R8C/5x Series

CAN Application note

R01AN0522EJ0110
Rev.1.10
Mar 15. 2012

Abstract

This document describes the procedures for CAN communication using the R8C/5x Series.

The R8C/5x Series has a channel CAN module (CAN0).

Products

R8C/5x Series

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.

The variables j, k and n are used in this document. Those variables represent the following.

j: Mailbox number (j = 0 to 15)

k: Mask register number (k = 0 to 3)

n: CFIDCRn register number (n = 0, 1)

Contents

1. Hardware

1.1 Pins Used

Table 1.1 lists the Pins Used and Their Functions.

Table 1.1 Pins Used and Their Functions

Pin Name	I/O	Function
P6_1/CTX_0	Output	CAN_0 data output port
P6_2/CRX_0/CLK_1	Input	CAN_0 data input port

Note: Set the CPE bit to 1 in CCTLR register.

2. Software

2.1 Initial Settings

The following settings are required for CAN communication:

- CAN Communication Speed Configuration [Refer to section 2.2 “CAN Communication Speed Configuration”.]
  - Clock setting
  - Bit timing setting
  - Baud rate setting
- Acceptance filter setting: [Refer to section 2.8 “Using an Acceptance Filter”.]

2.2 CAN Communication Speed Configuration

2.2.1 CAN Bit Timing

The CAN bit timing settings for the CAN module of the R8C/5x Series affect the three segments that make up each bit of the communication frame.

Figure 2.1 shows the segment structure and the sampling point.

Of the three segments, time segment 1 (TSEG1) and time segment 2 (TSEG2) specify the sampling point. The sampling timing can be adjusted by changing the values of TSEG1 and TSEG2.

The minimum unit in which timing settings are specified is called a time quanta (Tq). The Tq is determined by the clock frequency input to the CAN module and the baud rate prescaler frequency division value.

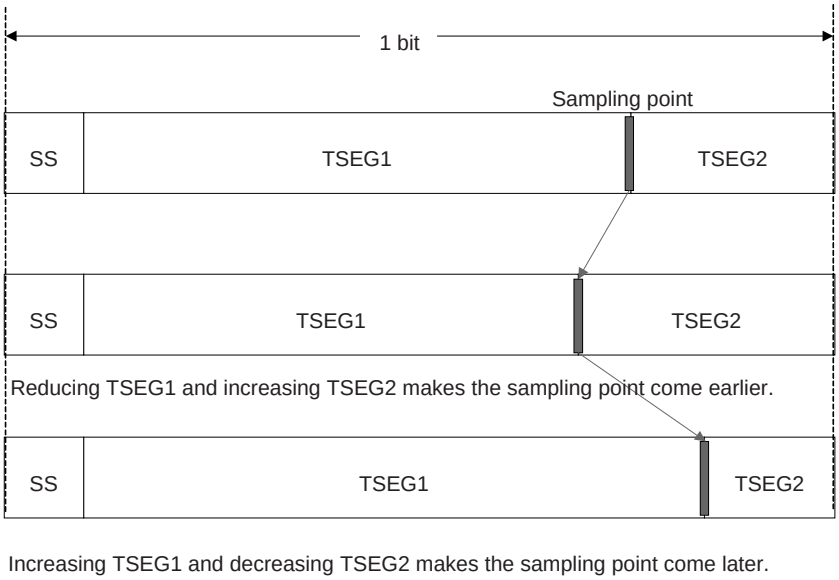


Figure 2.1 Segment Structure of Bit and Sampling Point

(1) SS: Synchronization Segment
This segment is used for synchronization by monitoring the edge transition from recessive to dominant within the interframe space.*[1]

(2) TSEG1: Time Segment 1
This segment's purpose is to absorb the physical delay inherent in the CAN network and to compensate for phase error*[2] occurring during resynchronization. The physical delay inherent in the network is equal to two times the sum of the bus delay, the input comparator delay, and the output driver delay.

(3) TSEG2: Time Segment 2
This segment's purpose is to compensate for phase error*[2] occurring during resynchronization.

(4) SJW: Resynchronization Jump Width
This is the maximum width of compensation to correct synchronization misalignment due to phase error*[2].

Notes: 1. The interframe space comprises three conditions: intermission, suspend transmission, and bus idle. All nodes can start transmission during the bus idle condition.
2. Phase error refers to synchronization misalignment between nodes, during message transmission or reception, due to factors such as a shift in the oscillator frequency or delay in the signal transfer path.

Table 2.1 lists the Settings and Restrictions for Each Segment.

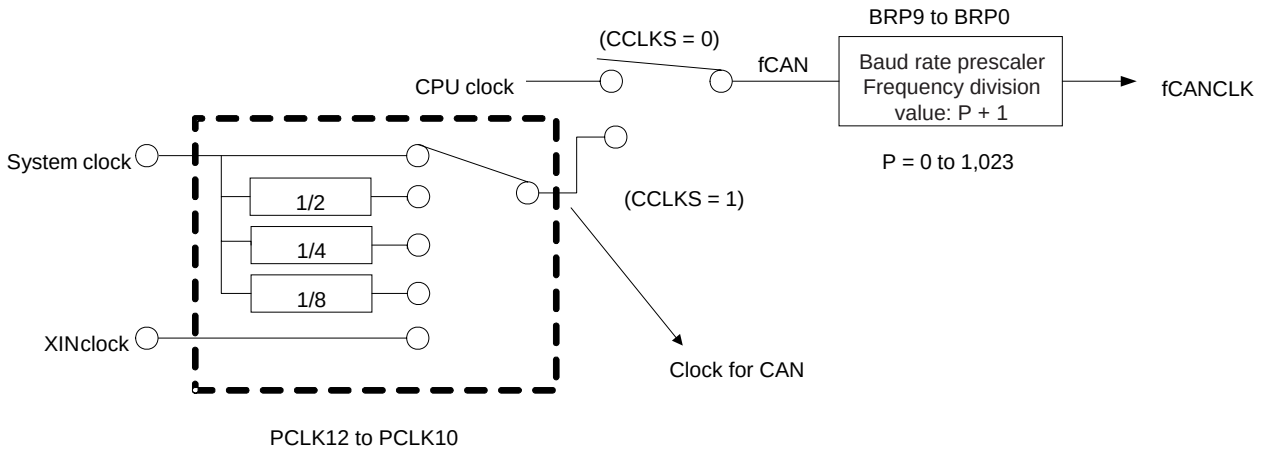
Table 2.1 Settings and Restrictions for Each Segment

Segment	Setting Range
SS	1 Tq, fixed
TSEG1	Setting within range of 4 to 16 Tq
TSEG2	Setting within range of 2 to 8 Tq
SJW	1 to 4 Tq

Notes: 1. SS + TSEG1 + TSEG2 = 8 to 25 Tq
2. TSEG1 > TSEG2 ≥ SJW (however, TSEG2 ≥ 2 when SJW = 1)

2.2.2 Transfer Speed

The transfer speed is determined by fCAN, the baud rate prescaler frequency division value, and the Tq count per bit. Figure 2.2 is a Block Diagram of the CAN Module System Clock.



fCAN: CAN system clock
P: Value selected using bits BRP0 to BRP9 in CBCR register (P = 0 to 1023)
fCANCLK: CAN communication clock (fCANCLK = fCAN / (P + 1))

CCLKS: Bit in CCLKR register (CCLKR)
PCLK12 to PCLK10: Bits in PCLKR1 register 1
BRP9 to BRP0: Bits in CBCR

Notes: 1. Set the PRC0 bit in PRCR register to 1 (write enable) before rewriting the PCLKR1 register
2. Write to this register in CAN reset mode. To set the CCLKS bit to 1, set the CPU clock to equal to or faster than the CAN clock.

Figure 2.2 Block Diagram of CAN Module System Clock Generator Circuit

Table 2.2 shows the main formula for calculating the transfer speed and actual examples, and Table 2.3 shows bit timing setting examples.

Table 2.2 Formula for Calculating Transfer Speed and Actual Examples

Formula for calculating transfer speed	$\frac{\text{fCAN}}{\text{Baud rate prescaler frequency division value}^{*1} \times \text{Tq count per bit}} = \frac{\text{fCANCLK}}{\text{Tq count per bit}}$					
fCAN / Transfer Speed	**20 MHz**		**16 MHz**		**8 MHz**	
	Tq	**P*2 + 1**	**Tq**	**P*2 + 1**	**Tq**	**P*2 + 1**
1 Mbps	10 Tq 20 Tq	2 1	8 Tq 16 Tq	2 1	8 Tq	1
500 Kbps	10 Tq 20 Tq	4 2	8 Tq 16 Tq	4 2	8 Tq 16 Tq	2 1
250 Kbps	10 Tq 20 Tq	8 4	8 Tq 16 Tq	8 4	8Tq 16 Tq	4 2
83.3 Kbps	8 Tq 10 Tq 16 Tq 20 Tq	30 24 15 12	8 Tq 16 Tq	24 12	8 Tq 16 Tq	12 6
33.3 Kbps	8 Tq 10 Tq 20 Tq	75 60 30	8 Tq 10 Tq 16 Tq 20 Tq	60 48 30 24	8 Tq 10 Tq 16 Tq 20 Tq	30 24 15 12

Notes: 1. Baud rate prescaler frequency division value = P + 1 (P = 0 to 1,023)

2. P: Value specified by Bits BRP0 to BRP9 in CBCR_0 register

Table 2.3 Bit Timing Setting Examples

1 Bit	Setting (Tq)				Sampling Point* (%)
	SS	TSEG1	TSEG2	SJW	
8 Tq	1	4	3	1	62.50
	1	5	2	1	75.00
10 Tq	1	6	3	1	70.00
	1	7	2	1	80.00
12 Tq	1	8	3	1	75.00
	1	9	2	1	83.33
15 Tq	1	10	4	1	73.33
	1	11	3	1	80.00
16 Tq	1	10	5	1	68.75
	1	11	4	1	75.00
20 Tq	1	12	7	1	65.00
	1	13	6	1	70.00
24 Tq	1	15	8	1	66.66

Note: The point at which the level of a bit is determined.

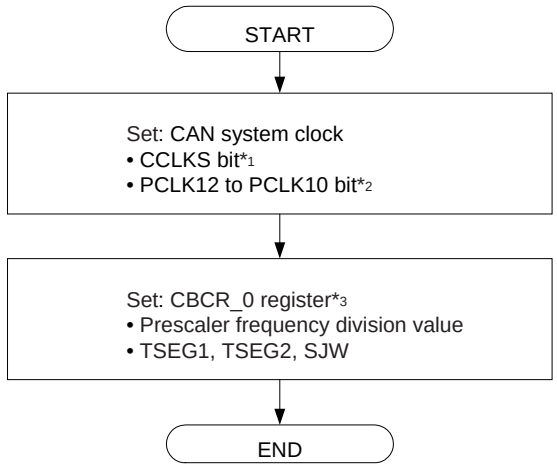
When the sampling point is set to 75%

2.2.3 CAN Bit Timing and Transfer Speed Settings

Figure2.1 shows the procedure for setting the CAN bit timing and transfer speed.

These settings must be performed during CAN configuration.

See section 2.3.1 for details on the CAN configuration procedure.



Notes: 1. Write to this register in CAN reset mode. To set the CCLKS bit to 1, set the CPU clock to equal to or faster than the CAN clock.
2. Set the PRC0 bit in the PRCR register to 1 (write enabled) before rewriting the PCLKR1 register.
3. The size of CAN bit configuration register is 24 bit.

Figure 2.3 CAN Bit Timing and Transfer Speed Setting Procedure

2.3 Transmission and Reception of CAN Messages

The three procedures listed below are used to transmit and receive CAN messages.

(1) CAN Configuration Procedure
During CAN configuration, CAN transfer speed, control mode, acceptance filter, and interrupt settings are made.

(2) Mailbox Configuration Procedure
The mailbox settings for the transmission and reception modes are made in the CAN message control register (CMCTLj_0) corresponding to each mailbox.
Table 3 lists the correspondences between the CAN message control register (CMCTLj_0) settings and the transmission/reception modes.

(3) Data Processing Procedure
This is the message processing that takes place during message transmission and reception as well as at normal reception complete.

These procedures are used in normal mailbox mode (MBM bit = 0). For details on FIFO mailbox mode (MBM bit = 1), see section 2.4.2.

Table 2.4 Correspondences of CAN Message Control Register Settings and Transmission/Reception Modes

TRMREQ*	RECREQ*	ONESHOT*	Mailbox Transmission/Reception Mode Setting
0	0	0	Mailbox unusable or in transmission-abort state or in reception-abort state
0	0	1	Setting possible only in one-shot mode when transmission from or reception to the programmed mailbox has been aborted
0	1	0	Mailbox set as receive mailbox for data frame or remote frame
0	1	1	Mailbox set as one-shot receive mailbox data frame or remote frame
1	0	0	Mailbox set as transmit mailbox for data frame or remote frame
1	0	1	Mailbox set as one-shot transmit mailbox for data frame or remote frame
1	1	0	Setting prohibited
1	1	1	Setting prohibited

Note: Bit in CMCTLj_0 register.

Keep the following points in mind when setting a mailbox as a receive mailbox or one-shot receive mailbox:

(1) Make sure to set the corresponding CAN message control register (CMCTLj_0) to 00h before setting a mailbox as a receive mailbox or one-shot receive mailbox.

(2) When a message is received, it is stored in the first mailbox that matches the conditions, according to the reception mode setting and acceptance filter processing results. In addition, the mailbox with the lower mailbox number is given priority when storing received messages.

(3) In CAN operation mode, the CAN module does not receive the data transmitted by itself, even if the ID/mask of a receive mailbox matches transmitted data. In self-test mode, however, the CAN module receives the transmit data. In this case, the CAN module returns an ACK.

Keep the following point in mind when setting a mailbox as a transmit mailbox or one-shot transmit mailbox:

(1) Before setting a mailbox as a transmit mailbox or one-shot transmit mailbox, make sure to set the corresponding CAN message control register (CMCTLj_0) to 00h and confirm that the mailbox is not in the middle of abort processing.

2.3.1 CAN Configuration

CAN configuration comprises the following three modes:

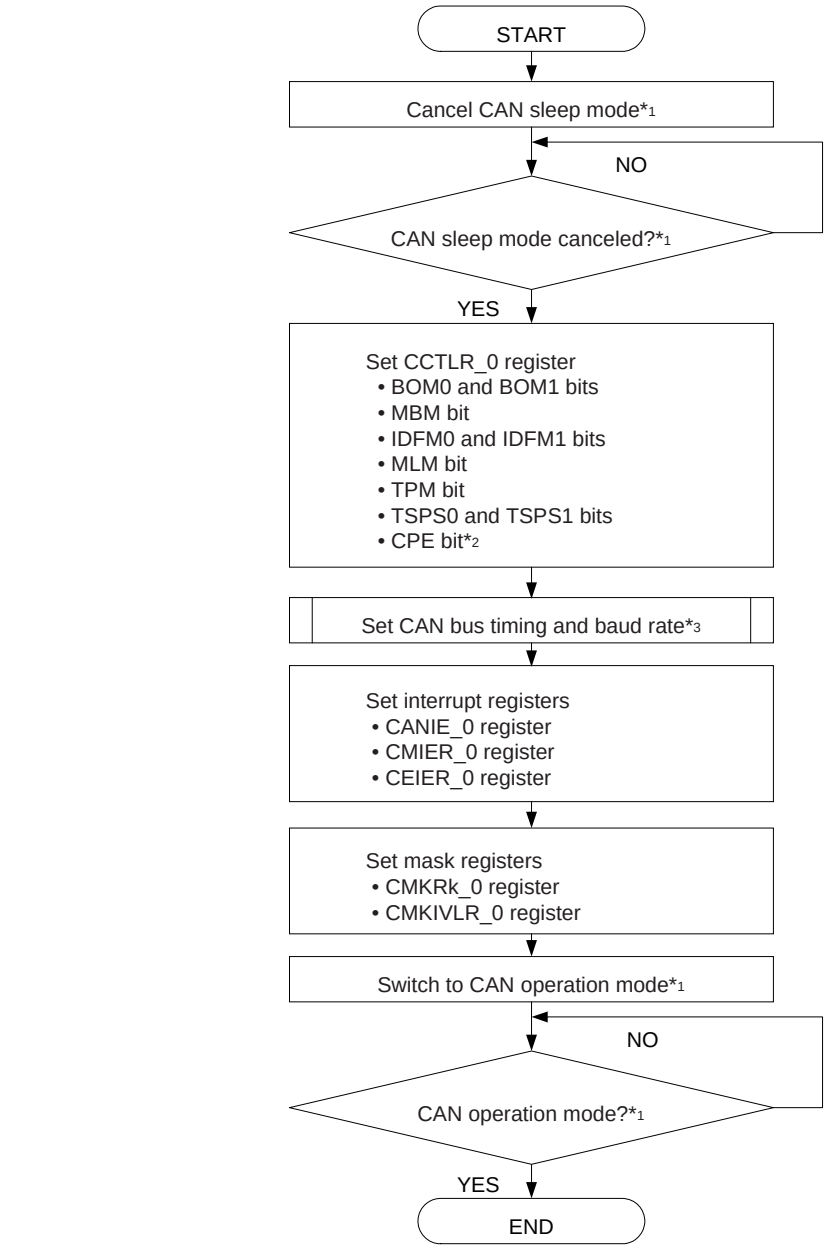
(1) Configuration after a Hardware Reset
This configuration mode is used after a hardware reset.

(2) Configuration after CAN Reset Mode
This configuration mode is used after transitioning to CAN reset mode.
The CAN module is reset, so settings must be made again.
This configuration mode must be used if it is necessary to change the transfer speed.

(3) Configuration after CAN Halt Mode
This configuration mode is used after transitioning to CAN halt mode.
The CAN module is not reset, so settings do not need to be made again.
This configuration mode must be used if it is necessary to stop communication temporarily.

(1) Configuration after a Hardware Reset

Figure 2.4 shows the CAN configuration procedure after a hardware reset.

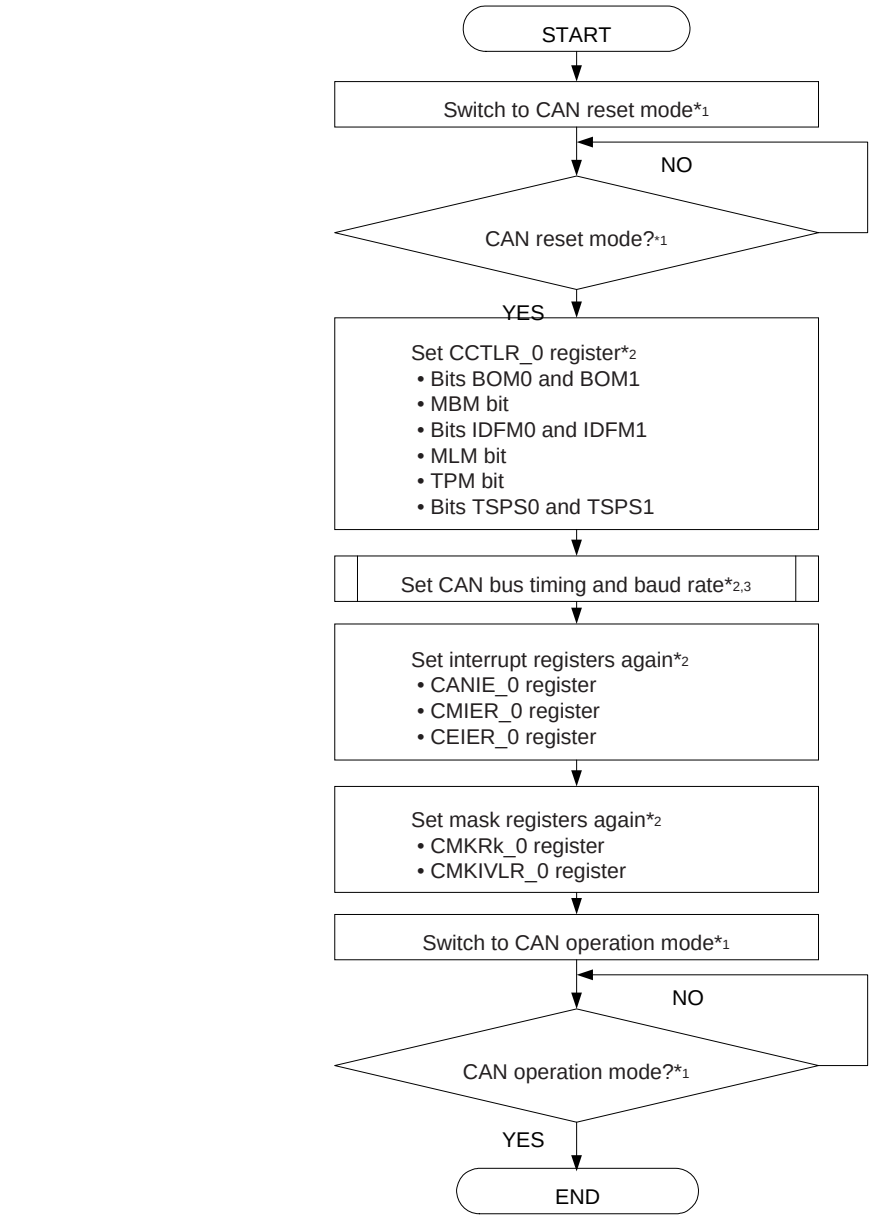


Notes: 1. When bits CANM1, CANM0 and SLPM are changed, check the CAN_0 status register to ensure that the mode has been switched.
Do not change bits CANM1, CANM0 and SLPM until the mode has been switched.
2. The CPE in the CCTLR_0 register is to 1.
3. Please refer to [2.2 CAN Communication Speed Configuration].

Figure 2.4 CAN Configuration Procedure after a Hardware Reset

(2) Configuration after CAN Reset Mode

Figure 2.5 shows the CAN configuration procedure after transitioning to CAN reset mode.

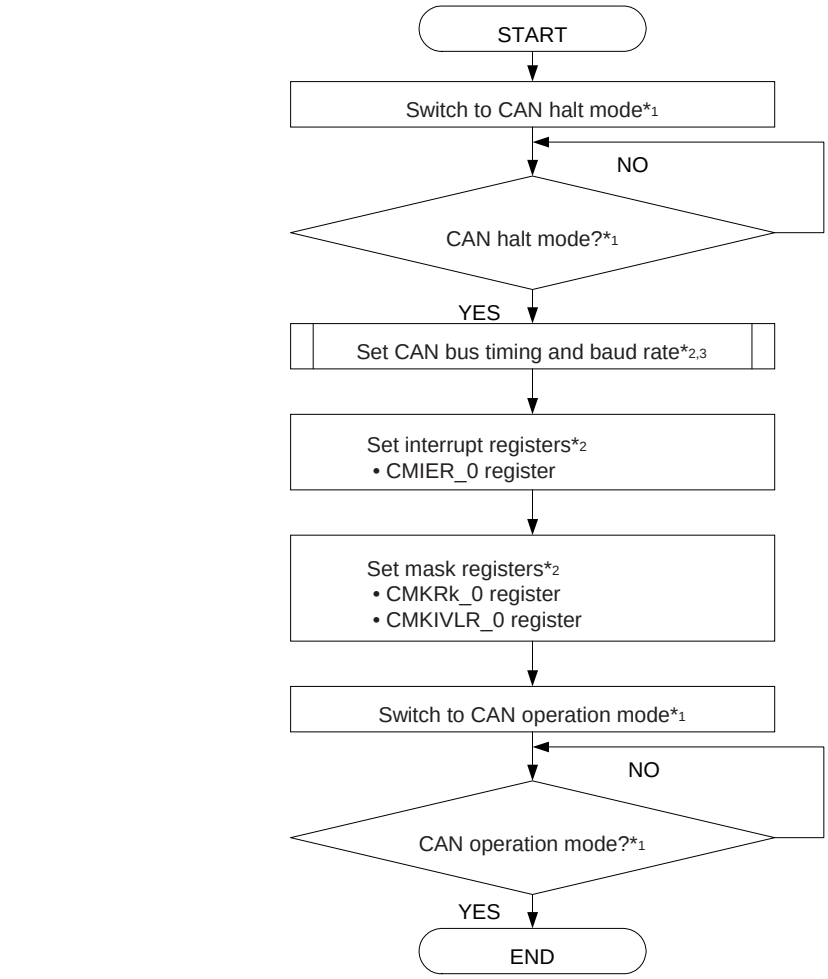


Notes: 1. When bits CANM0, CANM1 and SLPM are changed, check the CSTR_0 register to ensure that the mode has been switched.
Do not change bits CANM0, CANM1 and SLPM until the mode has been switched.
2. This step is not always necessary because these registers are not reset after entering CAN reset mode.
3. Please refer to [2.2 CAN Communication Speed Configuration].

Figure 2.5 CAN Configuration Procedure after Transitioning to CAN Reset Mode

(3) Configuration after CAN Halt Mode

Figure 2.6 shows the CAN configuration procedure after transitioning to CAN halt mode.



Notes: 1. When bits CANM1 and CANM0, check the CSTR_0 register to ensure that the mode has been switched.
Do not change bits CANM1 and CANM0 until the mode has been switched.
2. This step is not always necessary because these registers are not reset after entering CAN halt mode.
3. Please refer to [2.2 CAN Communication Speed Configuration].

Figure 2.6 CAN Configuration Procedure after Transitioning to CAN Halt Mode

2.3.2 Message Transmission

Each CAN module supports 16 mailboxes. The all mailboxes can be used for transmission in transmit modes.

(1) Normal Transmission Mode

When a mailbox is set to normal transmission mode, data frames or remote frames set to that mailbox can be transmitted.

It is possible to confirm whether or not a normal transmission completed successfully by checking the SENTDATA flag of the corresponding mailbox. The SENTDATA flag is set to 1 when a normal transmission completes successfully. When the mailbox loses an arbitration conflict or an error occurs during transmission, the message is retained (the CAN module transmits the message again).

(2) One-Shot Transmission Mode

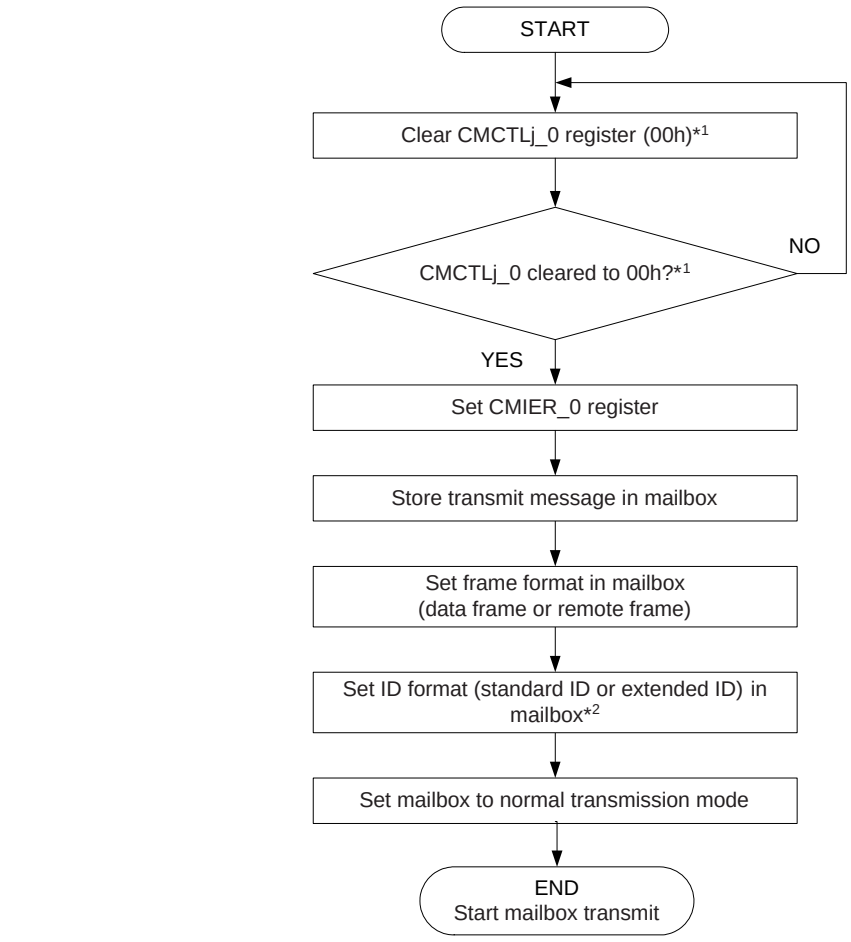
When a mailbox is set to one-shot transmission mode, data frames or remote frames set to that mailbox can be transmitted. When the ONESHOT bit is set to 1, the mailbox attempts to transmit a message one time only (The CAN module does not transmit the message again if a CAN bus error or CAN bus arbitration lost occurs).

It is possible to confirm whether or not a one-shot transmission completed successfully by checking the SENTDATA flag or the transmission abort complete flag (TRMABT) of the corresponding mailbox. The SENTDATA flag is set to 1 when a one-shot transmission completes successfully. The TRMABT flag is set to 1 when the corresponding mailbox loses an arbitration conflict or an error occurs during transmission.

(1) Normal Transmission Request

Figure 2.7 shows the normal transmission request procedure.

This process should be carried out when there is no outstanding transmission or reception request for the corresponding mailbox (CMCTLj_0 register = 00h and abort processing is not underway).

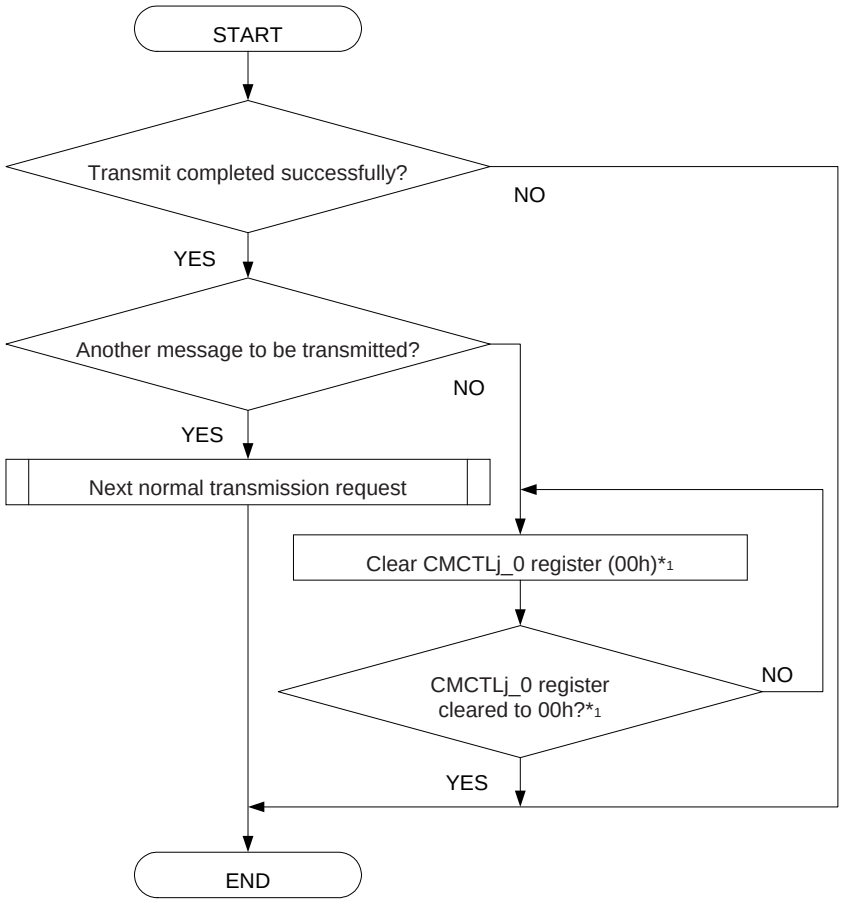


Notes: 1. To make new transmit settings after a transmission completes successfully, it is necessary first to clear the CMCTLj_0 register to 00h. Perform this step without fail.
2. Set the IDE bit as appropriate only when the bits IDFM1 and IDFM0 in the CCTLR_0 register is set to mixed ID mode.
Otherwise, set the IDE bit to 0.

Figure 2.7 Normal Transmission Request Procedure

(2) Normal Transmission Complete Processing

Figure 2.8 shows the necessary procedure following normal transmission complete. For details on performing the succeeding normal transmission request, see section (1) Normal Transmission Request.



Note: 1. To make new transmit settings after a transmission completes successfully, it is necessary first to clear the CMCTLj_0 register to 00h. Perform this step without fail.

Figure 2.8 Normal Transmission Complete Procedure

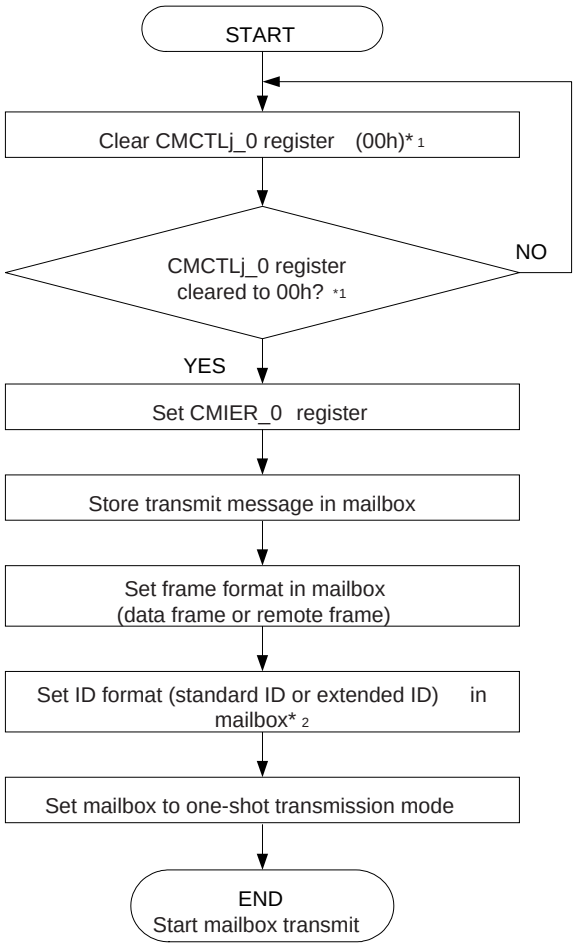
(3) One-Shot Transmission Request

When the ONESHOT bit is set to 1 in transmission mode, the CAN module transmits once only from the corresponding mailbox.

Figure 2.9 shows the one-shot transmission request procedure.

This process should be carried out when there is no outstanding transmission or reception request for the corresponding mailbox (CMCTLj_0 register = 00h and abort processing is not underway).

It is possible to confirm whether or not a one-shot transmission completed successfully by checking the SENTDATA flag or the TRMABT flag of the corresponding mailbox. The SENTDATA flag is set to 1 when a one-shot transmission completes successfully. The TRMABT flag is set to 1 when the corresponding mailbox loses an arbitration conflict or an error occurs during transmission.



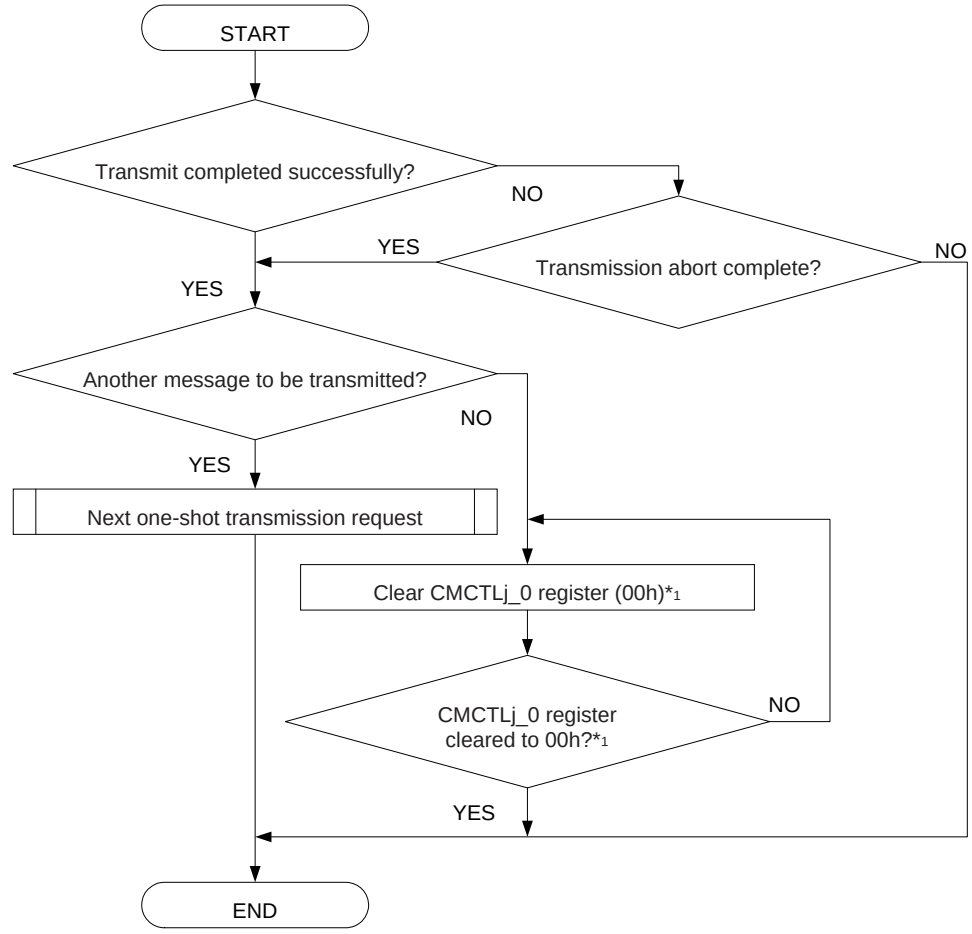
Notes: 1. To make transmit settings after a transmission completes successfully, it is necessary first to clear the CMCTLj_0 register to 00h. Perform this step without fail.

2. Set the IDE bit as appropriate only when bits IDFM1 and IDFM0 in the CCTLR_0 register is set to mixed ID mode. Otherwise, set the IDE bit to 0.

Figure 2.9 One-Shot Transmission Request Procedure

(4) One-Shot Transmission Complete Processing

Figure 2.10 shows the necessary procedure following one-shot transmission complete. This procedure requires that polling be performed. An interrupt cannot be used because no CAN_0 transmission complete interrupt is generated when transmission terminates due to loss of an arbitration conflict or an error. For details on performing the succeeding one-shot transmission request for a mailbox set to one-shot transmission mode, see section (3) One-Shot Transmission Request.



Notes: 1. To make transmit settings after a transmission completes successfully, it is necessary first to clear the CMCTLj_0 register to 00h. Perform this step without fail.

Figure 2.10 One-Shot Transmission Complete Procedure

(5) Transmission-Abort

When two or more nodes start transmission simultaneously, the node(s) whose message(s) have lower CAN ID priority lose the arbitration conflict. (The message is aborted in the case of a one-shot transmission and retained in the case of a normal transmission.) Transmission of a message cannot complete successfully unless the message wins the arbitration conflict or is transmitted when the CAN bus is idle.

The transmission-abort function enables discarding of messages that are being retransmitted in this type of circumstance. It is possible to confirm whether or not transmission-abort completed successfully by checking the SENTDATA flag or the TRMABT flag of the corresponding mailbox.

The SENTDATA flag is set to 1 when a transmission completes successfully.

The TRMABT flag is set to 1 in the following case.

• Following a transmission abort request, when the transmission abort is completed before starting transmission.

• Following a transmission abort request, when the CAN module detects CAN bus arbitration lost or a CAN bus error.

• In one-shot transmission mode (RECREQ bit = 0, TRMREQ bit = 1, and ONESHOT = 1), when the CAN module detects CAN bus arbitration lost or a CAN bus error.

The TRMABT flag is not set to 1 when data transmission is completed. In this case, the SENTDATA flag is set to 1. Using the transmission-abort function is effective in cases such as when only limited time may be allocated for the transmission of a single message or when it is necessary to transmit an urgent, high-priority message.

Figure 2.11 shows application examples using the transmission-abort function, and figure 2.12 shows the transmission-abort procedure.

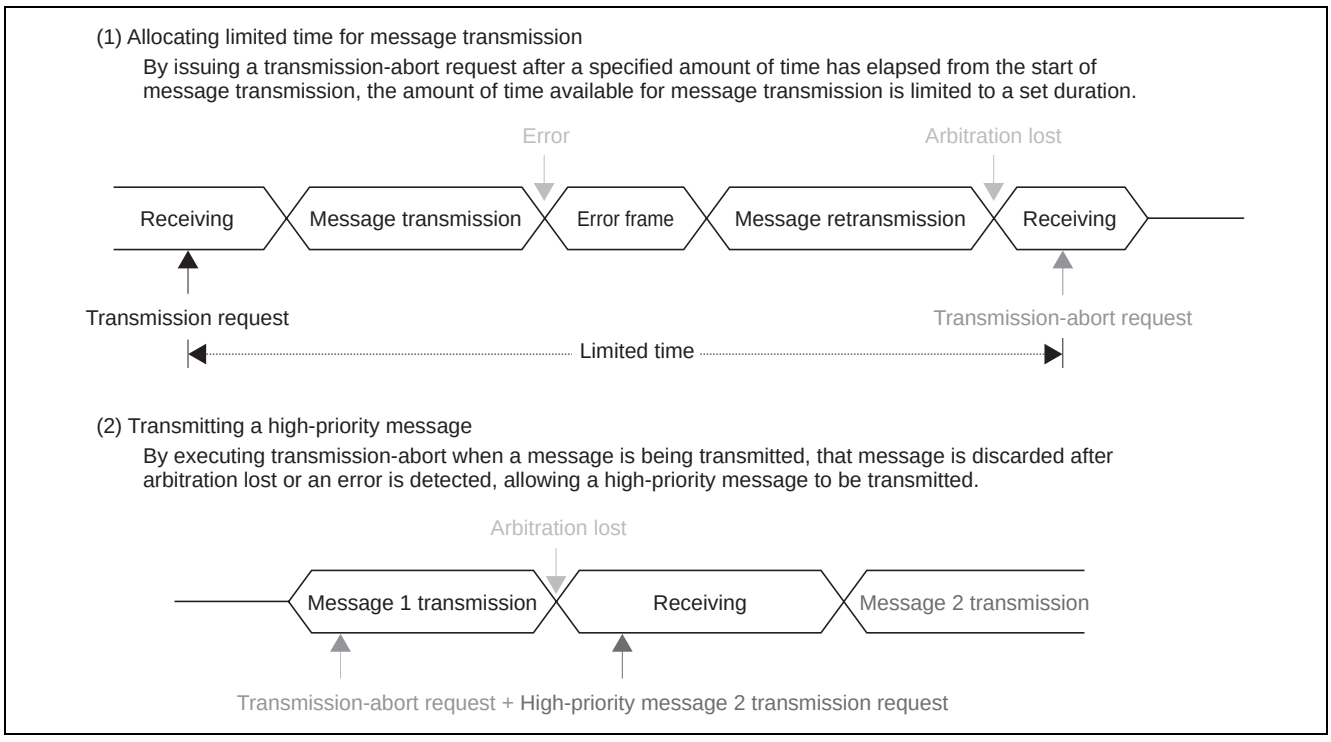
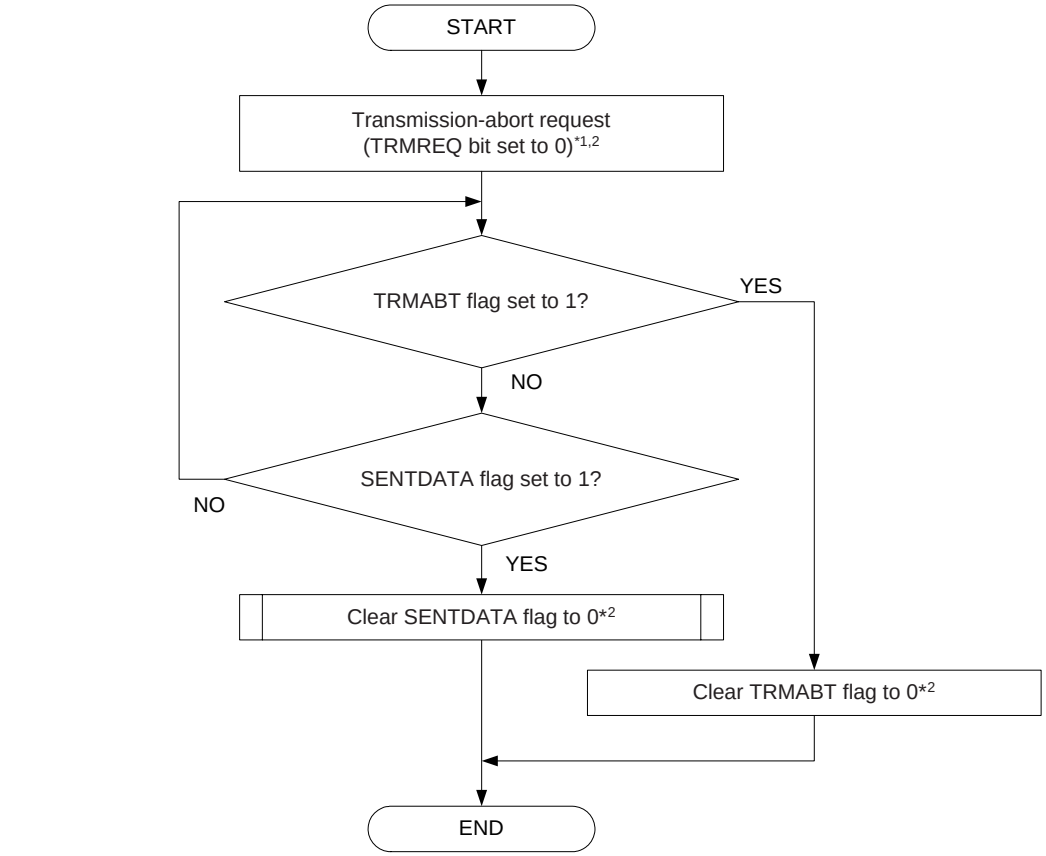


Figure 2.11 Transmission-Abort Function Application Examples



Notes:1. Do not change the one-shot setting during this process. (If the ONESHOT bit is set to 1, do not clear it to 0.)
2. When writing 0 to bits NEWDATA, SENTDATA, MSGLOST, TRMABT, RECREQ, and TREMREQ by a program, use the MOV instruction to ensure that only the specified bit is set to 0 and the other bits are set to 1.

Figure 2.12 Transmission-Abort Procedure

2.3.3 Message Reception

Each CAN module supports 16 mailboxes. The all mailboxes can be used for reception in receive modes.

Received message are always stored in the lowest-numbered mailbox among the mailboxes set to the matching ID.

The receiving mailbox can be selected by using an acceptance filter. See section 9 for details on the acceptance filter function.

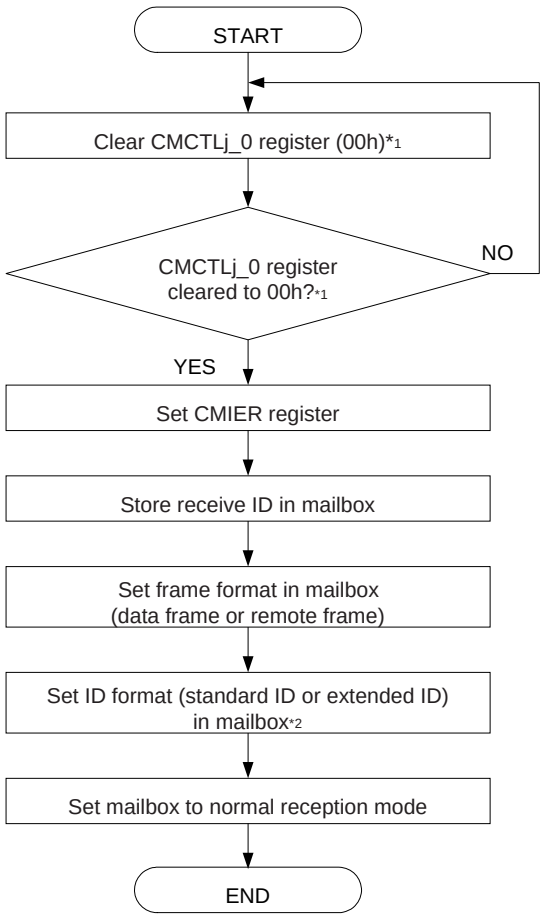
(1) Normal Reception Mode
When a mailbox is set to normal reception mode, data frames or remote frames with the same ID as the mailbox's ID setting (combined with the results of the applicable acceptance filter) can be received. If two or more mailboxes set to normal reception mode have the same ID, received messages are always stored in the lowest-numbered mailbox among the mailboxes set to the matching ID. This means it is possible that an overwrite or overrun may occur. (The overwrite mode or overrun mode can be selected by the MLM bit.)

(2) One-Shot Reception Mode
When a mailbox is set to one-shot reception mode, data frames or remote frames with the same ID as the mailbox's ID setting (combined with the results of the applicable acceptance filter) can be received. When the ONESHOT bit is set to 1, the CAN module receives a message only one time. If two or more mailboxes set to one-shot reception mode have the same ID, received messages are stored starting from the lowest-numbered mailbox and then working upward. In other words, the first received message is stored in the lowest-numbered mailbox. Then, if the first message has not yet been processed, the next received message is stored in the second-lowest-numbered mailbox.

(1) **Normal Reception Request**

Figure 2.13 shows the normal reception request procedure.

This process should be carried out when there is no outstanding transmission or reception request for the corresponding mailbox (CMCTLj_0 register = 00h and abort processing is not underway).



Notes: 1. To make new transmit settings after reception completes successfully, it is necessary first to clear the CMCTLj_0 register to 00h. Perform this step without fail.

2. The IDE bit is enabled when bits IDFM1 and IDFM0 in the CCTLR register are 10b (mixed ID mode). When bits IDFM1 and IDFM0 are 00b (standard ID mode) or 01b (extended ID mode), write 0 to the IDE bit.

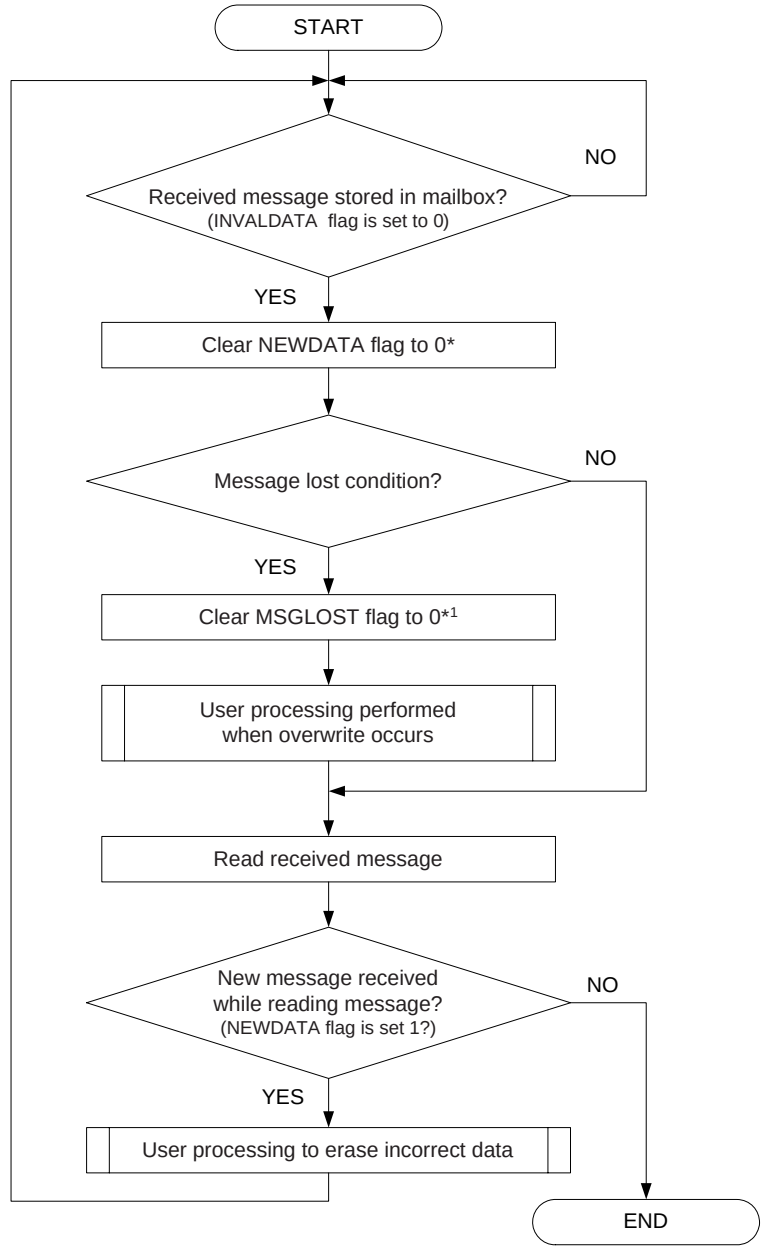
Figure 2.13 Normal Reception Request Procedure

(2) **Reception Complete (Overwrite Mode) with Mailbox Set to Normal Reception Mode**

In this mode, when a mailbox receives a new message before processing of the previously received message by software has completed, the old message in the mailbox is overwritten by the new one. Therefore, it is necessary for the software, after reading the received message from the mailbox, to confirm that the mailbox was not overwritten while the read operation was in progress. When a mailbox is overwritten while the NEWDATA flag is 1, the corresponding MSGLOST flag is set to 1.

When using the receive mailbox search function, perform the above process after checking to determine the numbers of mailboxes with unprocessed received messages. For details on the receive mailbox search function, see section 2.5.

Figure 2.14 shows the received message processing procedure when a mailbox is set to overwrite mode (MLM bit = 0) and normal reception mode.



Note: 1. When writing 0 to bits NEWDATA, SENTDATA, MSGLOST, TRMABT, RECREQ, and TRMREQ by a program, use the MOV instruction to ensure that only the specified bit is set to 0 and the other bits are set to 1.

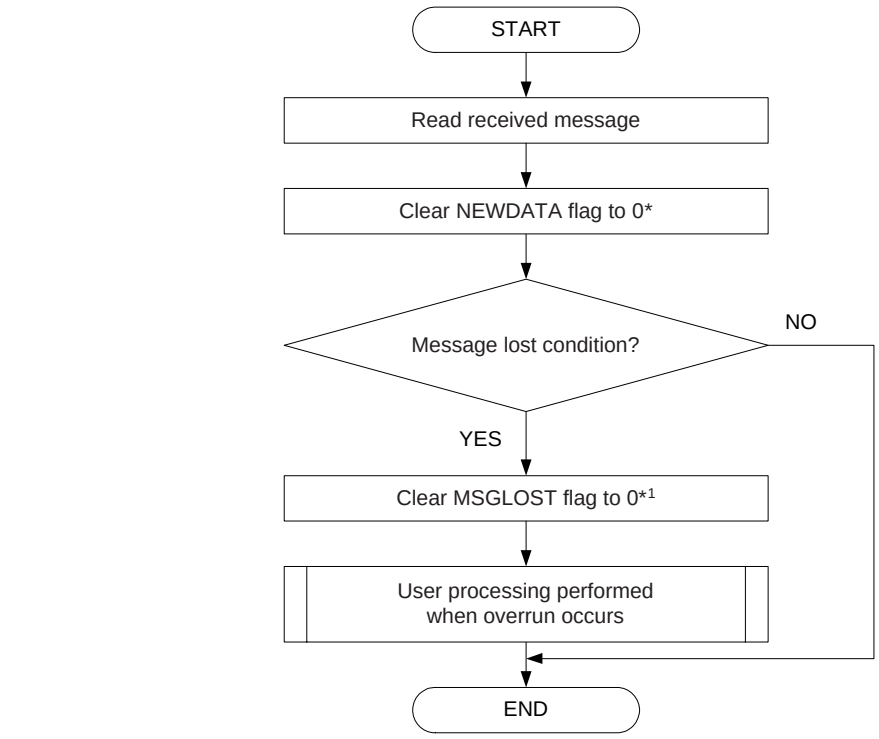
Figure 2.14 Message Reception Processing with Mailbox Set to Normal Reception Mode (Overwrite Mode)

(3) **Reception Complete (Overrun Mode) with Mailbox Set to Normal Reception Mode**

In this mode, when a mailbox receives a new message before processing of the previously received message by software has completed, the new message is discarded (not stored in the mailbox). In this case, the MSGLOST flag corresponding to the mailbox is set to 1 and an overrun interrupt is generated (if the overrun interrupt is enabled).

When using the receive mailbox search function, perform the above process after checking to determine the numbers of mailboxes with unprocessed received messages. For details on the receive mailbox search function, see section 2.5.

Figure 2.15 shows the received message processing procedure when a mailbox is set to overrun mode (MLM bit = 1) and normal reception mode.



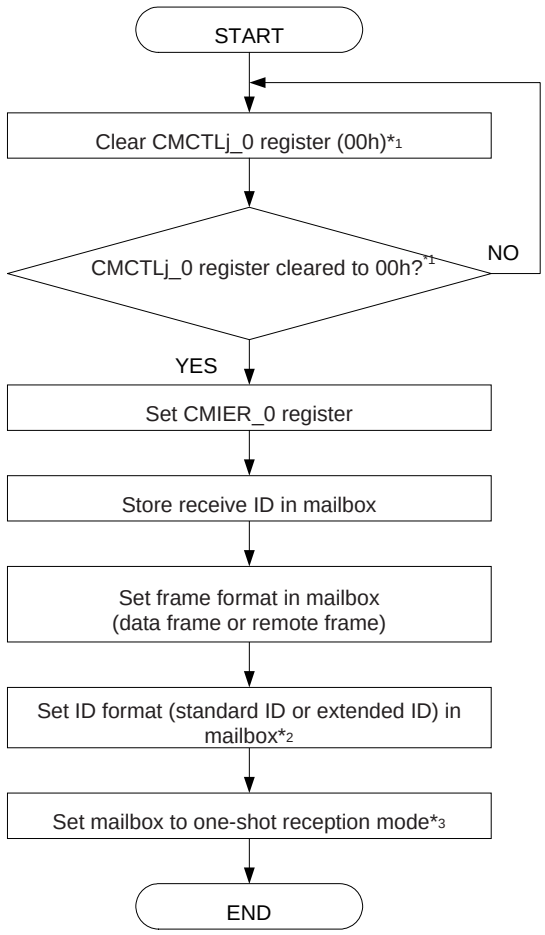
Note: 1. When writing 0 to bits NEWDATA, SENTDATA, MSGLOST, TRMABT, RECREQ, and TRMREQ by a program, use the MOV instruction to ensure that only the specified bit is set to 0 and the other bits are set to 1.

Figure 2.15 Message Reception Processing with Mailbox Set to Normal Reception Mode (Overrun Mode)

(4) **One-Shot Reception Request**

Figure 2.16 shows the one-shot reception request procedure.

This process should be carried out when there is no outstanding transmission or reception request for the corresponding mailbox (CMCTLj_0 register) = 00h and abort processing is not underway).



Notes: 1. To make new transmit settings after reception completes successfully, it is necessary first to clear the CMCTLj_0 register to 00h. Perform this step without fail.
2. The IDE bit is enabled when bits IDFM1 and IDFM0 in the CCTLR register are 10b (mixed ID mode). When bits IDFM1 and IDFM0 are not 10b, write 0 to these bits.
3. To enter one-shot receive mode, write 1 to the ONESHOT bit at the same time as setting the RECREQ bit to 1.

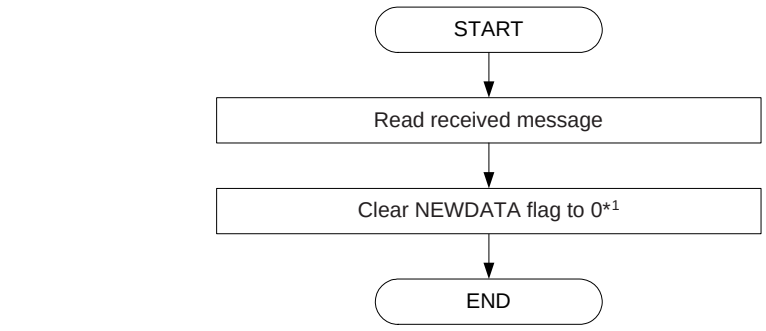
Figure 2.16 One-Shot Reception Request Procedure

(5) **Reception Complete with Mailbox Set to One-Shot Reception Mode**

In one-shot reception mode, the setting of the MLM bit is irrelevant and a message lost condition is never generated. This is because a mailbox that has received a message receives no further messages until the NEWDATA flag is cleared to 0.

When using the receive mailbox search function, perform the above process after checking to determine the numbers of mailboxes with unprocessed received messages. For details on the receive mailbox search function, see section 2.5

Figure 2.17 shows the received message processing procedure when a mailbox is set to one-shot reception mode.



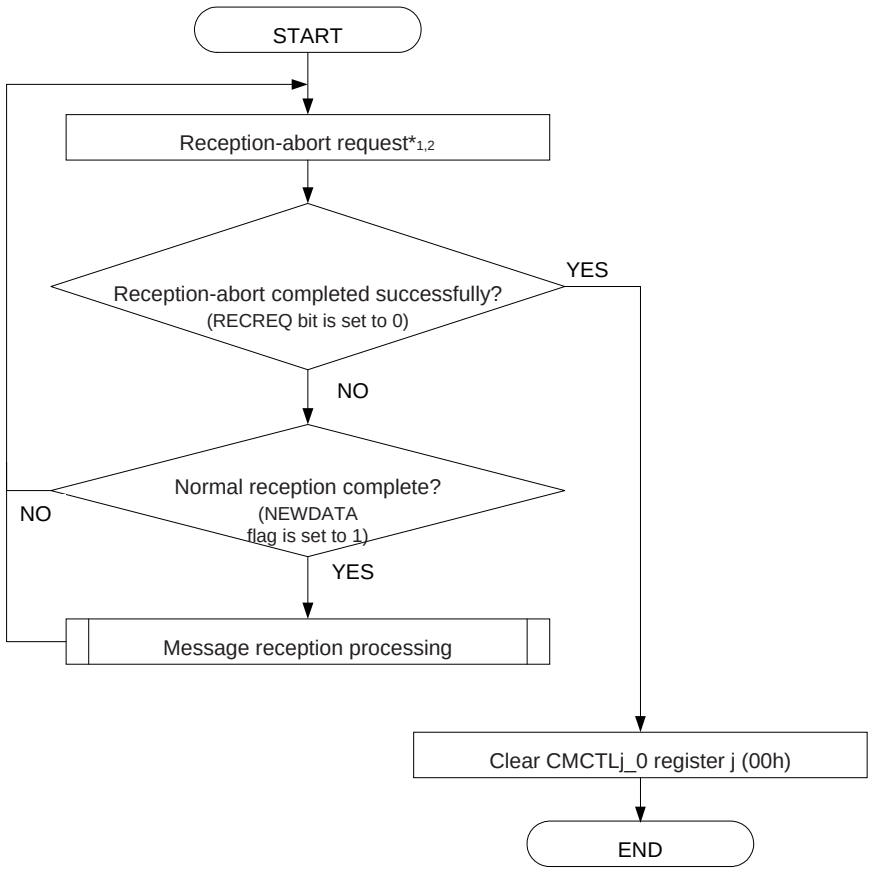
Note: 1. When writing 0 to bits NEWDATA, SENTDATA, MSGLOST, TRMABT, RECREQ, and TRMREQ by a program, use the MOV instruction to ensure that only the specified bit is set to 0 and the other bits are set to 1

Figure 2.17 Message Reception Processing with Mailbox Set to One-Shot Reception Mode

(6) **Reception-Abort**

A reception-abort is executed when the RECREQ bit, NEWDATA flag, and MSGLOST flag in the CMCTLj_0 register are cleared to 0 simultaneously. (Wait for the abort operation to complete before clearing the ONESHOT bit to 0.)

Figure 2.18 shows the reception-abort procedure.



Notes: 1. When writing 0 to bits NEWDATA, SENTDATA, MSGLOST, TRMABT, RECREQ, and TRMREQ by a program, use the MOV instruction to ensure that only the specified bit is set to 0 and the other bits are set to 1

2. To exit one-shot receive mode, write 0 to the ONESHOT bit after writing 0 to the RECREQ bit and confirming it has been set to 0.

Figure 2.18 Reception Abort Procedure

2.4 Mailbox Modes

During CAN configuration, either of the following two modes can be selected by setting the MBM bit in the CCTLR_0 register:

(1) Normal Mailbox Mode
All mailboxes are set to normal transmit or receive.

(2) FIFO Mailbox Mode
Registers CMB0_0 to CMB7_0 are set to normal transmit or receive, registers CMB8_0 to CMB11_0 are set as a transmit FIFO, and registers CMB12_0 to CMB15_0 are set as a receive FIFO.

2.4.1 Normal Mailbox Mode

All mailboxes are set to normal transmit or receive.

Figure 2.19 shows the mailbox configuration in normal mailbox mode.

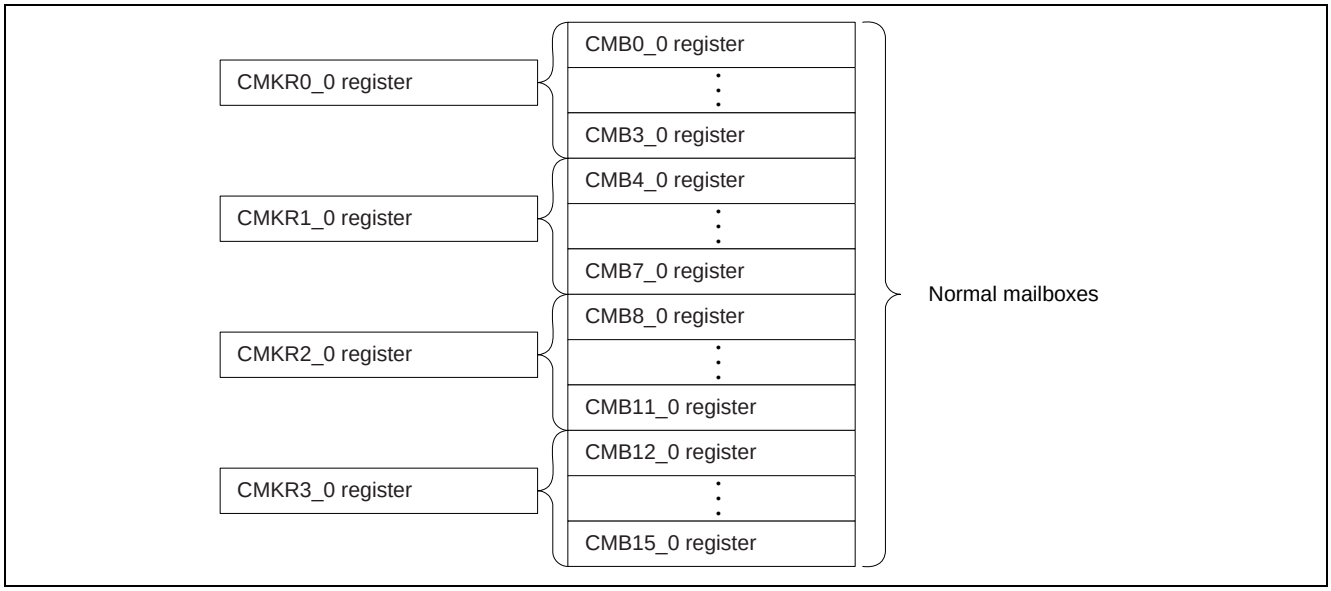


Figure 2.19 Mailbox Configuration in Normal Mailbox Mode

2.4.2 FIFO Mailbox Mode

Figure 2.20 shows the mailbox configuration in FIFO mailbox mode.

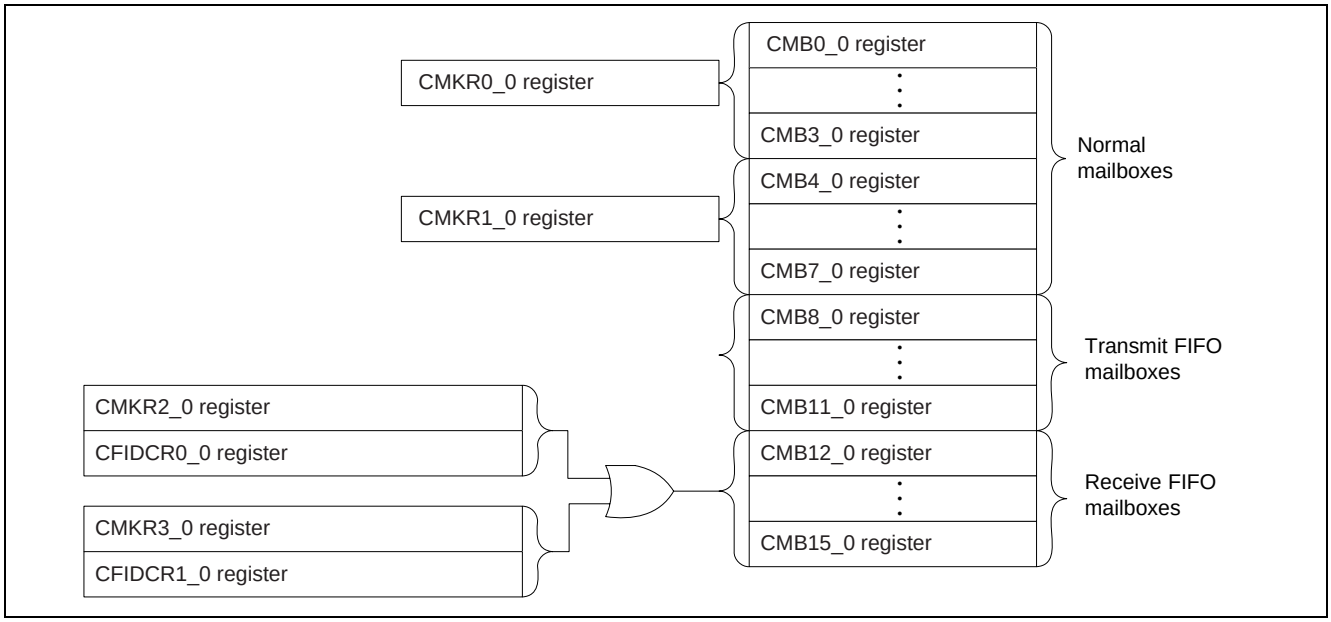


Figure 2.20 Mailbox Configuration in FIFO Mailbox Mode

Registers CMB8_0 to CMB11_0 are set as a transmit FIFO, and registers CMB12_0 to CMB15_0 are set as a receive FIFO. Transmit data is written into CMB8_0 register (CMB8_0 register is a window mailbox for the transmit FIFO). Receive data is read from CMB12_0 register (CMB12_0 register is a window mailbox for the receive FIFO). Updating of the CPU's pointers is accomplished by writing FFh to the CAN transmit/receive FIFO pointer control registers (CTFPCR_0 and CRFPCR_0). Registers CMB0_0 to CMB7_0 are set as normal transmit or receive mailboxes.

The following options can be selected by setting the corresponding bits in the CMIER_0 register.

- Transmit FIFO and receive FIFO interrupt disable/enable
- Transmit FIFO and receive FIFO interrupt generation source

The CMCTLj_0 register corresponding to registers CMB8_0 to CMB15_0 are not used in FIFO mailbox mode. Instead, the CTFCR_0 register and CRFCR_0 register are used.

Setting the TFE bit in the CTFCR_0 register to 1 causes registers CMB8_0 to CMB11_0 to function as a transmit FIFO. When the transmit FIFO enable bit (TFE) is cleared to 0, registers CMB8_0 to CMB11_0 do not function as a transmit FIFO. (The transmit FIFO is halted.)

When the TFE bit is cleared to 0 during transmission from a transmit FIFO mailbox, the transmit FIFO is emptied and any unsent messages it contained are lost after the next transmission complete, error, arbitration lost, or transition to CAN halt mode occurs.

The transmit FIFO mailboxes compose a four-stage FIFO. When there are no messages in the transmit FIFO, the TFEST bit in the CTFCR_0 register is set to 1. When all four of the mailboxes composing the transmit FIFO contain messages (in other words, if there are four unsent messages), the TFFST bit in the CTFCR_0 register is set to 1.

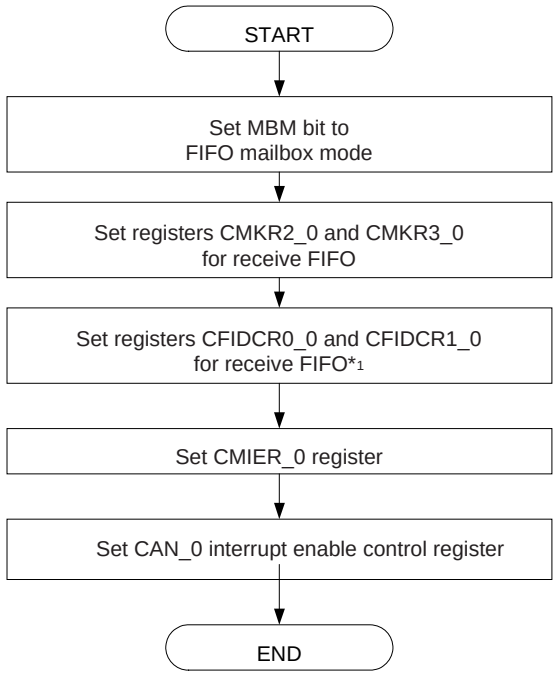
Setting the RFE bit in the CRFCR_0 register to 1 causes registers CMB12_0 to CMB15_0 to function as a receive FIFO. When the RFE bit is cleared to 0, registers CMB12_0 to CMB15_0 do not function as a receive FIFO. (The receive FIFO is halted.)

The receive FIFO mailboxes compose a four-stage FIFO. When there are no messages in the receive FIFO, the RFEST bit in the CRFCR_0 register is set to 1. When three of the four mailboxes composing the transmit FIFO contain messages, the RFWST bit in the CRFCR_0 register is set to 1. When all four of the mailboxes composing the receive FIFO contain messages, the RFFST bit in the CRFCR_0 register is set to 1. Furthermore, when a new message is received while the receive FIFO is full, the RFMLF flag in the CRFCR_0 register is set to 1. In this case, the new message is discarded (not stored in the mailbox) if the MLM bit in the CCTLR_0 register is set to overrun mode. If the MLM bit is set to overwrite mode, the first message received among those stored in the receive FIFO is overwritten by the new message. (The receive pointer are incremented automatically.)

FIFO mailbox mode uses two mask registers (CMKR2_0 and CMKR3_0) and two FIFO received ID compare registers (CFIDCR0_0 and CFIDCR1_0). If the ID of a message matches the combined results of one of the two mask registers and one of the two FIFO received ID compare registers, it is stored in the receive FIFO.

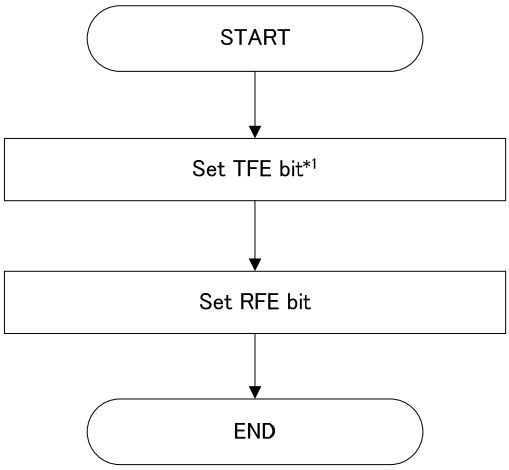
(1) FIFO Mailbox Mode Settings

Settings for FIFO mailbox mode are made by carrying out procedures in CAN reset mode and CAN operation mode. Figure 2.21 shows the procedure in CAN reset mode, and figure 2.22 shows the procedure in CAN operation mode.



Note 1. The IDE bit is enabled when bits IDFM1 and IDFM0 in the CCTLR register are 10b (mixed ID mode). When bits IDFM1 and IDFM0 are 00b (standard ID mode) or 01b (extended ID mode), write 0 to the IDE bit.

Figure 2.21 FIFO Mailbox Mode Setting Procedure in CAN Reset Mode

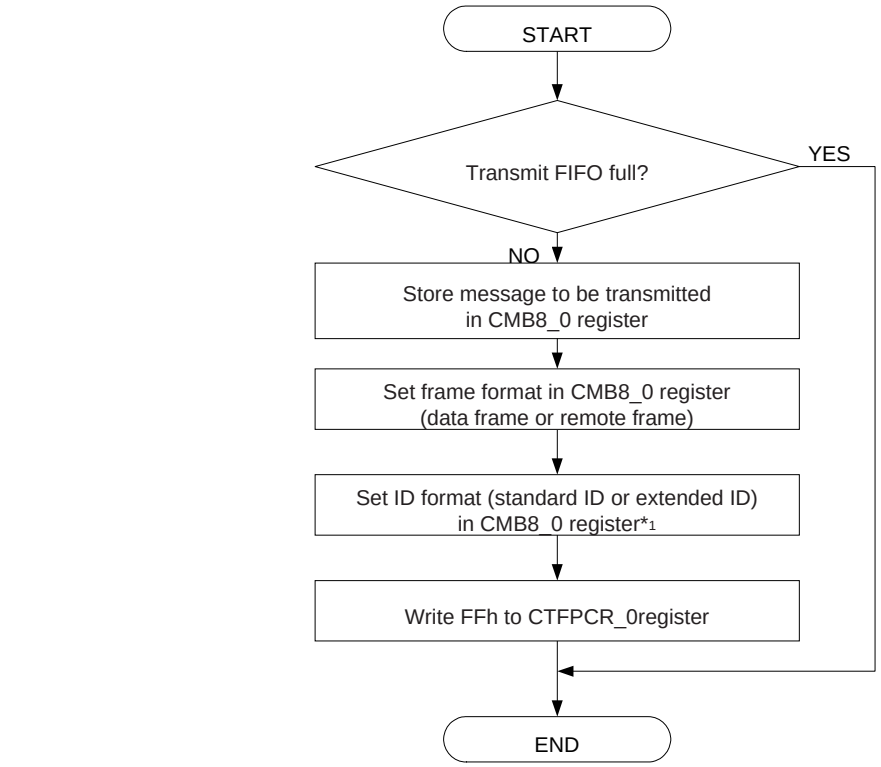


Note: 1. When carrying out this process after disabling the transmit FIFO by clearing the TFE bit to 0, confirm that the TFEST bit is set to 1 before resetting the TFE bit to 1.

Figure 2.22 FIFO Mailbox Mode Setting Procedure in CAN Operation Mode

(2) FIFO Transmission

Figure 2.23 shows the processing procedure for transmitting a message from the FIFO.



Note: 1. The IDE bit is enabled when bits IDFM1 and IDFM0 in the CCTLR register are 10b (mixed ID mode). When bits IDFM1 and IDFM0 are 00b (standard ID mode) or 01b (extended ID mode), write 0 to the IDE bit.

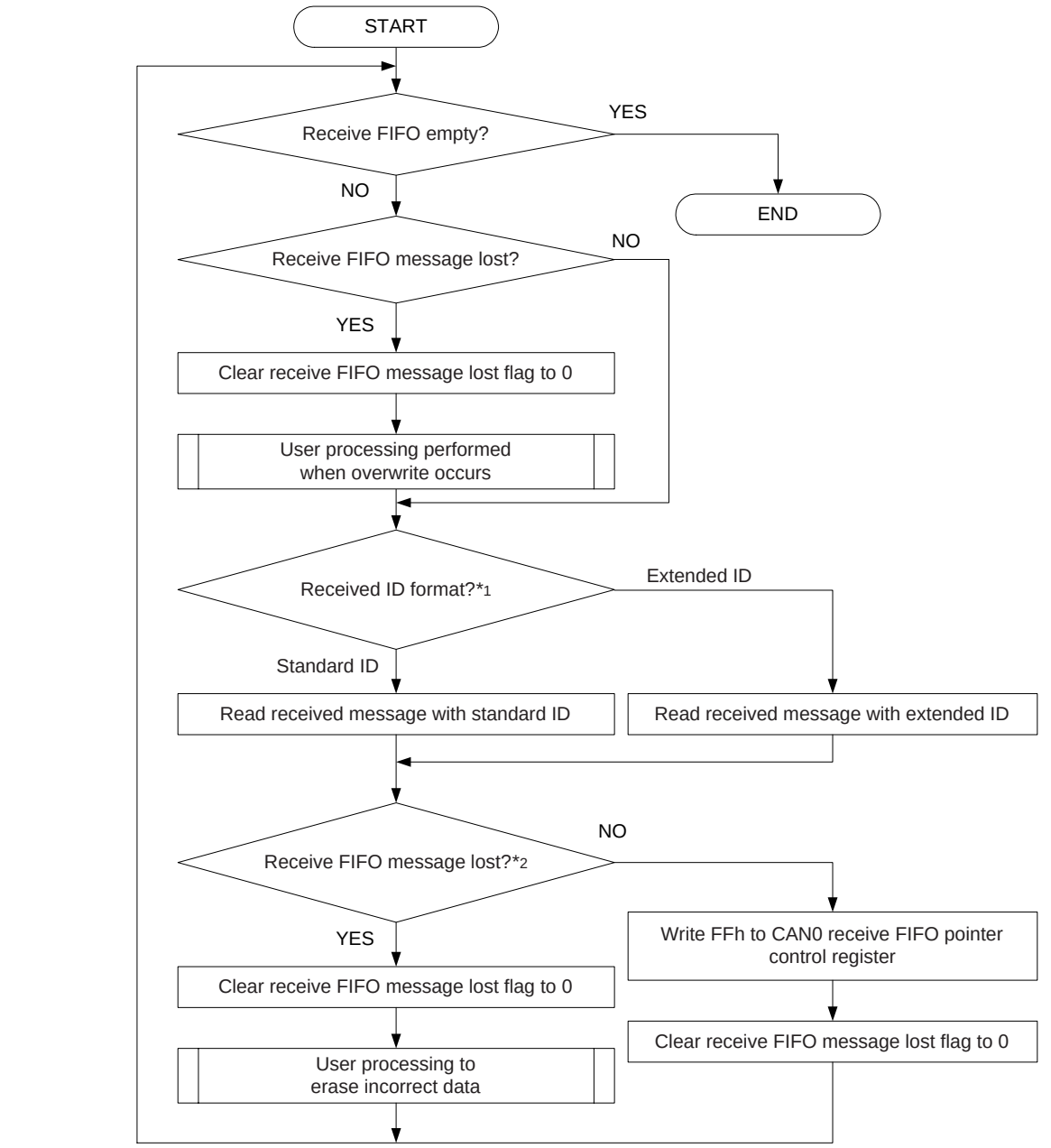
Figure 2.23 Transmission Processing Procedure in FIFO Mailbox Mode

The procedure for aborting transmission from a FIFO is similar to that for aborting transmission from a normal mailbox described in section2.3.2. To abort FIFO transmission, it is necessary to clear the TFE bit to 0 instead of the TRMREQ bit. In addition, the TFEST bit is set to 1 instead of the TRMABT flag.

(3) FIFO Reception (Overwrite Mode)

In overwrite mode it is necessary to consider the possibility of a message being overwritten while it is being read. If a message is overwritten while it is being read, it cannot be used as a normal message.

Figure 2.24 shows the processing procedure for receiving a message in the FIFO using overwrite mode.



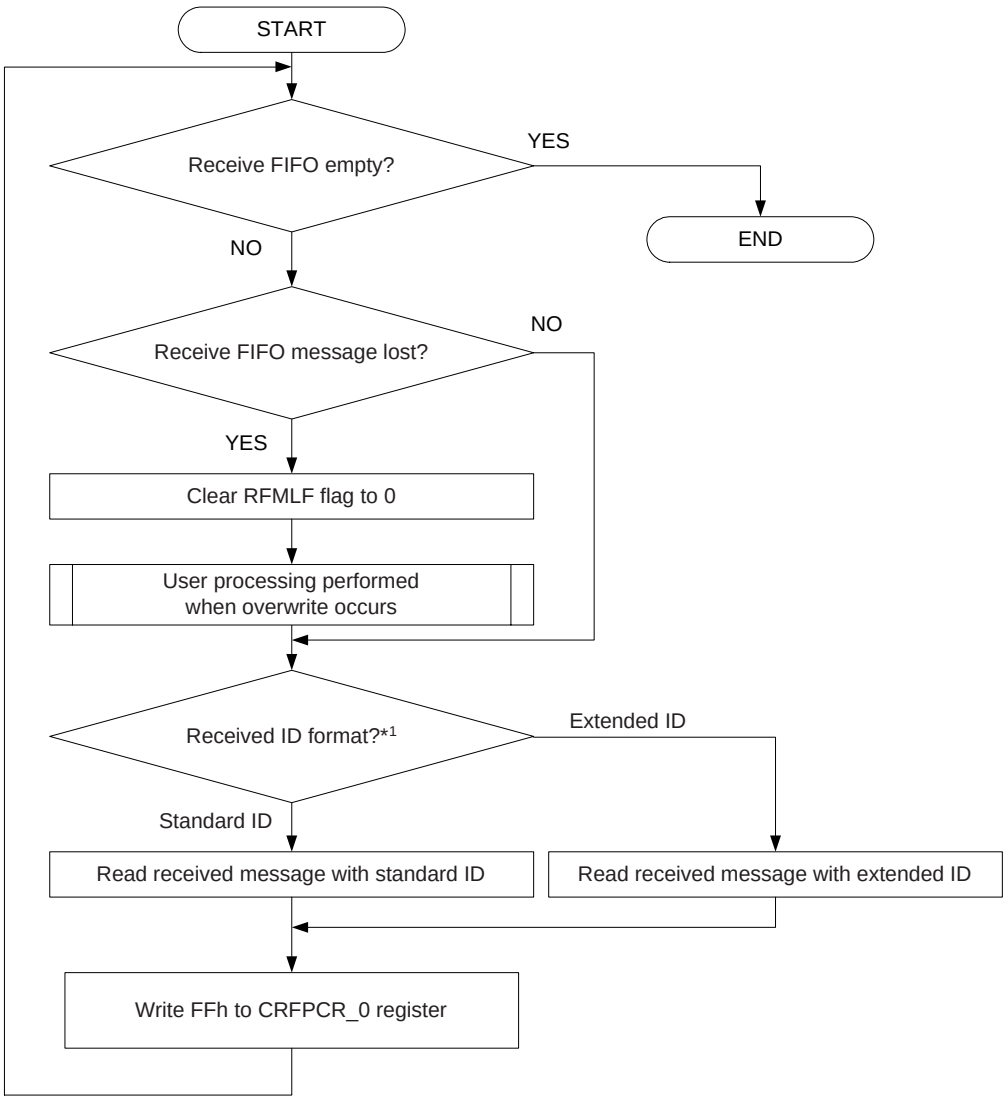
Notes: 1. This determination is necessary only when bits IDFM1 and IDFM0 in the CCTLR_0 register is set to mixed ID mode.
2. This determines whether or not the message was overwritten while the data was being read. If a message lost condition occurs, the message cannot be used as a normal message.

Figure 2.24 Reception Processing Procedure (Overwrite Mode) in FIFO Mailbox Mode

(4) FIFO Reception (Overrun Mode)

In overrun mode it is not necessary to consider the possibility of a message being overwritten by the CAN while it is being read by the CPU. The read value has never been overwritten, even if an overrun occurs.

Figure 2.25 shows the processing procedure for receiving a message in the FIFO using overrun mode.



Notes: 1. This determination is necessary only when bits IDFM1 and IDFM0 in the CCTLR_0 register is set to mixed ID mode.

Figure 2.25 Reception Processing Procedure (Overrun Mode) in FIFO Mailbox Mode

2.5 Mailbox Search Function

Usually it is necessary to search for the mailbox number after each transmission or reception operation completes when two or more transmit or receive mailboxes has been set up. This means that the load on the software grows as the number of mailboxes increases.

The mailbox search function can be used to reduce the load on the software.

The mailbox search mode provides an easy way to search for the mailbox number that triggered a reception complete or transmission complete message.

The following four modes are used for mailbox searches:

- Receive mailbox search mode
- Transmit mailbox search mode
- Message lost search mode
- Channel search mode

These modes can be used in both normal mailbox mode and FIFO mailbox mode.

For polling processing, checking the CSTR_0 register is recommended before using the mailbox search function.

Figure 2.26 shows an example.

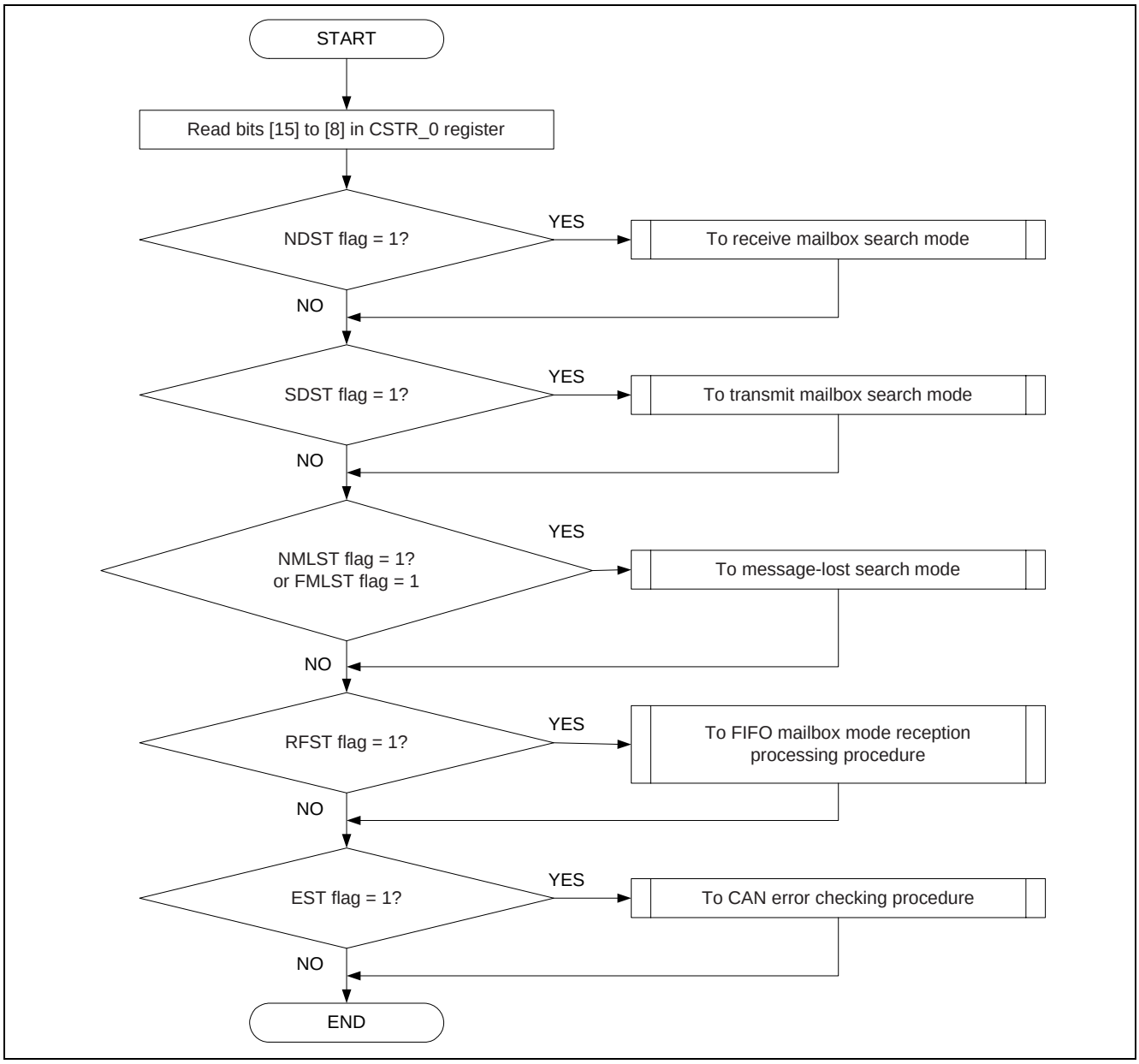


Figure 2.26 CSTR_0 Register Checking Example (Polling Operation)

2.5.1 Using the Mailbox Search Function

Table 2.5 lists the setting values of bits MBSM0 and MBSM1.

Table 2.5 Setting Values of Mailbox Search Mode Bits

CMSMR_0 register		Search Mode
b1	b0	
0	0	Receive mailbox search mode (default) (Search for NEWDATA flag)
0	1	Transmit mailbox search mode (Search SENTDATA flag)
1	0	Message lost search mode (Search MSGLOST flag)
1	1	Channel search mode

(1) Receive Mailbox Search Mode

This mode displays the lowest mailbox number in the receive-end state.

To use this mode, set the MBSM bit to 00b. The number of the mailbox in the reception complete state can then be read from the CMSSR_0 register. When two or more mailboxes are in the reception complete state (that is, when two or more mailboxes have their NEWDATA flags set to 1), the lowest mailbox number is read.

The NEWDATA flag is cleared to 0 by software as part of the reception complete processing. After this occurs, the mailbox search function can be used again to read the next mailbox number if there are other mailboxes in the reception complete state. When no other mailboxes are in the reception complete state, the SEST bit is set to 1.

In FIFO mailbox mode, mailbox number 12 is read for the receive FIFO mailbox when the RFEST bit is cleared to 0 (message present in receive FIFO mailbox).

Figure 2.27 shows the procedure for using receive mailbox search.

For details on reception complete processing, see section 2.3.3 or 2.4.2.

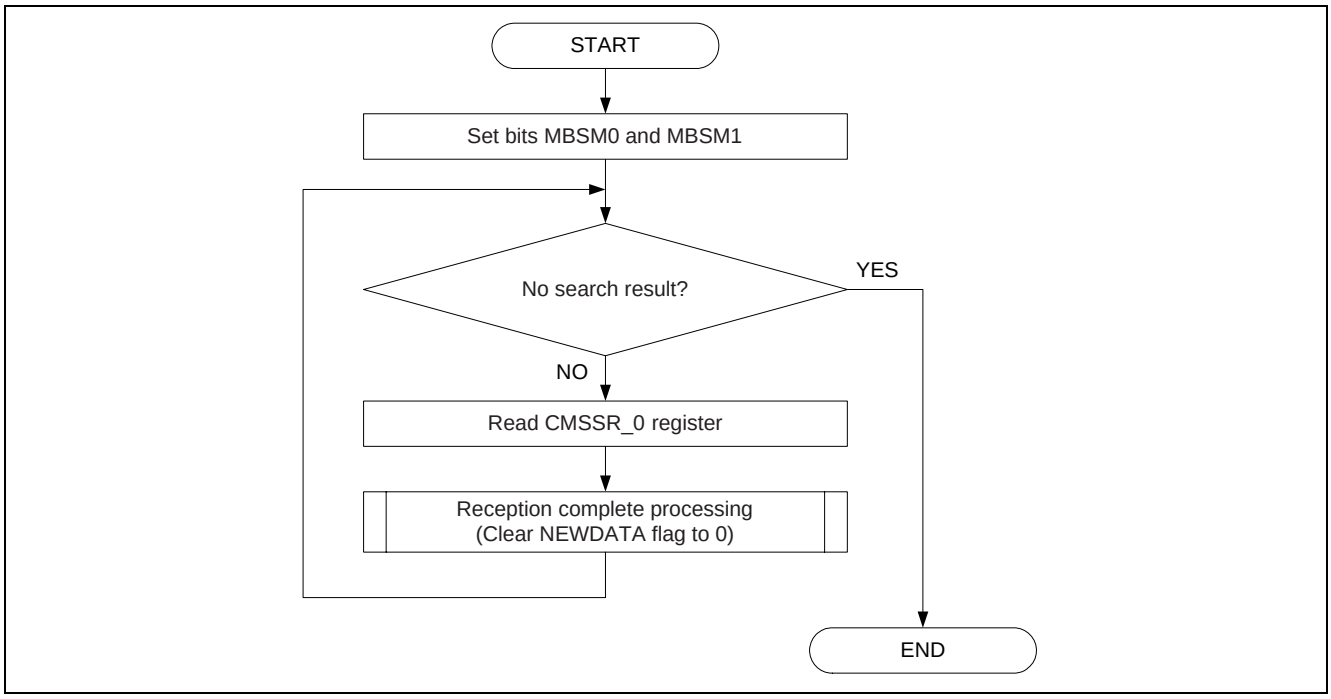


Figure 2.27 Receive Mailbox Search Procedure

(2) Transmit Mailbox Search Mode

This mode displays the number of the mailbox that successfully completed a transmission.

To use this mode, set the mailbox search mode select bits (MBSM) to 01b. The search result can be read from the CMSSR_0 register. When two or more mailboxes have successfully entered the transmission complete state (that is, when two or more mailboxes have their SENTDATA flags set to 1), the lowest mailbox number is read.

The SENTDATA flag is cleared to 0 by software as part of the transmission complete processing. After this occurs, the mailbox search function can be used again to read the next mailbox number if there are other mailboxes in the transmission complete state. When no other mailboxes are in the transmission complete state, the SEST bit is set to 1.

In FIFO mailbox mode, transmit FIFO mailboxes are not covered by the mailbox search function.

Figure 2.28 shows the procedure for using transmit mailbox search.

For details on continuing with a normal transmission request in normal mailbox mode, see section 4.2.1.

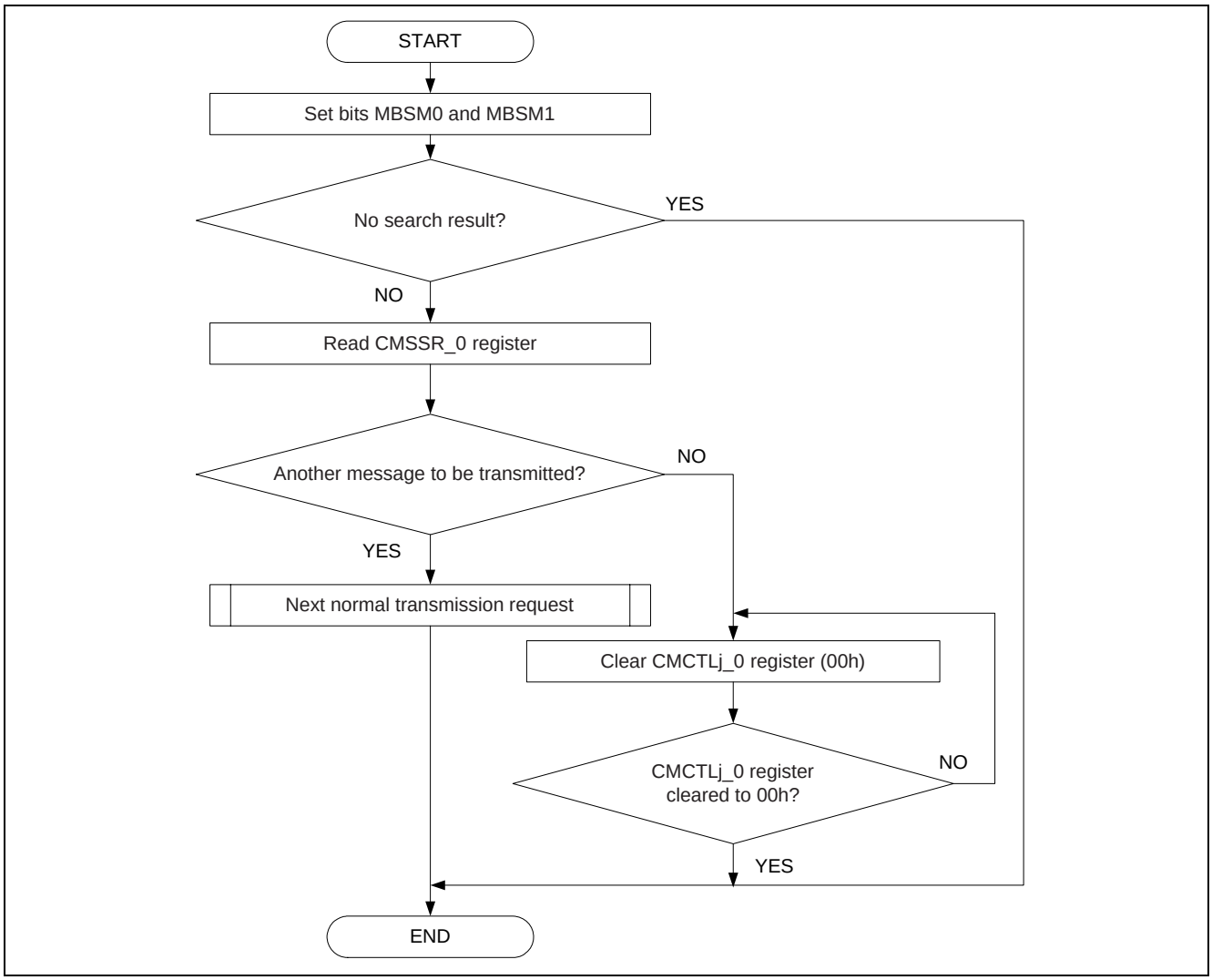


Figure 2.28 Transmit Mailbox Search Procedure

(3) Message Lost Search Mode

This mode displays the number of the mailbox that triggered a message lost condition.

To use this mode, set the MBSM bit to 10b. The search result can be read from the CMSSR_0 register. When two or more mailboxes are in the message lost state (that is, when two or more mailboxes have their MSGLOST flags or RFMLF flag set to 1), the lowest mailbox number is read.

The MSGLOST flag is cleared to 0 by software as part of the message lost processing. After this occurs, the mailbox search function can be used again to read the next mailbox number if there are other mailboxes in the message lost state. When no other mailboxes are in the message lost state, the SEST bit is set to 1.

In FIFO mailbox mode, mailbox number 12 is read when the RFMLF flag is set to 1 (message lost condition occurred in receive FIFO mailbox).

Figure 2.29 shows the procedure for using message lost search.

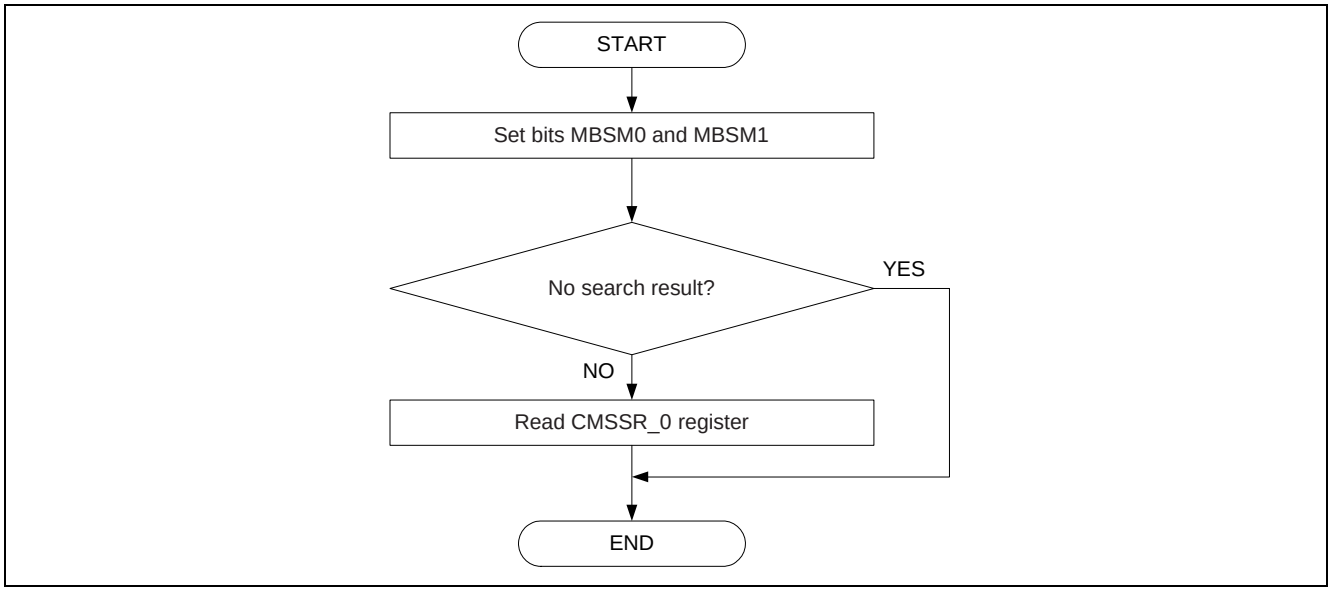


Figure 2.29 Message lost Search Procedure

(4) Channel Search Mode

The purpose and procedure for using the channel search mode differ from those of the other three modes. This mode does not search for a mailbox number. To use this mode, set bits MBSM1 and MBSM0 to 11b.

Set the channel search value (Table value) in the CCSSR_0 register. The encoded value can then be read from the CMSSR_0 register. When there are two or more channels, the channel numbers are read in order, starting from the lowest.

When the CMSSR_0 register is read, the search result is updated automatically. The next channel number can then be read if there are other channels. When there are no other channels, the SEST bit is set to 1.

Figures 2.30 and 2.31 show the procedure for using channel search.

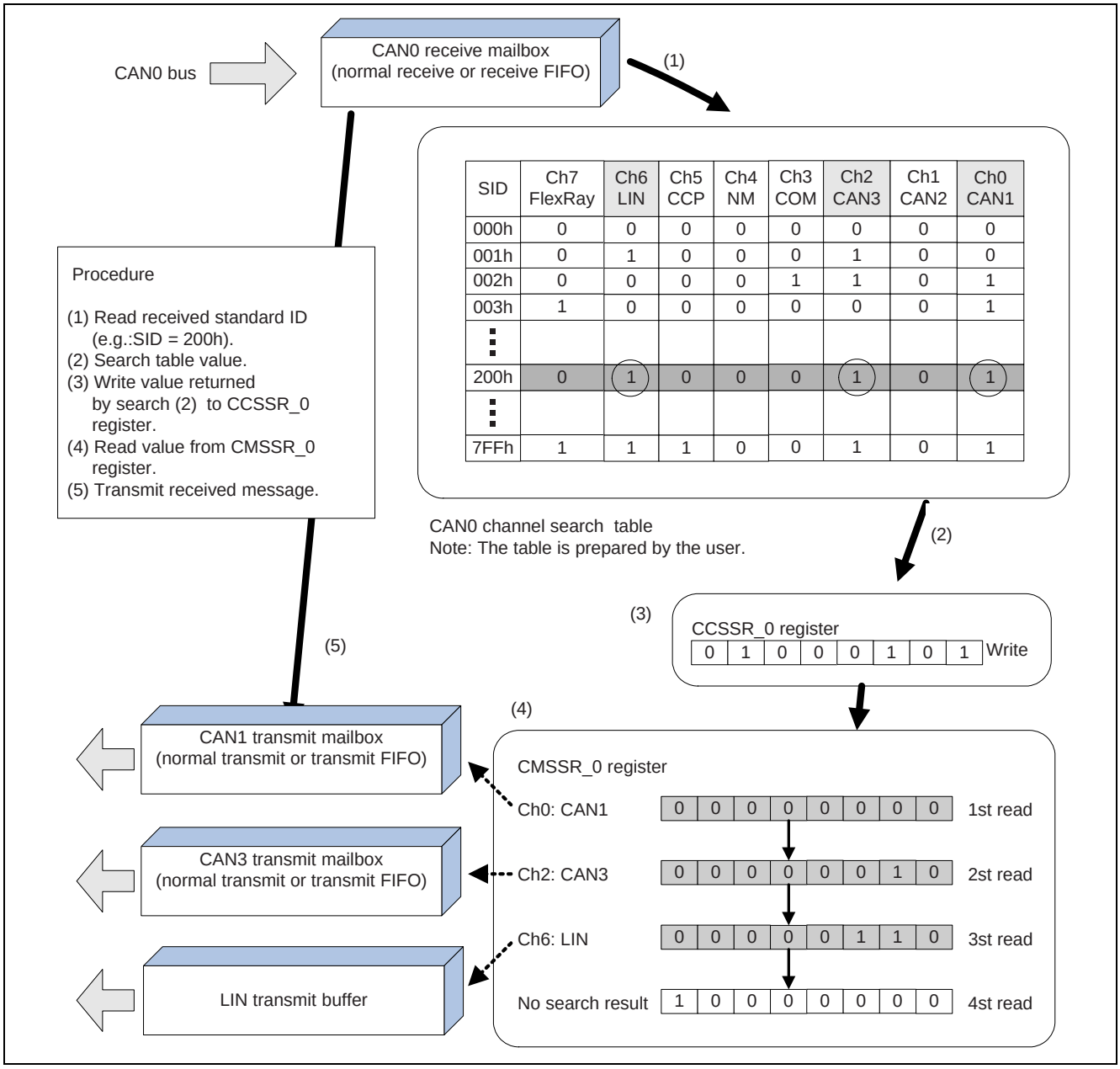


Figure 2.30 Outline of Channel Search Mode

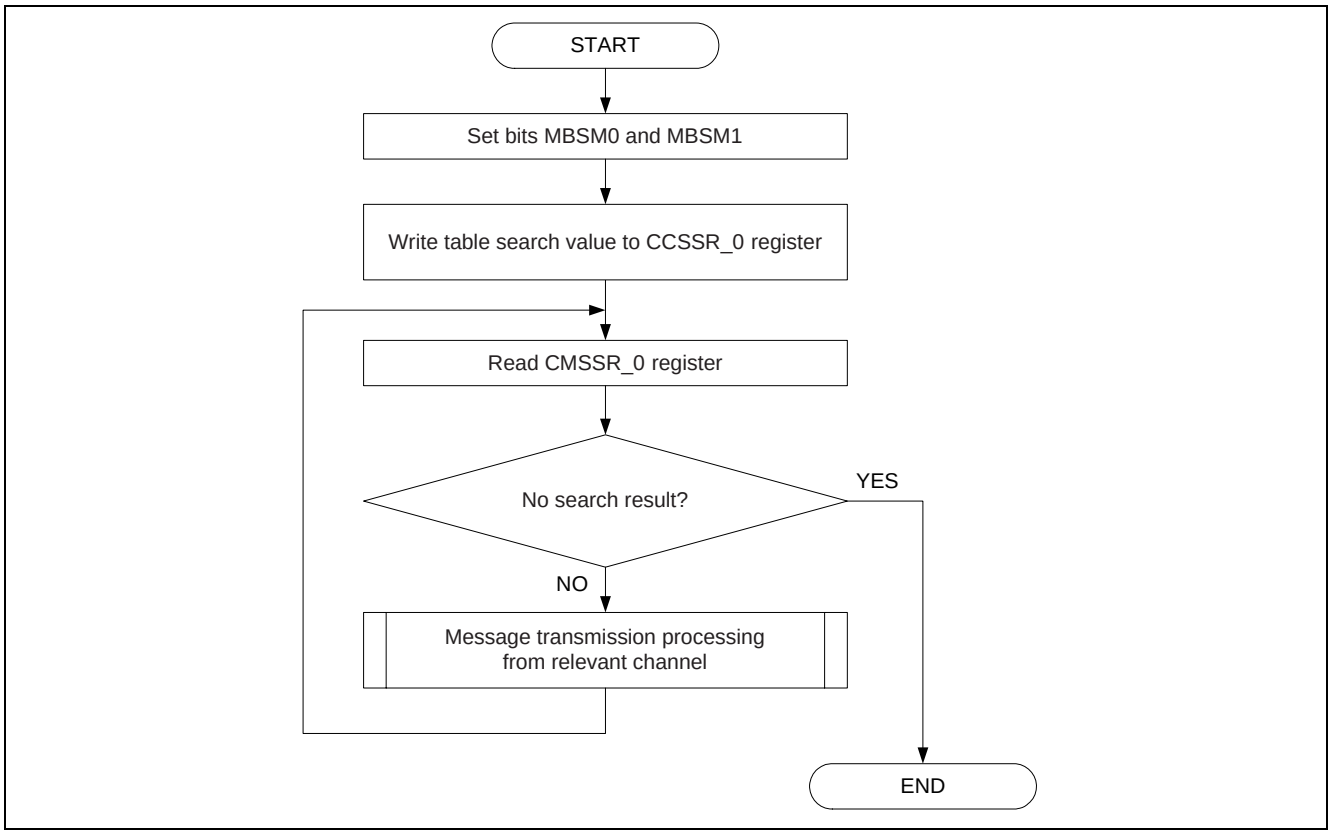


Figure 2.31 Channel Search Procedure

2.6 CAN Errors

When an error is detected due to a communication frame irregularity during transmission or reception by a mailbox, the transmit error counter or receive error counter value is incremented, depending on whether the error occurred during transmission or reception. When the transmit error counter or receive error counter value reaches 96 or greater, an error warning is detected and the EWIF flag is set to 1. When the transmit error counter or receive error counter value reaches 128 or greater, the CAN status changes from error-active to error-passive, the error-passive state is detected, and the EPIF flag is set to 1. When the transmit error counter value reaches 256 or greater, the CAN module enters the bus-off state, the bus-off state is detected, and the BOEIF flag is set to 1.

To use CAN error interrupts, set to 1 the bits in the CEIER_0 register corresponding to the error interrupts to be used. Whether or not a particular interrupt has occurred can then be confirmed by reading the CEIFR_0 register. Only make settings in the CEIER_0 register when in CAN reset mode. To use CAN error interrupts, it is necessary to make settings in the CANERIC_0 register beforehand.

2.6.1 CAN Error Checking

(1) Using Polling of the error state

The error state of the CAN module can be checked by polling the EPST flag and BOST flag in the CSTR_0 register.

Figure 2.32 shows the CAN error state checking procedure using polling.

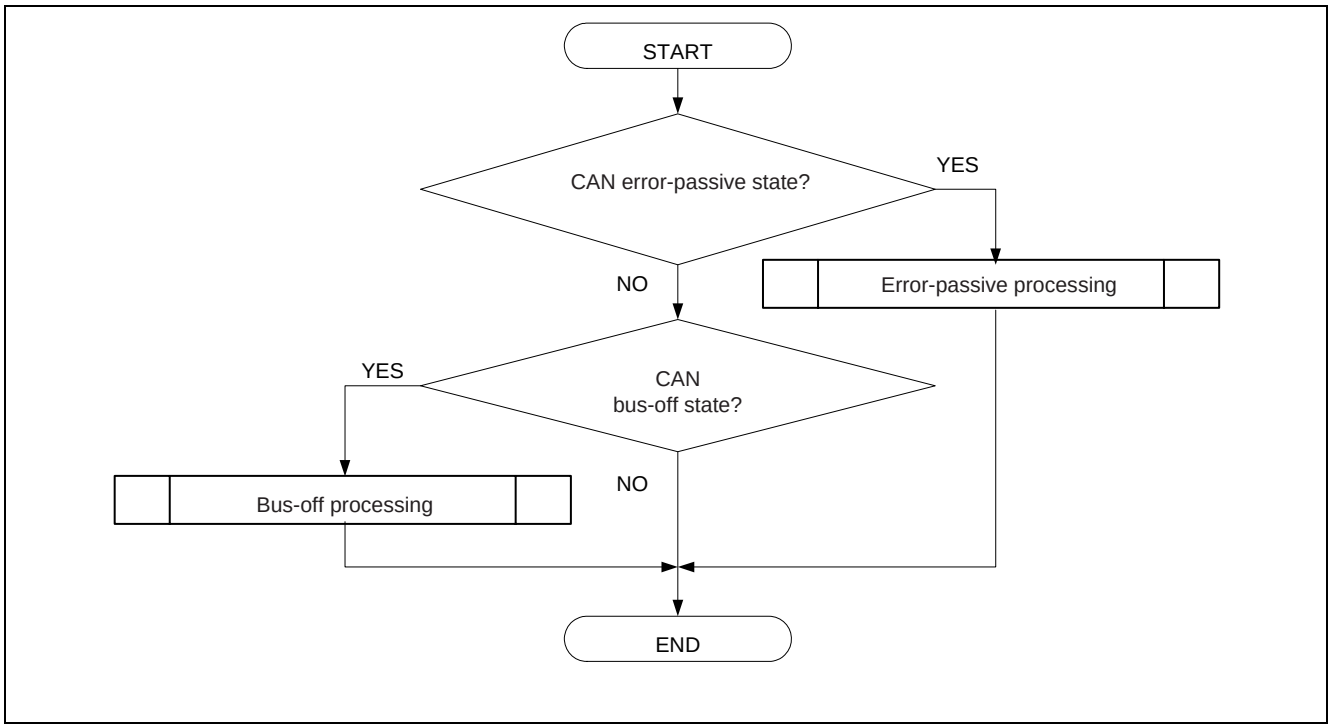


Figure 2.32 CAN Error State Checking Procedure (Using Polling)

(2) Using CAN Error Interrupts

CAN error interrupts can be used to check the error state by setting the interrupt priority level more than 000b in the CANERIC_0 register and enabling error interrupt in CANIE_0 register.

CAN error interrupt and CAN wake-up interrupt are assigned to same interrupt vector table. If the bit ERR is 1 when interruption is generated, it can be determined that it is CAN error interruption. The CAN error state is determined by reading the CEIFR_0 register as part of the CAN error interrupt routine.

Figure 2.33 shows the CAN error state checking procedure using CEIFR_0 register. For details on returning from the bus-off state, see section 2.7.

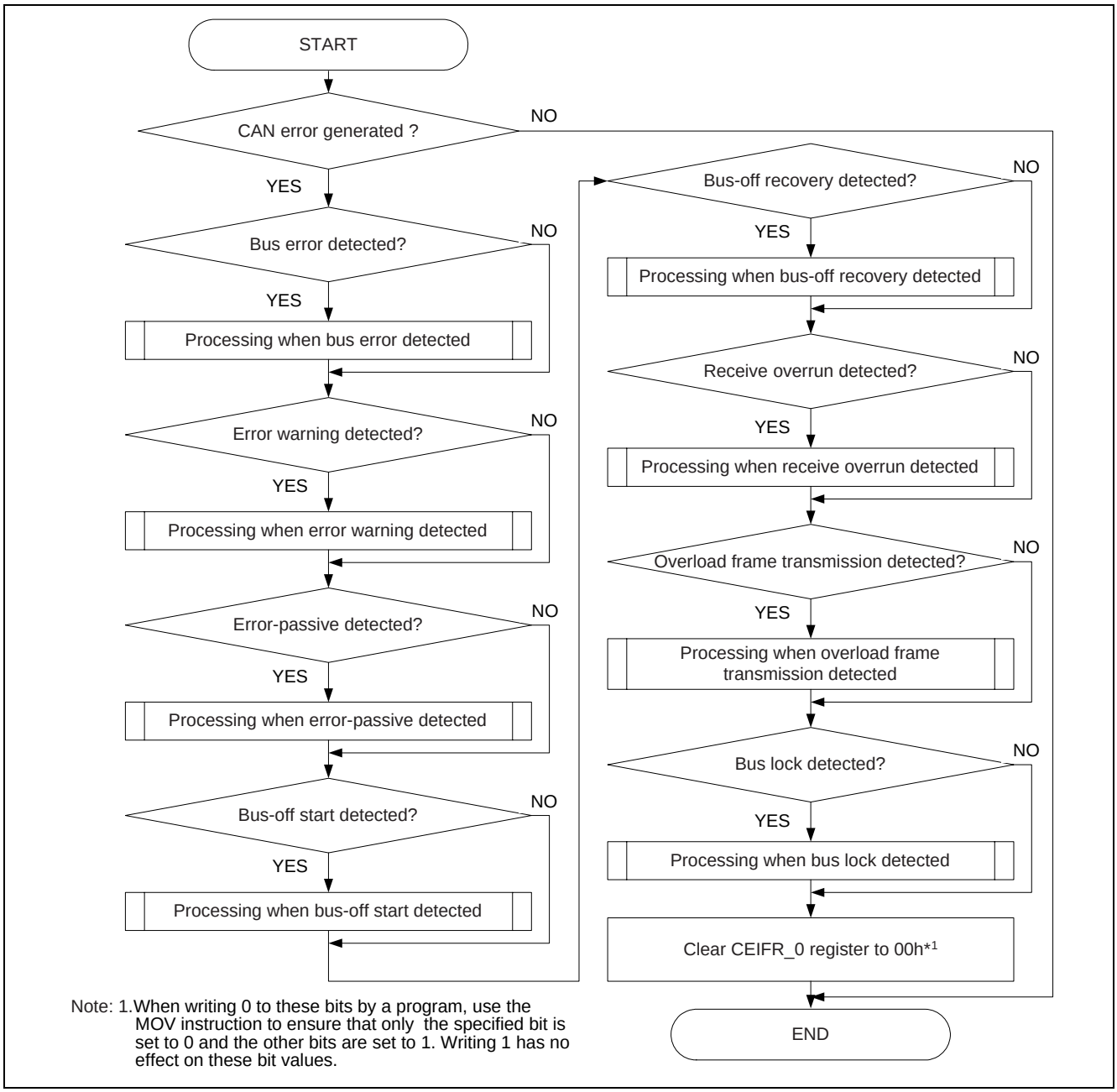


Figure 2.33 CAN Error State Checking Procedure (Using CAN Error Interrupts)

2.7 Bus-Off Recovery Modes

When CAN communication errors occur repeatedly, the CAN module transitions to the bus-off state, according to the fault restriction rules in the CAN specification.

The CAN module has five modes for returning from the bus-off state. Table 2.6 lists these modes and their associated bits and settings. Figure 2.34 shows transition to and from the bus-off state.

Table 2.6 Bus-Off-Return Modes

	Name	Description	Bit(s) Used	Bit Setting
(1)	Normal mode	After returning from the bus-off state, the CAN module immediately transitions to the error-active state so that CAN communication can begin without delay.*[1,2]	BOM*[5]	00b*[6]
(2)	Forcible return from bus-off	The CAN module immediately transitions to the error-active state, and CAN communication is possible after 11 successive recessive bits are detected.*[3]	BOM*[5]	00b*[6]
			RBOC*[5]	1*[7]
(3)	Entry to CAN halt mode automatically at bus-off start	The CAN module switches immediately to CAN halt mode when the bus-off state is reached.*[3]	BOM*[5]	01b*[6]
(4)	Entry to CAN halt mode automatically at bus-off end	The CAN module switches to CAN halt mode after returning from the bus-off state.*[1,2]	BOM*[5]	10b*[6]
(5)	Entry to CAN halt mode by a program request	When in the bus-off state, the CAN module switches immediately to CAN halt mode when the CAN operating mode select bits are set to 10b (CAN halt mode).*[3,4]	BOM*[5]	11b*[6]
			CANM*[5]	10b*[8]

Notes:
1. The CAN module returns from the bus-off state when 11 successive recessive bits are detected 128 times.
2. In this case the BORIF flag in the CEIFR_0 register is set to 1 (bus-off recovery detected).
3. In this case the BORIF flag is not set to 1.
4. When the bits CANM1 and CANM0 are not set to 10b (CAN halt mode) while in the bus-off state, operation is identical to (1).
5. Bits in CCTRLR_0 register.
6. Only make this setting when in CAN reset mode.
7. Only make this setting when in the bus-off state. After the forcible return from RBOC bit is set to 1 by a program, it is automatically cleared to 0.
8. After changing bits CANM1 and CANM0, make sure to check the CSTR_0 register.

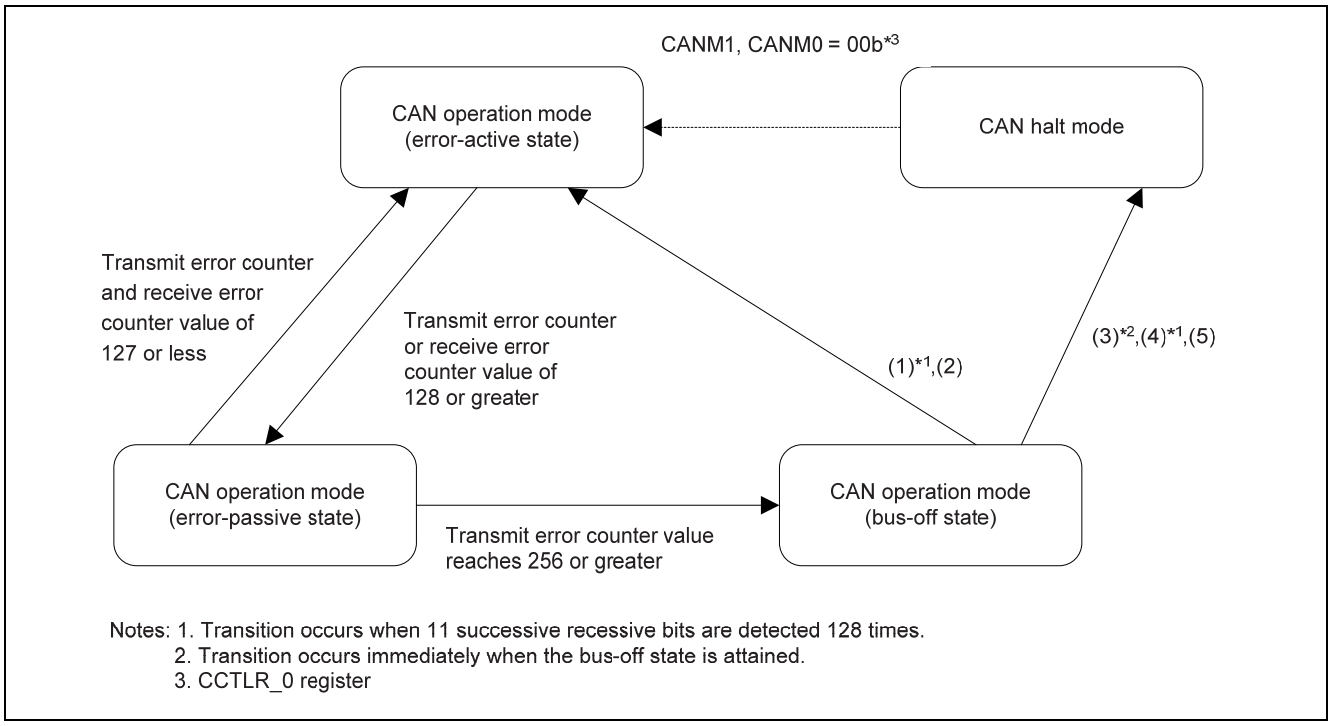


Figure 2.34 Transition to and from the Bus-Off State

2.8 Using an Acceptance Filter

An acceptance filter determines in hardware whether messages are received or discarded.

2.8.1 Standard ID and Extended ID

There are two CAN message ID formats: standard ID and extended ID. The former consists of 11 bits and the later consists of 29 bits.

Figure 2.35 shows bit maps of the standard ID and extended ID.

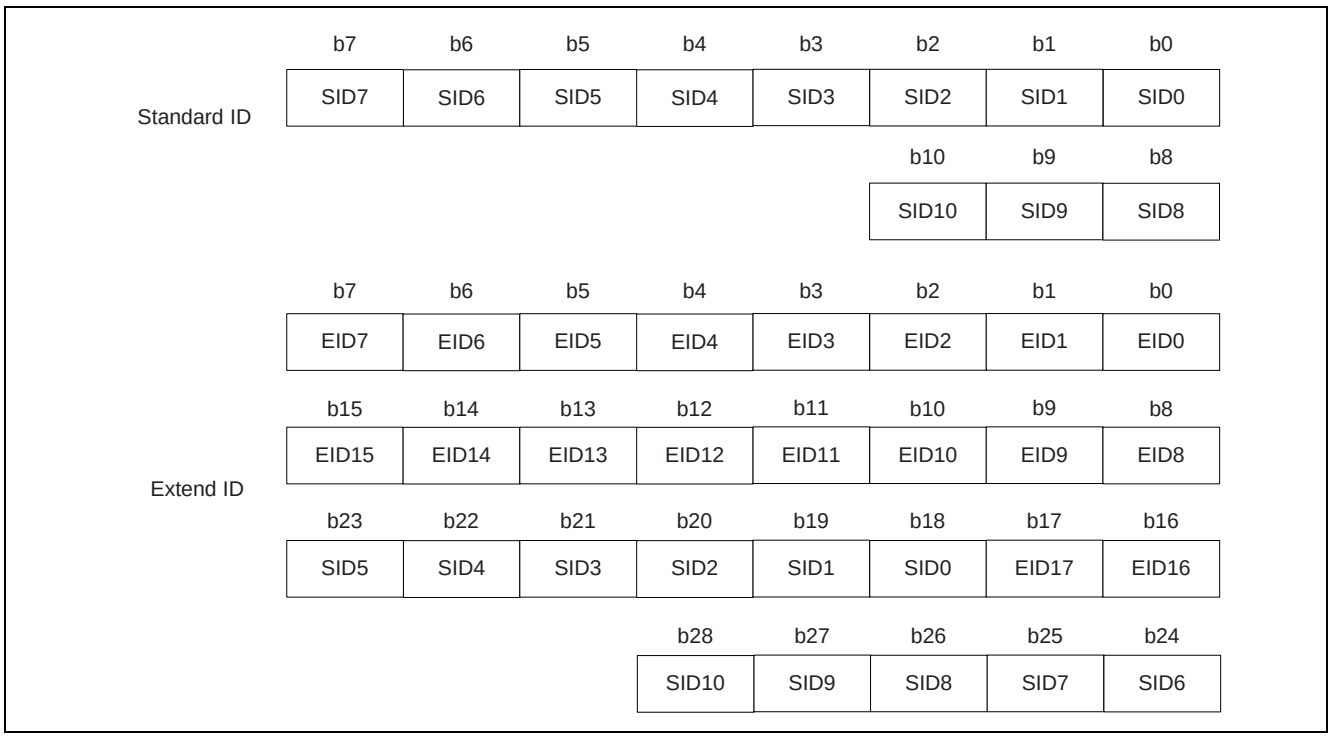


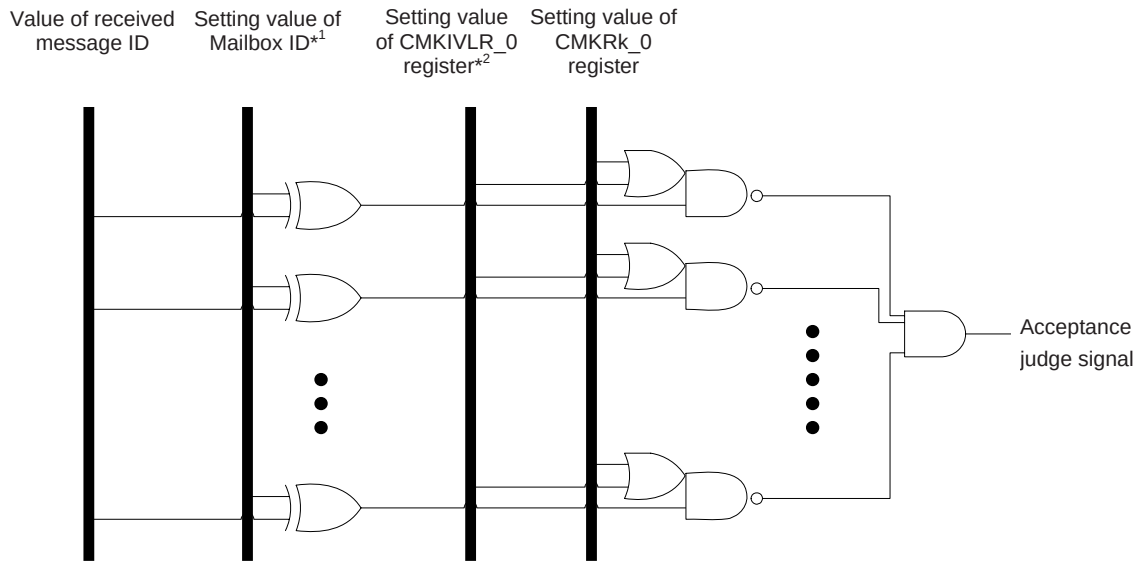
Figure 2.35 Bit Maps of Standard ID and Extended ID

2.8.2 Applying an Acceptance Filter to the Normal Receive Mailbox

An acceptance filter enables filtering by using CMKRk_0 register. For details on using an acceptance filter in FIFO mailbox mode, see section 2.8.3.

(1) Acceptance Filter Register Configuration

Figure 2.36 shows the ID and mask register configuration, and figure 2.37 shows a bit map.



Notes: 1. The values set in registers CFIDCR0_0 and CFIDCR1_0 are used in FIFO mailbox mode.
2. Invalid in FIFO mailboxes.

Figure 2.36 ID and Mask Register Configuration

Mailbox ID	Received Message ID Value Setting
CMKIVLR_0 register	0: Mask valid 1: Mask invalid
CMKRk_0 register	0: Corresponding ID bit is not compared 1: Corresponding ID bit is compared*
Acceptance signal value	0: Discard message 1: Receive message

Note: The compared IDs include the IDE bit and RTR bit. Note that the IDE bit is compared only when bits IDFM1 and IDFM0 in the CCTLR_0 register is set to mixed ID mode.

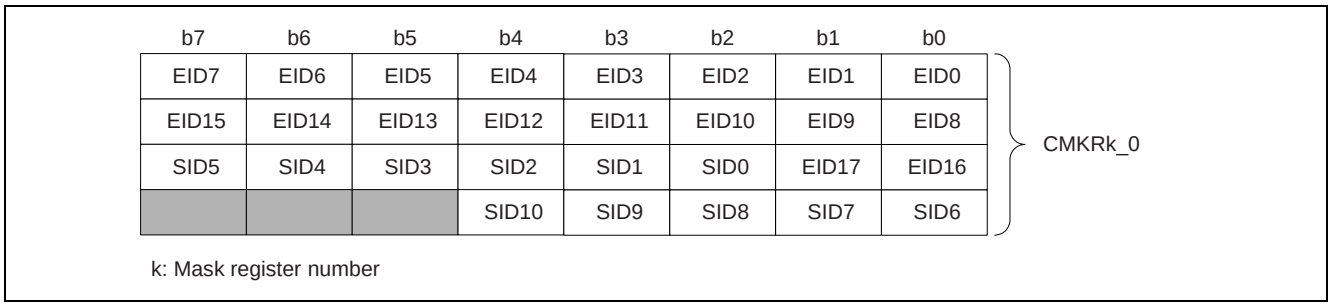


Figure 2.37 Bit Map

(2) Acceptance Filter Usage Examples

(a) **Usage Example 1**

Table 2.7 shows the register settings for receiving a standard data frame with ID 123h in mailbox [0].

Table 2.7 Acceptance Filter Usage Example 1

		IDE*1, RTR, SID10 to SID6	SID5 to SID0, EID17 to EID16	EID15 to EID8	EID7 to EID0
Mailbox [0]		0000100	100011XX	XXXXXXXX	XXXXXXXX
Mask register	CMKR0_0	--11111	111111XX	XXXXXXXX	XXXXXXXX
Received message	ID 123h	0000100	100011--	--------	--------

Note 1. The IDE bit is enabled when bits IDFM1 and IDFM0 in the CCTLR_0 register is 10b (mixed ID mode).
When bits IDFM1 and IDFM0 is not 10b, the IDE bit should be written with 0.

(b) **Usage Example 2**

Table 2.8 shows the register settings for receiving a standard remote frame with ID 123h in mailbox [0].

Table 2.8 Acceptance Filter Usage Example 2

		IDE*1, RTR, SID10 to SID6	SID5 to SID0, EID17 to EID16	EID15 to EID8	EID7 to EID0
Mailbox [0]		0100100	100011XX	XXXXXXXX	XXXXXXXX
Mask register	CMKR0_0	--11111	111111XX	XXXXXXXX	XXXXXXXX
Received message	ID 123h	0100100	100011--	--------	--------

Note 1. The IDE bit is enabled when bits IDFM1 and IDFM0 in the CCTLR_0 register is 10b (mixed ID mode).
When bits IDFM1 and IDFM0 is not 10b, the IDE bit should be written with 0.

(c) **Usage Example 3**

Table 2.9 shows the register settings for receiving two standard data frames with IDs 122h and 123h in mailbox [0].

Table 2.9 Acceptance Filter Usage Example 3

		IDE*1, RTR, SID10 to SID6	SID5 to SID0, EID17 to EID16	EID15 to EID8	EID7 to EID0
Mailbox [0]		0000100	10001XXX	XXXXXXXX	XXXXXXXX
Mask register	CMKR0_0	--11111	111110XX	XXXXXXXX	XXXXXXXX
Received message	ID 122h	0000100	100010--	--------	--------
	ID 123h	0000100	100011--	--------	--------

Note 1. The IDE bit is enabled when bits IDFM1 and IDFM0 in the CCTLR_0 register is 10b (mixed ID mode).
When bits IDFM1 and IDFM0 is not 10b, the IDE bit should be written with 0.

(d) **Usage Example 4**

Table 2.10 shows the register settings for receiving an extended data frame with ID 12345678h in mailbox [0].

Table 2.10 Acceptance Filter Usage Example 4

		IDE*1, RTR, SID10 to SID6	SID5 to SID0, EID17 to EID16	EID15 to EID8	EID7 to EID0
Mailbox [0]		1010010	00110100	01010110	01111000
Mask register	CMKR0_0	--11111	11111111	11111111	11111111
Received message	ID 12345678h	1010010	00110100	01010110	01111000

Note 1. The IDE bit is enabled when bits IDFM1 and IDFM0 in the CCTLR_0 register is 10b (mixed ID mode).
When bits IDFM1 and IDFM0 is not 10b, the IDE bit should be written with 0.

(e) **Usage Example 5**

Table 2.11 shows the register settings for receiving an extended remote frame with ID 12345678h in mailbox [0].

Table 2.11 Acceptance Filter Usage Example 5

		IDE*1, RTR, SID10 to SID6	SID5 to SID0, EID17 to EID16	EID15 to EID8	EID7 to EID0
Mailbox [0]		1110010	00110100	01010110	01111000
Mask register	CMKR0_0	--11111	11111111	11111111	11111111
Received message	ID 12345678h	1110010	00110100	01010110	01111000

Note: 1. The IDE bit is enabled when bits IDFM1 and IDFM0 in the CCTLR_0 register is 10b (mixed ID mode).
When bits IDFM1 and IDFM0 is not 10b, the IDE bit should be written with 0.

2.8.3 Applying an Acceptance Filter to the Receive FIFO

This acceptance filter mode is used with FIFO mailbox mode. Two filters can be applied to the receive FIFO. This extends the range of IDs that can be received by the receive FIFO. If there are 16 mailboxes for example, the two mask registers CMKR2_0 and CMKR3_0, and the two FIFO receive ID compare registers CFIDCR0_0 and CFIDCR1_0, would be used. In this acceptance filter mode the IDs of received messages are compared with registers CFIDCR0_0 and CFIDCR1_0 instead of the mailbox ID.

Figure 2.38 shows the ID and mask register configuration.

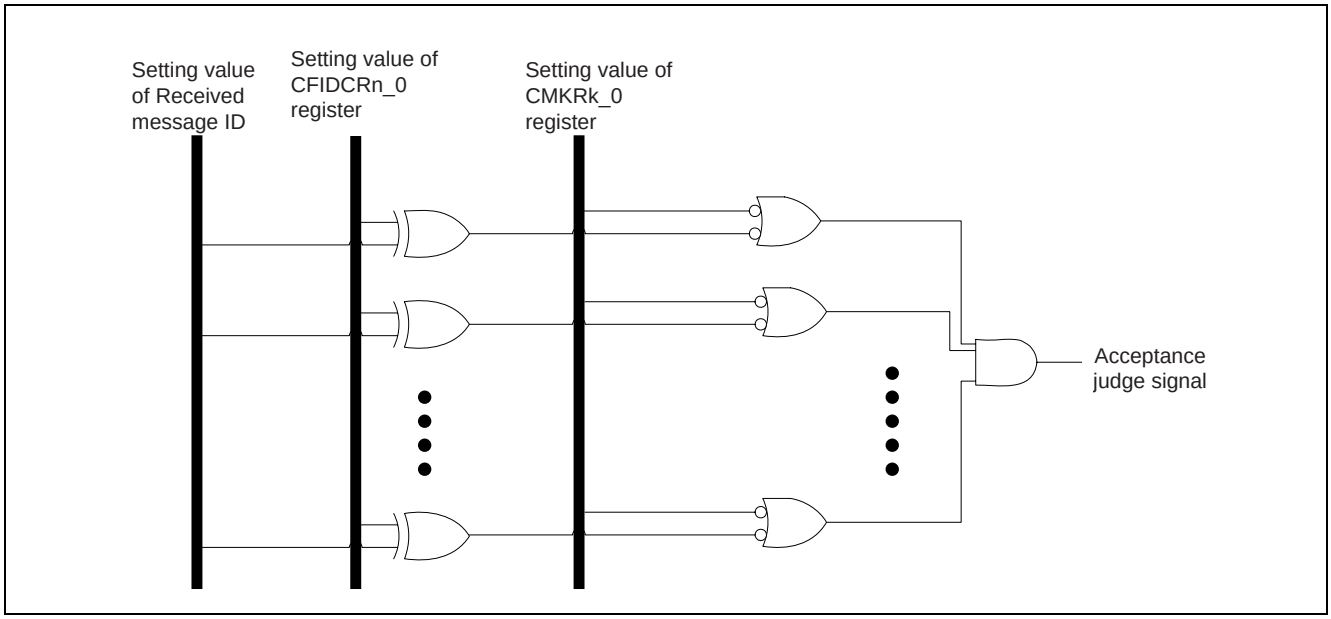


Figure 2.38 CFIDCRn_0 Register and CMKRk_0 Register Configuration

CFIDCRn_0 register	Received Message ID Value Setting
CMKRk_0 register	0: Corresponding ID bit is not compared 1: Corresponding ID bit is compared*
Acceptance signal value	0: Discard message 1: Receive message

Note: The compared IDs include the IDE bit and RTR bit. Note that the IDE bit is compared only when bits FM1 and IDFM0 in the CCTLR_0 register is set to mixed ID mode.

2.8.4 Acceptance Filter Support Unit (ASU)

The acceptance filter support unit determines whether a receive ID is valid or invalid by means of a table lookup (8 bit × 256). First, the IDs to be received are registered in a data table. Next, the ID of each received message is stored in the CAFSR_0 register, the decoded receive ID is read from the CAFSR_0 register, and a table lookup is performed. The acceptance filter support unit can only be used with standard frame IDs.
Using the acceptance filter support unit is effective in the following situations:

- When it is not possible to mask the IDs to be received by using an acceptance filter (e.g.: receive IDs 078h, 087h, and 111h)
- When the number of IDs to be received is extremely large and software filtering would take an excessive amount of time

(1) **Using the Acceptance Filter Support Unit**

This example shows how to use the acceptance filter support unit to allow reception of IDs 000h, 00Dh, 6F3h, 6F4h, and 6FFh.

(a) Data Table Settings
Create a data table in ROM or RAM for registering the IDs to be received. The data table may be mapped to any range of addresses.
In the data table, each vertical column contains the value of the upper 8 bits (SID10 to SID3) of a valid receive ID, and each horizontal row the value of the lower 3 bits (SID2 to SID0) decoded into 8 bits. Corresponding bits are set to 1 and the other bits are cleared to 0.

(b) Writing to the CAFSR_0 register
When CAN_0 receives a message, the received ID is written to the CAFSR_0 register.

(c) Reading from the CAFSR_0 register
The value of the upper 8 bits (SID10 to SID3) of the receive ID, and the value of the lower 3 bits (SID2 to SID0) decoded into 8 bits, are read from the CAFSR_0 register.

(d) Determining Validity of Received IDs
Using the values read from the CAFSR_0 register in step (c), a lookup is performed of the data table created in step (a) to determine whether the message is valid or invalid.

Figure 2.39 shows the data table configuration, and figure 2.40 shows the states when writing to and reading from the CAFSR_0 register.

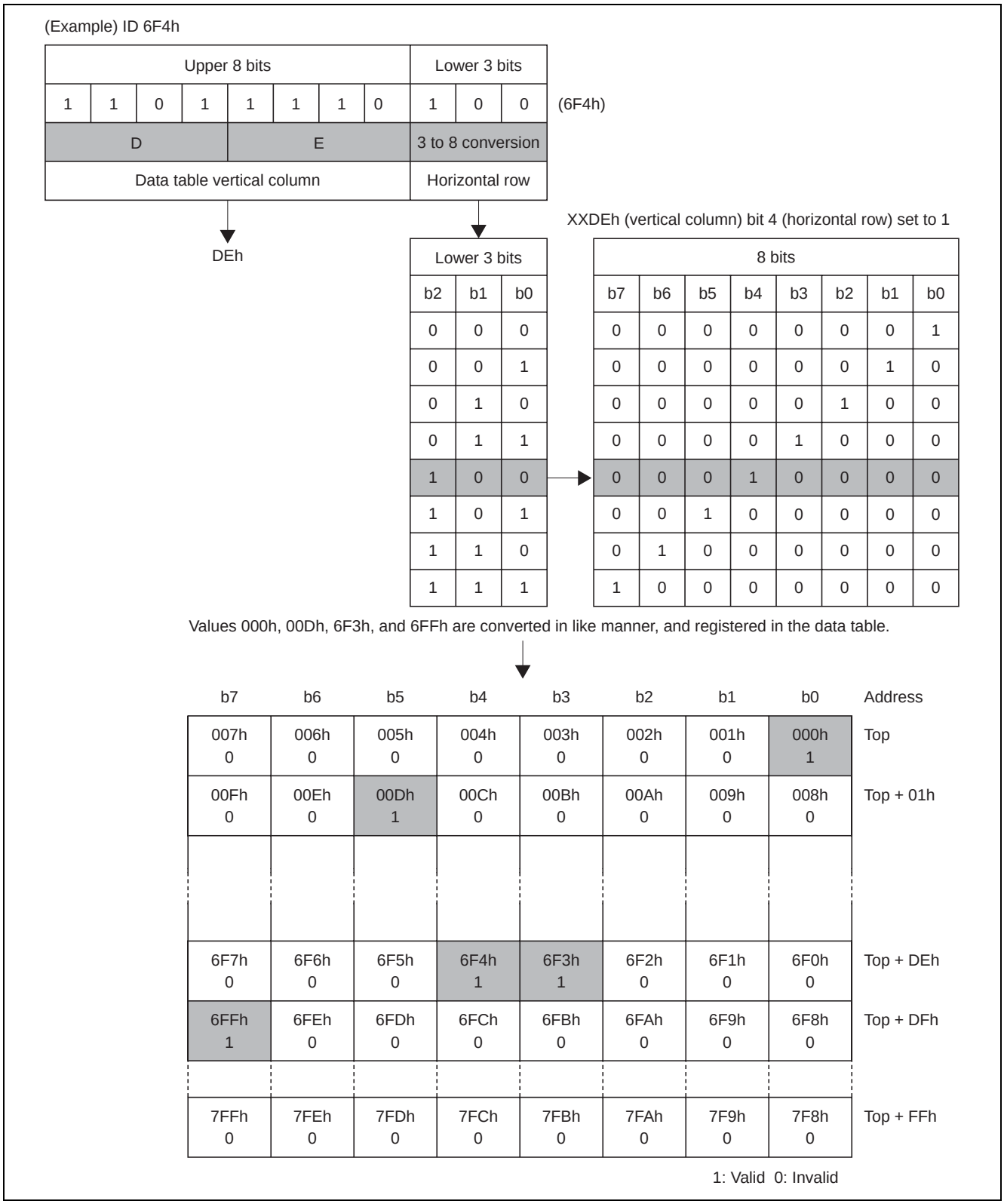


Figure 2.39 Data Table Configuration

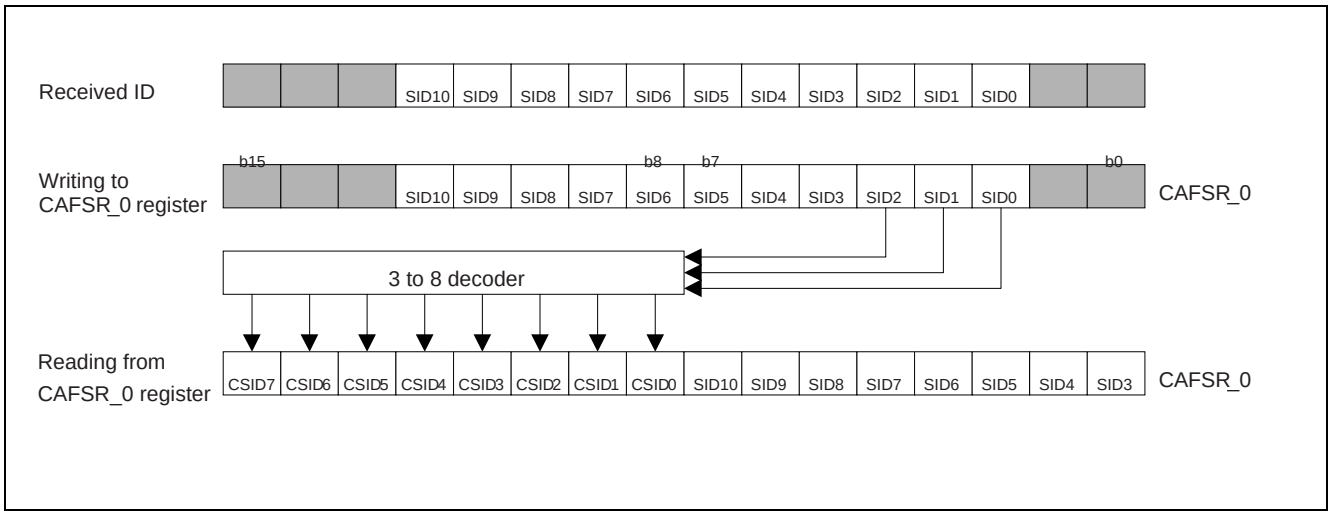
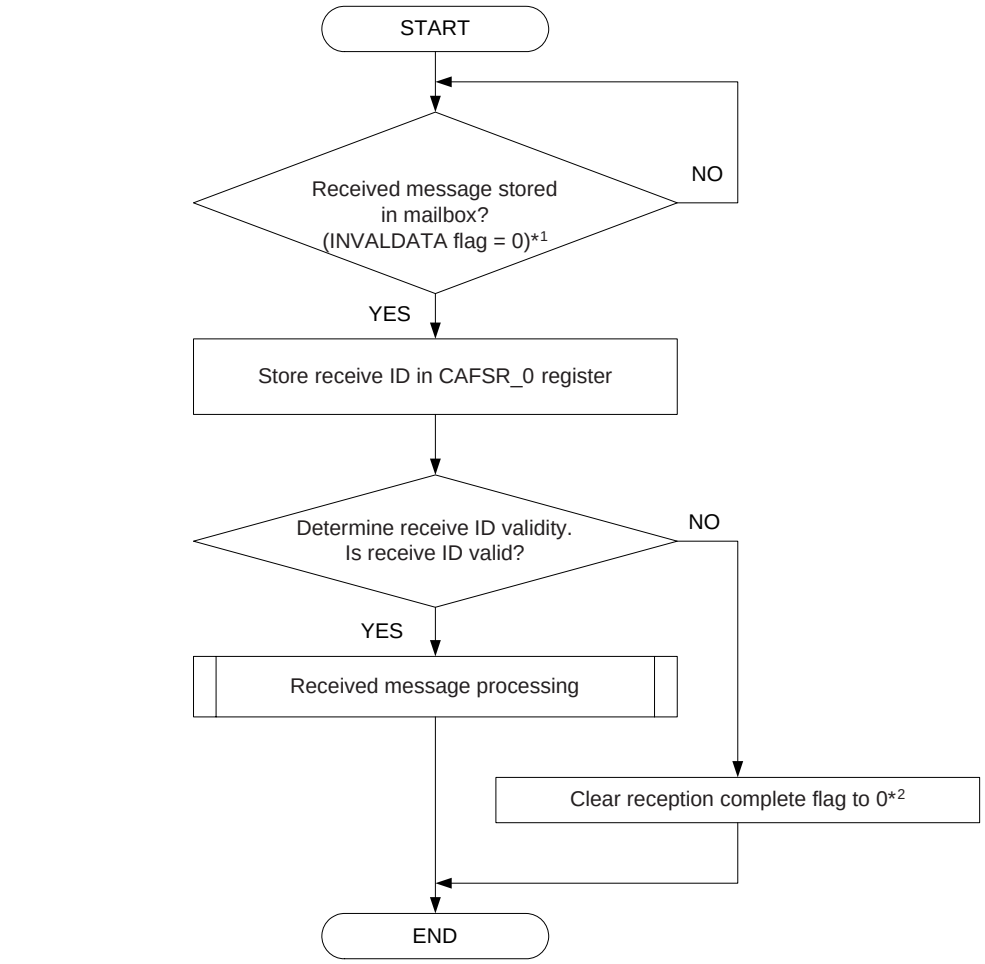


Figure 2.40 States when Writing to and Reading from CAFSR_0 Register

Figure 2.41 shows the usage procedure for the acceptance filter support unit.



Notes: 1. This process is not necessary in overrun mode or one-shot receive mode.
2. When writing 0 to bits NEWDATA, SENTDATA, MSGLOST, TRMABT, RECREQ, and TRMREQ by a program, use the MOV instruction to ensure that only the specified bit is set to 0 and the other bits are set to 1.

Figure 2.41 Acceptance Filter Support Unit Usage Procedure

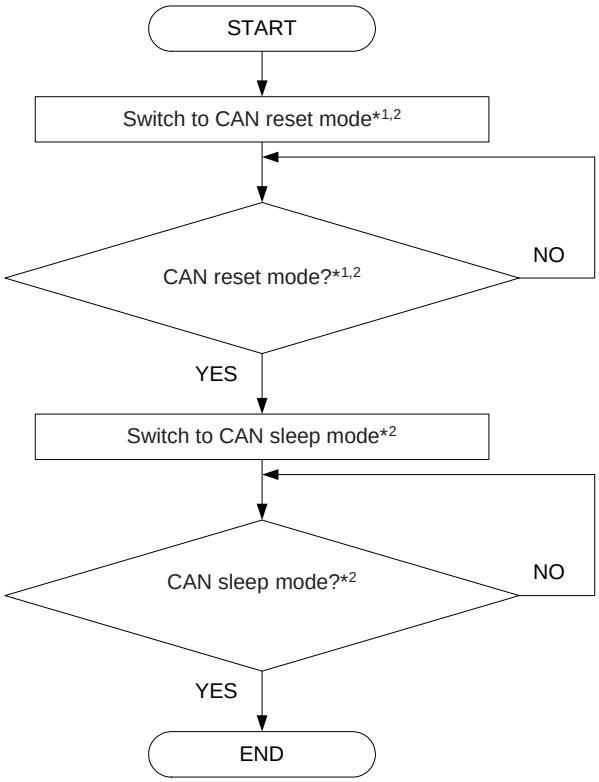
2.9 CAN Sleep Operation and CAN Wakeup Operation

2.9.1 CAN Sleep Operation

No clock is supplied to the CAN module when it is in CAN sleep mode, so the CAN module does not operate at all. Putting the CAN module into CAN sleep mode is recommended when the CAN module is not in use or when it is necessary to reduce the current consumption.

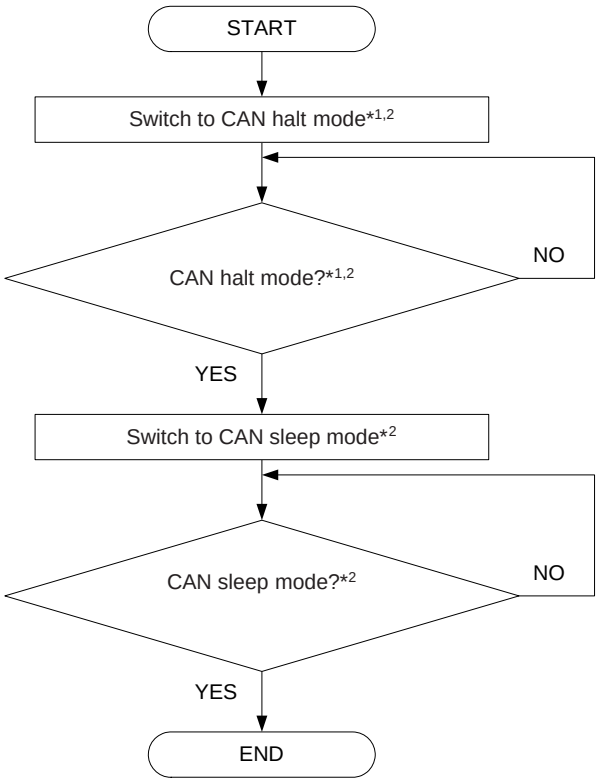
Always switch the CAN module to CAN reset mode or CAN halt mode before transitioning to CAN sleep mode. Figure 2.42 shows the procedure for switching the CAN module to CAN reset mode and then to CAN sleep mode, and figure 2.43 shows the procedure for switching the CAN module to CAN halt mode and then to CAN sleep mode.

When the CAN module is in CAN sleep mode, a further reduction in current consumption can be achieved by switching the MCU to wait mode or stop mode.



Note: 1. If a transition to CAN reset mode is triggered while the CAN module is transmitting, the reset does not occur until the transmit operation completes. Therefore, always check the CAN reset status flag.
2. When bits CANM0, CANM1, and SLPM are changed, check the CSTR register to ensure that the mode has been switched.

Figure 2.42 Procedure for Switching CAN Module to CAN Reset Mode and then CAN Sleep Mode



Note: 1. If a transition to CAN halt mode is triggered while the CAN module is transmitting or receiving, the switch to CAN halt mode does not occur until the transmit or receive operation completes. Therefore, always check the CAN halt status flag.
2. When bits CANM0, CANM1, and SLPM are changed, check the CSTR register to ensure that the mode has been switched.

Figure 2.43 Procedure for Switching CAN Module to CAN Halt Mode and then CAN Sleep Mode

2.9.2 CAN Wakeup Operation

The CAN module can be restored from CAN sleep mode by means of a CAN wakeup interrupt issued when the CAN receive line signal level falls.

To use CAN wakeup, set the CAN wakeup interrupt priority level more than 000b in the CANERIC_0 register[*1] and enable the wakeup interrupt by setting the WKUPIE in CANIE_0 register.

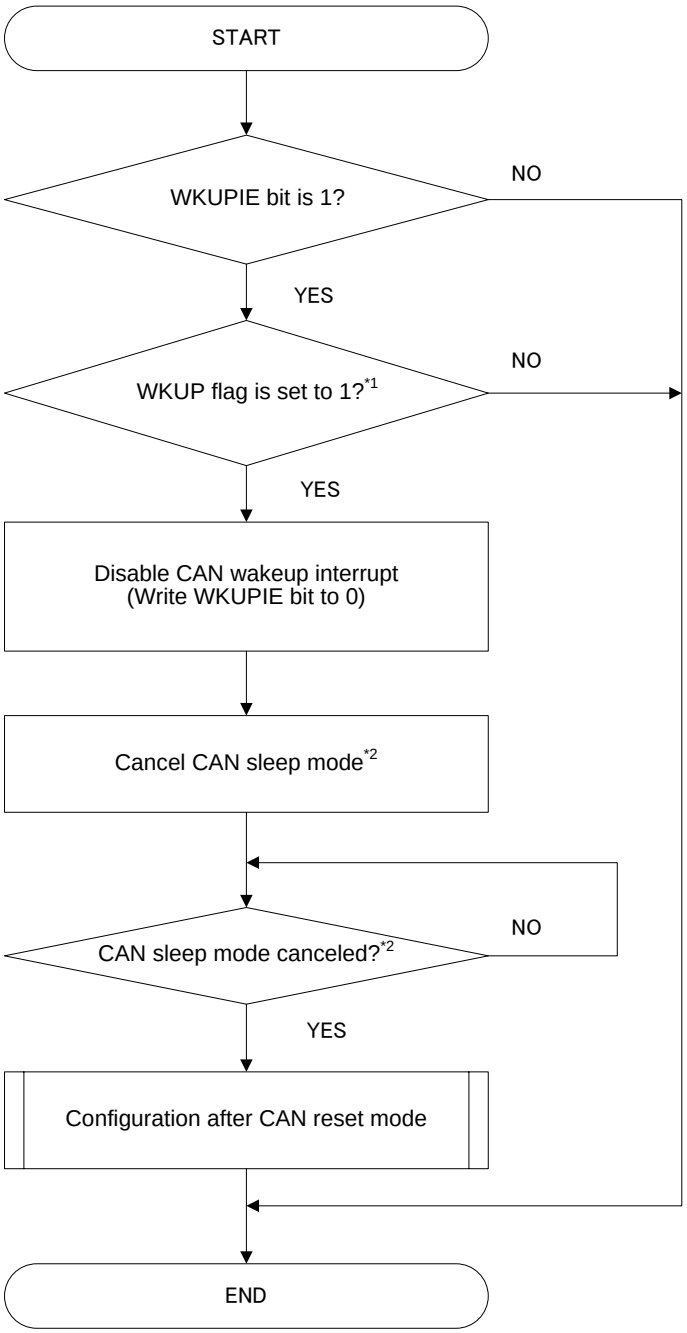
When CAN sleep mode is canceled by software, the CAN module reenters the mode (CAN reset mode or CAN halt mode) it was in before the transition to CAN sleep mode.

Figure 2.44 shows the CAN wakeup procedure when the CAN module was switched to CAN sleep mode from CAN reset mode, and figure 2.45 shows the CAN wakeup procedure when the CAN module was switched to CAN sleep mode from CAN halt mode.

In these examples, the CAN module is returned to CAN operation mode within the CAN wakeup interrupt routine.

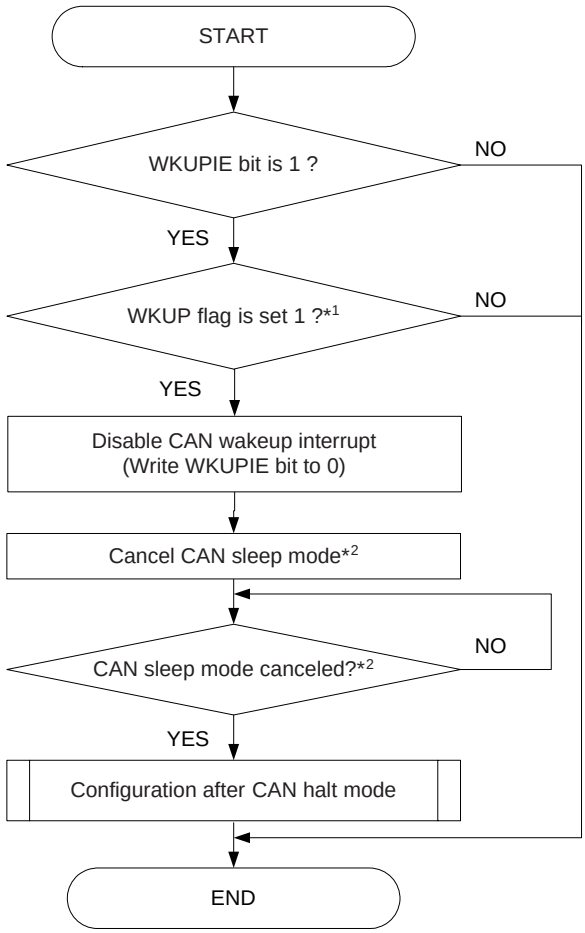
When using the CAN wake up interrupt, CPE bit in the CCTLR_0 register should be set to 1.

Note: 1. CAN wakeup interrupt request is generated, not only the time in the CAN sleep mod but when a message is detected on CAN bus.



Note:*1. When enable CAN wakeup interrupt at next, please clear WKUP flag to 0.
*2. When bits CANM0, CANM1, and SLPM are changed, check the CSTR register to ensure that the mode has been switched.

Figure 2.44 CAN Wakeup Procedure in CAN Reset Mode



Notes. 1. When enable CAN wakeup interrupt at next, please clear WKUP flag to 0
2. When bits CANM0, CANM1, and SLPM are changed, check the CSTR register to ensure that the mode has been switched.

Figure 2.45 CAN Wakeup Procedure in CAN Halt Mode

2.10 Test Modes

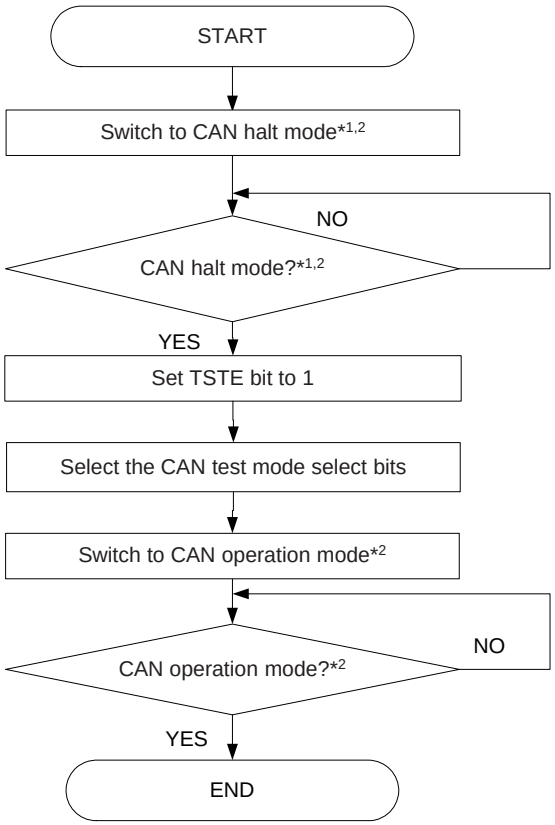
The following three test modes are provided for evaluation by the user:

- Listen-only mode
- Self-test mode 0 (external loop-back)
- Self-test mode 1 (internal loop-back)

Only select test modes when the CAN module is in CAN halt mode.

2.10.1 Test Mode Selection

Figure 2.46 shows the procedure for selecting a test mode.



Notes. 1. If a transition to CAN halt mode is triggered while the CAN module is transmitting or receiving, the switch to CAN halt mode does not occur until the transmit or receive operation completes.
2. When bits CANM0 and CANM1 are changed, check the CSTR register to ensure that the mode has been switched.

Figure 2.46 Test Mode Selection Procedure

2.10.2 Listen-Only Mode

The CAN specification (ISO 11898-1) recommends the implementation of an optional bus monitor mode. In listen-only mode, the CAN node can receive valid data frames and valid remote frames, but only recessive bits are sent via the CAN bus and transmit-start is not enabled. When the CAN engine receives a request to transmit a dominant bit (ACK bit, overload flag, or active error flag), the bit is rerouted internally so that the CAN engine can treat it as dominant even though the CAN bus remains in the recessive state.

Listen-only mode can be used for baud rate detection.

Do not initiate transmit requests from any mailbox when in listen-only mode.

To select listen-only mode, set bits TSTM1 and TSTM0 to 01b.

Figure 2.47 illustrates the operation of listen-only mode.

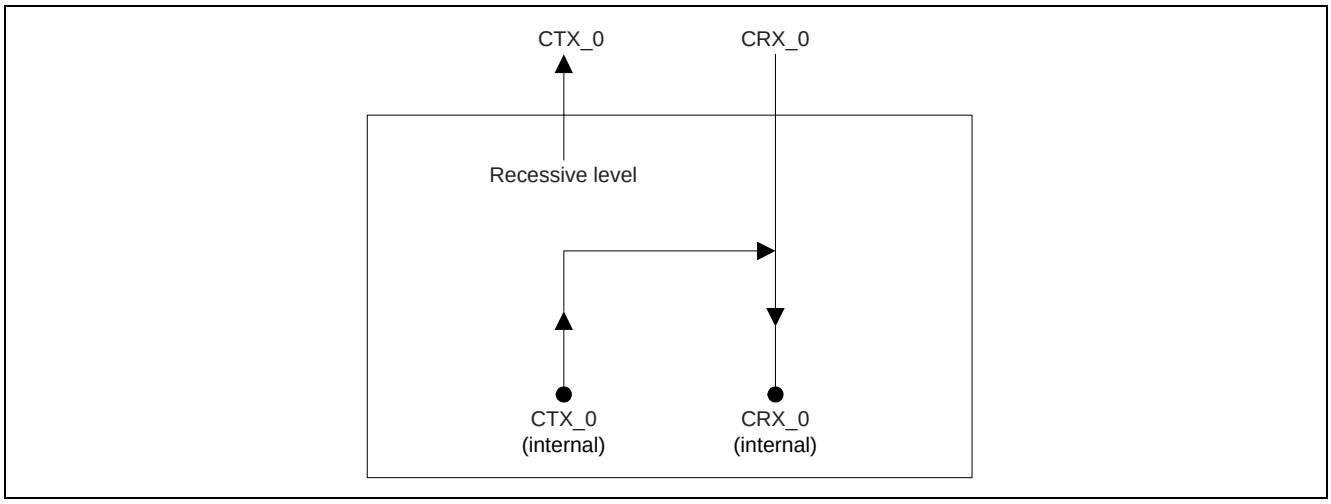


Figure 2.47 Listen-Only Mode Operation Illustration

2.10.3 Self-Test Mode 0 (External Loop-Back)

Self-test mode 0 is for CAN transceiver testing.

In self-test mode 0, the CAN engine handles messages it has transmitted itself as messages received via a CAN transceiver. Each transmitted message is stored in the receive buffer. Since this function is performed independently of the external device, the CAN engine generates its own ACK bits.

To use self-test mode 0, connect the CTX_0 and CRX_0 pins to a CAN transceiver.

To select self-test mode 0, set bits TSTM1 and TSTM0 to 10b.

Figure 2.48 illustrates the operation of self-test mode 0.

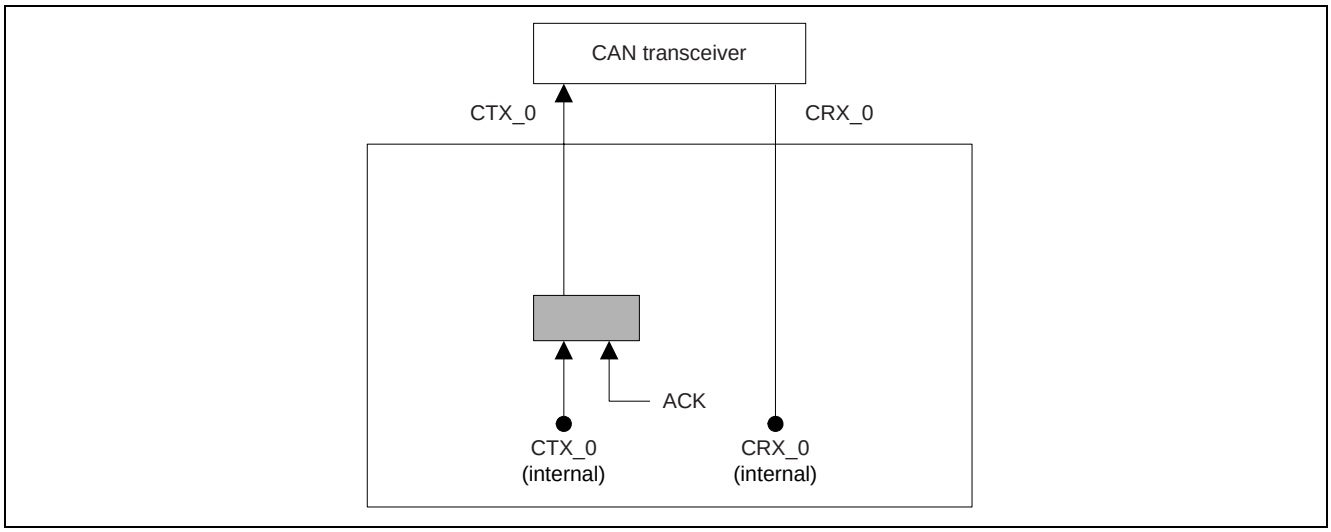


Figure 2.48 Self-Test Mode 0 Operation Illustration

2.10.4 Self-Test Mode 1 (Internal Loop-Back)

Self-test mode 1 is for self-testing.

In self-test mode 1, the CAN engine handles messages it has transmitted itself as received messages. Each transmitted message is stored in the receive buffer. Since this function is performed independently of any external device, the CAN engine generates its own ACK bits.

In self-test mode 1, the CAN engine performs internal feedback from the internal CTX_0 pin to the internal CRX_0 pin. The CAN engine ignores the actual value of input to the external CRX_0 pin. The external CTX_0 pin outputs only recessive bits. It is not necessary to connect the CTX_0 or CRX_0 pin to the CAN bus or to any device to use self-test mode 1.

To select self-test mode 1, set bits TSTM1 and TSTM0 to 11b.

Figure 2.49 illustrates the operation of self-test mode 1.

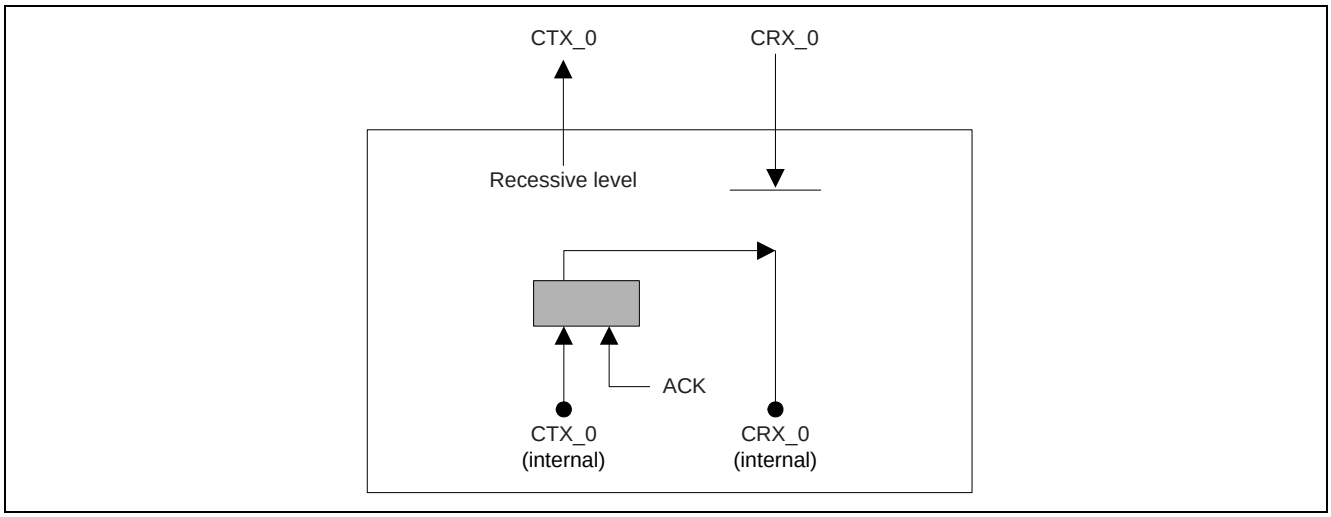


Figure 2.49 Self-Test Mode 1 Operation Illustration

2.11 CAN Interrupt

The CAN module provides the following 6 kinds of CAN interrupts:

- CAN reception complete interrupt
- CAN transmission complete interrupt
- CAN receive FIFO interrupt
- CAN transmit FIFO interrupt
- CAN error interrupt
- CAN wakeup interrupt

Every 2 kinds of CAN interrupts use 1 interrupt vector resource (share the same interrupt vector number and same interrupt control register).

- CAN reception complete interrupt & CAN receive FIFO interrupt share 1 interrupt vector resource.
- CAN transmission complete interrupt & CAN transmit FIFO interrupt share 1 interrupt vector resource.
- CAN error interrupt & CAN wakeup interrupt share 1 interrupt vector resource.

Figure 2.50 shows the relation of CAN interrupt control and Interrupt Request bit (IR) in the corresponding interrupt control register.

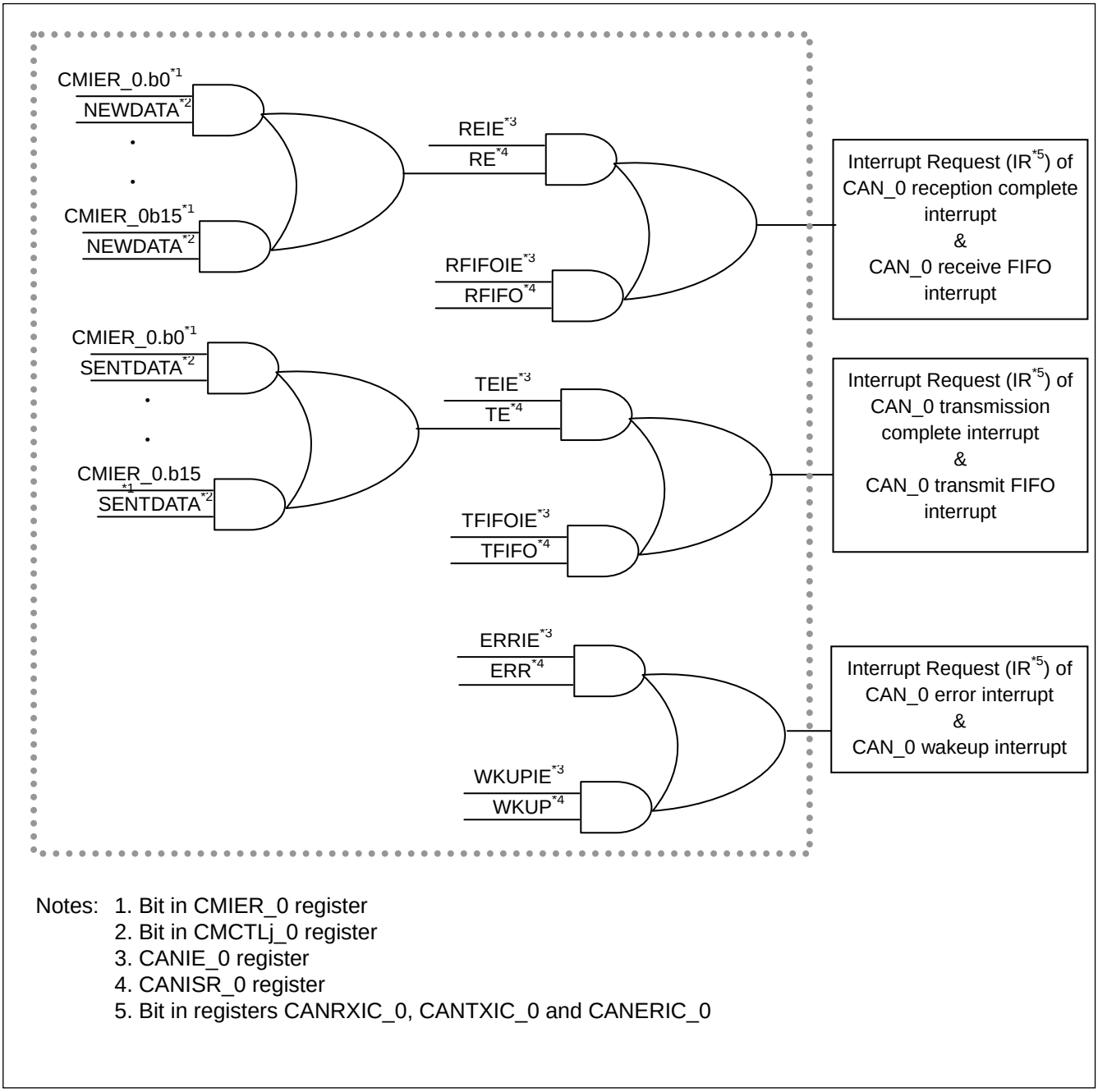


Figure 2.50 Relation between CAN interrupt control and interrupt request

2.11.1 Feature of R8C / 5X Series CAN Module Interrupt

Since each interrupt source of CAN module is generated by a combination of two interrupt request sources, the following differences from other maskable interrupts apply:

- When bits in the enable register are set to 1 and the corresponding bits in the status register are set to 1 (interrupt enabled), the IR bit in the interrupt control register is set to 1 (interrupt requested).
- When either bits in the status register or the corresponding bits in the enable register, or both are set to 0, the IR bit is set to 0 (no interrupt requested).
  That is, even if the interrupt is not acknowledged after the IR bit is set to 1, the interrupt request will not be retained.
  Also, if 0 is written to the IR bit, this bit is temporarily set to 0 (for five cycles of the CPU clock) and then set back to 1.
- Individual bits in the status register are not automatically set to 0 even if the interrupt is acknowledged.
  The IR bit is also not automatically set to 0 when the interrupt is acknowledged.
  Set individual bits in the status register to 0 in the interrupt routine.
  Refer to Table2.12 for how to set individual bits in the status register to 0.
- When multiple bits in the enable register are set to 1 and other request sources are generated after the IR bit is set to 1, the IR bit remains 1.
- When multiple bits in the enable register are set to 1, use the status register to determine which request source causes an interrupt.

The IR bit cannot be set to 0 when all the bits in the status register are not set to 0 in the interruption routine.

While IR is 1, when it escapes from the interruption routine, the same interrupt occurs again.

As with other maskable interrupts, CAN transmit, CAN receive, CAN error interrupts are controlled by the combination of the IR bit, bits ILVL0 to ILVL2, and IPL.

2.11.2 Conditions for Setting Each Flag in the CANISR_0 register to 1

A RFIFO flag is set to 1 under the following conditions.

- When a message is received to Receive FIFO at the timing of the b13 in the CMIER_0 register while the b12 in the CMIER_0 register is 1 (interrupt enable):

A TFIFO flag is set to 1 under the following conditions.

- When a message is transmitted to Transmit FIFO at the timing of the b9 in the CMIER_0 register while the b8 in the CMIER_0 register is 1 (interrupt enable):

A ERR flag is set to 1 under the following conditions.

- When one or more corresponding errors* occur while the corresponding bits in the CEIER_0 register is 1 (interrupt enable):

  Note: refer to bit explanation of CEIFR_0 register of the User' manual hardware for each error.

A WKUP flag is set to 1 under the following conditions.

- When a message (falling edge) is detected on CAN Bus:

A RE flag is set to 1 under the following conditions.

- When a message is received to the mailbox which is enabled interrupts by CMIER_0 register (NEWDATA flag is 1):

A TE flag is set to 1 under the following conditions.

- When a message is transmitted to the mailbox which is enabled interrupts by CMIER_0 register (SENTDATA flag is 1):

2.11.3 Conditions for Setting Each Flag in the CANISR_0 register to 0

Flags RFIFO, TFIFO, and ERR are set to 0 under the following conditions.

- After reset
- After reset When the CAN module enters CAN reset mode
- When 0 is written after reading 1

WKUP flag is set to 0 under the following conditions.

- After reset
- When 0 is written after reading 1

Flags RE and TE are read-only bits. These flags are set to 0 under the following conditions.

- After reset
- When the CAN module enters CAN reset mode
- RE flag: when all NEWDATA flags are to 0*
- TE flag: when all SENTDATA flags are to 0*

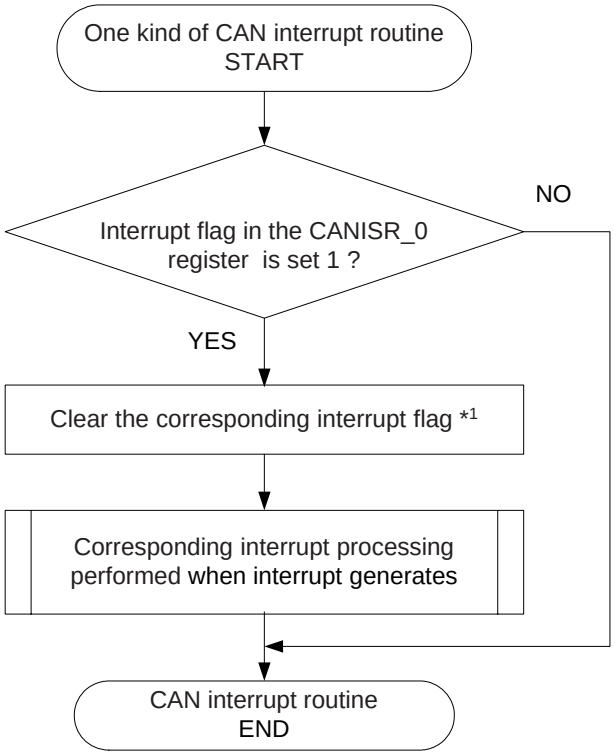
Note *: The corresponding flags are flags NEWDATA and SENTDATA in the mailbox whose interrupt enable bit in CMIER_0 register is 1 (interrupt enable). Although flags NEWDATA flag and SENTDATA are 1, when the interrupt enable bit is cleared to "0" (interrupt disable), it becomes nothing the interrupt request of the mailbox

Table 2.12 shows the Cleaning the CAN interrupt Request and Figure 2.51 shows the Procedure for CAN Interrupt Routine.

Table 2.12 Cleaning the CAN interrupt Request (IR bit)

Interrupt	Method of clearing the interrupt flag	Method of clearing the IR bit
CAN reception complete interrupt	By clearing all NEWDATA flags in the CMCTLj_0 register*2 (RE flag in the CANISR register is set to 0 with the above-mentioned procedure.)*3	Both of interruption flags are set to 0.
CAN receive FIFO interrupt	By clearing the REIFO flag in the CANISR_0 register*1,3	
CAN transmission complete interrupt	By clearing all SENTDATA flags in the CMCTLj_0 register*2 (TRE flag in the CANISR register is set to 0 with the above-mentioned procedure.) *3	Both of interruption flags are set to 0.
CAN transmit FIFO interrupt	By clearing the TFIFO flag in the CANISR_0 register*1,3	
CAN error interrupt	By clearing the ERR flag in the CANISR_0 register*1,3	Both of interruption flags are set to 0.
CAN wakeup interrupt	By clearing the WKUP flag in the CANISR_0 register*1	

Notes *1. These flags are set to 0 only when 0 is written after reading 1.
*2. When the corresponding bits in the CMIER_0 register are 1 (interrupt enabled).
*3. Flags RE, TE, RFIFO, TFIFO and ERR is to 0 also by enter CAN reset mode.



Note 1. In the case of RE flag and TE flag, this processing is unnecessary.
2. Figure 2.44 and 2.45 shows the procedure for CAN wakeup interrupt routine.

Figure 2.51 CAN interrupt routine procedure Example

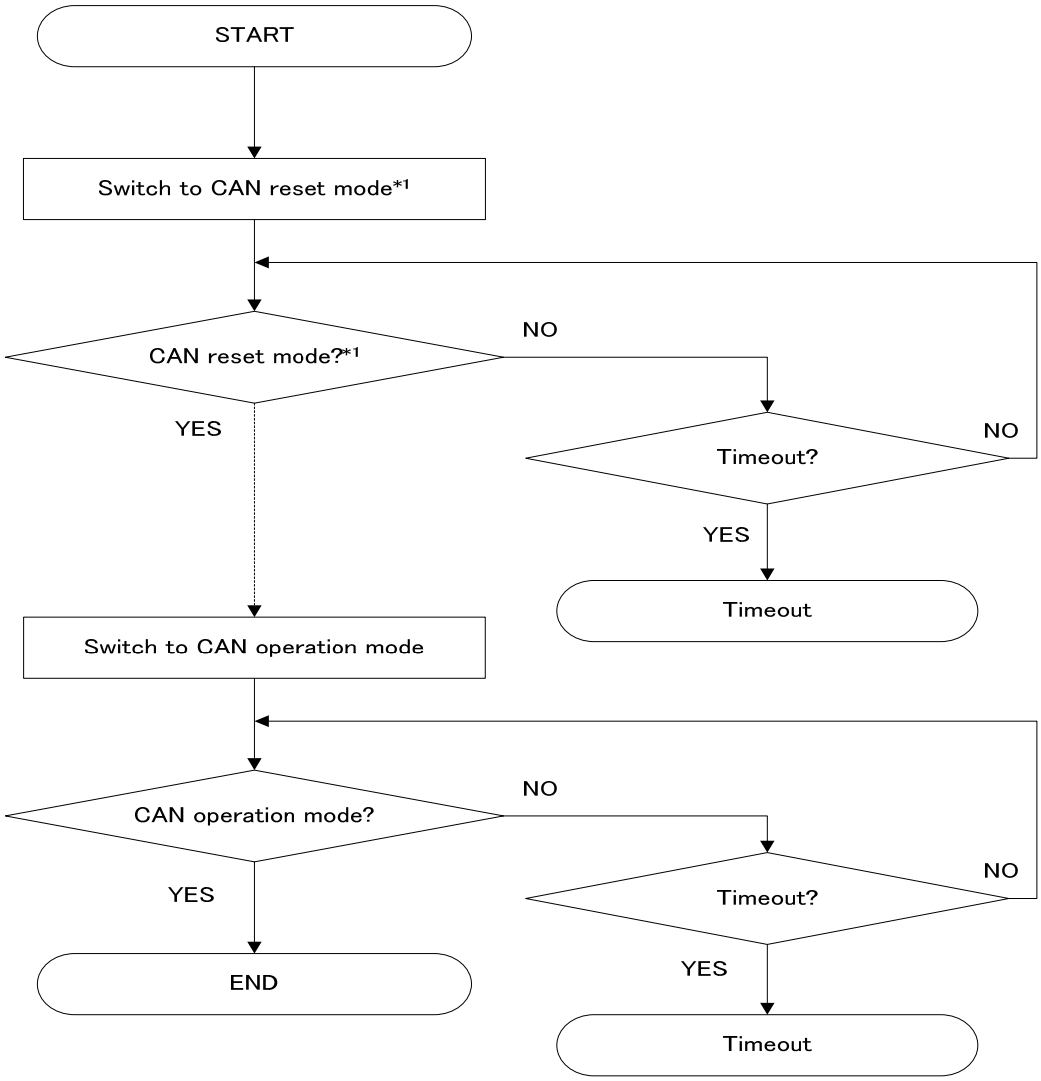
2.12 Notes on the Processing Flow

2.12.1 Infinite Loops

Since the notations in this document are abbreviated, they may contain instances where an infinite loop can occur in the processing flow. When creating the actual program code, make sure to insert loop time limits to prevent overruns from occurring.

Figure 2.52 shows a processing example using a loop time limit.

When the configuration after CAN reset mode



Note: 1. When the CAN module is reset during transmission of a CAN message, it waits for the transmission operation to complete and then transition to CAN reset mode. The time limit for the transition to CAN reset mode is the maximum time required to transmit one frame of approximately 160 bits (maximum number of bits in extended data frame (approximately 130 bits) + maximum number of stuff bits (approximately 30 bits)).

[Example]
Time required to transmit one frame (160 bits) at 500 Kbps: 320 µsec.
The time limit is set to 320 µsec., so processing breaks the loop after more than 320 µsec. elapse.

Figure 2.52 Processing Example with Loop Time Limit Example

3. Reference Documents

R8C/5x Series User's Manual Hardware Rev.1.00

The latest version can be downloaded from the Renesas Electronics website.

Technical Update/Technical News

The latest information can be downloaded from the Renesas Electronics website.

Website and Support

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

Inquiries
http://www.renesas.com/inquiry

REVISION HISTORY	R8C/5x Series Application Note CAN Application note

Rev.	Date	Description	
		Page	Summary
1.10	Mar 15, 2012	—	First edition issued

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this manual, refer to the relevant sections of the manual. If the descriptions under General Precautions in the Handling of MPU/MCU Products and in the body of the manual differ from each other, the description in the body of the manual takes precedence.

1. Handling of Unused Pins

   Handle unused pins in accord with the directions given under Handling of Unused Pins in the manual.

   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment when power is supplied.

   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.

   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.

   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.

   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

5. Differences between Products

   Before changing from one product to another, i.e. to one with a different part number, confirm that the change will not lead to problems.

   — The characteristics of MPU/MCU in the same group but having different part numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different part numbers, implement a system-evaluation test for each of the products.

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.

RENESAS

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 1.1