO
YES
Set destination for storing received frames to NORMAL_BUFF
Set destination for storing received frames to BROAD_BUFF
Set destination for storing received frames to GROUP_BUFF
Set receive master address in destination for storing received frames (IEMA1, IEMA2→storage destination)
Set receive control field in destination for storing received frames (IERCTL→storage destination)
Set receive message length in destination for storing received frames (IERFBL→storage destination)
Set CRA of DTC (IERBFL→R_CRA)
Set SAR of DTC (Address of storage destination for received data→T_SAR)
RTS


rxe_sub
Clear interrupt source
(Clear RxE flag in IETSR to 0)
Disableready-to-receive
interrupt
Halt DTC operation
(Clear RxRDYE bit in IEIER to 0
Clear DTCEG6 in DTCERG to 0)
Timing error?
(RTME flag in IEREF = 1?)
YES
NO
Set error code (H'01)
Clear error
(Clear RTME flag to 0)
Transfer byte error?
(DLE flag in IEREF = 1?)
YES
Set error code (H'02)
Clear max. transfer byte
exceeded flag
(Clear DLE flag in IEREF to 0)
Is frame being
transmitted
broadcast mode?
(RSS flag in
IEFLG = 0?)
NO
YES
YES
Parity error?
(PE flag in IEREF = 1?)
NO
Set error code (H'03)
Clear error
(Clear PE flag to 0)
NO
Overrun error?
(OVE flag in IEREF = 1?)
YES
Set error code (H'04)
Clear error
(Clear OVE flag to 0)
(Clear RxRDY flag to 0)
Set error code in
IEBUS_ERROR
RTS

6. Program Listings

```
1     ;---------------------------------------------------------------------
2     ;--Master transmission/slave reception example using internal IEBus --
3     ;-- controller of H8S/2258F                                         --
4     ;--IEBus usage conditions                                           --
5     ;--1. System clock: 12.58 MHz                                       --
6     ;--2. Communications mode: Mode 1                                   --
7     ;--3. Local address: H'AAA                                          --
8     ;--4. Number of retransmissions in case of arbitration loss: 3      --
9     ;--5. Data transmission and reception performed by DTC              --
10    ;--6. _TX and _RX pins set to negative logic I/O                    --
11    ;---------------------------------------------------------------------
12    .cpu        2000A:24
13    .include    "2258FIE.H"
1  I1  ;*****************************************************
2  I1  ;**2258F_IEBus.H                                     **
3  I1  ;**Defines registers related to the IEBus controller  **
4  I1  ;*****************************************************
5  I1  ;*System control related*
6  I1  MDCR        .equ    H'00FFFDE7   ;Mode control register
7  I1
8  I1  ;*Module stop control registers A to C*
9  I1  MSTPCRA     .equ    H'00FFFDE8
10 I1  MSTPA6      .equ    6           ;DTC module stop bit
11 I1  MSTPCRB     .equ    H'00FFFDE9
12 I1  MSTPCRC     .equ    H'00FFFDEA
13 I1  MSTPC3      .equ    3           ;IEBus module stop bit
14 I1
15 I1  ;*DTC enable registers*
16 I1  DTCERG      .equ    H'00FFFE1C
17 I1  DTCEG6      .equ    6           ;IEBus RxRDY source
18 I1  DTCEG5      .equ    5           ;IEBus TxRDY source
19 I1  DTVECR      .equ    H'00FFFE1F
20 I1
21 I1  ;*Port G registers*
22 I1  PGDDR       .equ    H'00FFFE3F
23 I1  PGDR        .equ    H'00FFFF0F
24 I1
25 I1  ;*IEBus controller registers*
26 I1  IECTR       .equ    H'00FFF800  ;IEBus control register
27 I1  IEE         .equ    7
28 I1  IOL         .equ    6
29 I1  DEE         .equ    5
30 I1  CKS         .equ    4
31 I1  RE          .equ    3
32 I1  LUEE        .equ    2
33 I1  IECMR       .equ    H'00FFF801  ;IEBus command register
34 I1  IEMCR       .equ    H'00FFF802  ;IEBus master control register
35 I1  SS          .equ    7
36 I1  IEAR1       .equ    H'00FFF803  ;IEBus local address register 1
37 I1  STE         .equ    0
38 I1  IEAR2       .equ    H'00FFF804  ;IEBus local address register 2
39 I1  IESA1       .equ    H'00FFF805  ;IEBus slave address setting register 1
```

```
40 I1  IESA2       .equ    H'00FFF806  ;IEBus slave address setting register 2
41 I1  IETBFL      .equ    H'00FFF807  ;IEBus transmit message length register
42 I1  IETBR       .equ    H'00FFF808  ;IEBus transmit buffer register
43 I1  IEMA1       .equ    H'00FFF809  ;IEBus receive master address register 1
44 I1  IEMA2       .equ    H'00FFF80A  ;IEBus receive master address register 2
44 I1  IEMA2       .equ    H'00FFF80A  ;IEBus receive master address register 2
45 I1  IERCTL      .equ    H'00FFF80B  ;IEBus receive control field register
46 I1  IERBFL      .equ    H'00FFF80C  ;IEBus receive message length register
47 I1  IERBR       .equ    H'00FFF80D  ;IEBus receive buffer register
48 I1  IELA1       .equ    H'00FFF80E  ;IEBus lock address register 1
49 I1  IELA2       .equ    H'00FFF80F  ;IEBus lock address register 2
50 I1  IEFLG       .equ    H'00FFF810  ;IEBus general flag register
51 I1  CMX         .equ    7
52 I1  MRQ         .equ    6
53 I1  SRQ         .equ    5
54 I1  SRE         .equ    4
55 I1  LCK         .equ    3
56 I1  RSS         .equ    1
57 I1  GG          .equ    0
58 I1  IETSR       .equ    H'00FFF811  ;IEBus transmit/runaway status register
59 I1  TxRDY       .equ    7
60 I1  IRA         .equ    3
61 I1  TxS         .equ    2
62 I1  TxF         .equ    1
63 I1  TxE         .equ    0
64 I1  IEIET       .equ    H'00FFF812  ;IEBus transmit/runaway interrupt enable register
65 I1  TxRDYE      .equ    7
66 I1  IRAE        .equ    3
67 I1  TxSE        .equ    2
68 I1  TxFE        .equ    1
69 I1  TxEE        .equ    0
70 I1  IETEF       .equ    H'00FFF813  ;IEBus transmit error flag register
71 I1  AL          .equ    4
72 I1  UE          .equ    3
73 I1  TTME        .equ    2
74 I1  RO          .equ    1
75 I1  ACK         .equ    0
76 I1  IERSR       .equ    H'00FFF814  ;IEBus receive status register
77 I1  RxRDY       .equ    7
78 I1  RxS         .equ    2
79 I1  RxF         .equ    1
80 I1  RxE         .equ    0
81 I1  IEIER       .equ    H'00FFF815  ;IEBus receive interrupt enable register
82 I1  RxRDYE      .equ    7
83 I1  RxSE        .equ    2
84 I1  RxFE        .equ    1
85 I1  RxEE        .equ    0
86 I1  IEREF       .equ    H'00FFF816  ;IEBus receive error flag register
87 I1  OVE         .equ    3
88 I1  RTME        .equ    2
89 I1  DLE         .equ    1
90 I1  PE          .equ    0
15     ;----------------------------------------------------------------------
16     ;--Vector table                                                      --
```

```
17    ;---------------------------------------------------------------------
18              .section    VECT,code,locate=0
19              .data.l     preset          ;Power-on reset
20              .org       H'1A0
21              .data.l     rxsti           ;RxSTI interrupt
22              .data.l     rxrdyi          ;RxRDYI interrupt
23              .data.l     txrdyi          ;TxRDYI interrupt
24              .data.l     txsti           ;TxSTI interrupt
25    ;---------------------------------------------------------------------
26    ;--DTC vector table                                                 --
27    ;---------------------------------------------------------------------
28       .org        H'4D2
29       .data.w     (R_MRA - H'FF0000)   ;Start DTC with RxRDYI interrupt
30       .org        H'4D4
31       .data.w     (T_MRA - H'FF0000)     ;Start DTC with TxRDYI interrupt
32
33       .section    PROG,code,locate=H'500
34    ;---------------------------------------------------------------------
35    ;-preset                                                            --
36    ;-Power-on reset processing                                         --
37    ;-[Input]: None                                                     --
38    ;-[Output]: None                                                    --
39    ;---------------------------------------------------------------------
40    preset              .equ   $
41              mov.l     #H'FFEFC0,ER7     ;Set stack pointers
42              mov.b     @MDCR,R0L         ;Latch mode
43              jsr       @io_int           ;Initialize IEBus, etc.
44              andc.b    #H'7F,CCR         ;Clear interrupt mask
45              jsr       @iebus_main
46    preset01
47              bra       preset01          ;Infinite loop
48    ;---------------------------------------------------------------------
49    ;-io_int                                                            --
50    ;-Initialize I/O, IEBus, and DTC                                    --
51    ;-[Input]: None                                                     --
52    ;-[Output]: None                                                    --
53    ;---------------------------------------------------------------------
54    io_int              .equ   $
55              sub.l     ER0,ER0
56              mov.l     #ram_wroks,ER1
57    io_int01
58    mov.l     ER0,@ER1                  ;Clear work area of internal RAM to 0
59              adds.l    #4,ER1
60              cmp.l     #ram_wroke,ER1
61              bcs       io_int01
62
63              bclr.b    #2,@PGDR          ;Fix _TX pin in idle state
64              bclr.b    #2,@PGDDR
65
66              jsr       @dtc_int
67              jsr       @iebus_int
68              rts
69    ;---------------------------------------------------------------------
70    ;-dtc_int                                                           --
```

```
71      ;-Initialize DTC related registers in internal RAM                --
72      ;-[Input]: None                                                   --
73      ;-[Output]: None                                                  --
74      ;-----------------------------------------------------------------
75      dtc_int             .equ   $
76              bclr.b      #MSTPA6,@MSTPCRA  ;Cancel module stop mode for DTC
77                                            ;Initialize DTC registers used for master transmission
78              mov.b       #B'10000000,R0L   ;
79              mov.b       R0L,@T_MRA
80
81              mov.b       #B'00000000,R0L   ;
82              mov.b       R0L,@T_MRB
83
84              mov.l       #IETBR,ER0        ;Set IETBR as destination address
85              mov.w       R0,@T_DAR_L
86              mov.w       E0,R0
87              mov.b       R0L,@T_DAR_H
88
89
90                                            ;Initialize DTC registers used for slave reception
91              mov.b       #B'00100000,R0L   ;
92              mov.b       R0L,@R_MRA
93
94              mov.b       #B'00000000,R0L   ;
95              mov.b       R0L,@R_MRB
96
97              mov.l       #IERBR,ER0        ;Set IERBR as source address
98              mov.w       R0,@R_SAR_L
99              mov.w       E0,R0
100             mov.b       R0L,@R_SAR_H
101             rts
102     ;-----------------------------------------------------------------
103     ;-iebus_int                                                       --
104     ;-Initialize IEBus                                                --
105     ;-[Input]: None                                                   --
106     ;-[Output]: None                                                  --
107     ;-----------------------------------------------------------------
108     iebus_int
109             bclr.b      #MSTPC3,@MSTPCRC   ;Cancel module stop mode for IEBus controller
110
111             mov.b       #B'00101000,R0L    ;Enable receive operation
112             mov.b       R0L,@IECTR
113                                            ;Master transmission settings
114             mov.b       #B'10111111,R0L    ;Set number of retransmissions in case of arbitration
                                                loss to 3
115             mov.b       R0L,@IEMCR         ;Control field is H'0F
116
117             mov.b       #(H'A0 | H'04),R0L ;Lower 4 bits of local address: H'A
118             mov.b       R0L,@IEAR1         ;Operating mode: Mode 1
119
120             mov.b       #H'AA,R0L          ;Upper 8 bits of local address: H'AA
121             mov.b       R0L,@IEAR2
122
123                                            ;Enable receive interrupts
```

```
124             mov.b     #B'00000111,R0L    ;Receive start interrupt
125             mov.b     R0L,@IEIER         ;Receive normal end interrupt
126                                          ;Receive error interrupt
127
128                                          ;Enable transmit interrupts
129             mov.b     #B'00001111,R0L    ;Runaway interrupt
130             mov.b     R0L,@IEIET         ;Transmit start interrupt
131                                          ;Transmit normal end interrupt
132                                          ;Transmit error interrupt
133
134             bset.b    #IEE,@IECTR        ;Start IEBus operation
135
136             rts
137     ;-------------------------------------------------------------------------------
138     ;-iebus_main                                                                   --
139     ;-Transmit frame to the slave 3 times and receive other frame from the master 3 times--
140     ;-[Input]: None                                                                --
141     ;-[Output]: None                                                               --
142     ;-------------------------------------------------------------------------------
143     iebus_main          .equ      $
144               mov.b     @TRANS_FRAMECNT,R0L    ;Frame transmitted 3 times?
145               cmp.b     #3,R0L
146               beq       iebus_main_05
147     iebus_main_01
148               jsr       @master_trans          ;Transmit 1 frame
149     iebus_main_02
150               btst.b    #0,@IEBUS_CONDITION    ;Recognize start of master transmission
151               bne       iebus_main_03          ;
152               btst.b    #1,@IEBUS_CONDITION    ;Recognize start of slave reception
153               bne       iebus_main_04
154               mov.b     @IEBUS_ERROR,R0L       ;
155               bne       iebus_main_01          ;
156               bra       iebus_main_02
157     iebus_main_03
158               btst.b    #0,@IEBUS_CONDITION    ;Recognize end of master transmission
159               bne       iebus_main_03
160               mov.b     @IEBUS_ERROR,R0L       ;Recognize transmit error
161               beq       iebus_main             ;
162               mov.b     #0,R0L
163               mov.b     R0L,@IEBUS_ERROR       ;
164               bra       iebus_main_01          ;Retransmission processing
165     iebus_main_04
166               btst.b    #1,@IEBUS_CONDITION    ;Recognize end of slave reception
167               bne       iebus_main_04
168               mov.b     @IEBUS_ERROR,R0L       ;Recognize receive error
169               beq       iebus_main_02
170               mov.b     #0,R0L
171               mov.b     R0L,@IEBUS_ERROR       ;Handle receive error within interrupt
172               bra       iebus_main_02
173     iebus_main_05
174               mov.b     @RECEIV_FRAMECNT,R0L   ;Frame received 3 times?
175               cmp.b     #3,R0L
176               beq       iebus_main_08
177     iebus_main_06
```

```
178             btst.b    #1,@IEBUS_CONDITION    ;Recognize start of slave reception
179             beq       iebus_main_06
180     iebus_main_07
181             btst.b    #1,@IEBUS_CONDITION    ;Recognize end of slave reception
182             bne       iebus_main_07
183             mov.b     @IEBUS_ERROR,R0L       ;Recognize receive error
184             beq       iebus_main_05
185             mov.b     #0,R0L
186             mov.b   R0L,@IEBUS_ERROR         ;Handle receive error within interrupt
187             bra       iebus_main_05
188     iebus_main_08
189             rts
190     ;---------------------------------------------------------------------
191     ;-master_trans                                                      --
192     ;-Perform master transmission                                       --
193     ;-[Input]: None                                                     --
194     ;-[Output]: Output @IEBUS_CMD command                               --
195     ;---------------------------------------------------------------------
196     master_trans      .equ       $
197             jsr       @mtrans_setup          ;Setup for master transmission
198             mov.b     #H'02,R0L              ;Transfer request as master
199             mov.b     R0L,@IEBUS_CMD
200             jsr       @iebuscmd_input
201             rts
202     ;---------------------------------------------------------------------
203     ;-mtrans_setup                                                      --
204     ;-Setup for performing master transmission                          --
205     ;-[Input]: None                                                     --
206     ;-[Output]: None                                                    --
207     ;---------------------------------------------------------------------
208     mtrans_setup       .equ       $
209             mov.l     #transdata,ER1         ;Search which frame will be transmitted
210             mov.b     @TRANS_FRAMECNT,R0L
211             sub.b     R0H,R0H
212     mtrans_setup01
213             cmp.b     R0H,R0L
214             beq       mtrans_setup02
215             add.l     #37,ER1
216             inc.b     R0H
217             bra       mtrans_setup01
218     mtrans_setup02
219             mov.b     @ER1+,R0L              ;Set broadcast/individual
220             beq       mtrans_setup03
221             bclr.b    #SS,@IEMCR
222             bra       mtrans_setup04
223     mtrans_setup03
224             bset.b    #SS,@IEMCR
225     mtrans_setup04                           ;Set slave address for transmission destination
226             mov.b    @ER1+,R0L
227             mov.b    R0L,@IESA2
228             mov.b    @ER1+,R0L
229             mov.b    R0L,@IESA1
230
231             mov.b    @IEMCR,R0H             ;Set control field
```

```
232             mov.b    #H'F0,R0L
233             and.b    R0L,R0H
234             mov.b    @ER1+,R0L
235             or.b     R0L,R0H
236             mov.b    R0H,@IEMCR
237     mtrans_setup05
238             sub.w    R0,R0                  ;Set transmit message length and CRA of DTC
239             mov.b    @ER1+,R0L
240             mov.b    R0L,@IETBFL
241             dec.b    R0L
242             mov.w    R0,@T_CRA
243     mtrans_setup06
244             mov.b    @ER1+,R0L
245             mov.b    R0L,@IETBR             ;Write 1st byte of data to IETBR
246             bclr.b   #TxRDY,@IETSR          ;Set 1st byte of transmission data
247                                             ;Transfer 2nd and subsequent bytes using DTC
248             mov.w    R1,@T_SAR_L            ;Set SAR of DTC
249             mov.w    E1,R0                  ;Storage destination for transmission data
250             mov.b    R0L,@T_SAR_H
251     mtrans_setup07
252             rts
253     ;---------------------------------------------------------------------
254     ;-iebuscmd_input                                                     --
255     ;-Input commands to IEBus                                            --
256     ;-[Input]: @IEBUS_CMD command                                       --
257     ;-[Output]: None                                                    --
258     ;---------------------------------------------------------------------
259     iebuscmd_input    .equ    $
260             btst.b   #CMX,@IEFLG            ;What for CMX flag to be cleared to 0
261             bne       iebuscmd_input
262             mov.b     @IEBUS_CMD,R0L
263             mov.b     R0L,@IECMR            ;Input command
264             rts
265
266     ;---------------------------------------------------------------------
267     ;-txrdyi interrupt                                                   --
268     ;-Halt transmit-data-ready interrupt (halt data transfer by DTC)    --
269     ;-[Input]: None                                                     --
270     ;-[Output]: None                                                    --
271     ;---------------------------------------------------------------------
272     txrdyi   .equ       $
273             bclr.b    #TxRDYE,@IEIET        ;Halts interrupt
274             rte
275     ;---------------------------------------------------------------------
276     ;-txsti                                                             --
277     ;-IEBus transmission related interrupts                             --
278     ;-[Input]: None                                                     --
279
280     ;-[Output]: See output values for individual subroutines            --
281     ;---------------------------------------------------------------------
282     txsti    .equ       $
283             push.l    ER0
284             push.l    ER1
285
```

```
286             btst.b    #IRA,@IETSR           ;Detect interrupt source
287             bne       txsti01
288             btst.b    #TxS,@IETSR
289             bne       txsti02
290             btst.b    #TxF,@IETSR
291             bne       txsti03
292
293             jsr       @txe_sub              ;To transmit error routine
294             bra       txsti04
295     txsti01
296             jsr       @ira_sub              ;To IEBus runaway routine
297             bra       txsti04
298     txsti02
299             jsr       @txs_sub              ;To transmit start routine
300             bra       txsti04
301     txsti03
302             jsr       @txf_sub              ;To transmit end routine
303     txsti04
304             pop.l     ER1
305             pop.l     ER0
306             rte
307     ;---------------------------------------------------------------------
308     ;-ira_sub                                                          --
309     ;-IEBus runaway routine                                            --
310     ;-[Input]: None                                                    --
311     ;-[Output]:@IEBUS_ERROR=H'0A                                       --
312     ;-         @IEBUS_CONDITION=0                                      --
313     ;---------------------------------------------------------------------
314     ira_sub           .equ       $
315           bclr.b    #IRA,@IETSR             ;Clear interrupt source
316           bclr.b    #TxRDYE,@IEIET          ;Halt transmission data setup interrupt
317           bclr.b    #DTCEG5,@DTCERG         ;Halt DTC operation
318           bclr.b    #RxRDYE,@IEIER          ;Halt ready-to-receive interrupt
319           bclr.b    #DTCEG6,@DTCERG         ;Halt DTC
320
321           bset.b   #MSTPC3,@MSTPCRC         ;Set module stop mode for IEBus (reset)
322           jsr      @iebus_int               ;Reinitialize IEBus
323           mov.b    #H'0A,R0L
324           mov.b    R0L,@IEBUS_ERROR         ;Set error code
325           sub.b    R0L,R0L                  ;End transmit/receive
326           mov.b    R0L,@IEBUS_CONDITION
327           rts
328     ;-----------------------------------------------------------------------------------
329     ;-txs_sub                                                                        --
330     ;-Processing of IEBus transmit start interrupt (start data transmission using DTC)  --
331     ;-[Input]: None                                                                  --
332     ;-[Output]:@IEBUS_CONDITION=H'01                                                 --
333     ;-----------------------------------------------------------------------------------
334     txs_sub           .equ       $
335             bclr.b    #TxS,@IETSR
336             bset.b    #DTCEG5,@DTCERG        ;Enable DTC operation
337             bset.b    #TxRDYE,@IEIET         ;Enable TxRDYI interrupt
338             bset.b    #0,@IEBUS_CONDITION    ;Start transmission
339             rts
```

```
340     ;---------------------------------------------------------------------
341     ;-txf_sub                                                           --
342     ;-End IEBus transmission. Count number of frames transmitted        --
343     ;-[Input]: None                                                     --
344     ;-[Output]:@TRANS_FRAMECNT++                                        --
345     ;-        @IEBUS_CONDITION=0                                        --
346     ;---------------------------------------------------------------------
347     txf_sub            .equ       $
348           bclr.b    #TxF,@IETSR              ;Clear interrupt source
349           mov.b     @TRANS_FRAMECNT,R0L
350           inc.b     R0L
351           mov.b     R0L,@TRANS_FRAMECNT
352           bclr.b    #0,@IEBUS_CONDITION      ;End transmission
353           rts
354     ;---------------------------------------------------------------------
355     ;-txe_sub                                                           --
356     ;-Transmit error processing of IEBus                                --
357     ;-[Input]:No                                                        --
358     ;-[Output]:@IEBUS_ERROR                                             --
359     ;-                   H'05:Arbitration error                         --
360     ;-                   H'06:Underrun error                            --
361     ;-                   H'07:Timing error                              --
362     ;-                   H'08:Transfer byte over                        --
363     ;-                   H'09:ACK error                                 --
364     ;-         @IEBUS_CONDITION=0                                       --
365     ;---------------------------------------------------------------------
366     txe_sub  .equ      $
367           bclr.b    #TxE,@IETSR              ;Clear interrupt source
368           bclr.b    #TxRDYE,@IEIET           ;Halt transmit-data-ready interrupt
369           bclr.b    #DTCEG5,@DTCERG          ;Halt DTC
370
371           btst.b    #AL,@IETEF               ;Search error contents
372           bne       txe_sub01
373           btst.b    #UE,@IETEF
374           bne       txe_sub02
375           btst.b    #TTME,@IETEF
376           bne       txe_sub03
377
378           bclr.b    #RO,@IETEF               ;Clear transfer byte over
379           mov.b     #H'08,R0L                ;Set error code
380           bra       txe_sub04                ;Transfer byte over caused by ACK error?
381                                              ;Verify
382     txe_sub01
383           bclr.b    #AL,@IETEF               ;Clear arbitration loss error
384           mov.b     #H'05,R0L                ;Set error code
385           bra       txe_sub05
386     txe_sub02
387           bclr.b    #UE,@IETEF               ;Clear underrun error
388           mov.b     #H'06,R0L                ;Set error code
389           bra       txe_sub05
390     txe_sub03
391           bclr.b    #TTME,@IETEF             ;Clear timing error
392           mov.b     #H'07,R0L                ;Set error code
393           bra       txe_sub05
```

```
394     txe_sub04
395           btst.b    #ACK,@IETEF
396           beq       txe_sub05
397           bclr.b    #ACK,@IETEF              ;Clear ACK error
398           mov.b     #H'09,R0L                ;Set error code
399     txe_sub05
400           bclr.b    #0,@IEBUS_CONDITION      ;End transmission
401           mov.b     R0L,@IEBUS_ERROR
402           rts
403     ;---------------------------------------------------------------------
404     ;-rxrdyi interrupt                                                  --
405     ;-Halt receive data setup interrupt (end data reception using DTC)  --
406     ;-[Input]: None                                                     --
407     ;-[Output]: None                                                    --
408     ;---------------------------------------------------------------------
409     rxrdyi   .equ      $
410              bclr.b    #RxRDY,@IERSR          ;Clear interrupt source
411              bclr.b    #RxRDYE,@IEIER         ;Halt ready-to-transmit interrupt
412              rte
413     ;---------------------------------------------------------------------
414     ;-rxsti interrupt                                                   --
415     ;-IEBus reception related interrupt                                 --
416     ;-[Input]: None                                                     --
417     ;-[Output]: See output values for individual subroutines            --
418     ;---------------------------------------------------------------------
419     rxsti    .equ        $
420              push.l    ER0
421              push.l    ER1
422     btst.b   #RxS,@IERSR                      ;Search for interrupt request source
423              bne       rxsti01
424              btst.b    #RxF,@IERSR
425              bne       rxsti02
426              jsr       @rxe_sub               ;To rxe_sub
427              bra       rxsti03
428     rxsti01
429              jsr       @rxs_sub               ; To rxs_sub
430              bra       rxsti03
431     rxsti02
432              jsr       @rxf_sub               ; To rxf_sub
433     rxsti03
434              pop       ER1
435              pop       ER0
436              rte
437     ;-------------------------------------------------------------------------------
438     ;-rxs_sub                                                                    --
439     ;-Processing of IEBus receive start interrupt (start data reception using DTC) --
440     ;-[Input]: None                                                              --
441     ;-[Output]:@IEBUS_CONDITION=H'02                                             --
442     ;-------------------------------------------------------------------------------
443     rxs_sub  .equ     $
444              bclr.b    #RxS,@IERSR
445              jsr       @sreceiv_setup
446
447              bset.b    #DTCEG6,@DTCERG        ;Enable DTC start
```

```
448             bset.b    #RxRDYE,@IEIER         ;Enable RxRDY interrupt
449             bset.b    #1,@IEBUS_CONDITION    ;
450     rxs_sub01
451             rts
452     ;---------------------------------------------------------------------
453     ;-sreceiv_setup                                                     --
454     ;-Setup for performing slave reception                              --
455     ;-[Input]: None                                                     --
456     ;-[Output]: None                                                    --
457     ;---------------------------------------------------------------------
458     sreceiv_setup    .equ    $
459           btst.b    #RSS,@IEFLG         ;Determine broadcast/individual
460           beq       sreceiv_setup02     ;Select receive buffer
461           mov.l     #NORMAL_BUFF,ER1    ;Use normal communication receive buffer
462           bra       sreceiv_setup04
463     sreceiv_setup02
464           btst.b    #GG,@IEFLG          ;Verify general broadcast or group broadcast
465           beq       sreceiv_setup03
466           mov.l     #BROAD_BUFF,ER1     ;Use general broadcast communication receive buffer
467           bra       sreceiv_setup04
468     sreceiv_setup03
469           mov.l     #GROUP_BUFF,ER1     ;Use group broadcast communication receive buffer
470     sreceiv_setup04
471           mov.b     @IEMA2,R0L          ;Set master address in receive buffer
472           mov.b     R0L,@ER1
473           inc.l     #1,ER1
474           mov.b     @IEMA1,R0L
475           mov.b     R0L,@ER1
476           inc.l     #1,ER1
477
478           mov.b     @IERCTL,R0L         ;Set control field in receive buffer
479           mov.b     R0L,@ER1
480           inc.l     #1,ER1
481
482           sub.w     R0,R0
483           mov.b     @IERBFL,R0L         ;Set message length in receive buffer
484           mov.b     R0L,@ER1
485           mov.w     R0,@R_CRA           ;Set CRA (transfer counter) for bytes received by DTC
486
487           inc.l     #1,ER1
488           mov.w     R1,@R_DAR_L         ;Set DAR of DTC
489           mov.w     E1,R0
490           mov.b     R0L,@R_DAR_H
491           rts
492     ;---------------------------------------------------------------------
493     ;-rxf_sub                                                           --
494     ;-IEBus receive end interrupt, count the number of frames received  --
495     ;-[Input]: None                                                     --
496     ;-[Output]:@RECEIV_FRAMECNT++                                       --
497     ;-         @IEBUS_CONDITION=0                                       --
498     ;---------------------------------------------------------------------
499     rxf_sub  .equ     $
500             bclr.b    #RxF,@IERSR
501             mov.b     @RECEIV_FRAMECNT,R0L
```

```
            inc.b     R0L
            mov.b     R0L,@RECEIV_FRAMECNT
            bclr.b    #1,@IEBUS_CONDITION        ;End reception
            rts
    ;---------------------------------------------------------------------
    ;-rxe_sub                                                          --
    ;-IEBus receive error interrupt                                    --
    ;-[Input]: None                                                    --
    ;-[Output]:@IEBUS_ERROR                                            --
    ;-                     H'01: Timing error                          --
    ;-                     H'02: Transfer byte count error             --
    ;-                     H'03: Parity error                          --
    ;-                     H'04: Overrun error                         --
    ;-        @IEBUS_CONDITION=0                                       --
    ;---------------------------------------------------------------------
    rxe_sub  .equ       $
          bclr.b    #RxE,@IERSR                 ;Clear RxE
          bclr.b    #RxRDYE,@IEIER              ;Halt receive ready interrupt
          bclr.b    #DTCEG6,@DTCERG             ;Halt DTC

          btst.b    #RTME,@IEREF                ;Search error contents
          bne       rxe_sub01
          btst.b    #DLE,@IEREF                 ;Transfer byte over
          bne       rxe_sub02
          btst.b    #RSS,@IEFLG                 ;During normal communication reception,
          bne       rxe_sub06                   ;discard frame being received in case of
                                                 parity error or overrun error.
          bra    rxe_sub03                      ;(Wait for transfer byte over error.)
                                                 ;During broadcast communication reception,
                                                 ;clear error immediately
    rxe_sub01
          mov.b     #H'01,R0L                  ;Set error code
          bclr.b    #RTME,@IEREF               ;Clear timing error
          bra       rxe_sub05
    rxe_sub02
          mov.b     #H'02,R0L                  ;Set error code
          bclr.b    #DLE,@IEREF                ;Clear transfer byte error
    rxe_sub03
          btst.b    #PE,@IEREF
          beq       rxe_sub04
          mov.b     #H'03,R0L                  ;Set error code
          bclr.b    #PE,@IEREF                 ;Clear parity error
          bra       rxe_sub05
    rxe_sub04
          btst.b    #OVE,@IEREF
          beq       rxe_sub05
          mov.b     #H'04,R0L                  ;Set error code
          bclr.b    #OVE,@IEREF                ;Clear overrun error
          bclr.b    #RxRDY,@IERSR
    rxe_sub05
          mov.b     R0L,@IEBUS_ERROR
          bclr.b    #1,@IEBUS_CONDITION        ;End reception
    rxe_sub06
          rts
```

```
555     ;---------------------------------------------------------------------
556     ;Transmission data table                                             --
557     ;The following frames are transmitted to the slave                   --
558     ;---------------------------------------------------------------------
559     transdata .equ    $
560
561     tdata0    .data.b  H'00                                    ;Normal communication
562               .data.b  H'BB,H'B0                               ;Slave address
563               .data.b  H'0F                                    ;Control field
564               .data.b  H'20                                    ;Message length
565               .data.b  H'11,H'11,H'11,H'11,H'11,H'11,H'11,H'11 ;Data (32 bytes)
566               .data.b  H'11,H'11,H'11,H'11,H'11,H'11,H'11,H'11
567               .data.b  H'11,H'11,H'11,H'11,H'11,H'11,H'11,H'11
568               .data.b  H'11,H'11,H'11,H'11,H'11,H'11,H'11,H'11
569
570     tdata1    .data.b  H'01                                    ;General broadcast communication
571               .data.b  H'FF,H'F0                               ;Slave address
572               .data.b  H'0F                                    ;Control field
573               .data.b  H'20                                    ;Message length
574               .data.b  H'22,H'22,H'22,H'22,H'22,H'22,H'22,H'22 ;Data (32 bytes)
575               .data.b  H'22,H'22,H'22,H'22,H'22,H'22,H'22,H'22
576               .data.b  H'22,H'22,H'22,H'22,H'22,H'22,H'22,H'22
577               .data.b  H'22,H'22,H'22,H'22,H'22,H'22,H'22,H'22
578
579     tdata2    .data.b  H'01                                    ;Group broadcast communication
580               .data.b  H'BB,H'B0                               ;Slave address
581               .data.b  H'0F                                    ;Control field
582               .data.b  H'20                                    ;Message length
583 .              data.b  H'33,H'33,H'33,H'33,H'33,H'33,H'33,H'33 ;Data (32 bytes)
584               .data.b  H'33,H'33,H'33,H'33,H'33,H'33,H'33,H'33
585               .data.b  H'33,H'33,H'33,H'33,H'33,H'33,H'33,H'33
586               .data.b  H'33,H'33,H'33,H'33,H'33,H'33,H'33,H'33
587     ;---------------------------------------------------------------------
588     ;--Variable definitions (internal RAM area)                          --
589     ;---------------------------------------------------------------------
590               .section     RAM_TOP,data,locate=H'FFB000
591     ram_wroks           .equ     $
592     NORMAL_BUFF         .res.b   36     ;Normal communication receive buffer
593     BROAD_BUFF          .res.b   36     ;Simultaneous broadcast communication receive buffer
594     GROUP_BUFF          .res.b   36     ;Group broadcast communication receive buffer
595     IEBUS_ERROR         .res.b   1      ;IEBus error variable
596     IEBUS_CMD           .res.b   1      ;IEBus command storage
597     TRANS_FRAMECNT      .res.b   1      ;Transmit frame counter
598     RECEIV_FRAMECNT     .res.b   1      ;Receive frame counter
599     IEBUS_CONDITION     .res.b   1      ;IEBus transmit/receive status monitor
600                .section     RAM_DTC,data,locate=H'FFEC00
601     ;--DTC register information during master transmission
602     T_MRA               .res.b   1      ;MRA
603     T_SAR_H             .res.b   1      ;SAR
604     T_SAR_L             .res.w   1      ;
605     T_MRB               .res.b   1      ;MRB
606     T_DAR_H             .res.b   1      ;DAR
607     T_DAR_L             .res.w   1      ;
608     T_CRA               .res.w   1      ;CRA
```

```
609     T_CRB              .res.w   1      ;CRB
610     ;--DTC register information during slave reception
611     R_MRA              .res.b   1      ;MRA
612     R_SAR_H            .res.b   1      ;SAR
613     R_SAR_L            .res.w   1      ;
614     R_MRB              .res.b   1      ;MRB
615     R_DAR_H            .res.b   1      ;DAR
616     R_DAR_L            .res.w   1      ;
617     R_CRA              .res.w   1      ;CRA
618     R_CRB              .res.w   1      ;CRB
619     ram_wroke     .equ         $
620             .end
```

Revision Record

		Description	
Rev.	**Date**	**Page**	**Summary**
1.00	Mar.09.05	—	First edition issued

Keep safety first in your circuit designs!

1. Renesas Technology Corp. puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage.
   Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corp. product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corp. or a third party.
2. Renesas Technology Corp. assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corp. without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor for the latest product information before purchasing a product listed herein.
   The information described here may contain technical inaccuracies or typographical errors.
   Renesas Technology Corp. assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
   Please also pay attention to information published by Renesas Technology Corp. by various means, including the Renesas Technology Corp. Semiconductor home page (http://www.renesas.com).
4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corp. assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corp. semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
6. The prior written approval of Renesas Technology Corp. is necessary to reprint or reproduce in whole or in part these materials.
7. If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
   Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
8. Please contact Renesas Technology Corp. for further details on these materials or the products contained therein.