H8S Family

Using the HCAN (4): Standard Format, 8 Bytes of Data, with Priority

Introduction

The Controller Area Network (HCAN) module is used to control the Controller Area Network (CAN), which provides a means for real-time communications in automobiles and industrial equipment systems.

This application note presents an example of communications operation using the H8S/2636's on-chip HCAN module and is offered to users for reference in the software and hardware design processes.

Although the operation of the sample application and programs provided in this application note has been confirmed, please verify operation in your environment before actually using them.

Target Device

H8S/2636

Contents

1. Specifications

Between two H8S/2636 devices, eight bytes of data are transmitted in a standard format message from multiple mailboxes in order of message identifier priority. The messages are received solely by mailbox 0.

(1) Specifications common to the transmitter and receiver

- Channel 0 (HCAN0) is used
- Baud rate: 250 Kbps (in 20-MHz operation)

(2) Specifications of the transmitter

- Uses mailboxes 1 to 15
- Transmits messages in order of priority given to the message identifiers
- Data length is eight bytes for each mailbox, and message identifiers and data for transmission are as shown in table 1

Table 1 Setting Values for Mailboxes

Mailbox No.	Identifier (11 Bits)	Data (8 Bytes)	Priority
1	H'666	H'1111 1111 1111 1111	12
2	H'2AA	H'2222 2222 2222 2222	5
3	H'777	H'3333 3333 3333 3333	14
4	H'333	H'4444 4444 4444 4444	6
5	H'088	H'5555 5555 5555 5555	1
6	H'4CC	H'6666 6666 6666 6666	9
7	H'199	H'7777 7777 7777 7777	3
8	H'7FF	H'8888 8888 8888 8888	15
9	H'111	H'9999 9999 9999 9999	2
10	H'444	H'AAAA AAAA AAAA AAAA	8
11	H'555	H'BBBB BBBB BBBB BBBB	10
12	H'6EE	H'CCCC CCCC CCCC CCCC	13
13	H'3BB	H'DDDD DDDD DDDD DDDD	7
14	H'222	H'EEEE EEEE EEEE EEEE	4
15	H'5DD	H'FFFF FFFF FFFF FFFF	11

- Transmits all messages in sequence at a time
- Polls the transmission-complete flag during transmission
- After confirming that the transmission-complete flag has been set, clears the flag as the final operation

Specifications of the receiver:

- Uses mailbox 0
- Does not set the message identifier mask and receives all messages
- Uses the receive message interrupt (IRRI)
  (a) The DTC is activated on a receive message interrupt and stores the received data in on-chip RAM.
  (b) The DTC is used in block-transfer mode and transfers 15 blocks, each consisting of eight bytes.
  (c) After the DTC transfer has ended, the receive message interrupt routine clears the reception-complete flag and disables the receive message interrupt, after which the operation is over.

2. Functional Descriptions of the Transmitter and Receiver

Tables 2 and 3 list the function assignments of the relevant pins and registers.

Table 2 Function Assignment for the HCAN Module

Pin Usage			**Function**
Pin	HTxD0		Used for message transmission by the HCAN module (pin 97)
	HRxD0		Used for message reception by the HCAN module (pin 98)

Relevant Registers			**Function**
Registers common to transmission and reception	MSTPCRC		Module stop control register C Takes HCAN0 out of the module stop mode.
	IRR		Interrupt register Indicates the states of individual interrupt sources.
	BCR		Bit configuration register Configures the baud-rate prescaler for CAN and sets up the bit-timing parameters.
	MBCR		Mailbox configuration register Configures mailboxes for transmission or reception.
	MCR		Master control register Controls the CAN interface.
	GSR		General status register Indicates the CAN bus states.
	MCx[n]		Message control registers (x = mailbox number)
		n = 1	Sets the data length for data frames and remote frames.
		n = 2 to 4	Reserved
		n = 5	Holds standard ID bits (STD_ID2 to STD_ID0), extended ID bits (EXD_ID17 and EXD_ID16), RTR (indicates data frame or remote frame), and IDE (indicates standard format or extended format).
		n = 6	Holds standard ID bits (STD_ID10 to STD_ID3)
		n = 7	Holds extended ID bits (EXD_ID7 to EXD_ID0)
		n = 8	Holds extended ID bits (EXD_ID15 to EXD_ID8)
	MDx[n]		Message data registers (x = mailbox number)
		n = 1 to 8	Hold CAN message data for transmission or received CAN message data.
Transmission-related registers	TXPR		Transmit wait register After a message for transmission has been stored in the mailbox, the corresponding bit in this register is set, indicating a transmission-wait state.
	TXACK		Transmit acknowledge register Each bit in this register indicates whether or not the message in the corresponding mailbox has been transmitted normally.
Reception-related registers	RXPR		Receive complete register Each bit in this register indicates that a message has been received normally in the corresponding mailbox.
	LAFMH, LAFML		Local acceptance filter mask H, L Identifier filter mask settings for the mailboxes configured for reception.

Relevant Registers		Function
Interrupt-related registers	MBIMR	Mailbox interrupt mask register Enables or disables interrupt requests for the individual mailboxes.
	IMR	Interrupt mask register Enables or disables interrupt requests by the IRR interrupt flag.
	IPRM	Interrupt priority register Sets the priority level for HCAN interrupts.
	SYSCR	System control register Sets the interrupt control mode.

Table 2 Function Assignment for the DTC

DTC-Related Registers	Function
MSTPCRA	Module stop control register A Takes the DTC out of module stop mode.
MRA, MRB	DTC mode register A, B Control the operating mode of the DTC.
SAR	DTC source address register Specifies the address of the source data area for DTC transfer.
DAR	DTC destination address register Specifies the address of the destination data area for DTC transfer.
CRA	DTC transfer count register A Specifies the number of data transfers by the DTC.
CRB	DTC transfer count register B In block transfer mode, specifies the block length.
DTCERA to DTCERG	DTC enable register Selects the interrupt source that activates the DTC.
DTVECR	DTC vector register Enables or disables activation of the DTC by software and sets the vector number for the software activation interrupt.

3. Flowchart for the Transmitter

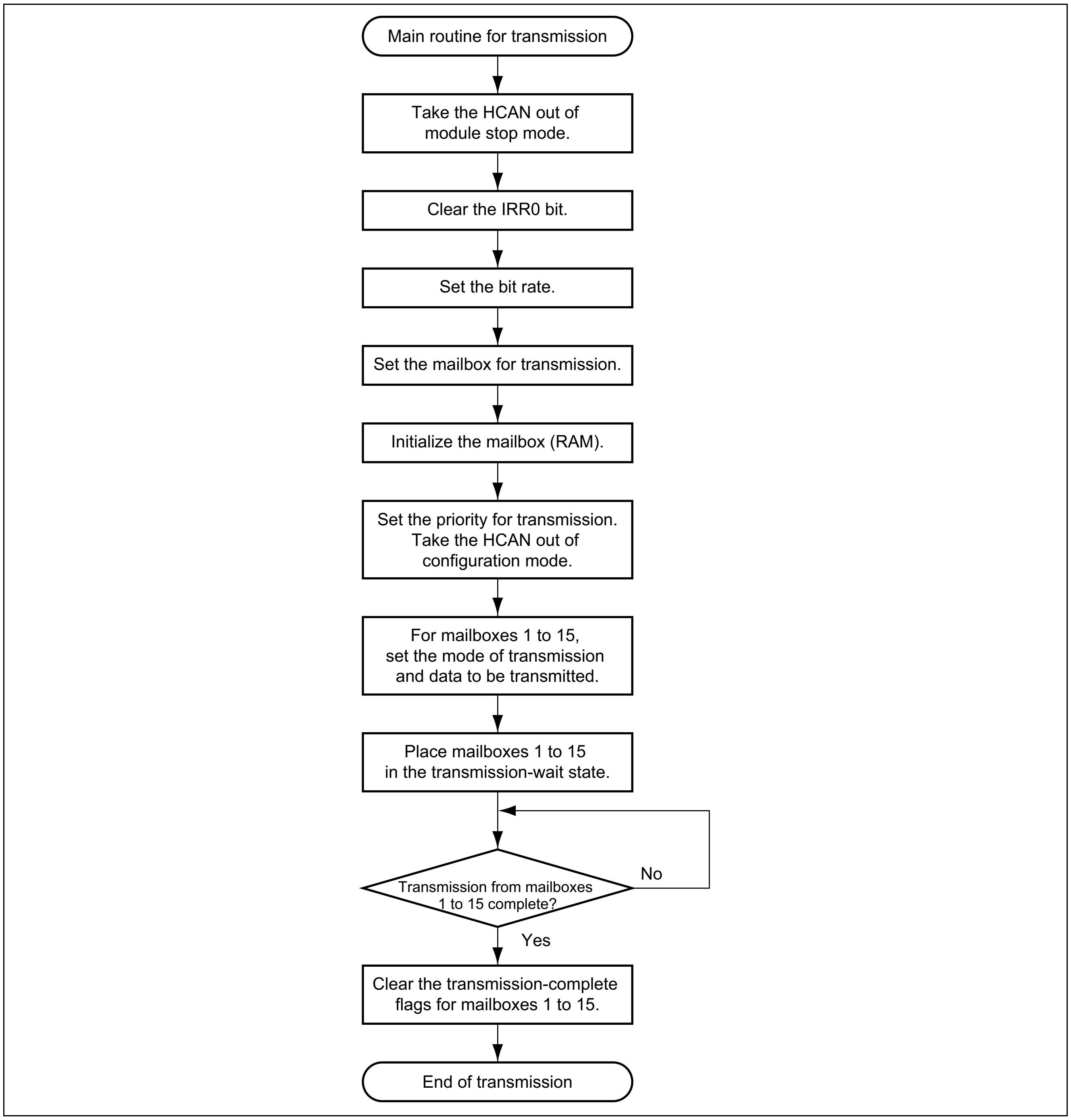
Main routine for transmission
Take the HCAN out of module stop mode.
Clear the IRR0 bit.
Set the bit rate.
Set the mailbox for transmission.
Initialize the mailbox (RAM).
Set the priority for transmission. Take the HCAN out of configuration mode.
For mailboxes 1 to 15, set the mode of transmission and data to be transmitted.
Place mailboxes 1 to 15 in the transmission-wait state.
Transmission from mailboxes 1 to 15 complete?
No
Yes
Clear the transmission-complete flags for mailboxes 1 to 15.
End of transmission

Figure 1 Flowchart for the Transmitter

4. Description of Software (Transmitter)

4.1 Module

Table 3 Description of Module

Module	Label	Function
Main Routine	t_main	Initialize the HCAN and makes settings for transmission.

4.2 Registers

Table 4 Description of Registers*

Register	Function	Setting	Used in
MSTP.CRC.BYTE	Takes HCAN0 out of module stop mode.	H'F7	Main routine
HCAN0.IRR.WORD	The reset interrupt flag in this register is cleared. (Clearing condition: writing a 1 to the bit)	H'0100	
HCAN0.BCR.WORD	Sets the bit rate to 250 Kbps when ϕ = 20 MHz	H'0334	
HCAN0.MBCR.WORD	Sets mailboxes 1 to 15 for transmission.	H'0100	
HCAN0.MCR.BYTE	Selects transmission in order of message identifier priority and takes the HCAN module out of configuration mode.	H'00	
HCAN0.GSR.BYTE	Checked to confirm that HCAN0 is out of configuration mode.	—	
HCAN0.MC[x][4]	For mailbox x, sets the frame type to data frame and the frame format to standard format. Also holds the message identifier bits, STD_ID2 to STD_ID0.	See table 1	
HCAN0.MC[x][5]	Holds the message identifier bits, STD_ID10 to STD_ID3	See table 1	
HCAN0.MC[x][0]	Sets the data length for transmission from mailbox x to eight bytes.	H'08	
HCAN0.MD[x][y]	Holds data for transmission from mailboxes x.	See table 1	
HCAN0.TXPR.WORD	Places mailboxes 1 to 15 in the transmission-wait state.	H'FEFF	
HCAN0.TXACK.WORD	Checked to see if the transmission-complete flags for mailboxes 1 to 15 are set; when set, the flags are cleared. (Clearing condition: writing a 1 to the bit)	H'FEFF	

Note: * The register names shown above are defined in a header file which is available for downloading from the following web page.
http://download.renesas.com/eng/mpumcu/sample_codes/h8sx_h8s_h8_family/io_register/index.html
x = 1 to 15, y = 0 to 7

5. Program Listing (Transmission)

```
/**********************************************************************************************/
/*  HCAN Transmission Program (No.4)                                                          */
/**********************************************************************************************/
#include <stdio.h>                      /* Header file for library functions                  */
#include <machine.h>                    /* Header file for library functions                  */
#include "2636S.h"                      /* Header file of peripheral register definitions     */
void t_main(void){
    unsigned char i,j;
/* Initialization */
    MSTP.CRC.BYTE = 0xF7;               /* Cancel module stop mode of HCAN                    */
    HCAN0.IRR.WORD = 0x0100;            /* Initialize reset flag for HCAN module              */
    HCAN0.BCR.WORD = 0x0334;            /* Bit rate: 250 kbps                                 */
    HCAN0.MBCR.WORD = 0x0100;           /* Set mailboxes 1 to 15 for transmission             */
    for(i=0; i<=15; i++){               /* Initialize mailboxes (RAM)                         */
        for(j=0; j<=7; j++){
            HCAN0.MC[i][j] = 0x00;
        }
    }
    for(i=0; i<=15; i++){               /* Initialize mailboxes (RAM)                         */
        for(j=0; j<=7; j++){
            HCAN0.MD[i][j] = 0x00;
        }
    }
    HCAN0.MCR.BYTE = 0x00;              /* Transmission in message identifier priority order  */
    while(HCAN0.GSR.BYTE & 0x08);       /* Configuration mode cancellation check              */
/* Transmit data setting */
/*******  Mail Box 1  *******/
    HCAN0.MC[1][4] = 0xC0;              /* Standard format, data frame, and identifier setting */
    HCAN0.MC[1][5] = 0xCC;              /* Identifier setting (STD: 0x666)                    */
    HCAN0.MC[1][0] = 0x08;              /* Data length: 8 bytes                               */
    HCAN0.MD[1][0] = 0x11;              /* Message data: 00010001                             */
    HCAN0.MD[1][1] = 0x11;              /* Message data: 00010001                             */
    HCAN0.MD[1][2] = 0x11;              /* Message data: 00010001                             */
    HCAN0.MD[1][3] = 0x11;              /* Message data: 00010001                             */
    HCAN0.MD[1][4] = 0x11;              /* Message data: 00010001                             */
    HCAN0.MD[1][5] = 0x11;              /* Message data: 00010001                             */
    HCAN0.MD[1][6] = 0x11;              /* Message data: 00010001                             */
    HCAN0.MD[1][7] = 0x11;              /* Message data: 00010001                             */
/*******  Mail Box 2  *******/
    HCAN0.MC[2][4] = 0x40;              /* Standard format, data frame, and identifier setting */
    HCAN0.MC[2][5] = 0x55;              /* Identifier setting (STD: 0x2AA)                    */
    HCAN0.MC[2][0] = 0x08;              /* Data length: 8 bytes                               */
    HCAN0.MD[2][0] = 0x22;              /* Message data: 00100010                             */
    HCAN0.MD[2][1] = 0x22;              /* Message data: 00100010                             */
    HCAN0.MD[2][2] = 0x22;              /* Message data: 00100010                             */
    HCAN0.MD[2][3] = 0x22;              /* Message data: 00100010                             */
    HCAN0.MD[2][4] = 0x22;              /* Message data: 00100010                             */
    HCAN0.MD[2][5] = 0x22;              /* Message data: 00100010                             */
    HCAN0.MD[2][6] = 0x22;              /* Message data: 00100010                             */
    HCAN0.MD[2][7] = 0x22;              /* Message data: 00100010                             */
```

```
/*******  Mail Box 3  *******/
    HCAN0.MC[3][4] = 0xE0;          /* Standard format, data frame, and identifier setting */
    HCAN0.MC[3][5] = 0xEE;          /* Identifier setting (STD: 0x777)                     */
    HCAN0.MC[3][0] = 0x08;          /* Data length: 8 bytes                                */
    HCAN0.MD[3][0] = 0x33;          /* Message data: 00110011                              */
    HCAN0.MD[3][1] = 0x33;          /* Message data: 00110011                              */
    HCAN0.MD[3][2] = 0x33;          /* Message data: 00110011                              */
    HCAN0.MD[3][3] = 0x33;          /* Message data: 00110011                              */
    HCAN0.MD[3][4] = 0x33;          /* Message data: 00110011                              */
    HCAN0.MD[3][5] = 0x33;          /* Message data: 00110011                              */
    HCAN0.MD[3][6] = 0x33;          /* Message data: 00110011                              */
    HCAN0.MD[3][7] = 0x33;          /* Message data: 00110011                              */
/*******  Mail Box 4  *******/
    HCAN0.MC[4][4] = 0x60;          /* Standard format, data frame, and identifier setting */
    HCAN0.MC[4][5] = 0x66;          /* Identifier setting (STD: 0x333)                     */
    HCAN0.MC[4][0] = 0x08;          /* Data length: 8 bytes                                */
    HCAN0.MD[4][0] = 0x44;          /* Message data: 01000100                              */
    HCAN0.MD[4][1] = 0x44;          /* Message data: 01000100                              */
    HCAN0.MD[4][2] = 0x44;          /* Message data: 01000100                              */
    HCAN0.MD[4][3] = 0x44;          /* Message data: 01000100                              */
    HCAN0.MD[4][4] = 0x44;          /* Message data: 01000100                              */
    HCAN0.MD[4][5] = 0x44;          /* Message data: 01000100                              */
    HCAN0.MD[4][6] = 0x44;          /* Message data: 01000100                              */
    HCAN0.MD[4][7] = 0x44;          /* Message data: 01000100                              */
/*******  Mail Box 5  *******/
    HCAN0.MC[5][4] = 0x00;          /* Standard format, data frame, and identifier setting */
    HCAN0.MC[5][5] = 0x11;          /* Identifier setting (STD: 0x088)                     */
    HCAN0.MC[5][0] = 0x08;          /* Data length: 8 bytes                                */
    HCAN0.MD[5][0] = 0x55;          /* Message data: 01010101                              */
    HCAN0.MD[5][1] = 0x55;          /* Message data: 01010101                              */
    HCAN0.MD[5][2] = 0x55;          /* Message data: 01010101                              */
    HCAN0.MD[5][3] = 0x55;          /* Message data: 01010101                              */
    HCAN0.MD[5][4] = 0x55;          /* Message data: 01010101                              */
    HCAN0.MD[5][5] = 0x55;          /* Message data: 01010101                              */
    HCAN0.MD[5][6] = 0x55;          /* Message data: 01010101                              */
    HCAN0.MD[5][7] = 0x55;          /* Message data: 01010101                              */
/*******  Mail Box 6  *******/
    HCAN0.MC[6][4] = 0x80;          /* Standard format, data frame, and identifier setting */
    HCAN0.MC[6][5] = 0x99;          /* Identifier setting (STD: 0x4CC)                     */
    HCAN0.MC[6][0] = 0x08;          /* Data length: 8 bytes                                */
    HCAN0.MD[6][0] = 0x66;          /* Message data: 01100110                              */
    HCAN0.MD[6][1] = 0x66;          /* Message data: 01100110                              */
    HCAN0.MD[6][2] = 0x66;          /* Message data: 01100110                              */
    HCAN0.MD[6][3] = 0x66;          /* Message data: 01100110                              */
    HCAN0.MD[6][4] = 0x66;          /* Message data: 01100110                              */
    HCAN0.MD[6][5] = 0x66;          /* Message data: 01100110                              */
    HCAN0.MD[6][6] = 0x66;          /* Message data: 01100110                              */
    HCAN0.MD[6][7] = 0x66;          /* Message data: 01100110                              */
```

```
/*******  Mail Box 7  *******/
    HCAN0.MC[7][4] = 0x20;           /* Standard format, data frame, and identifier setting */
    HCAN0.MC[7][5] = 0x33;           /* Identifier setting (STD: 0x199)                     */
    HCAN0.MC[7][0] = 0x08;           /* Data length: 8 bytes                                */
    HCAN0.MD[7][0] = 0x77;           /* Message data: 01110111                              */
    HCAN0.MD[7][1] = 0x77;           /* Message data: 01110111                              */
    HCAN0.MD[7][2] = 0x77;           /* Message data: 01110111                              */
    HCAN0.MD[7][3] = 0x77;           /* Message data: 01110111                              */
    HCAN0.MD[7][4] = 0x77;           /* Message data: 01110111                              */
    HCAN0.MD[7][5] = 0x77;           /* Message data: 01110111                              */
    HCAN0.MD[7][6] = 0x77;           /* Message data: 01110111                              */
    HCAN0.MD[7][7] = 0x77;           /* Message data: 01110111                              */
/*******  Mail Box 8  *******/
    HCAN0.MC[8][4] = 0xE0;           /* Standard format, data frame, and identifier setting */
    HCAN0.MC[8][5] = 0xFF;           /* Identifier setting (STD: 0x7FF)                     */
    HCAN0.MC[8][0] = 0x08;           /* Data length: 8 bytes                                */
    HCAN0.MD[8][0] = 0x88;           /* Message data: 10001000                              */
    HCAN0.MD[8][1] = 0x88;           /* Message data: 10001000                              */
    HCAN0.MD[8][2] = 0x88;           /* Message data: 10001000                              */
    HCAN0.MD[8][3] = 0x88;           /* Message data: 10001000                              */
    HCAN0.MD[8][4] = 0x88;           /* Message data: 10001000                              */
    HCAN0.MD[8][5] = 0x88;           /* Message data: 10001000                              */
    HCAN0.MD[8][6] = 0x88;           /* Message data: 10001000                              */
    HCAN0.MD[8][7] = 0x88;           /* Message data: 10001000                              */
/*******  Mail Box 9  *******/
    HCAN0.MC[9][4] = 0x20;           /* Standard format, data frame, and identifier setting */
    HCAN0.MC[9][5] = 0x22;           /* Identifier setting (STD: 0x111)                     */
    HCAN0.MC[9][0] = 0x08;           /* Data length: 8 bytes                                */
    HCAN0.MD[9][0] = 0x99;           /* Message data: 10011001                              */
    HCAN0.MD[9][1] = 0x99;           /* Message data: 10011001                              */
    HCAN0.MD[9][2] = 0x99;           /* Message data: 10011001                              */
    HCAN0.MD[9][3] = 0x99;           /* Message data: 10011001                              */
    HCAN0.MD[9][4] = 0x99;           /* Message data: 10011001                              */
    HCAN0.MD[9][5] = 0x99;           /* Message data: 10011001                              */
    HCAN0.MD[9][6] = 0x99;           /* Message data: 10011001                              */
    HCAN0.MD[9][7] = 0x99;           /* Message data: 10011001                              */
/*******  Mail Box 10  *******/
    HCAN0.MC[10][4] = 0x80;          /* Standard format, data frame, and identifier setting */
    HCAN0.MC[10][5] = 0x88;          /* Identifier setting (STD: 0x444)                     */
    HCAN0.MC[10][0] = 0x08;          /* Data length: 8 bytes                                */
    HCAN0.MD[10][0] = 0xAA;          /* Message data: 10101010                              */
    HCAN0.MD[10][1] = 0xAA;          /* Message data: 10101010                              */
    HCAN0.MD[10][2] = 0xAA;          /* Message data: 10101010                              */
    HCAN0.MD[10][3] = 0xAA;          /* Message data: 10101010                              */
    HCAN0.MD[10][4] = 0xAA;          /* Message data: 10101010                              */
    HCAN0.MD[10][5] = 0xAA;          /* Message data: 10101010                              */
    HCAN0.MD[10][6] = 0xAA;          /* Message data: 10101010                              */
    HCAN0.MD[10][7] = 0xAA;          /* Message data: 10101010                              */
```

```
/*******  Mail Box 11  *******/
    HCAN0.MC[11][4] = 0xA0;           /* Standard format, data frame, and identifier setting */
    HCAN0.MC[11][5] = 0xAA;           /* Identifier setting (STD: 0x555)                     */
    HCAN0.MC[11][0] = 0x08;           /* Data length: 8 bytes                                */
    HCAN0.MD[11][0] = 0xBB;           /* Message data: 10111011                              */
    HCAN0.MD[11][1] = 0xBB;           /* Message data: 10111011                              */
    HCAN0.MD[11][2] = 0xBB;           /* Message data: 10111011                              */
    HCAN0.MD[11][3] = 0xBB;           /* Message data: 10111011                              */
    HCAN0.MD[11][4] = 0xBB;           /* Message data: 10111011                              */
    HCAN0.MD[11][5] = 0xBB;           /* Message data: 10111011                              */
    HCAN0.MD[11][6] = 0xBB;           /* Message data: 10111011                              */
    HCAN0.MD[11][7] = 0xBB;           /* Message data: 10111011                              */
/*******  Mail Box 12  *******/
    HCAN0.MC[12][4] = 0xC0;           /* Standard format, data frame, and identifier setting */
    HCAN0.MC[12][5] = 0xDD;           /* Identifier setting (STD: 0x6EE)                     */
    HCAN0.MC[12][0] = 0x08;           /* Data length: 8 bytes                                */
    HCAN0.MD[12][0] = 0xCC;           /* Message data: 11001100                              */
    HCAN0.MD[12][1] = 0xCC;           /* Message data: 11001100                              */
    HCAN0.MD[12][2] = 0xCC;           /* Message data: 11001100                              */
    HCAN0.MD[12][3] = 0xCC;           /* Message data: 11001100                              */
    HCAN0.MD[12][4] = 0xCC;           /* Message data: 11001100                              */
    HCAN0.MD[12][5] = 0xCC;           /* Message data: 11001100                              */
    HCAN0.MD[12][6] = 0xCC;           /* Message data: 11001100                              */
    HCAN0.MD[12][7] = 0xCC;           /* Message data: 11001100                              */
/*******  Mail Box 13  *******/
    HCAN0.MC[13][4] = 0x60;           /* Standard format, data frame, and identifier setting */
    HCAN0.MC[13][5] = 0x77;           /* Identifier setting (STD: 0x3BB)                     */
    HCAN0.MC[13][0] = 0x08;           /* Data length: 8 bytes                                */
    HCAN0.MD[13][0] = 0xDD;           /* Message data: 11011101                              */
    HCAN0.MD[13][1] = 0xDD;           /* Message data: 11011101                              */
    HCAN0.MD[13][2] = 0xDD;           /* Message data: 11011101                              */
    HCAN0.MD[13][3] = 0xDD;           /* Message data: 11011101                              */
    HCAN0.MD[13][4] = 0xDD;           /* Message data: 11011101                              */
    HCAN0.MD[13][5] = 0xDD;           /* Message data: 11011101                              */
    HCAN0.MD[13][6] = 0xDD;           /* Message data: 11011101                              */
    HCAN0.MD[13][7] = 0xDD;           /* Message data: 11011101                              */
/*******  Mail Box 14  *******/
    HCAN0.MC[14][4] = 0x40;           /* Standard format, data frame, and identifier setting */
    HCAN0.MC[14][5] = 0x44;           /* Identifier setting (STD: 0x222)                     */
    HCAN0.MC[14][0] = 0x08;           /* Data length: 8 bytes                                */
    HCAN0.MD[14][0] = 0xEE;           /* Message data: 11101110                              */
    HCAN0.MD[14][1] = 0xEE;           /* Message data: 11101110                              */
    HCAN0.MD[14][2] = 0xEE;           /* Message data: 11101110                              */
    HCAN0.MD[14][3] = 0xEE;           /* Message data: 11101110                              */
    HCAN0.MD[14][4] = 0xEE;           /* Message data: 11101110                              */
    HCAN0.MD[14][5] = 0xEE;           /* Message data: 11101110                              */
    HCAN0.MD[14][6] = 0xEE;           /* Message data: 11101110                              */
    HCAN0.MD[14][7] = 0xEE;           /* Message data: 11101110                              */
```

```
/*******  Mail Box 15  *******/
    HCAN0.MC[15][4] = 0xA0;            /* Standard format, data frame, and identifier setting */
    HCAN0.MC[15][5] = 0xBB;            /* Identifier setting (STD: 0x5DD)                     */
    HCAN0.MC[15][0] = 0x08;            /* Data length: 8 bytes                                */
    HCAN0.MD[15][0] = 0xFF;            /* Message data: 11111111                              */
    HCAN0.MD[15][1] = 0xFF;            /* Message data: 11111111                              */
    HCAN0.MD[15][2] = 0xFF;            /* Message data: 11111111                              */
    HCAN0.MD[15][3] = 0xFF;            /* Message data: 11111111                              */
    HCAN0.MD[15][4] = 0xFF;            /* Message data: 11111111                              */
    HCAN0.MD[15][5] = 0xFF;            /* Message data: 11111111                              */
    HCAN0.MD[15][6] = 0xFF;            /* Message data: 11111111                              */
    HCAN0.MD[15][7] = 0xFF;            /* Message data: 11111111                              */
/* Message transmission */
    HCAN0.TXPR.WORD = 0xFEFF;          /* Place mailboxes 1 to 15 in transmission wait state  */
    while((HCAN0.TXACK.WORD & 0xFEFF) != 0xFEFF); /* Wait until transmission is complete      */
/* Transmission-complete flag clearing */
    HCAN0.TXACK.WORD &= 0xFEFF;        /* Clear transmission-complete flag                    */
    while(1);
}
```

6. Flowchart for the Receiver

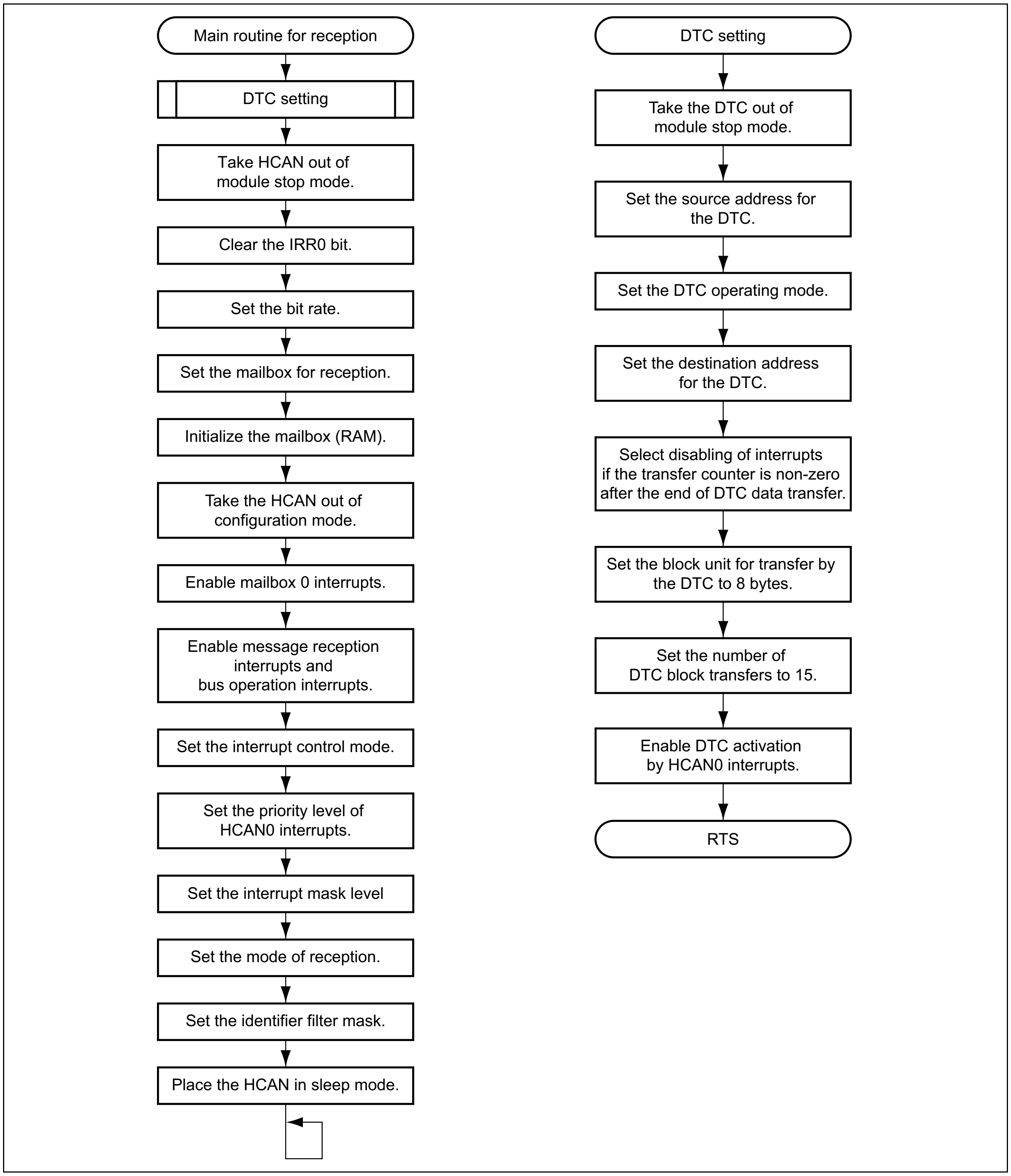
Main routine for reception
DTC setting
Take HCAN out of module stop mode.
Clear the IRR0 bit.
Set the bit rate.
Set the mailbox for reception.
Initialize the mailbox (RAM).
Take the HCAN out of configuration mode.
Enable mailbox 0 interrupts.
Enable message reception interrupts and bus operation interrupts.
Set the interrupt control mode.
Set the priority level of HCAN0 interrupts.
Set the interrupt mask level
Set the mode of reception.
Set the identifier filter mask.
Place the HCAN in sleep mode.
DTC setting
Take the DTC out of module stop mode.
Set the source address for the DTC.
Set the DTC operating mode.
Set the destination address for the DTC.
Select disabling of interrupts if the transfer counter is non-zero after the end of DTC data transfer.
Set the block unit for transfer by the DTC to 8 bytes.
Set the number of DTC block transfers to 15.
Enable DTC activation by HCAN0 interrupts.
RTS

Figure 2 Flowchart for the Receiver

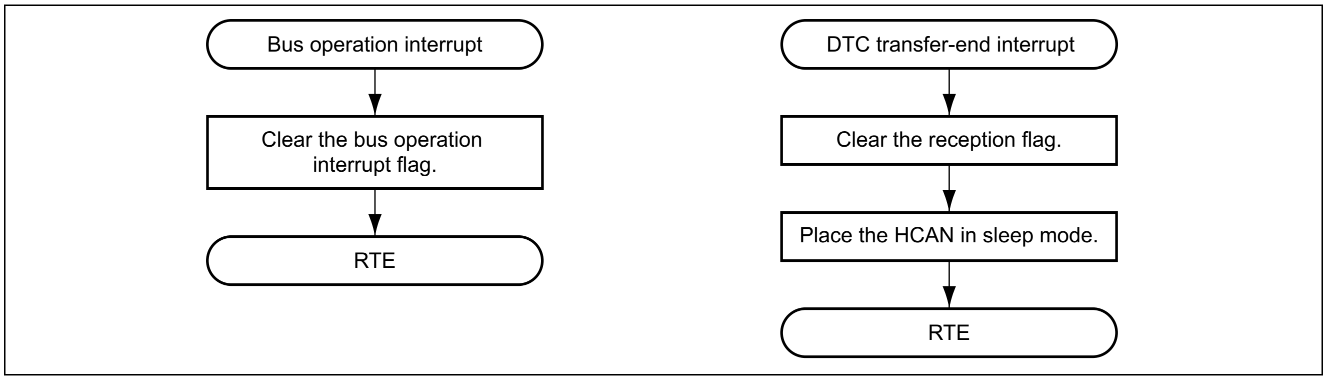
Bus operation interrupt
Clear the bus operation interrupt flag.
RTE
DTC transfer-end interrupt
Clear the reception flag.
Place the HCAN in sleep mode.
RTE

Figure 3 Flowchart of Interrupt Routines for the Receiver

7. Description of Software (Receiver)

7.1 Modules

Table 5 Description of Modules

Module	Label	Function
Main Routine	r_main	Initializes the HCAN and makes settings for reception.
Bus operation interrupt routine	OVR0_IRR12	Clears the bus operation interrupt flag.
DTC transfer-end interrupt routine	DTCend_RM0	Clears the reception flag and places the HCAN in sleep mode.

7.2 Registers

Table 6 Description of Registers*

Register	Function	Setting	Used in
MAILBOX.MDATA[1][8]	Storage for the received data Address range: H'FFE000 to H'FFE007	—	Main routine
MSTP.CRA.BYTE	Takes the DTC out of module stop mode.	H'3F	
DTC_SAR	Sets the first address of the message data area for mailbox 0 as the source address for transfer.	H'FFF8B0	
DTC_MRA	Sets the incrementation of both DTC_SAR and DTC_DAR after each transfer, and selects block transfer mode and byte-sized transfer.	H'AA	
DTC_DAR	Sets the first address of the received-data storage area as the destination address for transfer.	H'FFE000	
DTC_MRB	Disables interrupts to the CPU if the transfer counter value is non-zero after the end of DTC transfer.	H'00	
DTC_CRA	Sets the size of a block (8 bytes).	H'0808	
DTC_CRB	Sets the number of block transfers (15 times).	H'000F	
DTC.DTCEG.BYTE	Selects the DTC activation source (RM0).	H'04	
MSTP.CRC.BYTE	Takes HCAN0 out of module stop mode.	H'F7	
HCAN0.IRR.WORD	The reset interrupt flag in this register is cleared. (Clearing condition: writing a 1 to the bit)	H'0100	
HCAN0.BCR.WORD	Sets the bit rate to 250 Kbps when ϕ = 20 MHz	H'0334	
HCAN0.MBCR.WORD	Sets mailbox 0 for reception.	H'0100	

Register	Function	Setting	Used in
HCAN0.MCR.BYTE	Takes HCAN0 out of configuration mode and places it in sleep mode.	H'FE and H'A0	Main routine
HCAN0.GSR.BYTE	Checked to confirm that HCAN0 is out of configuration mode.	—	
HCAN0.MBIMR.WORD	Enables interrupt requests of mailbox 0.	H'FEFF	
HCAN0.IMR.WORD	Enables message reception and bus operation interrupts.	H'FCEF	
SYSCR.BYTE	Sets the interrupt control mode.	H'20	
INTC.IPRM.BYTE	Sets the priority level of HCAN interrupts.	H'07	
HCAN0.MC[0][4]	For mailbox 0, sets the frame type to data frame and the frame format to standard format. Also holds the message identifier bits, STD_ID2 to STD_ID0 (for message ID = H'555).	H'A0	
HCAN0.MC[0][5]	Holds the message identifier bits, STD_ID10 to STD_ID3 (for message ID = H'555).	H'AA	
HCAN0.LAFMH.WORD	Sets no identifier filter masking.	H'FFFF	
HCAN0.IRR.WORD	The bus-operation interrupt flag in this register is cleared.	H'0010	Bus-operation interrupt routine
HCAN0.RXPR.WORD	The reception-complete flag for mailbox 0 in this register is cleared. (Clearing condition: writing a 1 to the bit)	H'FFFF	DTC transfer-end interrupt routine

Note: * The register names shown above are defined in a header file which is available for downloading from the following web page.
http://download.renesas.com/eng/mpumcu/sample_codes/h8sx_h8s_h8_family/io_register/index.html

8. Program Listing (Reception)

```
/************************************************************************************************/
/*  HCAN Reception Program (No.4)                                                               */
/************************************************************************************************/
#include <stdio.h>                      /* Header file for library functions                   */
#include <machine.h>                    /* Header file for library functions                   */
#include "2636S.h"                      /* Header file of peripheral register definitions      */
/************************************************************************************************/
/*  Definitions of Constants                                                                    */
/************************************************************************************************/
volatile struct MB{                     /* struct MAILBOX0-15                                  */
    unsigned char MDATA[15][8];         /* Storage of received data                            */
};
#define MAILBOX     (volatile struct MB        *)0xFFE000) /* First address of                 */
                                                            /*            received data storage */
#define DTC_SAR     (*(volatile unsigned long  *)0xFFEBC0) /* DTC register info setting        */
#define DTC_MRA     (*(volatile unsigned char  *)0xFFEBC0) /* DTC register info setting        */
#define DTC_DAR     (*(volatile unsigned long  *)0xFFEBC4) /* DTC register info setting        */
#define DTC_MRB     (*(volatile unsigned char  *)0xFFEBC4) /* DTC register info setting        */
#define DTC_CRA     (*(volatile unsigned short *)0xFFEBC8) /* DTC register info setting        */
#define DTC_CRB     (*(volatile unsigned short *)0xFFEBCA) /* DTC register info setting        */

void r_main(void){
    unsigned char i,j;

/* DTC initialization */
    MSTP.CRA.BYTE = 0x3F;               /* Cancel module stop mode of DTC                      */
    DTC_SAR = (long)(&HCAN0.MD[0][0]);  /* Set transfer source address                         */
    DTC_MRA = 0xAA;                     /* SAR and DAR incremented after transfer;             */
                                        /*                             set block transfer mode */
    DTC_DAR = (long)(&MAILBOX.MDATA[0][0]);/* Set transfer destination address (on-chip RAM)   */
    DTC_MRB = 0x00;                     /* Disable interrupt after end of DTC transfer         */
                                        /*                     if transfer counter is non-zero */
    DTC_CRA = 0x0808;                   /* Block transfer size: 8 bytes                        */
    DTC_CRB = 0x000F;                   /* Number of block transfers: 15                       */
    DTC.DTCEG.BYTE |= 0x04;             /* Enable activation of DTC by HCAN0 interrupt (RM0)*/
/* HCAN initialization */
    MSTP.CRC.BYTE = 0xF7;               /* Cancel module stop mode of HCAN                     */
    HCAN0.IRR.WORD = 0x0100;            /* Initialize reset flag for HCAN module               */
    HCAN0.BCR.WORD = 0x0334;            /* Bit rate: 250 kbps                                  */
    HCAN0.MBCR.WORD = 0x0100;           /* Set mailbox 0 for reception                         */
    for(i=0; i<=15; i++){               /* Initialize mailboxes (RAM)                          */
        for(j=0; j<=7; j++){
            HCAN0.MC[i][j] = 0x00;
        }
    }
    for(i=0; i<=15; i++){               /* Initialize mailboxes (RAM)                          */
        for(j=0; j<=7; j++){
            HCAN0.MD[i][j] = 0x00;
        }
    }
    HCAN0.MCR.BYTE &= 0xFE;             /* Cancel configuration mode                           */
    while(HCAN0.GSR.BYTE & 0x08);       /* Configuration mode cancellation check               */
```

```
/* Interrupt settings */
    HCAN0.MBIMR.WORD = 0xFEFF;          /* Enable mailbox 0 interrupt requests             */
    HCAN0.IMR.WORD = 0xFCEF;            /* Enable message reception and                    */
                                        /*                        bus operation interrupts */
    SYSCR.BYTE |= 0x20;                 /* Set interrupt control mode 2                    */
    INTC.IPRM.BYTE = 0x07;              /* Set the priority level of HCAN0 interrupts to 7 */
    set_imask_exr(0);                   /* Set interrupt request mask level                */
/* Reception data settings */
    HCAN0.MC[0][4] = 0xA0;              /* Standard format, data frame, and                */
                                        /*                              identifier setting */
    HCAN0.MC[0][5] = 0xAA;              /* Identifier setting                              */
    HCAN0.LAFMH.WORD = 0xFFFF;          /* Mailbox 0 receives data for any identifiers     */
/* HCAN sleep mode setting */
    HCAN0.MCR.BYTE |= 0xA0;             /* Put HCAN in sleep mode; enable recovery by      */
                                        /*                         bus operation interrupt */
    while(1);
}
/*************************************************************************************/
/*  Bus-Operation Interrupt Routine                                                  */
/*************************************************************************************/
#pragma interrupt(OVR0_IRR12)
void OVR0_IRR12(void){
    HCAN0.IRR.WORD &= 0x0010;           /* Clear IRR12 (bus-operation interrupt flag)      */
}
/*************************************************************************************/
/*  DTC Transfer-End Interrupt Routine                                               */
/*************************************************************************************/
//#pragma interrupt(DTCend_RM0)
void DTCend_RM0(void){
    HCAN0.RXPR.WORD &= 0xFFFF;          /* Clear IRR1 (reception message interrupt flag)   */
/* HCAN sleep mode setting */
    HCAN0.MCR.BYTE |= 0xA0;             /* Put HCAN in sleep mode                          */
}
```

9. Waveforms during Operation (Transmission and Reception)

Figure 4 show the first and last waveforms seen when this application is executed.

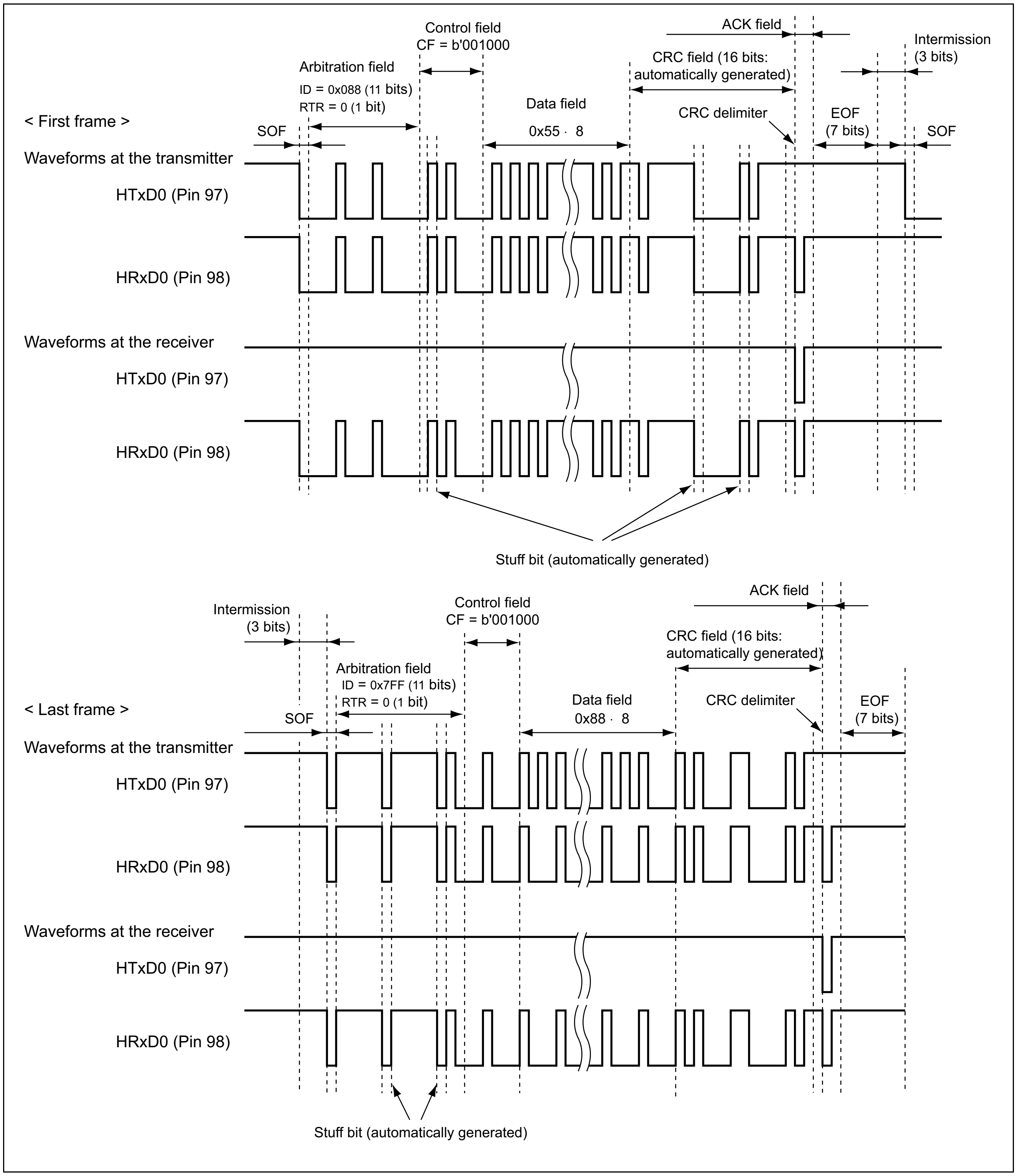

Figure 4 Waveforms during Operation

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Jul.22.05	—	First edition issued