RL78/F24

RS-CANFD Lite Configuration, Reception and Transmission

R01AN6464EJ0100
Rev.1.00
2022.09.30

Introduction

This application note describes the procedures for configuring a control area network (CAN) bus for the RL78/F24 microcontrollers. This application note also describes the procedures for receiving and transmitting messages on the CAN Bus. Regarding the settings to each register, refer to the cautions and notes in the latest User's Manual: Hardware.

Target Device

This application note is applied to the RL78/F24 microcontrollers.

The table below lists the variables used in this document.

Target devices and their corresponding variables

	Index	Target MCUs
		RL78/F24
CAN receive rule entry register number (GAFLIDiL, GAFLIDiH, GAFLMiL, GAFLMiH, GAFLP0iL, GAFLP0iH and GAFLP1iL registers)	i	i=0 to 15
PNF receive rule entry register number (GPFLIDjL, GPFLIDjH, GPFLMjL, GPFLMjH, GPFLP0jL, GPFLP0jH, GPFLP1jL, GPFLPTjL, GPFLPTjH, GPFLPD0jL, GPFLPD0jH, GPFLPD1jL, GPFLPD1jH, GPFLPM0jL, GPFLPM0jH, GPFLPM1jL and GPFLPM1jH registers)	j	j=0, 1
Receive FIFO buffer number	k	k=0, 1
Transmit buffer number	m	m=0 to 3
Receive buffer number	n	n=0 to 15
Data field register number (RFDFk_pL, RFDFk_pH, CFDFpL, CFDFpH, TMDFm_pL, TMDFm_pH, RMDFn_pL and RMDFn_pH)	p	p=0 to 15
CAN RAM test register number (RPGACCrL and RPGACCrH register)	r	r=0 to 63

Contents

1. CAN Configuration

1.1 CAN Configuration

With CAN configuration, the functions needed for CAN communication are configured. Carry out the CAN configuration before CAN communication starts or restarts after a microcontroller unit (MCU) is reset, the CAN module is software reset, any bus error is detected, or a wake-up signal is generated.

The CAN configuration can be performed in the following modes. Regarding the CAN status (mode), see “1.2 CAN Status (Mode) Transitions”.

- Global reset mode
- Channel reset mode
- Channel halt mode

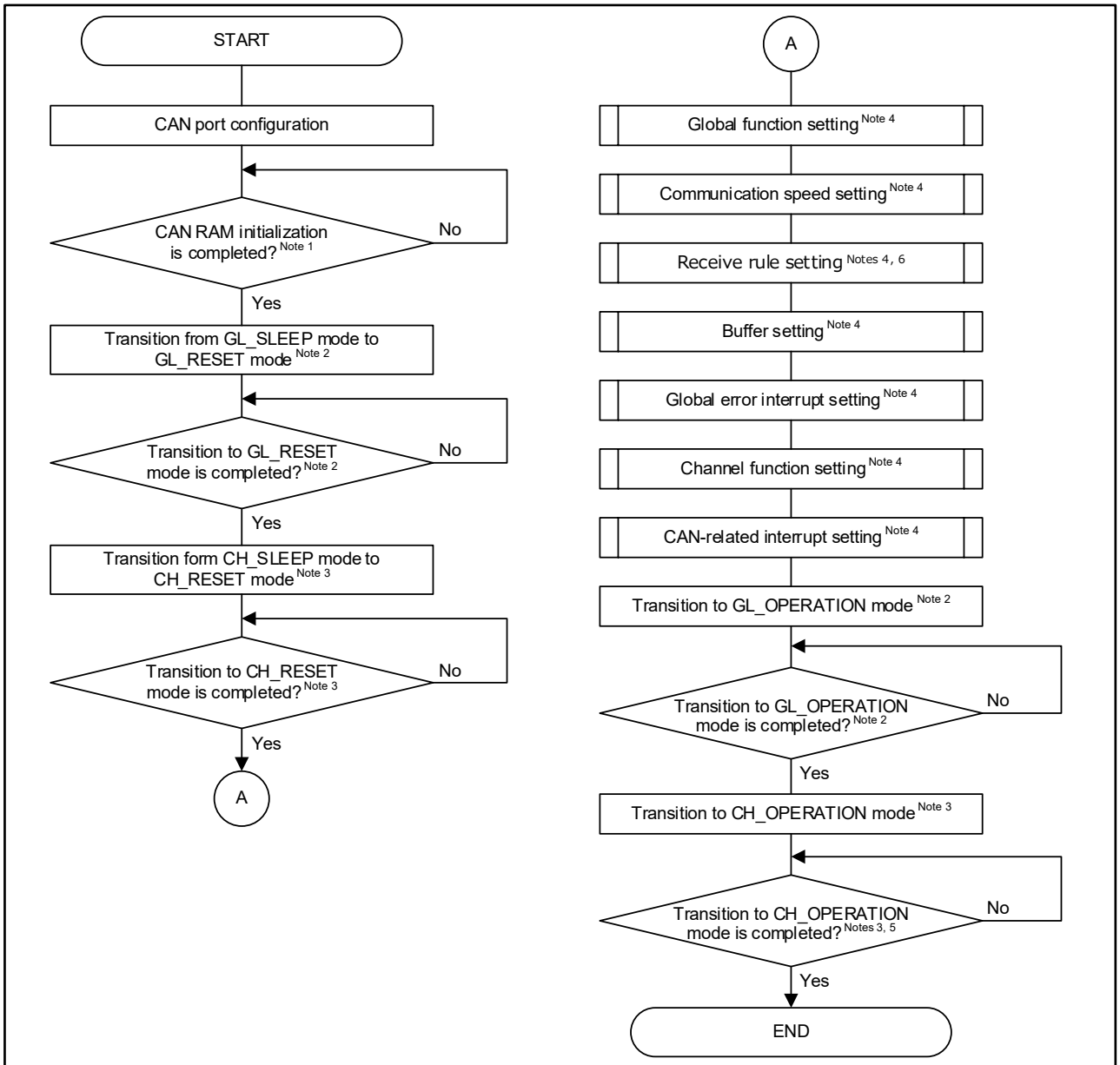
Note: After the CAN module is enabled (set the CAN0EN bit in the PER2 register to 1)

The functions below need to be set with the CAN configuration. For details, refer to the following sections.

- CAN status (mode) transition
- Communication speed
- Global functions
- Receive rule table
- Buffers
- Global error interrupts
- Channel functions
- PNF receive rules table

1.1.1 CAN configuration after CAN module is enabled

Initialize the entire CAN module after the CAN module is enabled. Figure 1.1 show the CAN configuration procedures after the CAN module is enabled.



Note 1: After the CAN module is enabled (set the CAN0EN bit in the PER2 register to 1), do not access CAN RAM before the CAN RAM is initialized (set the GRAMINIT bit in the GSTS register to 1).

Note 2: When changing a global mode (the GSLPR bit and GMDC[1:0] bits in the GCTRL register), check the GSTS register to see if the transition is completed. Do not modify the value of the GMDC[1:0] bits until the mode transition is completed.

Note 3: When changing a channel mode (the CSLPR bit and CHMDC[1:0] bits in the C0CTRL register), check the C0STSL register to see if the transition is completed. Do not modify the value of the CHMDC[1:0] bits until the mode transition is completed.

Note 4: For detailed information on the processing for each function, refer to the following sections.

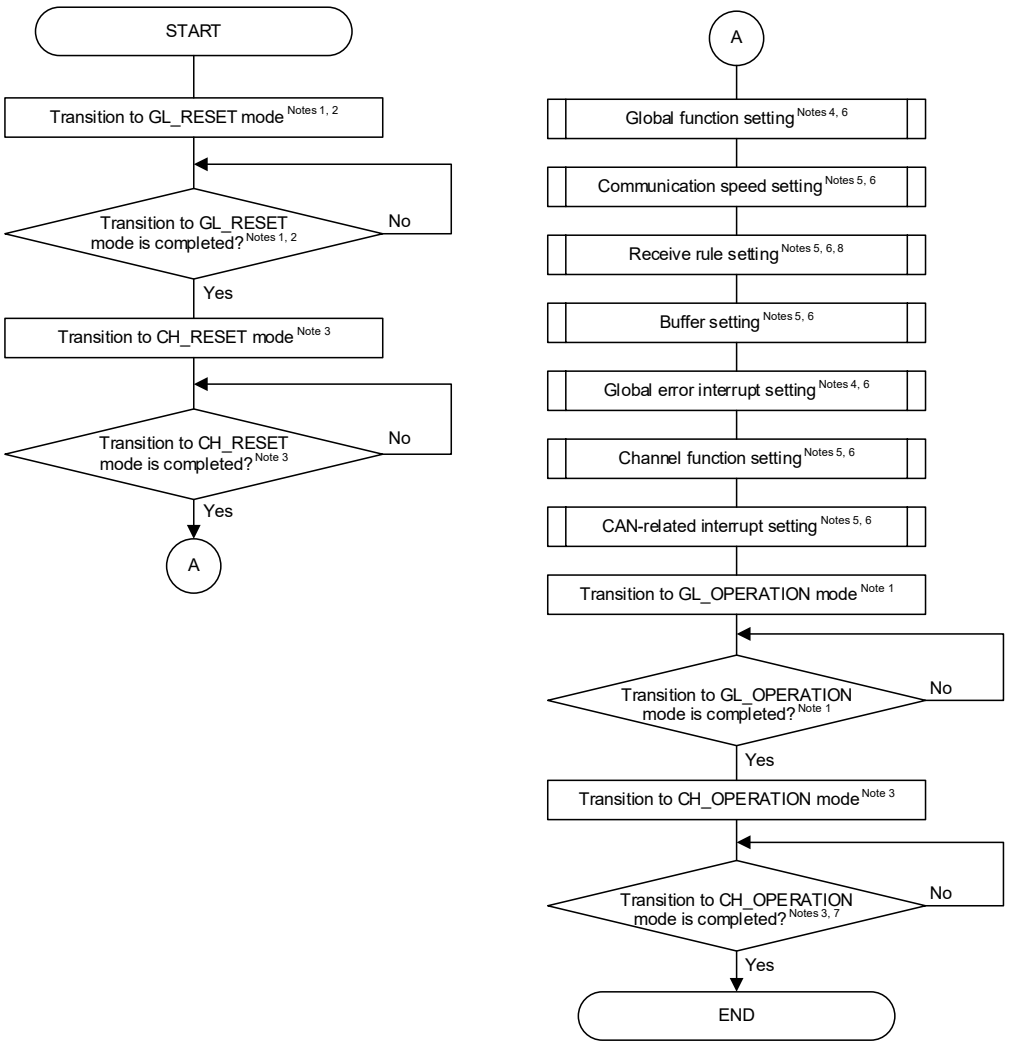
Note 5: When 11 consecutive recessive bits are detected after a transition to channel operation mode is completed, communication will be available (the COMSTS flag in the C0STSL register is set to 1) and transmission/reception will be enabled on the CAN Bus as an active node, which means message transmission/reception is enabled.

Note 6: When using PNF, set up the receive rules and the PNF receive rules.

Figure 1.1 Configuration Procedures After CAN Module is Enabled

1.1.2 CAN configuration after transition to global reset mode

Figure 1.2 show the initialization procedure for the entire CAN configuration after the transition to global reset mode is completed.



Note 1: When changing a global mode (the GSLPR bit and GMDC[1:0] bits in the GCTRL register), check the GSTS register to see if the transition is completed. Do not modify the value of the GMDC[1:0] bits until the mode transition is completed.

Note 2: A transition of global modes may affect the channel mode. For details, see “1.2.3 Shifts in Channel Modes due to a Transition of Global Modes”.

Note 3: When changing a channel mode (the CSLPR bit and CHMDC[1:0] bits in the C0CTRL register), check the C0STSL register to see if the transition is completed. Do not modify the value of the CHMDC[1:0] bits until the mode transition is completed.

Note 4: These settings are not necessarily required because the bit values are reset even after the transition to global reset mode.

Note 5: These settings are not necessarily required because the bit values are not reset even after the transition to channel reset mode.

Note 6: For detailed information on the processing for each function, refer to the following sections.

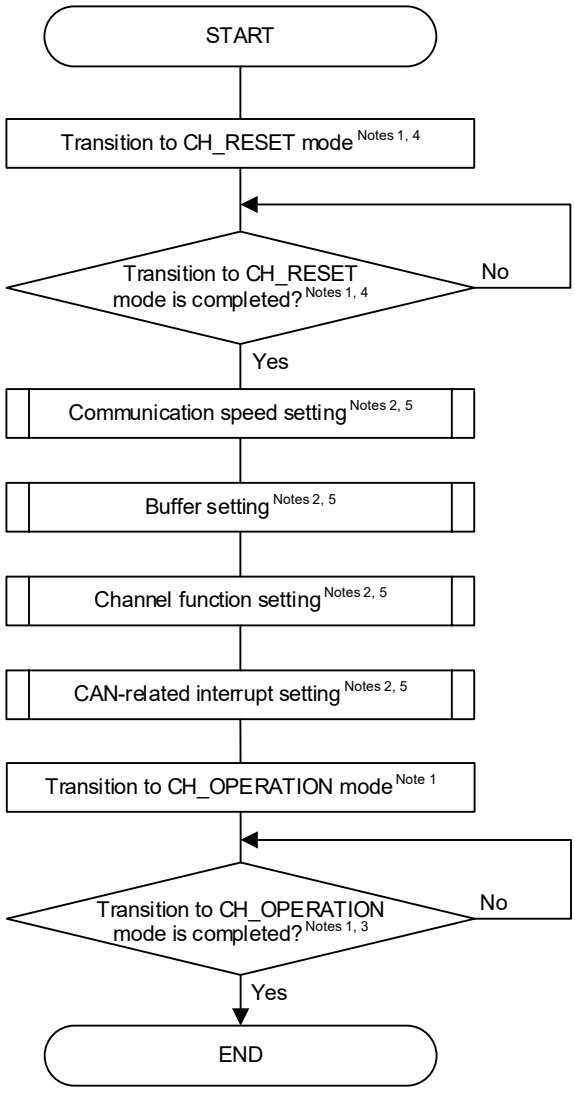
Note 7: When 11 consecutive recessive bits are detected after a transition to channel operation mode is completed, communication will be available (the COMSTS flag in the C0STSL register is set to 1) and transmission/reception will be enabled on the CAN Bus as an active node, which means message transmission/reception is enabled.

Note 8: When using PNF, set up the receive rules and the PNF receive rules.

Figure 1.2 Configuration Procedures After Transition to Global Reset Mode

1.1.3 CAN Configuration After Transition to Channel Reset Mode

Figure 1.3 shows the initialization procedures for the CAN channel(s) after the transition to channel reset mode is completed.



Note 1: When changing a channel mode (the CSLPR bit and CHMDC[1:0] bits in the C0CTRL register), check the C0STSL register to see if the transition is completed. Do not modify the value of the CHMDC[1:0] bits until the mode transition is completed.

Note 2: These settings are not necessarily required because the bit values are not reset even after the transition to channel reset mode.

Note 3: When 11 consecutive recessive bits are detected after a transition of channel operation mode is completed, communication will be available (the COMSTS flag in the C0STSL register is set to 1), and transmission/reception will be enabled on the CAN Bus as an active node, which means message transmission/reception is enabled.

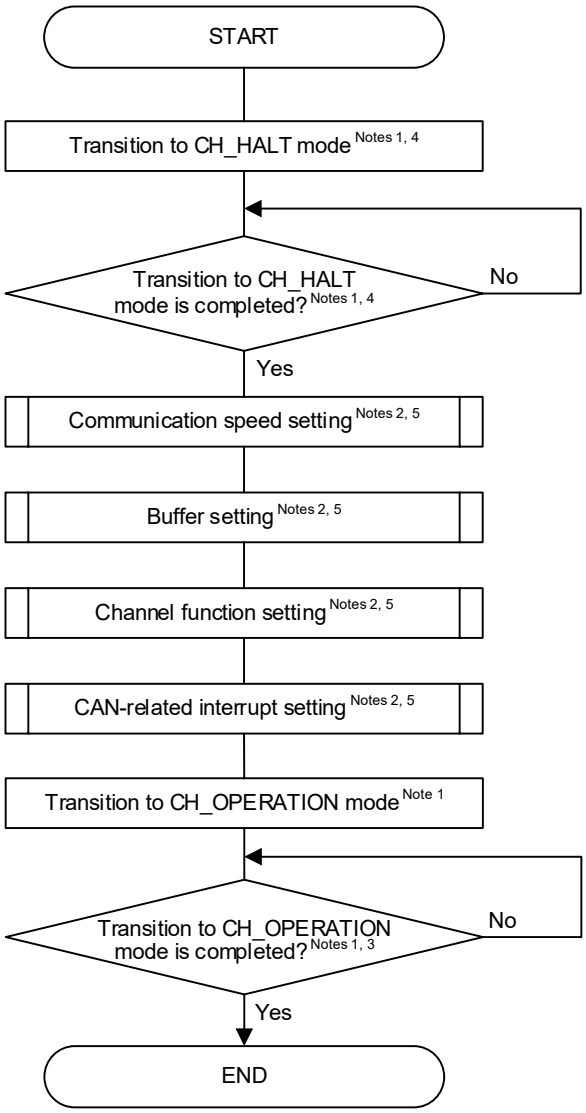
Note 4: The transition to channel reset mode is completed even though communication is not completed. To realize the transition to channel reset mode after communication is completed, set the CAN Bus to channel halt mode and confirm that the communication has been completed and the transition to channel halt mode has been completed. Then, execute the transition to channel reset mode.

Note 5: For detailed information on the processing for each function, refer to the following sections.

Figure 1.3 Configuration Procedures After Transition to Channel Reset Mode

1.1.4 CAN Configuration After Transition to Channel Halt Mode

Figure 1.4 shows the initialization procedures for the CAN channel(s) after the transition to channel halt mode is completed.



Note 1: When changing a channel mode (the CSLPR bit and CHMDC[1:0] bits in the C0CTRL register), check the C0STSL register to see if the transition is completed. Do not modify the value of the CHMDC[1:0] bits until the transition is completed.

Note 2: These settings are not necessarily required because the bit values are note reset even after a transition to channel halt mode.

Note 3: When 11 consecutive recessive bits are detected after a transition to channel operation mode is completed, communication will be available (the COMSTS flag in the C0STSL register is set to 1) and transmission/reception will be enabled on the CAN Bus as an active node, which means message trasnsmission/reception is enabled.

Note 4: While the CAN Bus is locked at the dominant level (the BLF flag in the C0ERFLL register is set to 1), a transition to channel halt mode is not available. In this case, transition the mode of the CAN module to channel reset mode.

Note 5: For detailed information on the processing for each function, refer to the following sections.

Figure 1.4 Configure Procedures After Transition to Channel Halt Mode

1.2 CAN Status (Mode) Transitions

The CAN module has four global modes to control the status of the entire CAN module and four channel modes to control individual channel status.

The CAN module has the following modes:

- Global mode
  - Global sleep mode
  - Global reset mode
  - Global halt mode
  - Global operating mode
- Channel mode
  - Channel sleep mode
  - Channel reset mode
  - Channel halt mode
  - Channel operation mode

1.2.1 Global modes

These are modes to control the entire CAN module.

Figure 1.5 shows the transitions of global modes.

Note that a transition of global modes may shift a channel mode. For details, see “1.2.3 Shifts in Channel Modes due to a Transition of Global Modes”.

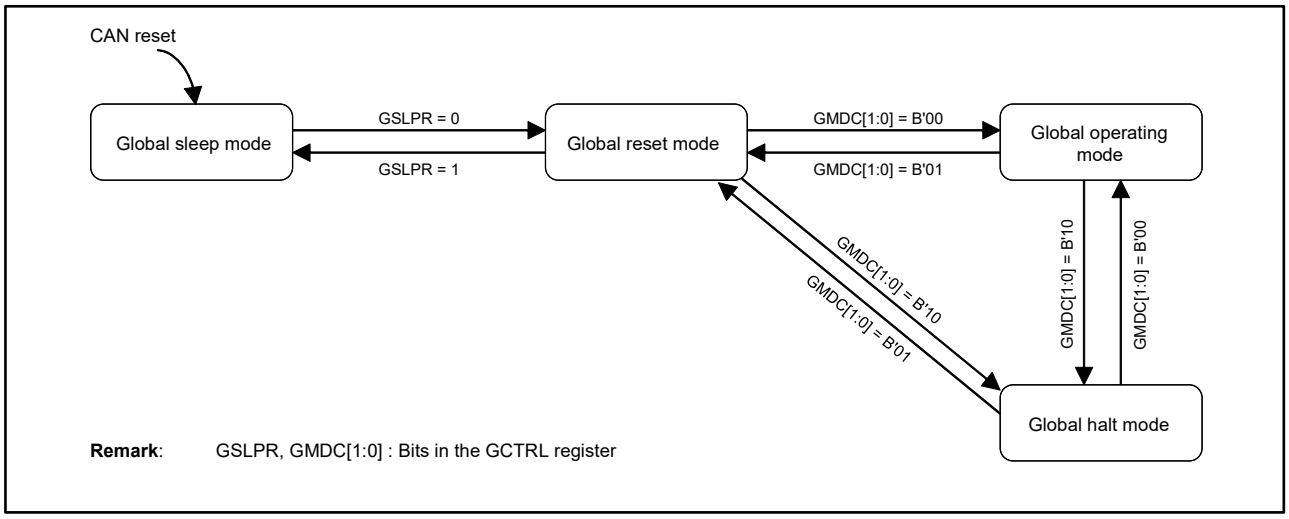


Figure 1.5 Transitions of global modes

(1) Global sleep mode

In this mode, the clock of the CAN module is stopped. Therefore, power consumption can be reduced. Read access to CAN-related registers is enabled but write access to the registers is prohibited. The values of the registers are retained.

(2) Global reset mode

This is a mode to perform settings for the entire CAN module. After the transition to global reset mode, some registers will be initialized. Table 1.2 and Table 1.3 list the registers to be initialized in this mode.

(3) Global halt mode

This is a mode to perform settings to test-related registers. After the transition to global halt mode, CAN communication (among all channels) will be stopped.

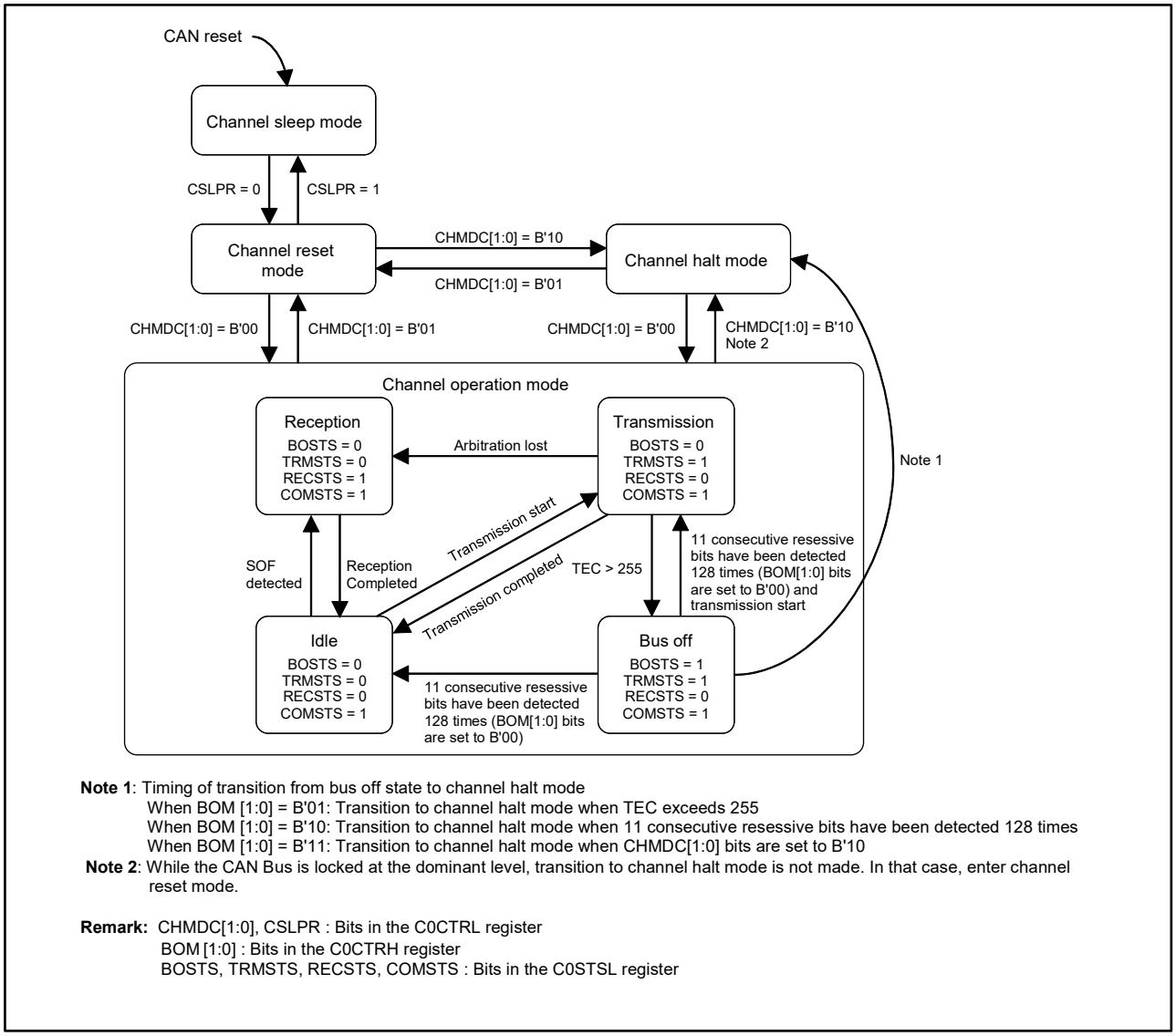
(4) Global operating mode

This is a mode to activate the entire CAN module. For CAN communication, the CAN module needs to be transitioned to global operating mode.

1.2.2 Channel modes

These are modes to control the channel(s).

Figure 1.6 shows the transitions of channel modes.



Note 1: Timing of transition from bus off state to channel halt mode
When BOM [1:0] = B'01: Transition to channel halt mode when TEC exceeds 255
When BOM [1:0] = B'10: Transition to channel halt mode when 11 consecutive resessive bits have been detected 128 times
When BOM [1:0] = B'11: Transition to channel halt mode when CHMDC[1:0] bits are set to B'10

Note 2: While the CAN Bus is locked at the dominant level, transition to channel halt mode is not made. In that case, enter channel reset mode.

Remark: CHMDC[1:0], CSLPR : Bits in the C0CTRL register
BOM [1:0] : Bits in the C0CTRH register
BOSTS, TRMSTS, RECSTS, COMSTS : Bits in the C0STSL register

Figure 1.6 Transitions of channel modes

(1) Channel sleep mode

In this mode, clock supply to the channel is stopped. Therefore, power consumption can be reduced. Read access to CAN-related registers of a corresponding channel is enabled, but write access to the registers is prohibited. The values of the registers are retained.

(2) Channel reset mode

This is a mode to perform settings of the channel. After the transition to channel reset mode, some channel-related registers will be initialized. Table 1.3 lists the registers to be initialized in this mode.

(3) Channel halt mode

This is a mode to perform setting of the registers related to channel tests. After the transition to channel halt mode, the corresponding CAN communication stops.

(4) Channel operation mode

This is a mode to perform CAN communication. The (Each) channel has the following communication status during CAN communication.

- Idle: Neither reception nor transmission is in progress.
- Reception: Receiving a message transmitted from a different (another) node
- Transmission: Transmitting a message
- Bus off: Isolated from CAN communication.

1.2.3 Shifts in Channel Modes due to a Transition of Global Modes

A transition of global modes may shift a channel mode. Table 1.1 shows the transitions of channel modes due to a transition of global modes. Figure 1.7 illustrates the transitions of channel modes due to a transition of global modes.

Table 1.1 Transitions of Channel Modes Due to Setting of Global Modes

Channel mode before global mode setting	Channel mode after global mode setting			
	Global operating	Global halt	Global reset	Global sleep
Channel operation	Channel operation	**Channel halt**	**Channel reset**	*Transition prohibited*
Channel halt	Channel halt	Channel halt	**Channel reset**	*Transition prohibited*
Channel reset	Channel reset	Channel reset	Channel reset	**Channel sleep**
Channel sleep	Channel sleep	Channel sleep	Channel sleep	Channel sleep

Note: **bold**: channel modes to be transitioned due to a transition of global modes
italic: limitations

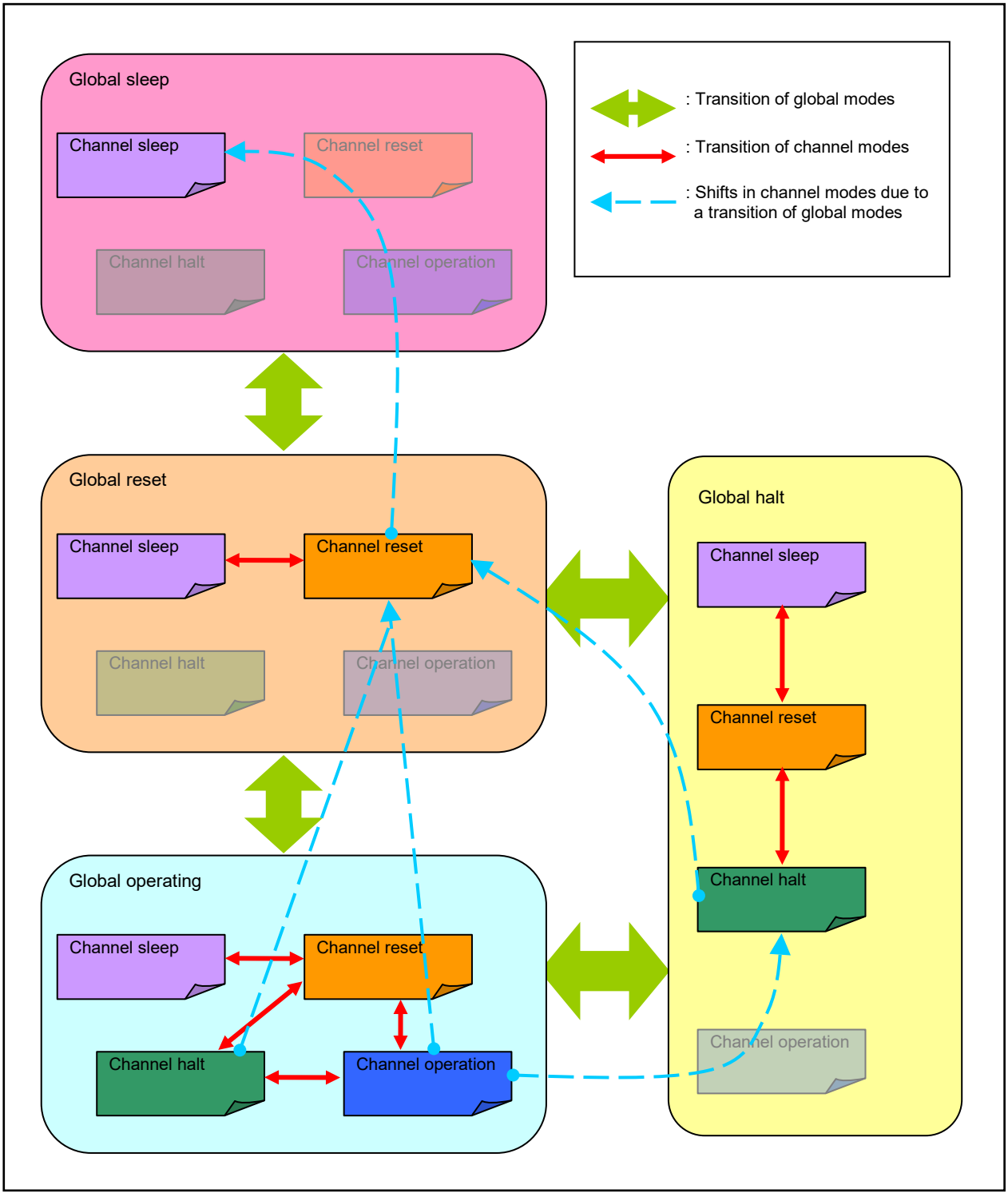


Figure 1.7 Transitions of Global Modes and Channel modes

Table 1.2 Registers to be Initialized Due to Transition to Global Reset Mode and Channel Reset Mode

Registers	Bits/flags
C0CTRL register	CHMDC[1:0]
C0CTRH register	CTMS[1:0], CTME, BFT, ROM
C0STSL register	CHLTSTS, EPSTS, BOSTS, TRMSTS, RECSTS, COMSTS, ESIF
C0STSH register	REC[7:0], TEC[7:0]
C0ERFLL register	ADERR, B0ERR, B1ERR, CERR, AERR, FERR, SERR, ALF, BLF, OVLF, BORF, BOEF, EPF, EWF, BEF
C0ERFLH register	CRCREG[14:0]
CFCCL register	When the common FIFO buffer is in transmit mode : CFE
CFSTS register	When the common FIFO buffer is in transmit mode : CFMC[5:0], CFTXIF, CFRXIF, CFMLT, CFFLL, CFEMP
FESTS register	When the common FIFO buffer is in transmit mode : CFEMP
FFSTS register	When the common FIFO buffer is in transmit mode : CFFLL,
TMCm register	TMOM, TMTAR, TMTR
TMSTSm register	TMTARM, TMTRM, TMTRF[1:0], TMTSTS
TMTRSTS register	TMTRSTSm
TMTARSTS register	TMTARSTSm
TMTCSTS register	TMTCSTSm
TMTASTS register	TMTASTSm
THLCC register	THLE
THLSTS register	THLMC[3:0], THLIF, THLELT, THLFLL, THLEMP
GTINTSTS register	TSIF, TAIF, CFTIF, THIF
C0FDCTRH register	PNMDC[1:0]
C0FDSTSL register	TDCVF, PNSTS[1:0], SOCO, EOCO, TDCR[7:0]
C0FDSTSH register	SOC[7:0], EOC[7:0]
C0FDCRCL register	CRCREG[15:0]
C0FDCRCH register	SCNT[3:0], CRCREG[20:16]

Table 1.3 Registers to be Initialized Due to Transition Only to Global Reset Mode

Register	Bits/flags
GSTS register	GHLTSTS
GERFLL register	EEF, CMPOF, THLES, MES, DEF
GTSC register	TS[15:0]
RMND register	RMNSn
RFCCk register	RFE
RFSTSk register	RFMC[5:0], RFIF, RFMLT, RFFLL, RFEMP
CFCCL register	When the common FIFO buffer is in receive mode : CFE
CFSTS register	When the common FIFO buffer is in receive mode : CFMC[5:0], CFFLL, CFEMP, CFTXIF, CFRXIF, CFMLT
FESTS register	RFkEMP, When the common FIFO buffer is in receive mode : CFEMP
FFSTS register	RFkFLL, When the common FIFO buffer is in receive mode : CFFLL,
FMSTS register	CFMLT, RFkMLT
RFISTS register	RFkIF
GTSTCFG register	RTMPS[3:0]
GTSTCTR register	RTME

1.3 Communication Speed

Set the CAN communication speed. To determine the communication speed, the following settings are needed.

- Bit timing
- Communication speed calculation

1.3.1 Setting of CAN Bit Timing

In this CAN module, one bit of a communication frame consists of three segments: a synchronization segment (SS), a time segment 1 (TSEG1), and a time segment 2 (TSEG2).

Figure 1.8 shows the structure of the bit segments and a sample point.

The sample point is specified by both the time segment 1 (TSEG1) and time segment 2 (TSEG2). The sample timing can be determined by changing the values of the segments.

The smallest unit for the sample timing is one time quantum (Tq) that is obtained by a clock frequency input to the CAN module and a baud rate prescaler value.

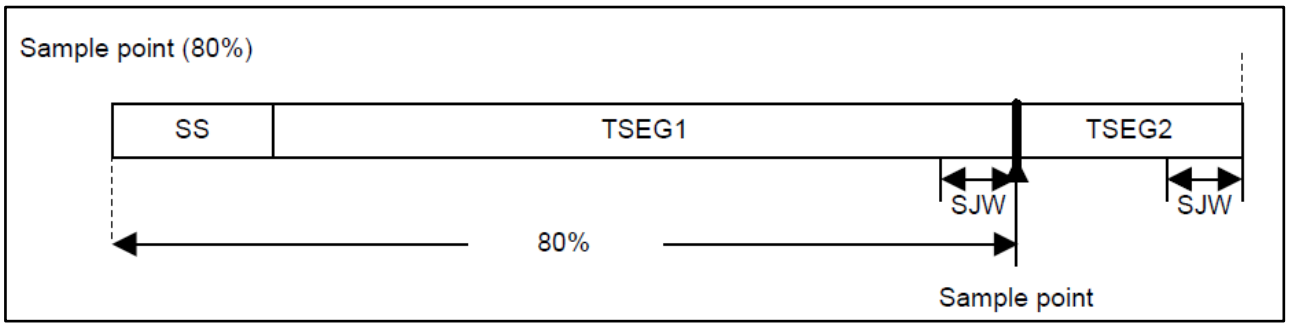


Figure 1.8 Structure of Bit Segments and Sample Point

- SS: Synchronization segment
  SS performs synchronization by monitoring an edge from a recessive bit to a dominant bit in the interframe space.

  The interframe space consists of Intermission, Suspend transmission, and Bus idle. During Bus Idle, all nodes can start transmission.
- TSEG1: Time segment 1
  TSEG1 absorbs the physical delay on the CAN Bus. The physical delay on the CAN Bus is twice the total of the following three delays: a delay on the CAN Bus, a delay in the input comparator, and a delay in the output driver.
- TSEG2: Time segment 2
  TSEG2 compensates for the phase error due to clock frequency errors.
- SJW: Resynchronization jump width
  SJW is a length to extend or reduce a time segment to compensate an error in phase due to the phase error.

(1) Conditions for setting bit timing

The following are the settings to each segment and the limitation.

The settings to each segment:

- Nominal bit rate

  SS=1 Tq fixed

  Set TSEG1 to a range of 2 Tq to 256 Tq.

  Set TSEG2 to a range of 2 Tq to 128 Tq.

  Set SJW to a range of 1 Tq to 128 Tq.

  Set “SS+TSEG1+TSEG2” to a range of 8 Tq to 385 Tq.

- Data bit rate

  SS=1 Tq fixed

  Set TSEG1 to a range of 2 Tq to 32 Tq.

  Set TSEG2 to a range of 2 Tq to 16 Tq.

  Set SJW to a range of 1 Tq to 16 Tq.

  Set “SS+TSEG1+TSEG2” to a range of 5 Tq to 49 Tq.

Limitation on TSEG1 and TSEG2:

- Nominal bit rate

  TSEG1 > TSEG2 ≥ SJW (However, when SJW=1, TSEG2 ≥ 2.)

- Data bit rate

  TSEG1 ≥ TSEG2 ≥ SJW (However, when SJW=1, TSEG2 ≥ 2.)

1.3.2 Communication Speed Calculation

The communication speed is determined by the CAN clock (f_{CAN}) which is a clock source for the CAN module, the baud rate prescaler value, and Tq count per bit time. Either one of the following clocks can be used as f_{CAN}: the CPU/peripheral hardware clock (the clock with less jitter) or the X1 clock. Regarding the f_{CAN} settings, see “1.4.5 CAN Clock Source Setting”.

Table 1.4 and Table 1.5 indicates a formula to calculate the communication speed and examples of communication speed. Table 1.6 lists the bit time settings.

Table 1.4 Communication Speed Calculation and Examples of Communication Speed (Nominal)

Formula for communication speed	**f_{CAN} / (Baud rate prescaler ratio [Note] x (Tq count of 1-bit time))**							
f_{CAN} / Communication speed	40MHz	32MHz	30MHz	24MHz	20MHz	16MHz	10MHz	8MHz
1Mbps	8Tq(5) 20Tq(2)	8Tq(4) 16Tq(2)	10Tq(3) 15Tq(2)	8Tq(3) 12Tq(2) 24Tq(1)	10Tq(2) 20Tq(1)	8Tq(2) 16Tq(1)	10Tq(1)	8Tq(1)
500Kbps	8Tq(10) 20Tq(4)	8Tq(8) 16Tq(4)	10Tq(6) 15Tq(4) 20Tq(3)	8Tq(6) 12Tq(4) 24Tq(2)	10Tq(4) 20Tq(2)	8Tq(4) 16Tq(2)	10Tq(2) 20Tq(1)	8Tq(2) 16Tq(1)
250Kbps	8Tq(20) 20Tq(8)	8Tq(16) 16Tq(8)	10Tq(12) 15Tq(8) 20Tq(6)	8Tq(12) 12Tq(8) 24Tq(4)	10Tq(8) 20Tq(4)	8Tq(8) 16Tq(4)	10Tq(4) 20Tq(2)	8Tq(4) 16Tq(2)
125Kbps	8Tq(40) 20Tq(16)	8Tq(32) 16Tq(16)	10Tq(24) 15Tq(16) 20Tq(12)	8Tq(24) 12Tq(16) 24Tq(8)	10Tq(16) 20Tq(8)	8Tq(16) 16Tq(8)	10Tq(8) 20Tq(4)	8Tq(8) 16Tq(4)
83.3Kbps	8Tq(60) 12Tq(40) 16Tq(30) 24Tq(20)	8Tq(48) 12Tq(32) 16Tq(24) 24Tq(16)	8Tq(45) 10Tq(36) 12Tq(30) 15Tq(24) 20Tq(18) 24Tq(15)	8Tq(36) 12Tq(24) 16Tq(18) 24Tq(12)	8Tq(30) 10Tq(24) 12Tq(20) 15Tq(16) 16Tq(15) 20Tq(12) 24Tq(10)	8Tq(24) 12Tq(16) 16Tq(12) 24Tq(8)	8Tq(15) 10Tq(12) 12Tq(10) 15Tq(8) 20Tq(6) 24Tq(5)	8Tq(12)
33.3Kbps	8Tq(150) 12Tq(100) 16Tq(75) 20Tq(60) 24Tq(50)	8Tq(120) 10Tq(96) 12Tq(80) 15Tq(64) 16Tq(60) 20Tq(48) 24Tq(40)	10Tq(90) 12Tq(75) 15Tq(60) 20Tq(45)	8Tq(90) 10Tq(72) 12Tq(60) 15Tq(48) 16Tq(45) 20Tq(36) 24Tq(30)	8Tq(75) 10Tq(60) 12Tq(50) 15Tq(40) 20Tq(30) 24Tq(25)	8Tq(60) 10Tq(48) 12Tq(40) 15Tq(32) 16Tq(30) 20Tq(24) 24Tq(20)	10Tq(30) 12Tq(25) 15Tq(20) 20Tq(15)	8Tq(30)

Note: Baud rate prescaler ratio = P+1 (P=0 to 1023)

P: the value set to the NBRP[9:0] bit in the C0NCFGL register

Remark: Figures in parentheses indicate baud rate prescaler values.

Table 1.5 Communication Speed Calculation and Examples of Communication Speed (Nominal and Data)

Formula for communication speed	f_{CAN} / Baud rate prescaler ratio Note x (Tq count of 1-bit time)	
f_{CAN} / Communication speed	40MHz	20MHz
Nominal : 1Mbps Data : 8Mbps	Nominal 40Tq(1) Data 5Tq(1)	Nominal 20Tq(1) Data Not possible
Nominal : 1Mbps Data : 5Mbps	Nominal 40Tq(1) Data 8Tq(1)	Nominal 20Tq(1) Data Not possible
Nominal : 500Kbps Data : 2Mbps	Nominal 80Tq(1) Data 20Tq(1)	Nominal 40Tq(1) Data 10Tq(1)

Note: Baud rate prescaler ratio = P+1

P: Nominal - the value set to the NBRP[9:0] bit in the C0NCFGL register (P=0 to 1023)

Data - the value set to the DBRP[7:0] bit in the C0DCFGL register (P=0 to 255)

Remark: Figures in parentheses indicate baud rate prescaler values.

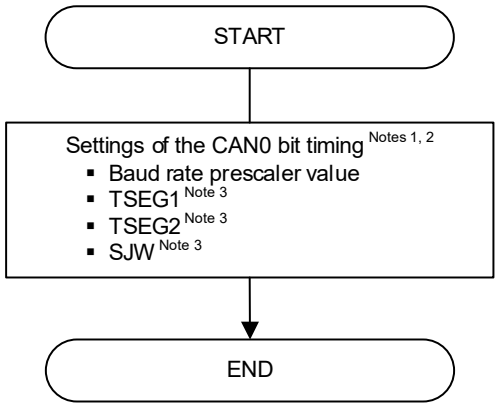
Table 1.6 Example of Bit Timing Settings

1 bit	Set value (Tq)				Sample point Note (%)
	SS	TSEG1	TSEG2	SJW	
5Tq	1	2	2	1	60.00
8Tq	1	4	3	1	62.50
	1	5	2	1	75.00
10Tq	1	6	3	1	70.00
	1	7	2	1	80.00
12Tq	1	8	3	1	75.00
	1	9	2	1	83.33
15Tq	1	10	4	1	73.33
	1	11	3	1	80.00
16Tq	1	10	5	1	68.75
	1	11	4	1	75.00
20Tq	1	12	7	1	65.00
	1	13	6	1	70.00
	1	15	4	3	80.00
24Tq	1	15	8	1	66.67
	1	16	7	1	70.83
50Tq	1	39	10	4	80.00

Note: A position determining a level of one bit

1.3.3 Procedure for Setting CAN Bit Timing and Communication Speed

Figure 1.9 shows the procedures for setting CAN bit timing and communication speed.
These settings need to be performed with CAN configuration.
Regarding the CAN configuration, see “1.1 CAN Configuration”.



Note 1: Modify the values of the C0CFGL and C0CFGH registers, C0DCFGL and C0DCFGH registers when the CAN module is in channel reset mode or channel halt mode. Also, these settings need to be done prior to a transition to channel operation mode.

Note 2: Set the bit timing for the nominal bit rate and data bit rate. However, when in Classical CAN only mode, you do not need to set the bit timing of the data bit rate.

Note 3: The following conditions need to be satisfied for the settings for TSEG1, TSEG2, and SJW.

Nominal bit rate :
TSEG1 > TSEG2 ≥ SJW (However, when SJW=1, TSEG2 ≥ 2.)

Data bit rate :
TSEG1 ≥ TSEG2 ≥ SJW (However, when SJW=1, TSEG2 ≥ 2.)

Figure 1.9 Procedures for Setting CAN Bit Timing and Communication Speed

1.4 Global Function

The following functions are set as a global function common to the entire CAN module (all channels).

- Transmit priority
- DLC check
- DLC replacement
- Mirror function
- CAN clock source
- CAN-FD message payload overflow
- Timestamp clock
- Interval timer prescaler
- RES bit protocol exception detection
- Timestamp capture

1.4.1 Setting of Transmit Priority

Set the transmit priority for the case in which a transmission request is issued from two or more transmit buffers of the same channel.

The transmit priority is common to the channel (all channels) and setting the priority for individual channel is unavailable. There are the following two options to determine the priority.

- ID priority
  A message is transmitted according to the priority of stored message IDs. The ID priority conforms to the CAN Bus arbitration rules specified in the CAN specifications.
  The targets for the priority determination are IDs of the messages stored in the transmit buffers and common FIFO buffer (transmit mode).
  With the common FIFO buffer, the oldest (stored earlier) messages in the common FIFO buffer are the targets for priority determination.
  When a message is being transmitted from the common FIFO buffer, the messages in the same common FIFO buffer that are to be transmitted next are the targets for the priority determination.
  When the same message ID is set to two or more buffers, the message in the transmit buffer having the minimum number among the messages will be transmitted first.

- Priority based on transmit buffer number
  The message in the transmit buffer of the minimum number among the transmit buffers having a transmit request is transmitted first.
  When the common FIFO buffer is linked to transmit buffers, the transmit priority is determined according to the buffer numbers of the transmit buffers.

  When messages are retransmitted as a result of arbitration lost or any error, transmit priority determination is made again regardless of the selected transmit priority method.

1.4.2 Setting of DLC Check Function

The setting of the DLC check function is described below.

When the DLC check function is enabled, DLC filter processing is carried out for the messages that have passed through the acceptance filter processing.

When the DLC check function is disabled, the DLC filter processing is not carried out.

When a DLC value of a received message is equal to or larger than the DLC value specified in the receive rule, the DLC filter processing will be carried out for the received message. Meanwhile, when a DLC value of a received message is smaller than the DLC value specified in the receive rule, the DLC filter processing will not be carried out for the received message. In this case, the message will not be stored in the receive buffer or common FIFO buffer, which means a DLC error has occurred.

For detailed information on the receive rules, see “1.5 Receive Rule Table”.

1.4.3 Setting of DLC Replacement Function

The setting of the DLC replacement function is described below.

The DLC replacement function is enabled only when the DLC check function is enabled.

When the DLC filter processing is carried out for a message while the DLC replacement function is enabled, the DLC value specified in the receive rule is stored in the buffer instead of the DLC value of the received message. In this case, H’00 is stored in data bytes that exceed the DLC value in the receive rule.

When the DLC replacement function is disabled, all data bytes of the received message are stored in the buffer.

For detailed information on the receive rules, see “1.5 Receive Rule Table”.

Table 1.7 DLC Filtering and DLC Replacement Functions

GCFGL register		DLC of a received message /DLC of a receive rule	Received message	
DCE bit	**DRE bit**		**Processing**	**DLC to be stored**
0 (DLC check disabled)	0 (DLC replacement disabled)	DLC of a received message < DLC of a receive rule	Stored in the buffer. Note 1	DLC of a received message
		DLC of a received message ≥ DLC of a receive rule		
		DLC of receive rules = 0		
	1 (DLC replacement enabled)	DLC of a received message < DLC of a receive rule		
		DLC of a received message ≥ DLC of a receive rule		
		Receive rule DLC=0		
1 (DLC check enabled)	0 (DLC replacement disabled)	DLC of a received message < DLC of a receive rule	Discarded (DLC error)	-
		DLC of a received message ≥ DLC of a receive rule	Stored in the buffer.	DLC of a received message
		Receive rule DLC=0	Stored in the buffer.	DLC of a received message
	1 (DLC replacement enabled)	DLC of a received message < DLC of a receive rule	Discarded (DLC error)	-
		DLC of a received message ≥ DLC of a receive rule	Stored in the buffer.	DLC of a received rule Note 2
		Receive rule DLC=0	Stored in the buffer.	DLC of a received message

Note 1: DLC check itself will not be carried out.
Note 2: A value of H'00 will be stored in data bytes exceeding the DLC value specified in the receive rule.

1.4.4 Setting of Mirror Function

The setting of the mirror function is described below.

When the mirror function is enabled, a CAN node can receive a message transmitted from the transmitting node itself (the same node).

When receiving a message transmitted from another (a different) CAN node while the mirror function is enabled, receive rules in which the mirror function is disabled are used for processing the received message.

When receiving a message transmitted from the transmitting node itself, receive rules in which the mirror function is enabled are used for processing the received message.

For detailed information on the receive rules, see "1.5 Receive Rule Table".

Table 1.8 Messages Target for Data Processing Based on the Mirror Function

MME bit in the GCFGL register	GAFLLB bit in the GAFLIDiH register	Messages target for data processing according to the receive rule
0(Mirror function disabled)	0	Messages transmitted from other (different) CAN nodes
	1	No target message
1(Mirror function enable)	0	Messages transmitted from other (different) CAN nodes
	1	Messages transmitted from the transmitting CAN node

1.4.5 CAN Clock Source Setting

The setting of the CAN clock (f_{CAN}) Notes 1, 2, 3 as a clock source is described below. The following clocks can be used as a CAN clock source. The maximum usable value for f_{CAN} is 40MHz.

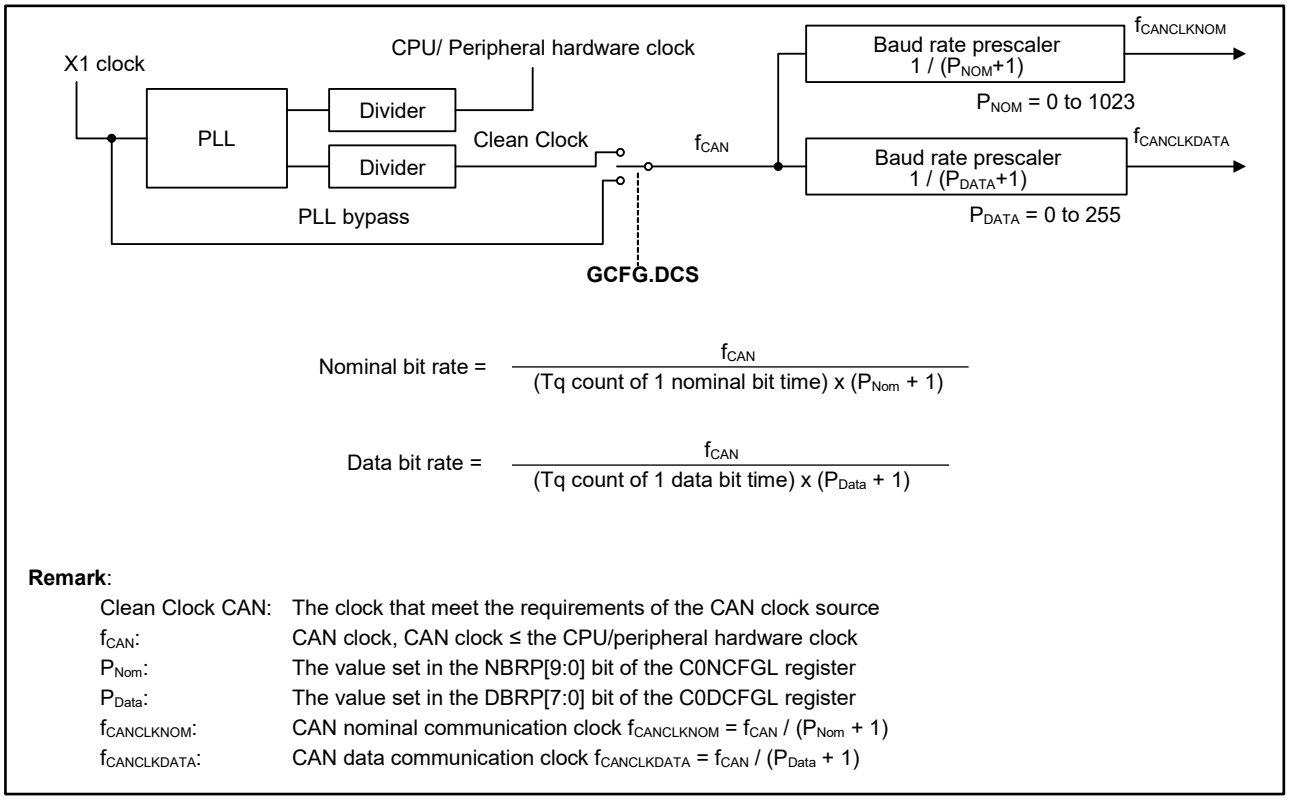
- The CPU/peripheral hardware clock
- X1 clock (fx)

Note 1: Regardless of which of the above choices is chosen as the CAN clock source, the CPU/peripheral hardware clock should be 1/2 of the Main System/PLL select clock (f_{MP}).

Note 2: Because the oscillation frequency accuracy of a high-speed on-chip oscillator (f_{IH}) does not meet the requirements of a CAN clock source, choose an X1 clock when using f_{IH} for the CPU/peripheral hardware clock (including a f_{IH} divider and a PLL clock sourced from f_{IH}).

Note 3: When using the X1 clcok as a clock source, make the X1 clock into the value less than or equal to the CPU/peripheral hardware clock (f_{CAN}).

Figure 1.10 illustrates the CAN system clock generator.



Remark:

Clean Clock CAN:	The clock that meet the requirements of the CAN clock source
f_{CAN}:	CAN clock, CAN clock ≤ the CPU/peripheral hardware clock
P_{Nom}:	The value set in the NBRP[9:0] bit of the C0NCFGL register
P_{Data}:	The value set in the DBRP[7:0] bit of the C0DCFGL register
$f_{CANCLKNOM}$:	CAN nominal communication clock $f_{CANCLKNOM} = f_{CAN} / (P_{Nom} + 1)$
$f_{CANCLKDATA}$:	CAN data communication clock $f_{CANCLKDATA} = f_{CAN} / (P_{Data} + 1)$

Figure 1.10 CAN clock generator

1.4.6 Setting CAN-FD Message Payload Overflow

The setting of operation when the payload length of a received message exceeds the payload size of the destination buffer are described below. The operation can be selected from the following:

- Received messages that overflow the payload are not stored in the buffer. (discarded)
- Received messages with payloads overflowing are stored in a buffer.
  At this time, the payload that exceeds the payload size of the buffer is truncated.

When a received message that overflows payload is stored in a buffer, the stored DLC stores the inbound DLC value or the DLC value of the receive rule in the buffer according to the DLC replacement function setting.

For information about setting the payload size for each buffer, see:

- 1.6.1 Setting of Receive Buffers (2) Setting of the size of payload
- 1.6.2 Setting of Receive FIFO Buffers (2) Setting of the size of payload
- 1.6.3 Setting of Common FIFO Buffer (2) Setting of the size of payload

1.4.7 Setting of Timestamp Clock

The settings of the clock source and division ratios used for the timestamp counter are described below.

The timestamp counter is a 16-bit free-running counter used for recording message receiving time and completion time of message transmission. The value of the timestamp counter is fetched at the start-of-frame (SOF) [Note] timing of a received message and then stored in a receive buffer or a FIFO buffer together with the message ID and its data. In addition, it is ingested at the SOF timing of a transmitted message and stored in the transmit history list buffer. The ingestion timing can be changed. For details, see “1.4.10 Setting of Timestamp Capture”.

The clock used for the timestamp counter can be selected from the following:

- The CPU/peripheral hardware clock
- CAN0 nominal bit time clock (Only when CAN0 is in classical CAN only mode)

Note: SOF: A field indicating a start of a frame

When the CAN0 nominal bit time clock is used as a clock source, the timestamp counter stops when the corresponding channel transitions to channel reset mode or channel halt mode. When the CPU/peripheral hardware clock is used as a clock source, the timestamp function (counter) is not affected by channel modes.

Figure 1.11 is a block diagram of the timestamp function.

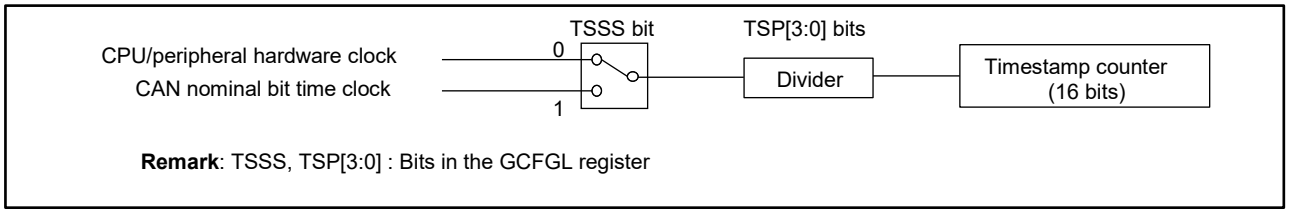


Figure 1.11 Timestamp Function

1.4.8 Interval Timer Prescaler Setting

Set the divider ratio when using the CPU/Peripheral Hardware Clock (f_{CLK}) divider clock as the count source. The interval timer operates with the interval transmission function of the common FIFO buffer.

When the CFTISS bit in the CFCCL is set to 0, f_{CLK} (the clock obtained by frequency-dividing f_{CLK}) is the clock source for the interval timer. f_{CLK} is divided by the ITRCP[15:0] bit in the GCFGH register, and is further divided by 10 if the CFITR bit in the CFCCL register is set to 1.

For detailed information on the interval timer function, see “1.6.3 (5) Setting of interval timer”.

1.4.9 Setting of RES Bit Protocol Exception Event Detection

Set the permission/prohibition of RES bit protocol exception event detection.

If the RES bit protocol exception event detection is set to permission, the RES bit of the CAN-FD frame is treated as protocol exception event detection when it is detected as recessive. No error is detected.

If the RES bit protocol exception event detection is set to prohibition, and the RES bit of the CAN-FD frame is detected as receive, it is considered a form error and an error frame is outputted.

1.4.10 Setting of Timestamp Capture

Set when timestamp values are captured. The capture can be selected from the following:

- Timestamp capture at the sample point of SOF bit (start of frame)
- Timestamp capture at frame valid indication
- Timestamp capture at the sample point of RES bit (SOF bit for Classical CAN frame)

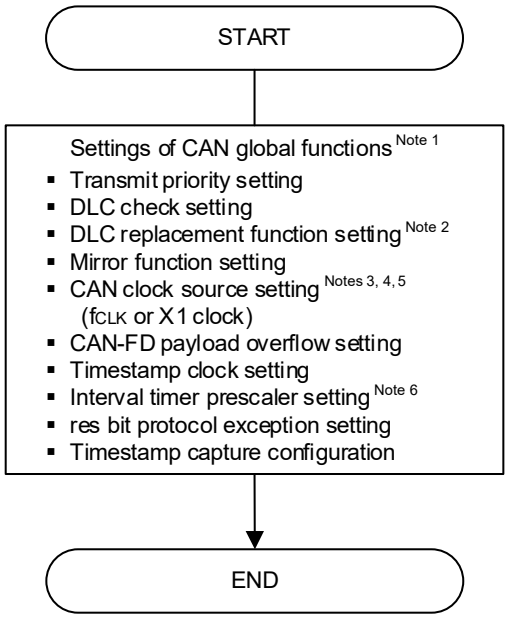
For detailed information on the timestamp function, see “1.4.7 Setting of Timestamp Clock”.

1.4.11 Setting of Global Functions

Figure 1.12 shows the procedures for setting the global functions.

The settings below need to be performed with the CAN configuration.

For detailed information on the CAN configuration procedure, see “1.1 CAN Configuration”.



Note 1: Modify the values of the GCFGL, GCFGH and GFDCFG registers when the CAN module is in global reset mode.

Note 2: Before disabling the DLC check function (set the DCE bit in the GCFGL register to 0), disable the DLC of the receive rule table (set the GAFLDLC[3:0] bits in the GAFLP0iL register to B’0000).

Note 3: Regardless of the choice of the CAN clock source, f_{CLK} should be set to $f_{MP}/2$.

Note 4: Because the oscillation frequency accuracy of the high-speed on-chip oscillator (f_{IH}) does not meet the requirements of the CAN clock source, choose an X1 clock when using f_{IH} for f_{CLK} (including the PLL clock with the f_{IH} divider and f_{IH} as the clock source).

Note 5: If you select the X1 clock, make sure that the X1 clock is less than or equal to f_{CLK}.

Note 6: When using the interval timer, do not set H'0000 to a prescaler of the interval timer (the ITRCP[15:0] bits in the GCFGH register).

Figure 1.12 Setting procedures for global function

1.5 Receive Rule Table

To filter the received messages, set the receive rule table.

With the data processing according to the receive rule table, the filtered messages are stored in the specified buffers. The data processing includes:

– acceptance filter processing
  – DLC filter processing
  – routing processing
  – label addition
  – mirror function

The following need to be specified in the receive rules. However, receive rules are common to CAN-FD frames and Classical CAN frames, and there is no CAN-FD format setting.

- number of receive rules,
- IDE bit, RTR (RRS) bit and IDs,
- messages target for the receive rules,
- IDE mask, RTR (RRS) mask, and ID mask,
- DLC check function,
- receive rule labels, and
- buffers to store messages

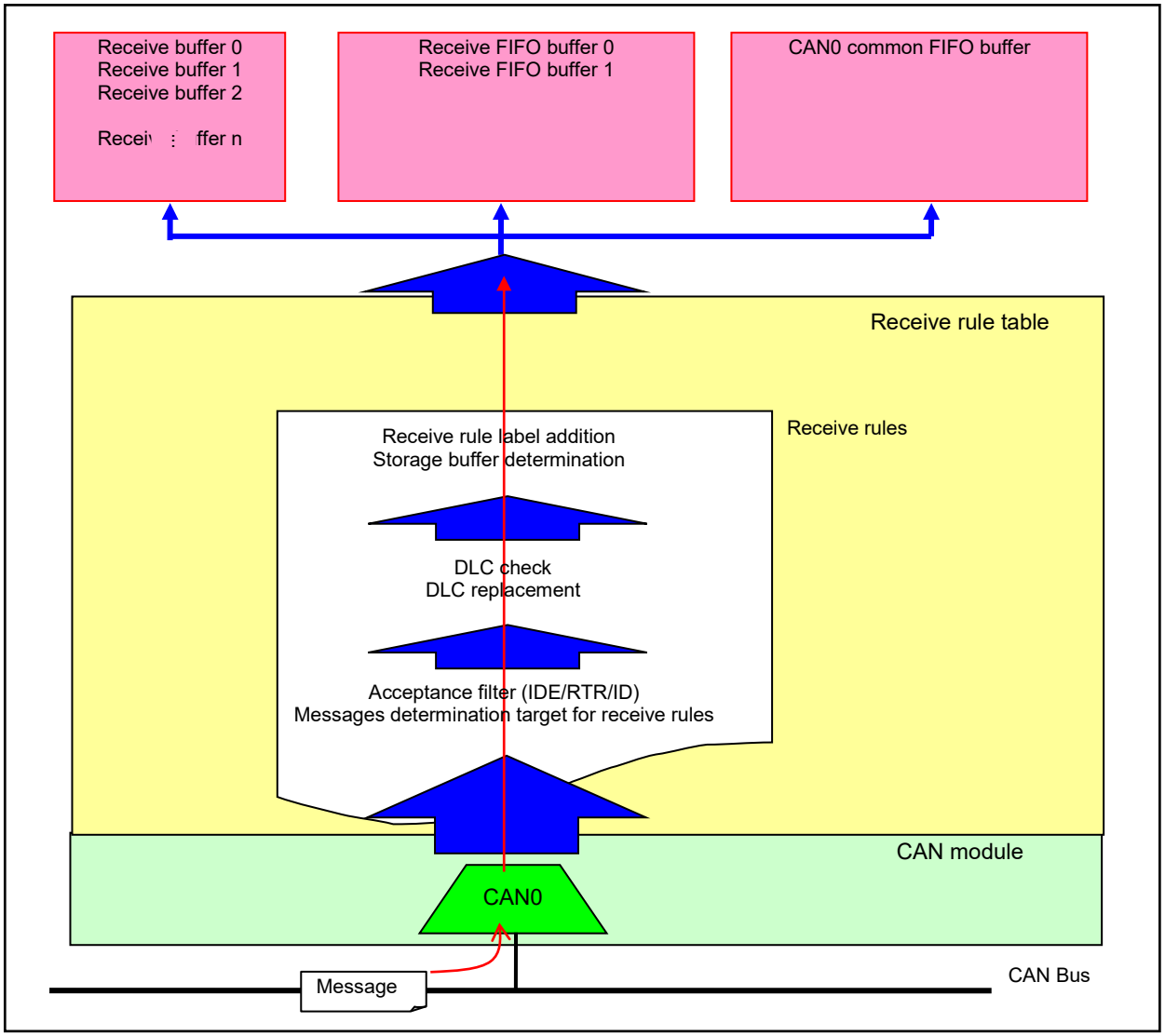


Figure 1.13 Filtering Based on Receive Rule Table

1.5.1 Setting of the Number of Receive Rules

Set the number of receive rules used for the (each) channel.

The number of receive rules for the entire CAN module is 16 in total.

The check begins with the receive rule with the smallest rule number and the processing is performed in ascending order. When the bits of the received message to be compared match the bits specified in the receive rule or when the comparison with the receive rules are completed without any match, the filter processing stops. If there is no matching receive rule, the received message is not stored in the receive buffer or FIFO buffer.

The following is the limitation on the receive rules.

- Limitaion

The number of CAN0 receive rules that can be registered ≤ 16

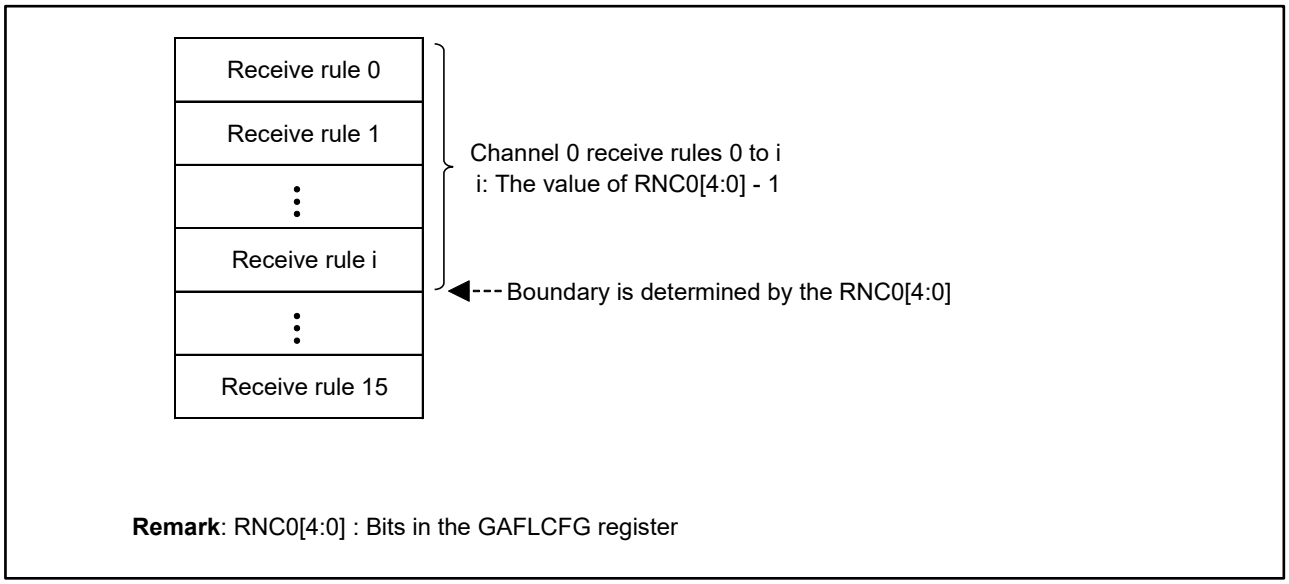


Figure 1.14 Registration of Receive Rules

1.5.2 Settings of IDE/RTR(RRS)/ID

Set the ID format (standard ID or extended ID), a frame format (data frame or remote frame), and a receive ID for a received message. The CAN-FD frame format does not define a remote frame. If set to receive a remote frame (a frame with the RTR bit value of 1), a frame with the RRS bit value of 1 is received.

1.5.3 Setting of Messages Target for Receive Rules

When the target is the message transmitted from a different (another) CAN node (set the GAFLLB bit in the GAFLIDiH register to 0), data processing according to the receive rules is carried out for the message transmitted from the different (another) CAN node.

When the target is the message transmitted from the same CAN node (the transmitting node itself) (set the GAFLLB bit in the GAFLIDiH register to 1) and the mirror function is enabled, data processing according to the receive rules is carried out for the message transmitted from the same CAN node (the transmitting node itself).

For details on the mirror function, see “1.4.4 Setting of Mirror Function”.

1.5.4 Settings to Mask IDE/RTR(RRS)/ID

Set values to mask the values set by the IDE and RTR (RRS) bits and ID data.

With this setting, the acceptance filtering is enabled for the bits that have not been masked with the IDE, RTR (RRS) and ID masks.

1.5.5 Setting of Values to be Compared with DLC Values

Specify the DLC values in the receive rule which are compared with the DLC values of messages received when the DLC check is enabled.

For detailed information on the DLC check, see “1.4.2 Setting of DLC Check Function”.

1.5.6 Setting of Receive Rule Label

Set 16-bit label and 2-bit label to be attached to a message that has passed through the DLC filter. The labels can be attached when the message is stored in a buffer.

The labels can be arbitrarily set. Also, the label of a received message can be freely used with a program. For example, if a channel number that a message is to be received is specified to the labels, it becomes possible to identify the channel (only one channel) that has received the same ID message stored in a receive FIFO buffer.

1.5.7 Setting of Buffers to Store Messages

Set the buffers to store the messages which have passed through the DLC filter.

The buffers below can be selected as a message storage buffer.

- Receive buffer n (Only one buffer can be selected for one receive rule.)
- Receive FIFO buffer k
- Common FIFO buffer (set to receive mode)

For one receive rule, a maximum of two buffers can be selected as a message storage buffer. However, the number of receive buffers that can be selected as a storage buffer is only one. That is, it is impossible to store messages in two receive buffers 0 and 1.

Combination example of message storage buffers

- Maximum of two buffers = one receive FIFO buffer k plus one receive buffer n
- Maximum of two buffers = one receive FIFO buffer k plus one common FIFO buffer

Possible/impossible settings

Possible: Storing messages in receive buffer 0 and receive FIFO buffer 0

Impossible: Storing messages in receive buffer 0 and receive buffer 1

Note: Storing messages in two receive buffers is impossible.

1.5.8 Application Examples of Receive Rules

The following are application examples of the receive rules.

- **Example 1**
  To receive the messages indicated in the table below, each register needs to be set as follows:
  - ID format : standard ID
  - Message format : data frame
  - Mirror function : reception of messages transmitted from a different (another) CAN node
  - Receive ID : 120h,121h,122h, 123h
  - DLC : a DLC value of a received message ≥ 6
  - Label : 16-bit 0010h, 12-bit 01b
  - Storage buffer : receive buffer 3, receive FIFO buffers 1

		GAFLIDE/GAFLIDEM	GAFLRTR/GAFLRTRM	GAFLLB	GAFLID/GAFLIDM			
GAFLIDiL, GAFLIDiH		0	0	0	B’00000	B’00000000	B’00000001	B’00100000
GAFLMiL, GAFLMiH		1	1	-	B’00000	B’00000000	B’00000111	B’11111100
Messages that can be received	**H’120**	0	0	0	B’-----	B’--------	B’-----001	B’00100000
	H’121				B’-----	B’--------	B’-----001	B’00100001
	H’122				B’-----	B’--------	B’-----001	B’00100010
	H’123				B’-----	B’--------	B’-----001	B’00100011

	GAFLDLC	GAFLPTR	GAFLIFL0	GAFLIFL1	GAFLRMV	GAFLRMDP	GPFLFDP0	GPFLFDP1	GAFLFDP8
GAFLMiH	-	-	-	0	-	-	-	-	-
GAFLP0iL, GAFLP0iH	6	H’0010	1	-	1	3	-	-	-
GAFLP1iL	-	-	-	-	-	-	0	1	0

- **Example 2**

To receive the messages indicated in the table below, each register needs to be set as follows:

- ID format : standard ID
- Message format : remote frame [Note], data frame
- Mirror function : Reception of a message transmitted from a different (another) CAN node.
- Receive ID : 130h
- DLC : The DLC check function is not used.
- Label : 16-bit 0130h, 2-bit 10b
- Storage buffer : receive FIFO buffer 0, common FIFO buffer 0

		GAFLIDE/GAFLIDEM	GAFLRTR/GAFLRTRM	GAFLLB	GAFLID/GAFLIDM			
GAFLIDiL, GAFLIDiH		0	0	0	B’00000	B’00000000	B’00000001	B’00110000
GAFLMiL, GAFLMiH		1	0	-	B’00000	B’00000000	B’00000111	B’11111111
Messages that can be received	**H’130 (Data)**	0	0	0	B’-----	B’--------	B’-----001	B’00110000
	H’130 (Rmt)	0	1	0	B’-----	B’--------	B’-----001	B’00110000

	GAFLDLC	GAFLPTR	GAFLIFL0	GAFLIFL1	GAFLRMV	GAFLRMDP	GPFLFDP0	GPFLFDP1	GAFLFDP8
GAFLMiH	-	-	-	1	-	-	-	-	-
GAFLP0iL, GAFLP0iH	0	H’0130	0	-	0	0	-	-	-
GAFLP1iL	-	-	-	-	-	-	1	0	1

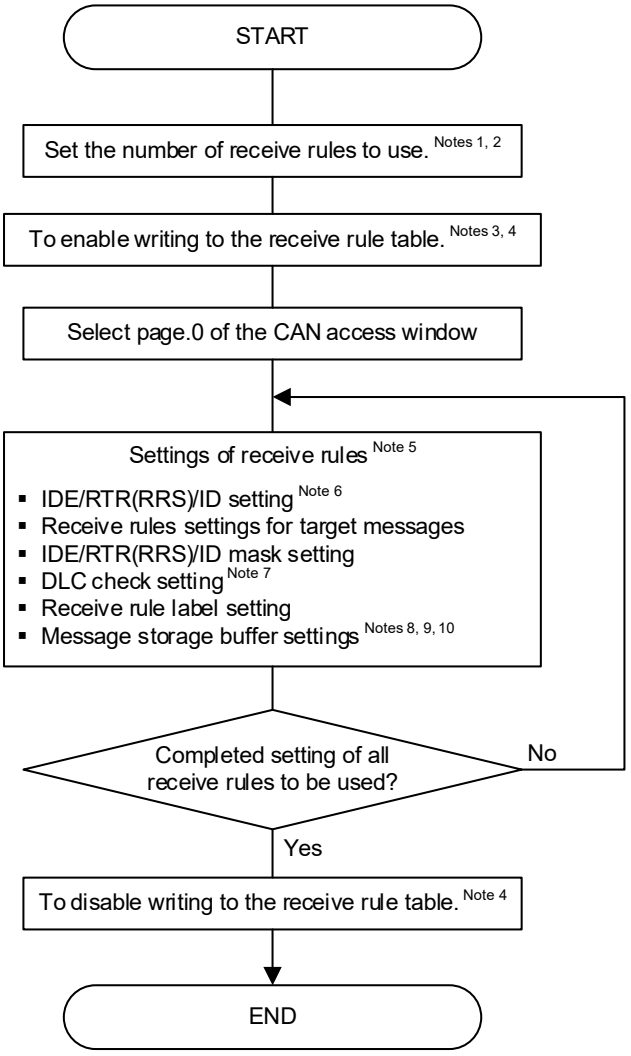
Note: Because remote frame is not defined, CAN-FD frame is set to receive a frame with an RRS bit value of 1.

1.5.9 Procedures for Setting Receive Rule Table

Figure 1.15 shows the procedures for setting the receive rule table.

These settings need to be performed with the CAN configuration.

For details on the CAN configuration procedure, see “1.1 CAN Configuration”.



Note 1: Modify the number of receive rules (the RNC[4:0] bits in the GAFLCFG register) when the CAN module is in global reset mode.

Note 2: Set the number of receive rules (RNC[4:0] bits) used for the (each) channel must be 16 or ress.

Note 3: To write to the receive rule table needs to be permited (set the AFLDAE bit in the GAFLECTR register to 1).

Note 4: After writing to the receive rule table is completed, set write to the receive rules table to prohibit (set the AFLADE bit in the GAFLECTR register to 0).

Note 5: Modify the receive rules (the GAFLIDiL/ register, GAFLMiL/H register, GAFLP0iL/H register, and GAFLP1iL register) when all of tha following conditions are met.

*Set write to the receive rules table to permit (the AFLDAE bit in the GAFLECTR register is 1).

*Select page 0 of access window for the CAN module (the RPAGE[1:0] bits in the CFDGRWC register are B’00).

*Set channel reset mode or channel halt mode.

Note 6: When the standard ID is selected, set the value of the standard ID to bits 10-0 of the ID data (the GAFLID[15:0] bits in the GAFLIDiL register) and also set 0 to bits 15-11 in the GAFLIDiL register and the GAFLID[28:16] bits in the GAFLIDiH register.

Figure 1.15 Setting Procedures for Receive Rule Table (1/2)

Note 7: This setting is valid only when the DLC check is enabled (set the DCE bit in the GCFGL register to 1).
Note 8: A maximum of two FIFO buffers can be selected. However, when storing a message in one receive buffer (set the GAFLRMV bit in the GAFLP0iL register to 1), the number of FIFO buffers that can be selected is only one.
Note 9: Select only one receive FIFO buffer and one common FIFO buffer which is set to receive mode.
Note 10: When selecting a receive buffer as a storage buffer, enable the receive buffer (set the GAFLRMV bit to 1) and set a buffer number which is smaller than the number of receive buffers to be used (the NRXMB[4:0] bits in the RMNB register).

Figure 1.15 Setting Procedures for Receive Rule Table (2/2)

1.6 Buffers and FIFO Buffers

The following buffers need to be set for message transmission/reception:

- receive buffer
- Receive FIFO buffer
- Common FIFO buffer
- Transmit buffer
- Transmit history list buffer

The below is the limitation on the receive buffer, receive FIFO buffer, and common FIFO buffer.

- the number of receive buffers
- the number of buffers in receive FIFO buffer 0
- the number of buffers in receive FIFO buffer 1
- the number of buffers in common FIFO buffer

  (the sum of the buffers above) ≤ 16 buffers

The maximum CAN RAM available for receive and FIFO buffers is 1216 bytes. This is the size of all 16 buffers, which are limited to the number of buffers, and can store a 64-byte payload. Therefore, there are no payload size the limitation associated with the CAN RAM size.

Figure 1.16 illustrates the buffer configuration.

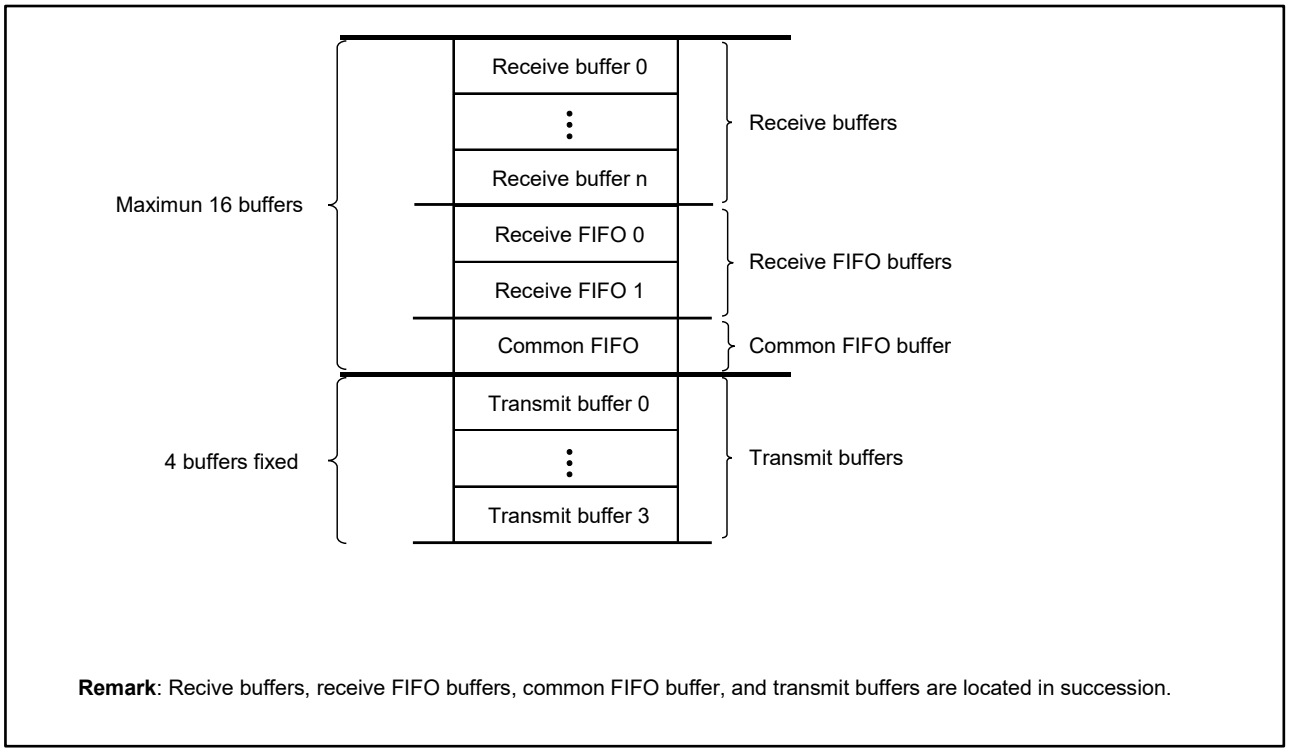


Figure 1.16 Buffer Configuration

1.6.1 Setting of Receive Buffers

The following need to be set to use the receive buffer.

- The number of buffers
- The size of payload
- Enable/disable interrupts

(1) Setting of the number of buffers

Specify the number of buffers to be allocated as a receive buffer. The number of buffers that can be allocated as a receive buffer is a range of 0 to 16. If the set number is 0, no receive buffer can be used.

(2) Setting of the size of payload

Specify the maximum of the payload size that can be stored in the receive buffer.

The maximum of the payload size can be selected from 8, 12, 16, 20, 24, 32, 48, and 64 bytes.

(3) Enable/disable interrupts

Enable/disable each receive buffer interrupt.

1.6.2 Setting of Receive FIFO Buffers

The following need to be set to use the receive FIFO buffer.

- The number of buffers
- Enable/disable interrupts and set interrupt sources.

(1) Setting of the number of buffers

Specify the number of buffers to be allocated as a receive FIFO buffer.

The CAN module has two receive FIFO buffers and a maximum of 16 buffers can be allocated to each of the receive FIFO buffers. However, the number of receive FIFO buffers which are allocated as a receive FIFO buffer can be selected from among 0 Note, 4, 8 and 16.

Note: If no receive FIFO buffer is used, set the number of receive FIFO buffers to 0 (write B'000 to the RFDC[2:0] bits in the RFCCk register).

(2) Setting of the size of payload

Specify the maximum of the payload size that can be stored in the receive FIFO buffer.

The maximum of the payload size can be selected from 8, 12, 16, 20, 24, 32, 48, and 64 bytes.

(3) Enable/disable interrupts and set interrupt sources

Enable/disable the receive FIFO interrupt and set its interrupt sources. The sources for the receive FIFO interrupt can be selected from among the following.

- A receive FIFO interrupt will be generated (the RFIM bit in the RFCCk register is set to 0) when the conditions set by the RFIGCV[2:0] bits in the RFCCk register is met:
  - RFIGCV bit = B'000: the receive FIFO buffer is 1/8 full Note
  - RFIGCV bit = B'001: the receive FIFO buffer is 2/8 full
  - RFIGCV bit = B'010: the receive FIFO buffer is 3/8 full Note
  - RFIGCV bit = B'011: the receive FIFO buffer is 4/8 full
  - RFIGCV bit = B'100: the receive FIFO buffer is 5/8 full Note
  - RFIGCV bit = B'101: the receive FIFO buffer is 6/8 full
  - RFIGCV bit = B'110: the receive FIFO buffer is 7/8 full Note
  - RFIGCV bit = B'111: the receive FIFO buffer is full
- Every time reception of one message is completed, a receive FIFO interrupt will be generated (the RFIM bit in the RFCCk register is set to 1).

Note: When the number of receive FIFO buffers is set to 4 (the value of the RFDC[2:0] bits is B'001), do not perform this setting.

1.6.3 Setting of Common FIFO Buffer

The following need to be set to use the common FIFO buffer.

- The number of buffers
- Enable/disable interrupts and set interrupt sources.
- Modes of the common FIFO buffer
- Interval timer (transmit mode)
- Transmit buffer link (transmit mode)

(1) Setting of the number of buffers

Set the number of common FIFO buffers.

One channel has one common FIFO buffer. A maximum of 16 buffers can be allocated to a common FIFO buffer. The number of buffers to be allocated can be selected from among 0 Note, 4, 8 and 16.

Note: If no common FIFO buffer is used, set the number of common FIFO buffers to 0 (write B'000 to the CFDC [2:0] bits in the CFCCH register).

(2) Setting of the size of payload

Specify the maximum of the payload size that can be stored in the common FIFO buffer.

The maximum of the payload size can be selected from 8, 12, 16, 20, 24, 32, 48, and 64 bytes.

(3) Enable/disable interrupts and set interrupt sources

Enable/disable interrupts for each common FIFO buffer, and set the sources for the interrupts. Table 1.9 lists the interrupt sources that can be set for each mode of the common FIFO buffer.

Table 1.9 Common FIFO Buffer Interrupt Sources

Mode of common FIFO	CFIM bit in the CFCCLk register	Interrupt sources
Receive mode	0	When the number of received messages amounts to the number specified by setting the CFIGCV[2:0] bits in the CFCCLk register, a common FIFO receive interrupt request will be generated. Values set to the CFIGCV[2:0] bits: • B'000: the common FIFO buffer is 1/8 full Note • B'001: the common FIFO buffer is 2/8 full • B'010: the common FIFO buffer is 3/8 full Note • B'011: the common FIFO buffer is 4/8 full • B'100: the common FIFO buffer is 5/8 full Note • B'101: the common FIFO buffer is 6/8 full • B'110: the common FIFO buffer is 7/8 full Note • B'111: the common FIFO buffer is full
	1	Every time one message reception is completed, a common FIFO receive interrupt request is generated.
Transmit mode	0	When the buffer becomes empty (contains no message) upon completion of message transmission, a common FIFO transmit completion interrupt request is generated.
	1	Every time one message transmission is completed, a common FIFO transmit interrupt request is generated.

Note: When the number of common FIFO buffers are set to 4 (the value of the CFDC[2:0] bits is B'001), do not perform this setting.

The common FIFO transmit interrupt triggers the CAN0 transmit interrupt. The following are the sources for the CAN0 transmit interrupt.

- CAN0 transmit complete interrupt
- CAN0 transmit abort interrupt
- CAN0 common FIFO transmit complete interrutp
- CAN0 transmit history interrupt

(4) Mode setting for common FIFO buffer

As a mode of the common FIFO buffer, receive or transmit mode can be selected.

- In receive mode, the buffer serves as a receive FIFO buffer.
- In transmit mode, the buffer serves as a transmit FIFO buffer.

(5) Setting of interval timer

Set the count sources and transmission intervals for the interval timer. The interval timer is enabled in transmit mode.

For the count sources for the interval timer and formulas to calculate the interval time, see Table 3.1 in “3.3.3 Interval Transmission Function”.

(6) Setting of transmit buffer link

Link the common FIFO buffer to a transmit buffer. This linking is enabled only in transmit mode.

1.6.4 Setting of Transmit Buffers

Enable/disable the transmit complete interrupt for each transmit buffer.

One channel has four transmit buffers that can be simply used as a transmit buffer or that can be linked to a common FIFO buffer (set to transmit mode).

When the transmit buffer is used to be linked to the common FIFO buffer (set to transmit mode), write H'00 to the corresponding TMCm register. Also, set the TMIEm bit in the corresponding TMIEC register to 0 (interrupt disabled).

The transmit complete interrupt triggers the CAN0 transmit interrupt. The following are the sources for the CAN0 transmit interrupt.

- CAN0 transmit complete interrupt
- CAN0 transmit abort interrupt
- CAN0 common FIFO transmit complete interrupt
- CAN0 transmit history interrupt

1.6.5 Setting of Transmit History List Buffers

The settings for transmit history list buffers are described below.

Each channel has a single transmit history list buffer that can contain history data of eight transmissions.

- Set the buffers that store transmission data.
- Enable/disable the interrupts and set the interrupt sources.

(1) Setting of storage buffers

Specify the transmit buffer whose transmit history data will be stored in a transmit history list buffer. The buffer to store the history data can be selected from among the following.

Whether to store message transmission history data can be set for each message transmission.

- Common FIFO buffers
- Transmit buffers, common FIFO buffers

(2) Enable disable interrupts and set interrupt sources

Enable/disable the transmit history interrupts and set interrupt sources. The transmit history interrupt is generated by the following conditions.

- When history data of six transmissions have been stored in the transmit history list buffer.
- Every time history data of one transmission have been stored.

The transmit history interrupt triggers generation of the CAN0 transmit interrupt. The CAN0 transmit interrupt is generated by the following:

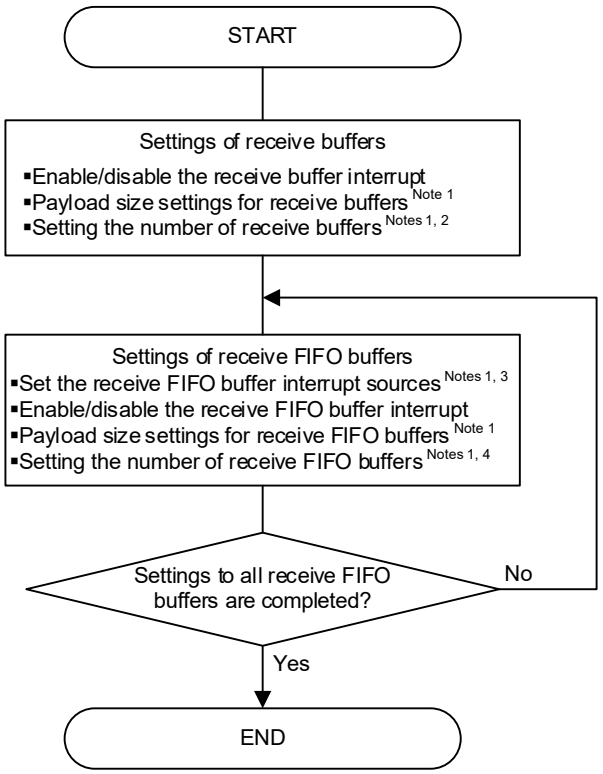
- CAN0 transmit complete interrupt
- CAN0 transmit abort interrupt
- CAN0 common FIFO transmit complete interrupt
- CAN0 transmit history interrupt

1.6.6 Procedures for Setting Buffers

Figure 1.17 shows the procedures for setting the receive buffer and the receive FIFO buffer. Figure 1.18 shows the procedures for setting the common FIFO buffer, the transmit buffer and the transmit history list buffer.

These settings need to be performed with the CAN configuration.

For details on the CAN configuration procedure, see “1.1 CAN Configuration”.



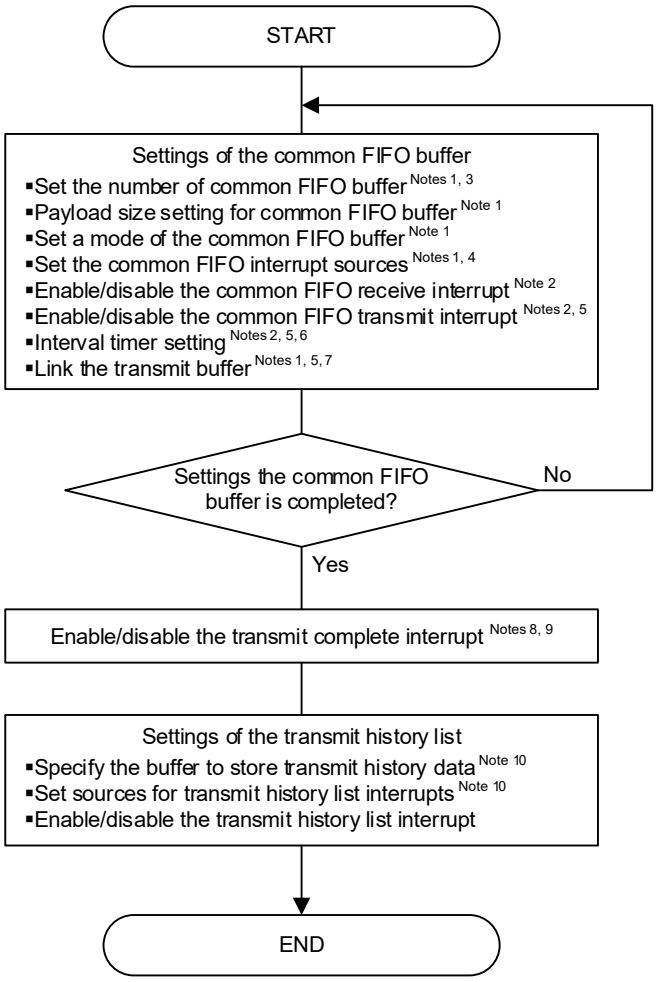
Note 1: Modify the values of the RMNB register and the values of the RFIGCV[2:0] bits, the RFIM bit, the RFDC[2:0] bits and RFPLS[2:0] bits in the RFCCk register when the CAN module is in global reset mode.

Note 2: Set the number of receive buffers within a range of 0 to 16. Note that when the number (the NRXMB[4:0] bits in the RMNB register) is set to 0, no receive buffer can be used.

Note 3: When the number of receive FIFO buffers is set to 4 (set the RFDC[2:0] bits to B’001), write B’001, B’011, B’101, or B’111 to the RFIGCV[2:0] bits.

Note 4: When the number of receive FIFO buffers is set to 0 (the RFDC[2:0] bits is set to B’000), do not use the receive FIFO buffers.

Figure 1.17 Procedures for Setting Receive Buffers and Receive FIFO Buffers



Note 1: Modify the values of the CFPLS[2:0] bits, the CFM bit, the CFIM bit, the CFIGCV[2:0] bits, the CFTML[1:0] bits and CFDC[2:0] bits in the CFCCL and CFCCH registers when the CAN module is in global reset mode.

Note 2: Modify the values of the CFITSS bit, the CFITR bit and the CFITT[7:0] bits in the CFCCL and CFCCH registers while the common FIFO buffer is not in use (when the CFE bit in the CFCCL register is set to 0).

Note 3: When the common FIFO buffer is not used, set the number of messages to be received to 0 (write B'000 to the CFDC[2:0] bits).

Note 4: When the number of common FIFO buffers is set to 4 (a value of B'001 is written to the CFDC[2:0] bits), write B'001, B'011, B'101, or B'111 to the CFIGCV[2:0] bits.

Note 5: This is only valid for the common FIFO buffer that is set to transmit mode.

Note 6: When the interval timer is not used, specify the transmission interval time to 0 with the CFITT[7:0] bits in the CFCCH register.

Note 7: Set a different value to each buffer of the same channel, which is linked to the common FIFO buffer (in transmit mode) (the CFTML[1:0] bits in the CFCCH register).

Note 8: Enable/disable transmit buffer interrupts with the TMIEm bit in the TMIEC register when transmission request is not issued to the transmit buffer (when the TMTRM flag in the TMSTSm register is set to 0).

Note 9: Set 0 to the TMIEm bit corresponding to the transmit buffer linked to the transmit FIFO buffer.

Note 10: Modify the values of the THLDTE and THLIM bits in the THLCC register when the CAN module is in channel reset mode.

Figure 1.18 Procedures for Setting Transmit Buffers, Common FIFO Buffers, and Transmit History List Buffers

1.7 Global Error Interrupt

The setting for global error interrupts is described below. When a corresponding interrupt enabled bit is enabled, an interrupt request is output from the CAN module. Also, the interrupt generation depends on the settings to the interrupt control registers of the interrupt controller.

1.7.1 Setting of Global Error Interrupt

The following are the generation sources for global error interrupts.

- DLC check error
- FIFO message lost
- TX history list entry lost
- CAN-FD message payload overflow

(1) DLC check error

When the DLC check is enabled and a DLC value of a received message which has passed through the acceptance filter is smaller than the DLC value specified in the receive rule, the value is detected as a DLC check error.

(2) FIFO message lost

When a receive FIFO buffer and a common FIFO buffer, both of which have been already full, attempt to store further messages in the buffers themselves, FIFO message lost error will be detected.

(3) TX history list entry lost

When a transmit history list buffer which has been already full attempts to store further transmit history data to the buffer itself, the transmit history list buffer overflows.

(4) CAN-FD message payload overflow

When the payload length of the received message exceeds the payload size of the destination buffer, the CAN-FD payload overflow is detected.

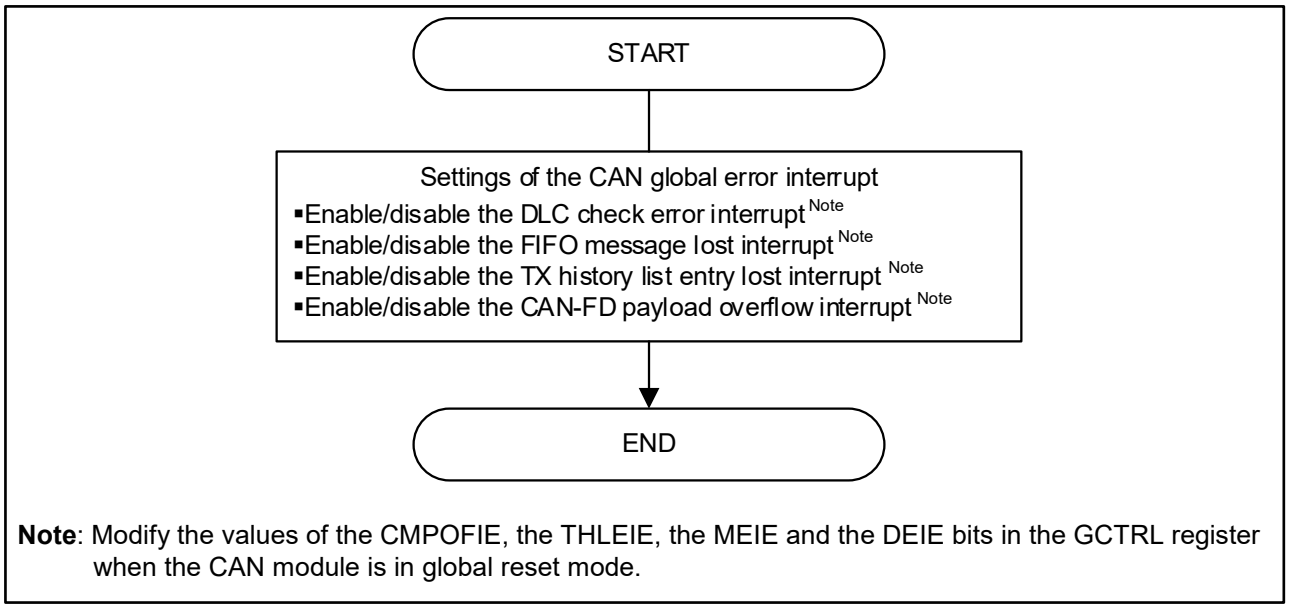
This detection occurs after the DLC check, so if a DLC check error is detected, the CAN-FD payload overflow is not detected.

1.7.2 Procedures for Setting Global Error Interrupts

Figure 1.19 shows the procedures for setting global error itnerrupts.

The following need to be performed with CAN configuration.

For details on the CAN configuration procedure, see “1.1 CAN Configuration”.



Note: Modify the values of the CMPOFIE, the THLEIE, the MEIE and the DEIE bits in the GCTRL register when the CAN module is in global reset mode.

Figure 1.19 Global Error Interrupt Setting Procedures

1.8 Channel Functions

Set the following functions of the channel(s):

- channel error interrupt
- transmit abort interrupt
- bus-off recovery mode
- error display mode
- communication test mode
- communication error occurrence counter
- PN mode recovery operation
- using CAN-FD/Classical CAN frames
- CAN-FD frame transmission/reception

1.8.1 CAN0 Error Interrupt

Enable/disable the CAN0 error interrupt. The following are the generation sources for the channel error interrupt.

- bus error
- error warning
- error passive
- bus-off entry
- bus-off recovery
- overload frame transmit
- bus lock
- arbitration lost
- communication error occurrence counter overflow
- successful communication occurrence counter overflow
- transceiver delay compensation violation

(1) Bus error

An interrupt will be generated in the following conditions:

  – When a form error is detected in the ACK delimiter (when the ADERR flag in the C0ERFLL register is set to 1),
  – When a recessive bit is detected although a dominant bit has been transmitted (when the B0ERR flag in the C0ERFLL register is set to 1),
  – When a dominant bit is detected although a recessive bit has been transmitted (when the B1ERR flag in the C0ERFLL register is set to 1),
  – When a CRC error is detected (when the CERR flag in the C0ERFLL register is set to 1),
  – When an ACK error is detected (when the AERR flag in the C0ERFLL register is set to 1),
  – When a form error is detected (when the FERR flag in the C0ERFLL register is set to 1), or

– When a stuff error is detected (when the SERR flag in the C0ERFLL register is set to 1).

(2) Error warning

An interrupt is generated when a value of a receive error counter or transmit error counter exceeds 95, which is an error warning state. This interrupt is generated only when a value of the receive error counter or transmit error counter exceeds 95 for the first time.

(3) Error passive

An interrupt is generated when a value of a receive error counter or transmit error counter exceeds 127, which is an error passive state. This interrupt is generated only when a value of the receive error counter or transmit error counter exceeds 127 for the first time.

(4) Bus off entry

An interrupt is generated when a value of the transmit error counter exceeds 255, which is a bus-off state.

When the recovery mode is set to transition to channel halt mode at bus-off entry (the value of the BOM[1:0] bits in the C0CTRH register is B'01), an interrupt is also generated at the bus off state.

(5) Bus off recovery

An interrupt is generated when recovery from the bus-off state is detected after 11 consecutive recessive bits have been detected 128 times. For details, see "1.8.3 Settings of Bus Off Recovery Mode".

(6) Overload frame transmit

When performing reception or transmission, an interrupt is generated when the conditions for overload frame transmit are detected.

(7) Bus lock

An interrupt is generated when the bus lock is detected.

The detection of 32 consecutive dominant bits on the CAN Bus in channel operation mode is regarded as the bus lock state.

(8) Arbitration lost

An interrupt is generated when arbitration lost is detected.

(9) Communication error occurrence counter overflow

When the communication error occurrence counter value reaches H'FF, an interrupt occurs if a CAN Bus error is detected under the set conditions. For details, see "1.8.6 Settings of Communication Error Occurrence Counter".

(10) Successful communication occurrence counter overflow

When the successful communication occurrence counter value reaches H'FF, an interrupt occurs when a message is received/transmitted complete.

(11) Transceiver delay compensation violation

If transceiver delay compensation exceeds the compensation maximum (6-data bit time-2f_{CAN}), an interrupt occurs as a transceiver delay compensation violation.

When setting to Classical CAN only mode, set the interrupt of transceiver delay compensation infringement to Prohibit.

1.8.2 CAN0 Transmit Abort Interrupt

Enable/disable the transmit abort interrupt. When the transmit abort interrupt is enabled, an interrupt is generated when transmit abort completion is detected.

The transmit abort interrupt triggers the CAN0 transmit interrupts. The following are the sources for the CAN0 transmit interrupt.

- CAN0 transmit complete interrupt
- CAN0 transmit abort interrupt
- CAN0 common FIFO transmit complete interrupt
- CAN0 transmit history interrupt

1.8.3 Settings of Bus Off Recovery Mode

Set the operation at the bus-off recovery. Table 1.10 lists the operations at the bus-off recovery. Also Figure 1.20 to Figure 1.23 illustrate the operations at the bus-off recovery.

Table 1.10 Operations at Bus Off Recovery

BOM[1:0] bits in the C0CTRH register	Operations	Bus off entry interrupt	Bus off recovery interrupt Note 1
B'00	ISO11898-1 specifications compliant	✓	✓ Note 2
B'01	Transition to channel halt mode at bus-off entry Notes 3, 4	✓	No generation
B'10	Transition to channel halt mode at bus-off end Notes 3, 4	✓	✓
B'11	Transition to channel halt mode (in the bus off state) by a program request	✓	✓ Note 5

Note 1: No interrupt will be generated if the transition to channel reset mode has been done before 11 consecutive recessive bits are detected 128 times (when the value of the CHMDC[1:0] bits in the C0CTRL register is set to B'01).

Note 2: When the CHMDC[1:0] bits in the C0CTRL register are set to B'10 (transition to channel halt mode) before 11 consecutive recessive bits have been detected 128 times, the CAN module will not transition to channel halt mode until 11 consecutive recessive bits have been detected 128 times. Also, no interrupt will be generated when the CAN module is forcibly returned from the bus-off state (the RTBO bit in the C0CTRL register is set to 1).

Note 3: If the transition to channel halt mode and write access to the CHMDC[1:0] bits by a program are performed simultaneously, the write access takes precedence.

Note 4: The automatic transition to channel halt mode is carried out only in channel operation mode (when the value of the CHMDC[1:0] bits is B'00).

Note 5: No interrupt will be generated when transition to channel halt mode is made by a program request before 11 consecutive recessive bits are 128 times during the bus-off state.

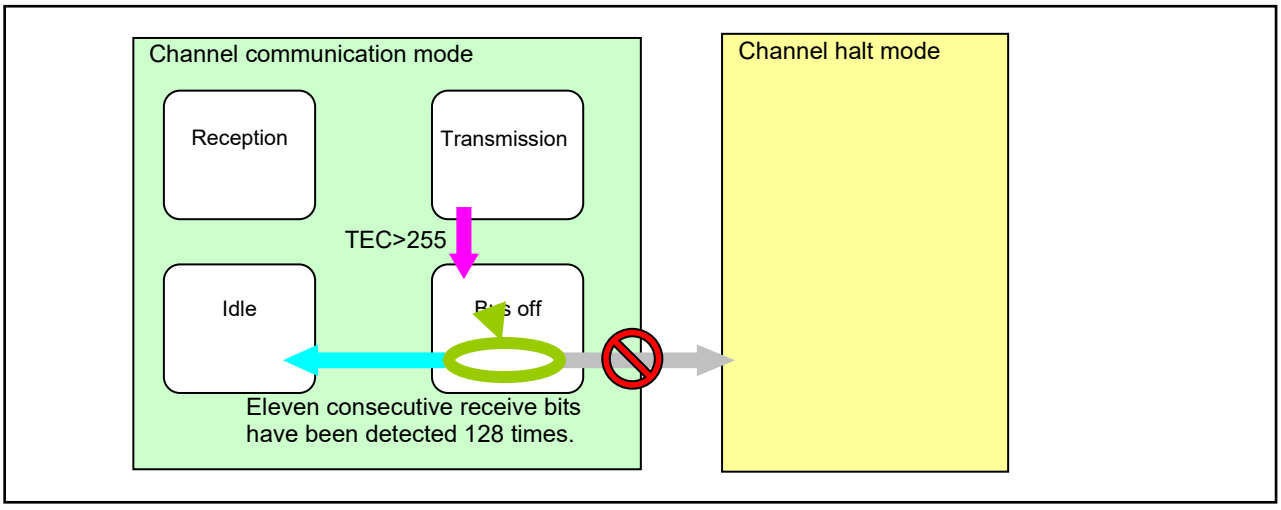


Figure 1.20 ISO11898-1 Specification Compliant Operation (when the value of the BOM[1:0] bits is B'00)

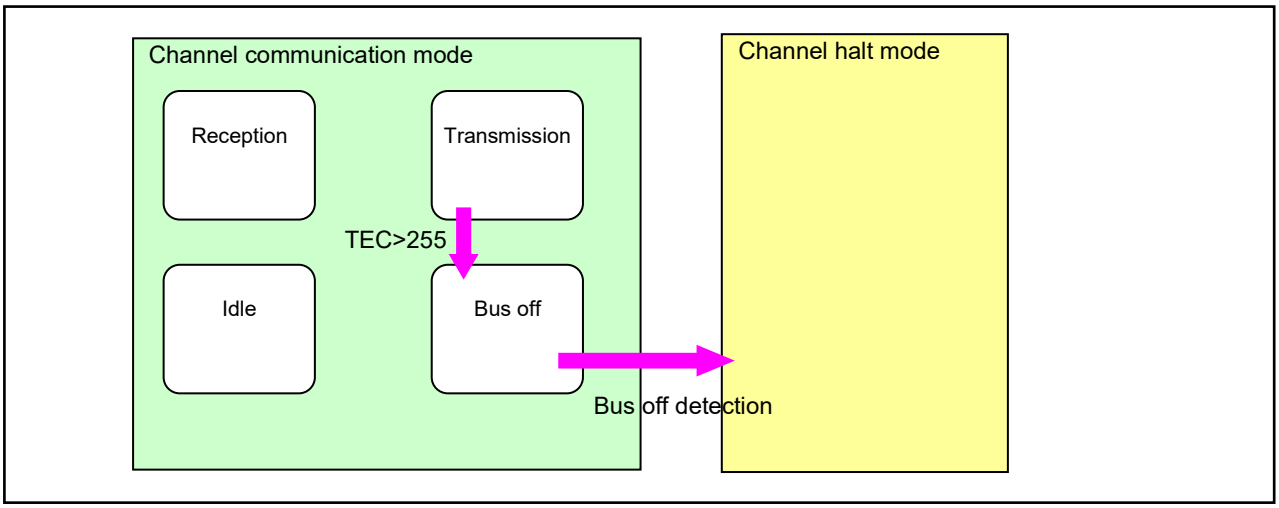


Figure 1.21 Operation at Transition to Channel Halt Mode at Bus Off Entry (when the value of the BOM[1:0] bits is B'01)

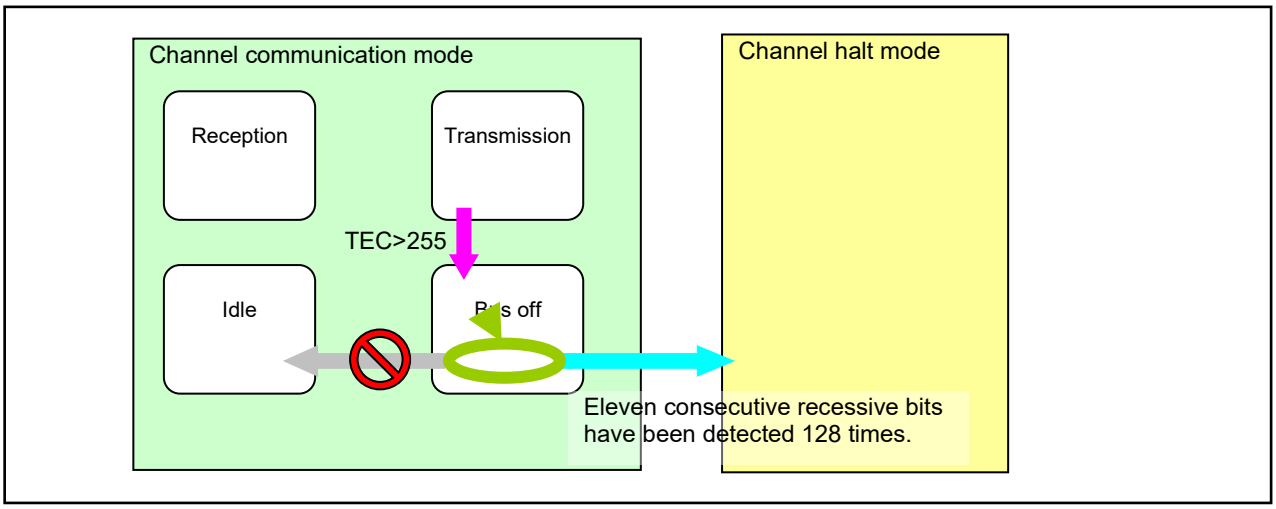


Figure 1.22 Operation at Transition to Channel Halt Mode at Bus Off End (when the value of the BOM[1:0] bits is B'10)

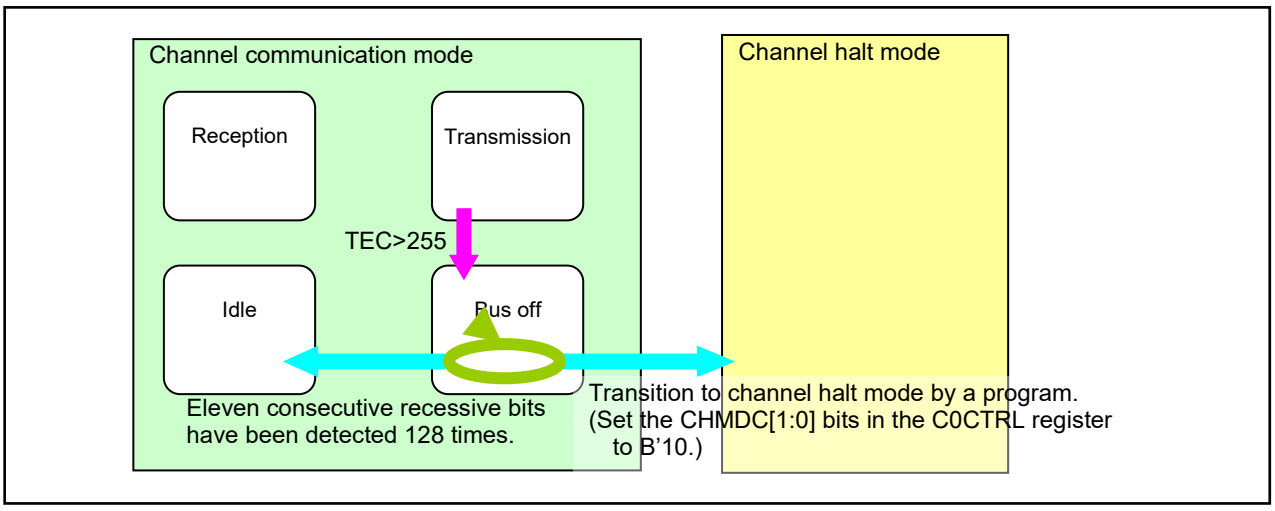


Figure 1.23 Operation at Transition to Channel Halt Mode Due to Request by the Program During Bus-off State (when the value of the BOM[1:0] bits is B'11)

1.8.4 Settings of Error Display Mode

When a CAN Bus error occurs, the error is indicated with bits 14-8 in the C0ERFLL register. The method for indicating the errors can be set as follows:

Indication of only the first error (Set the ERRD bit in the C0CTRH register to 0.)

Only the flag in which the first error has occurred is set to 1. If two or more errors occur simultaneously, all the flags in which the errors have been detected are set to 1.

Indication of all the errors occurred (Set the ERRD bit in the C0CTRH register to 1.)

All the flags in which the errors have occurred are set to 1 regardless of the error occurrence order.

Figure 1.24 illustrates the operations of the C0ERFLL register in each error indication mode.

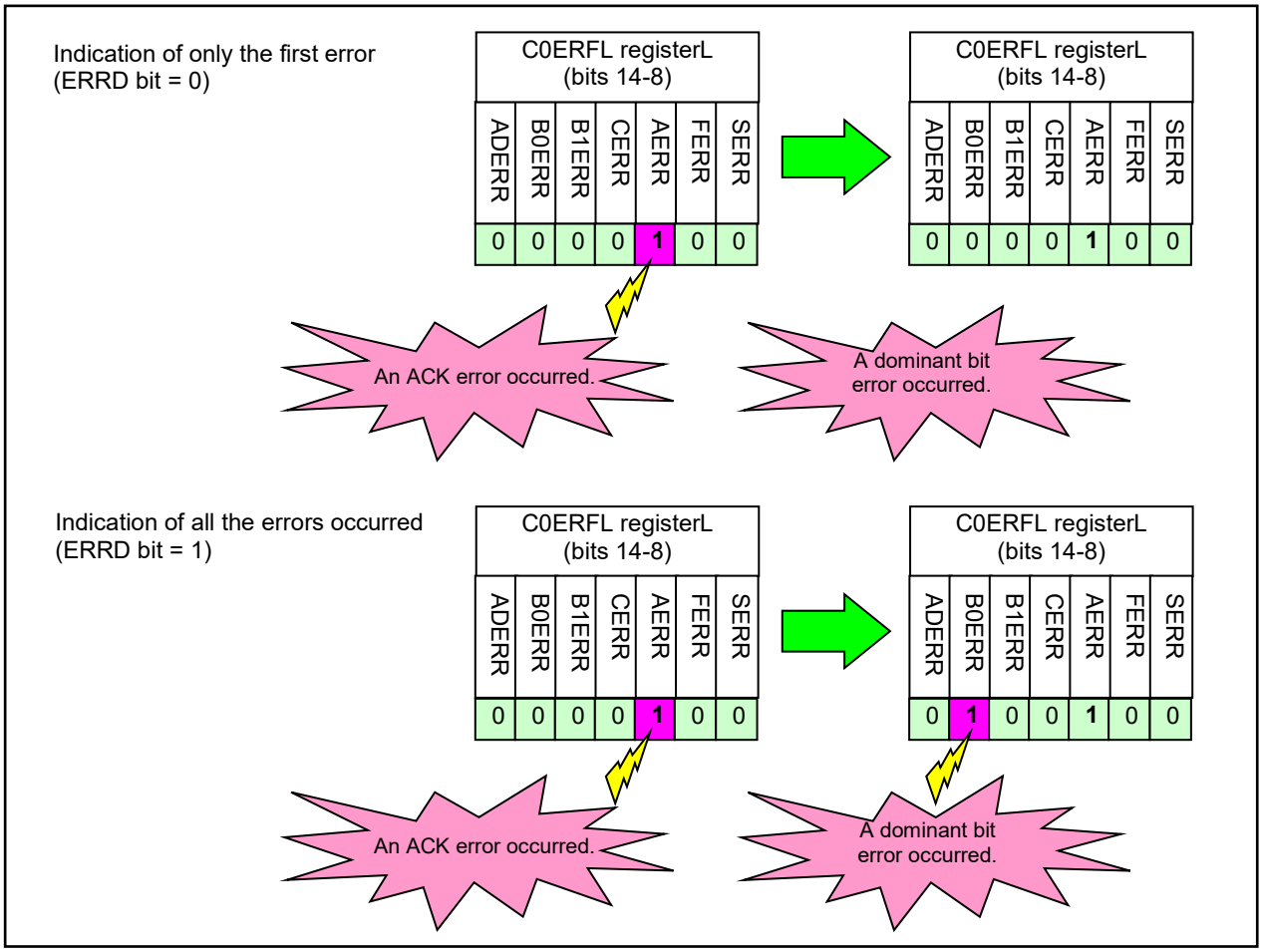


Figure 1.24 Error Indication

1.8.5 Settings of Communication Test Mode

Set the communication test mode. This communication test mode allows self-tests for CAN communication or RAM using the CAN transceiver or MCUs.

1.8.6 Settings of Communication Error Occurrence Counter

Set the counting conditions for the communication error occurrence counter. If a CAN Bus error is detected under the selected conditions, the communication error occurrence counter is counted. The conditions can be selected from the following:

- all transmitted/received CAN frames
- transmitted CAN frame
- received CAN frame
- all transmitted/received CAN-FD data bits
- transmitted CAN-FD data bit
- received CAN-FD data bit

1.8.7 Settings of PN Mode Recovery Operation

Set the PN mode recovery operation when reception of frames that have passed the PNF receive rule in PN mode is completed. The operation can be selected from the following:

- Transitioning to normal inbound rules
- Transitioning to normal inbound rules and PNF receive rules (no ID payload filtering only)
- Transitioning to normal inbound rules and PNF receive rules (payload filtering)
- Continuing PN mode (never return)

1.8.8 Settings of Using CAN-FD/Classical CAN Frames

Set the permission/prohibition transmitting/receiving of CAN-FD and Classical CAN frames. The modes can be selected from the following:

- CAN-FD mode:
  Transmit/Receive CAN-FD and Classical CAN frames.
- CAN-FD only mode:
  Transmit/Receive only CAN-FD frame.
  Classical CAN frames cannot be transmitted. (Transmitted as a CAN-FD frame.)
  If a Classical CAN frame is received, an error frame is transmitted.
- Classical CAN only mode:
  Transmit/Receive only Classical CAN frame.
  CAN-FD frames cannot be transmitted.
  If a Classical CAN frame is received, a form error or CRC error is detected.

1.8.9 Settings of CAN-FD Frame Transmission/Reception

Set the following functions for CAN-FD frame transmission/reception. If you are setting it to Classical CAN only mode, set it to the value after resetting.

- setting of transceiver delay compensation
- setting of the transmit error state indication (ESI) bit
- setting of permission/prohibition of the receive filter

(1) Setting of transceiver delay compensation

Set the permission/prohibition transceiver delay compensation settings, the SSP offset value, and the definition of the SSP position.

The transceiver delay compensation module compares the transmitted data to the received CAN Bus level and measures the delay width due to the transceiver's loop delay.

When using transceiver delay compensation, the definition of the SSP position can be selected from the following:

- The SSP position is defined by the sum of the measured delay and the configured SSP offset value (fixed value).
- The SSP position is defined only by the SSP offset value.

(2) Setting of the transmit error state indication (ESI) bit

Sets the ESI bit value of the transmitted CAN-FD frame. The option can be selected from the following:

- Option 1: ESI bit of the CAN-FD frame indicates the error state of the node.
- Option 2: If the node is not in an error passive state, the ESI bit indicates the error state of the CAN-FD message. If the node is in an error passive state, the ESI bit indicates the state of the node.

The ESI bit value is 0 to indicate an error active state and 1 to indicate an error passive state.

Only when the channel is in the error active state and option 2 is set, any ESI bit value set in the transmit and receive FIFO buffer or transmit buffer is transmitted as the ESI bit value of the transmit message.

Otherwise, an ESI bit value indicating the state of the channel is transmitted, regardless of the ESI bit value set in the transmit or receive FIFO buffer or the transmit buffer.

When in Classical CAN only mode, set it to Option 1.

(3) Setting of permission/prohibition of the receive filter

Set permission/prohibition of the receive filter.

When set the receive filter is set permission, edge filtering of the receive data is enabled when an idle state is detected. At this time, dominant levels less than 2 Tq are ignored and dominant levels of 2 Tq or higher are detected as edges.

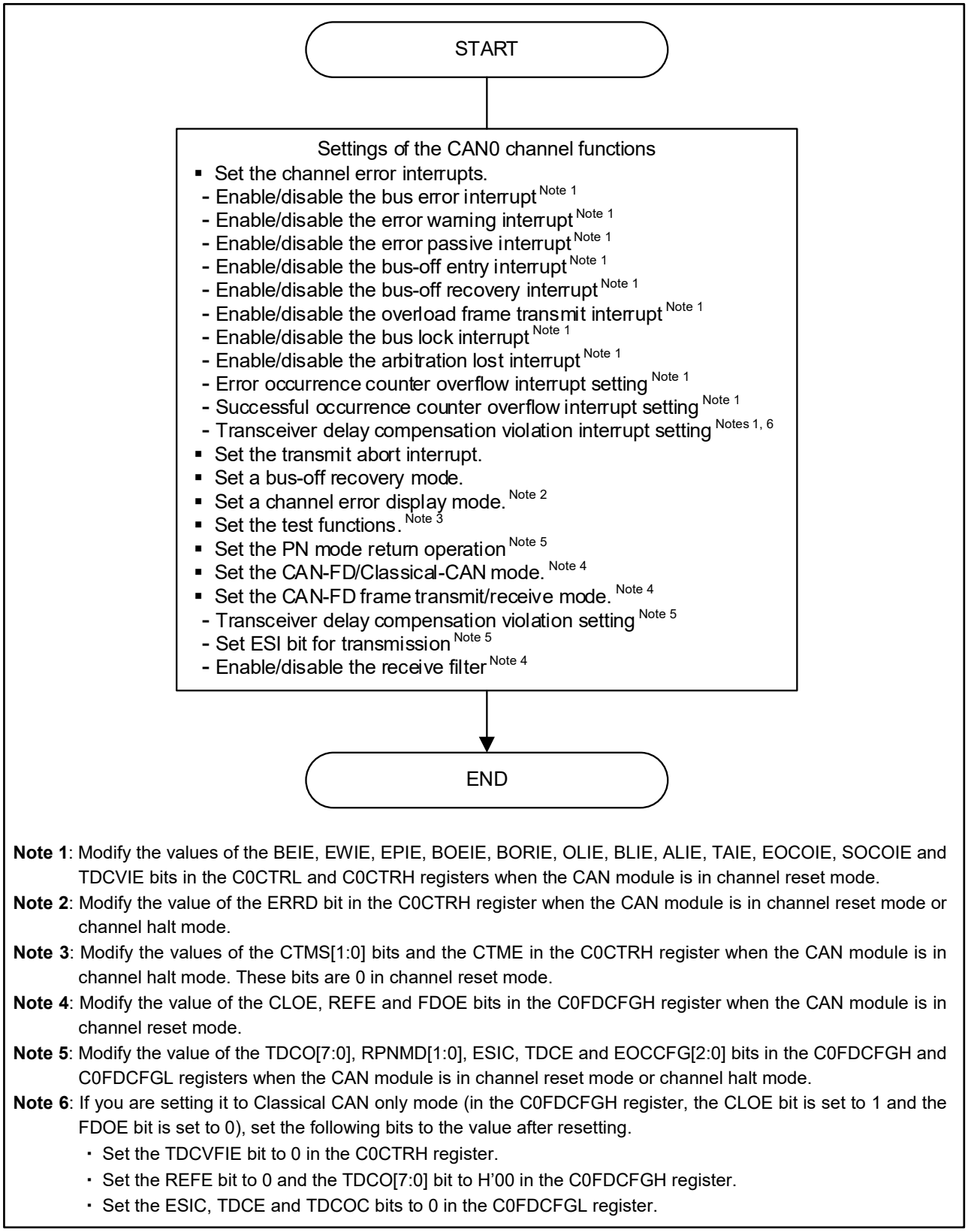
When in Classical CAN on-on mode, set the receive filter to prohibit.

1.8.10 Procedures for Setting Channel Functions

Figure 1.25 shows the procedures for setting channel functions.

These settings need to be performed with the CAN configuration.

For details on the CAN configuration procedures, see “1.1 CAN Configuration”.



Note 1: Modify the values of the BEIE, EWIE, EPIE, BOEIE, BORIE, OLIE, BLIE, ALIE, TAIE, EOCOIE, SOCOIE and TDCVIE bits in the C0CTRL and C0CTRH registers when the CAN module is in channel reset mode.

Note 2: Modify the value of the ERRD bit in the C0CTRH register when the CAN module is in channel reset mode or channel halt mode.

Note 3: Modify the values of the CTMS[1:0] bits and the CTME in the C0CTRH register when the CAN module is in channel halt mode. These bits are 0 in channel reset mode.

Note 4: Modify the value of the CLOE, REFE and FDOE bits in the C0FDCFGH register when the CAN module is in channel reset mode.

Note 5: Modify the value of the TDCO[7:0], RPNMD[1:0], ESIC, TDCE and EOCCFG[2:0] bits in the C0FDCFGH and C0FDCFGL registers when the CAN module is in channel reset mode or channel halt mode.

Note 6: If you are setting it to Classical CAN only mode (in the C0FDCFGH register, the CLOE bit is set to 1 and the FDOE bit is set to 0), set the following bits to the value after resetting.

- Set the TDCVFIE bit to 0 in the C0CTRH register.
- Set the REFE bit to 0 and the TDCO[7:0] bit to H’00 in the C0FDCFGH register.
- Set the ESIC, TDCE and TDCOC bits to 0 in the C0FDCFGL register.

Figure 1.25 Channel Function Setting Procedure

1.9 PNF Receive Rules Table

Set the pretend network filter (PNF) receive rules table for the PNF to perform filtering.

Initially in channel communication mode, PNF is stopped, and filtering is performed according to the Acceptance Filter List (AFL). For details of the receive rules handled by AFL, see “1.5 Receive Rule Table”. Here, “AFL receive rules” are used to distinguish them.

PNF compares the ID and payload values of the received message to the filter values and stores them in the buffer. Filtering by PNF receive rules is independent of filtering AFL receive rules.

Data processing using the PNF receive rule table stores the selected messages in the specified buffer. PNF data processing can be broadly divided into two parts: ID filtering and payload filtering. In addition to data processing using the AFL receive rule table, there is DLC filtering, routing processing, labeling processing, and mirroring functions.

The following settings must be made in the PNF receive rules. Note that PNF receive rules are common to CAN-FD frames and Classical CAN frames, and there is no FDF setting.

- the number of PNF receive rules
- ID Filters
  - IDE/RTR(RRS)/ID
    - messages for receive rules
    - IDE Mask/RTR (RRS) Mask/ID Mask
    - DLC check
    - the receive rule label
    - the storage buffer
- setting of permission/prohibition of the receive filter
  - AND/OR conditions for payload filters 0 and 1
    - payload comparison method
    - the comparison offset
  - comparison data and mask

For PNF filtering to work, the PNF operating state must be switched with channel reset mode deactivated. For details of the operating state of PNF, see “1.9.1 PNF Operating State”.

1.9.1 PNF Operating State

After the channel reset mode is released, AFL receive rule filtering is operating when the PNF operating state is after reset, and PNF receive rule filtering is stopped. To operate PNF, change the PNF operating state by rewriting the PNF mode control bit (the PNMDC[1:0] bits in the C0FDCTRH register).

When PNF is activated, it is selectable whether to stop filtering AFL receive rules when switching the PNF operating state.

(1) PNF Mode Status Flag

The PNF mode status flag (the PNSTS[1:0] bits in the C0FDSTSL register) indicates the PNF operating state. There are four types of filtering combinations of PNF receive rules and AFL receive rules.

Table 1.11 PNF Operating State

C0FDSTSL register	PNF receive rules		AFL receive rules
PNSTS[1:0] bits	ID filter	payload filter	ID filter
B’00 (Acceptance Filter Mode)	no operation	no operation	operation
B’01 (Pretended Network Filter ID only and Acceptance Filter Mode)	operation	no operation	
B’10 (Pretended Network Filter and Acceptance Filter Mode)		operation	
B’11 (Pretended Network Filter Mode)			no operation

After the channel reset mode is released, the PNF operating state is normal receive (the PNSTS[1:0] bits in the C0FDSTSL register is B'00) and the PNF is not operating. To operate PNF, the PNF mode control bit (the PNMDC[1:0] bits in the C0FDCTRH register) is rewritten to switch the operating state to something other than normal receive.

The operating state in which only PNF receive is performed (the PNSTS[1:0] bits is B'11) is called PN mode. This mode is for waking up from the CPU's halt mode. When the received message passes both the ID filter and the payload filter, PN mode automatically returns to the mode set to the return PNF mode select bit (the RPNMD[1:0] bits). For details of setting the return destination, see “1.8.7 Settings of PN Mode Recovery Operation”.

If a received message matches the ID filter in both a PNF receive rule and an AFL receive rule, the PNF action takes precedence.

(2) The buffer rules

When the received message passes the ID filter of the PNF receive rule, the payload filter compares it. If the payload matches the filter criteria, the received message is buffered. If they do not match, the received message is discarded.

If the received message matches the ID filter in both the PNF receive rule and the AFL receive rule, PNF is given priority and buffered based on the information in the PNF receive rule. In this case, even if the PNF does not pass the payload filter or DLC filter, the AFL receive rule does not buffer and the received message is discarded.

When the payload filter works, if it matches the PNF ID filter but the payload filter does not match, it will not be stored in the receive buffer or FIFO buffer.

Table 1.12 PNF Operating State and Storage Buffer

C0FDSTSL register	PNF receive rules		AFL receive rules	buffered or not buffered	buffer rules
PNSTS[1:0] bits	ID filter	payload filter	ID filter		
B’00 (Acceptance Filter Mode)	no operation	no operation	match	buffered	follow the AFL rules
			no match	not buffered	—
B’01 (Pretended Network Filter ID only and Acceptance Filter Mode)	match	no operation	(arbitrary)	buffered	follow the PNF rules
	no match		match	buffered	follow the AFL rules
			no match	not buffered	—
B’10 (Pretended Network Filter and Acceptance Filter Mode)	match	match	(arbitrary)	buffered	follow the PNF rules
		no match	(arbitrary)	not buffered	—
	no match	no check	match	buffered	follow the AFL rules
			no match	not buffered	—
B’11 (Pretended Network Filter Mode)	match	match	no operation	buffered	follow the PNF rules
		no match		not buffered	—
	no match	no check		not buffered	—

1.9.2 Setting of the Number of PNF Receive Rules

Set the number of PNF receive rules to use for the channel.

There are two PNF receive rules in the entire module. One rule consists of the ID filter setting and the payload filter setting.

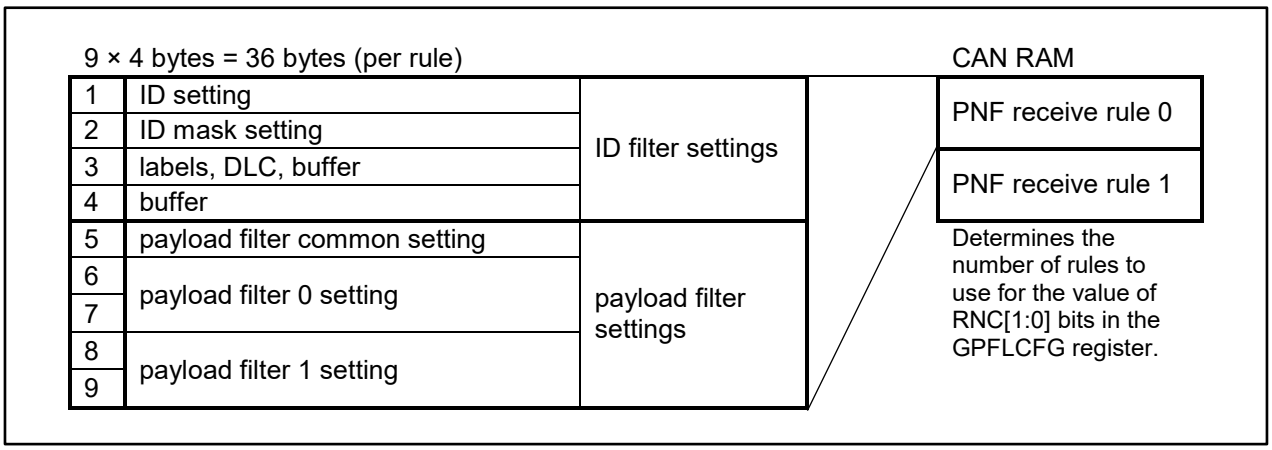


Figure 1.26 Configuring the PNF Receive Rules

The check process starts with the lowest numbered PNF receive rules in ascending order. The PNF filtering stops when all the bits to be compared in the received message match the ID filter settings of the PNF receive rule, or when all checks are completed without matching the PNF receive rule. If there is no matching the PNF receive rule, or if it is matched by the ID filter but not by the payload filter, it is not stored in the receive buffer or FIFO buffer.

The restrictions on the number of PNF receipt rules that can be registered are as follows.

- Limit the number of all PNF receive rules
  the number of CAN0 PNF receive rules ≤ 2

1.9.3 Setting of ID Filter

Set the receive ID and storage buffer of the received message. The settings are the same as those for the AFL receive rules table. For details, please see the following in this application note.

- 1.5.2 Settings of IDE/RTR(RRS)/ID
- 1.5.3 Setting of Messages Target for Receive Rules
- 1.5.4 Settings to Mask IDE/RTR(RRS)/ID
- 1.5.5 Setting of Values to be Compared with DLC Values
- 1.5.6 Setting of Receive Rule Label
- 1.5.7 Setting of Buffers to Store Messages

1.9.4 Setting of Payload Filter

The payload filter compares two locations of 32-bit data to the payload of the received message matched by the ID filter.

The payload filter sets the AND/OR conditions for two filters and one set of filter conditions per filter.

- AND/OR conditions for payload filters 0, 1
- Payload filter 0, 1
  - payload comparison method
    - Comparison Offset
    - Comparison data and mask

Because a single payload comparison is done in 4-byte increments, the following received messages will always fail the payload filter check, regardless of the filter conditions:

- data frames with less than 4 DLC
- remote frame

(1) Setting of AND/OR conditions for payload filters 0, 1

Set the pass conditions (AND/OR conditions) for the entire payload filter. Here are the methods that can be selected from:

- Pass both receive rule filters 0 and 1 (AND condition)
- Pass either receive rule filters 0 or 1 (OR condition)

It is not possible to operate only one of filters 0 and 1.

(2) Setting of the payload comparison method

Set the method by which payloads are compared. The different payload comparison method for each filter can be set. Here are the methods that can be selected from:

- Match filter
- Upper and lower limit filter

When using an upper and lower filter, the pass condition can be selected from the following:

- Within the upper and lower limits:
  the lower limit value ≤ payload data value ≤ the upper limit value
- Outside the upper and lower limits:
  the payload data value < the lower limit value or the upper limit value < the payload data value

In an upper and lower filter, the byte order of the payload is treated as little endian.
For example, if the comparison target is the payload data 0 byte to 3 byte and the values are H'A1, H'B2, H'C3, and H'D4 in order, the payload data value to be compared will be H'D4C3B2A1.

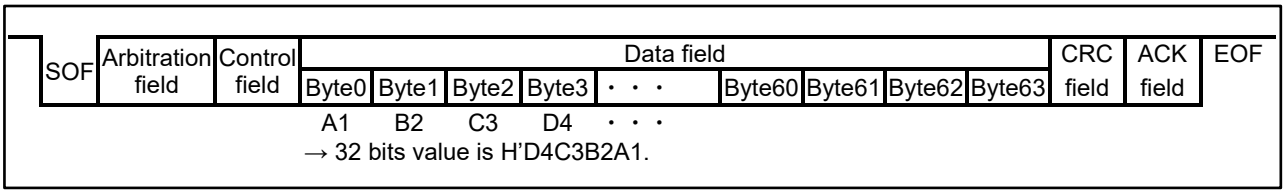


Figure 1.27 Comparison Payload Data Values for Received CAN Frames

(3) Setting of the comparison offset

Set the offset position of the payload to be compared. It can be set one offset position for a single filter, and two offset locations for the entire payload filter.

The offset indicates the position of the payload of the received message divided into 32 bits increments and is set in the range 0 to 15. The payload data 0 byte to 3 bytes is compared at offset 0, and the payload data 60 to 63 byte is used for comparison at offset 15.

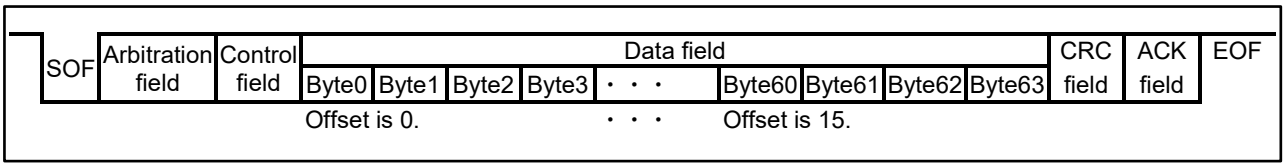


Figure 1.28 Payload Comparison Offset of Received CAN Frame

If the payload length of the received message is less than the byte to be compared specified at the offset, the payload condition at that offset is considered unsuccessful.

(4) Setting of comparison data and mask

Set the 32 bits data value under which the payload is compared. Two 32 bits data values must be set for a single filter.

When the comparison method is the match filter, set the expected 4 bytes payload data value and mask value. Bits with a mask value of “1” are the ones that the payload filter compares.

When the comparison method is an upper and lower bounds filter, set the upper and lower limits of 32 bits integers.

1.9.5 Example of Using PNF Receives Rules

Here is an example of using PNF inbound rules:

- Example 1
  The following is an example of each register when receiving a message:
  - ID format: Standard ID
  - Message format: Data frame
  - Mirror function: Receiving messages from other CAN nodes
  - Receiver ID: 120h, 121h, 122h and 123h
  - DLC: DLC ≥ 8 for received messages
  - Label: 16 bits 0010h, 2 bits 01b
  - Destination buffer: Receive FIFO buffer 1
  - Payload: Byte0 is H'06, Byte1's upper 4 bits is B'1011, Byte7 is H'5A

[ID filter]

		GAFLIDE/GAFLIDEM	GAFLRTR/GAFLRTRM	GAFLLB	GAFLID/GAFLIDM			
GAFLIDiL、GAFLIDiH		0	0	0	B’00000	B’00000000	B’00000001	B’00100000
GAFLMiL、GAFLMiH		1	1	—	B’00000	B’00000000	B’00000111	B’11111100
Messages that can be received	**H’120**	0	0	0	B’-----	B’--------	B’-----001	B’00100000
	H’121				B’-----	B’--------	B’-----001	B’00100001
	H’122				B’-----	B’--------	B’-----001	B’00100010
	H’123				B’-----	B’--------	B’-----001	B’00100011

	GAFLDLC	GAFLPTR	GAFLIFL0	GAFLIFL1	GAFLRMV	GAFLRMDP	GPFLFDP0	GPFLFDP1	GPFLFDP8
GAFLMiH	—	—	—	0	—	—	—	—	—
GAFLP0iL、GAFLP0iH	8	H’0010	1	—	0	0	—	—	—
GAFLP1iL	—	—	—	—	—	—	0	1	0

[payload filter]

Settings of the payload filter are as follows:

- AND/OR condition: Both receive rule filters 0 and 1 pass (AND condition)
- Payload filter 0: match filter, comparison position offset 0
  Comparative data H'06, H'B0, H'00, H00
  Compare the 8 bits of Byte0 with the upper 4 bits of Byte1
- Payload filter 1: match filter, comparison position offset 1
  Comparative data H'00, H'00, H'00, H'5A
  Compare the 8 bits of Byte7

[AND/OR condition]

	GPFLANDOR
GPFLPTjH	0

[Payload filter 0]

	GPFLRANG0	**GPFLOUT0**	**GPFLOFFSET0**	**FDATA0/FMASK0**			
				[31:24]	**[23:16]**	**[15:8]**	**[7:0]**
GPFLPTjH	0	0	0	–	–	–	–
GPFLPD0jL, GPFLPD0jH	–	–	–	B’00000000	B’00000000	**B’10110000**	**B’00000110**
GPFLPM0jH, GPFLPM0jL	–	–	–	B’00000000	B’00000000	**B’11110000**	**B’11111111**

[Payload filter 1]

	GPFLRANG1	**GPFLOUT1**	**GPFLOFFSET1**	**FDATA1/FMASK1**			
				[31:24]	**[23:16]**	**[15:8]**	**[7:0]**
GPFLPTjL	0	0	1	–	–	–	–
GPFLPD1jL, GPFLPD1jH	–	–	–	**B’01011010**	B’00000000	B’00000000	B’00000000
GPFLPM0jH, GPFLPM0jL	–	–	–	**B’11111111**	B’00000000	B’00000000	B’00000000

- Example 2
  The following is an example of each register when receiving a message:
  - ID format: Standard ID
  - Message format: Data frame
  - Mirror function: Receiving messages from other CAN nodes
  - Receiver ID: 130h
  - DLC: DLC check not used
  - Label: 16 bits 0130h, 2 bits 10b
  - Destination buffer: common FIFO buffer 0
  - Payload: Byte 0 to 3 (little endian) is 500 to 2000 or more than 20000

[ID filter]

		GAFLIDE/GAFLIDEM	GAFLRTR/GAFLRTRM	GAFLLB	GAFLID/GAFLIDM			
GAFLIDiL、GAFLIDiH		0	0	0	B’00000	B’00000000	B’00000001	B’00110000
GAFLMiL、GAFLMiH		1	1	—	B’00000	B’00000000	B’00000111	B’11111111
Messages that can be received	**H’130**	0	0	0	B’-----	B’--------	B’-----001	B’00110000

	GAFLDLC	GAFLPTR	GAFLIFL0	GAFLIFL1	GAFLRMV	GAFLRMDP	GPFLFDP0	GPFLFDP1	GPFLFDP8
GAFLMiH	—	—	—	1	—	—	—	—	—
GAFLP0iL、GAFLP0iH	0	H’0130	0	—	0	0	—	—	—
GAFLP1iL	—	—	—	—	—	—	0	0	1

[payload filter]

Settings of the payload filter are as follows:

- AND/OR condition: Either receive rule filters 0 or 1 pass (OR condition)
- Payload filter 0: upper and lower limit filter (within upper and lower limits)
  Comparison position offset 0
  Low limit 500 (H'1F4), upper limit 2000 (H'7D0)
- Payload filter 1: upper and lower limit filters (outside the upper and lower limits)
  Comparison position offset 0
  Lower limit 0 (H'0), upper limit 19999 (H'4E1F)

[AND/OR condition]

	GPFLANDOR
GPFLPTjH	1

[Payload filter 0]

	GPFLRANG0	**GPFLOUT0**	**GPFLOFFSET0**	**FDATA0/FMASK0**			
				[31:24]	**[23:16]**	**[15:8]**	**[7:0]**
GPFLPTjH	1	0	0	–	–	–	–
GPFLPD0jL, GPFLPD0jH	–	–	–	H’00	H’00	H’07	H’D0
GPFLPM0jH, GPFLPM0jL	–	–	–	H’00	H’00	H’01	H’F4

[Payload filter 1]

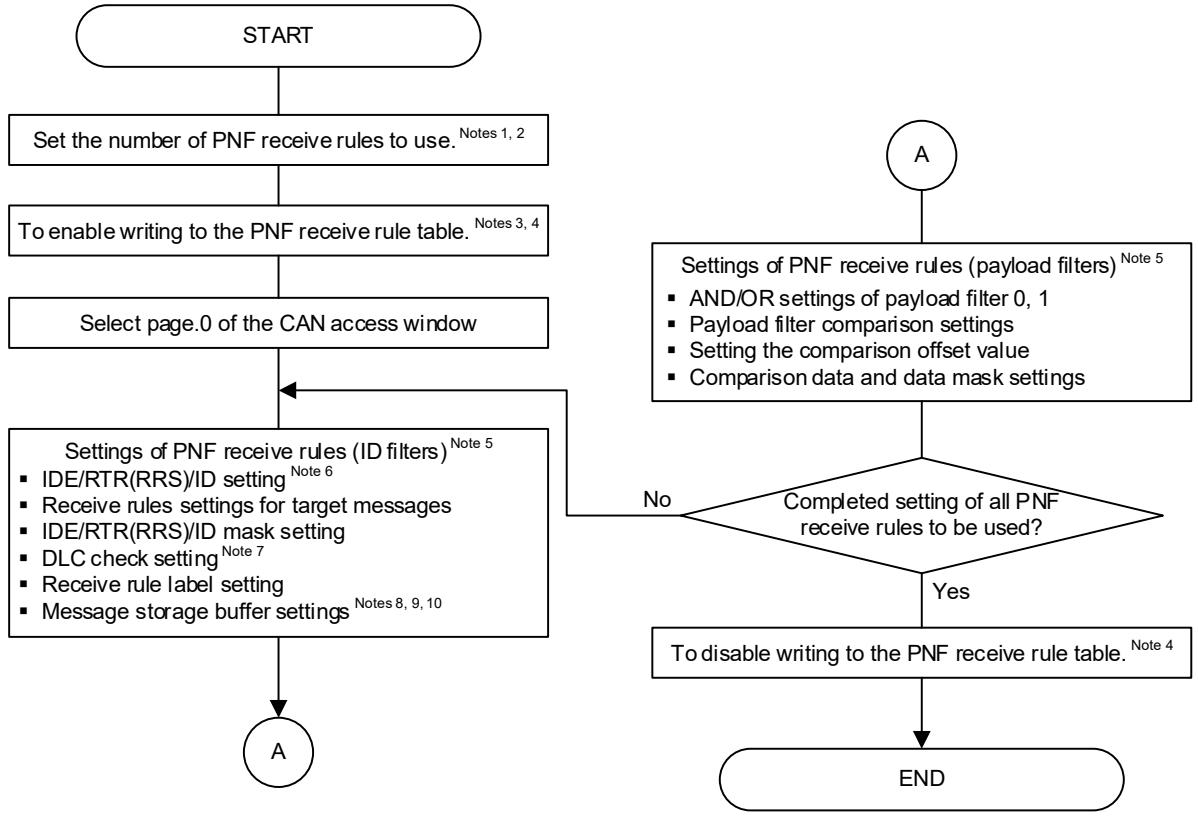
	GPFLRANG1	**GPFLOUT1**	**GPFLOFFSET1**	**FDATA1/FMASK1**			
				[31:24]	**[23:16]**	**[15:8]**	**[7:0]**
GPFLPTjL	1	1	0	–	–	–	–
GPFLPD1jL, GPFLPD1jH	–	–	–	H’00	H’00	H’4E	H’1F
GPFLPM0jH, GPFLPM0jL	–	–	–	H’00	H’00	H’00	H’00

1.9.6 Procedures for Setting Receive Rule Table

Figure 1.29 shows the procedures for setting the PNF receive rule table.

These settings need to be performed with the CAN configuration.

For details on the CAN configuration procedure, see “1.1 CAN Configuration”.



Note 1: Modify the number of receive rules (the RNC[1:0] bits in the GAFLCFG register) when the CAN module is in global reset mode.

Note 2: Set the number of PNF receive rules used for the channel must be 2 or ress.

Note 3: To write to the receive rule table needs to be permited (set the PFLDAE bit in the GPFLECTR register to 1).

Note 4: After writing to the PNF receive rule table is completed, set write to the receive rules table to prohibit (set the PFLADE bit to 0).

Note 5: Modify ID filter of the receive rules (the GPFLIDjL register, GPFLIDjH register, GPFLMjL register, GPFLMjH register, GPFLP0jL register, GPFLP0jH register, and GPFLP1jL register) and payload filter of the receive rules (the GPFLPTjL register, the GPFLPTjH register, the GPFLPD0jL register, the GPFLPD0jH register, the GPFLPM0jL register, GPFLPM0jH register, the GPFLPD1jL register, the GPFLPD1jH register, the GPFLPM1jL register and the GPFLPM1jH rigister) when all of tha following conditions are met.

*Set write to the PNF receive rules table to permit (the PFLDAE bit is 1).

*Select page 0 of access window for the CAN module (the RPAGE[1:0] bits in the CFDGRWC register are B’00).

*Set channel reset mode or channel halt mode.

Note 6: When the standard ID is selected, set the value of the standard ID to bits 10-0 of the ID data (the GPFLID[15:0] bits in the GPFLIDjL register) and also set 0 to bits 15-11 in the GPFLIDjL register and the GPFLID[28:16] bits in the GPFLIDjH register.

Note 7: This setting is valid only when the DLC check is enabled (set the DCE bit in the GCFGL register to 1).

Note 8: A maximum of two FIFO buffers can be selected. However, when storing a message in one receive buffer (set the GPFLRMV bit in the GPFLP0jL register to 1), the number of FIFO buffers that can be selected is only one.

Figure 1.29 Setting Procedures for PNF Receive Rule Table (1/2)

Note 9: Select only one receive FIFO buffer and one common FIFO buffer which is set to receive mode.
Note 10: When selecting a receive buffer as a storage buffer, enable the receive buffer (set the GPFLRMV bit to 1) and set a buffer number which is smaller than the number of receive buffers to be used (the NRXMB[4:0] bits in the RMNB register).

Figure 1.29 Setting Procedures for PNF Receive Rule Table (2/2)

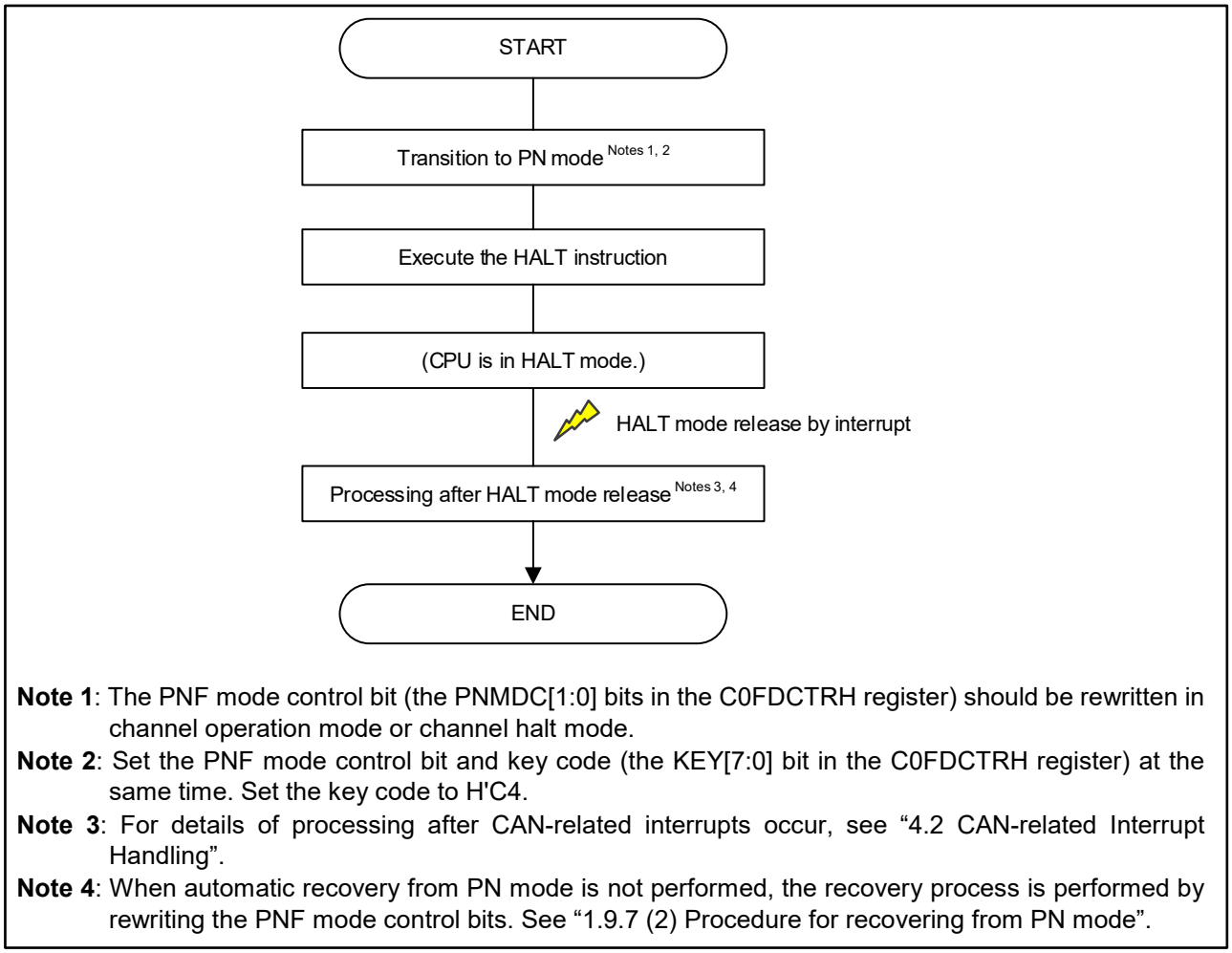
1.9.7 Procedures for PNF Operating State Transition

This section describes the procedure for transitioning to and recovering to PN mode.

(1) Procedure for transitioning to PN mode

Figure 1.30 shows the procedure for transitioning to PN mode and CPU HALT mode.

If the received message matches the PNF's ID filter and payload filter, it is buffered. When receive interrupt generation is permitted for the buffer to be stored, the receive interrupt causes the halt mode to be removed.



Note 1: The PNF mode control bit (the PNMDC[1:0] bits in the C0FDCTRH register) should be rewritten in channel operation mode or channel halt mode.

Note 2: Set the PNF mode control bit and key code (the KEY[7:0] bit in the C0FDCTRH register) at the same time. Set the key code to H'C4.

Note 3: For details of processing after CAN-related interrupts occur, see “4.2 CAN-related Interrupt Handling”.

Note 4: When automatic recovery from PN mode is not performed, the recovery process is performed by rewriting the PNF mode control bits. See “1.9.7 (2) Procedure for recovering from PN mode”.

Figure 1.30 Procedure for Transitioning to PN Mode and HALT Mode

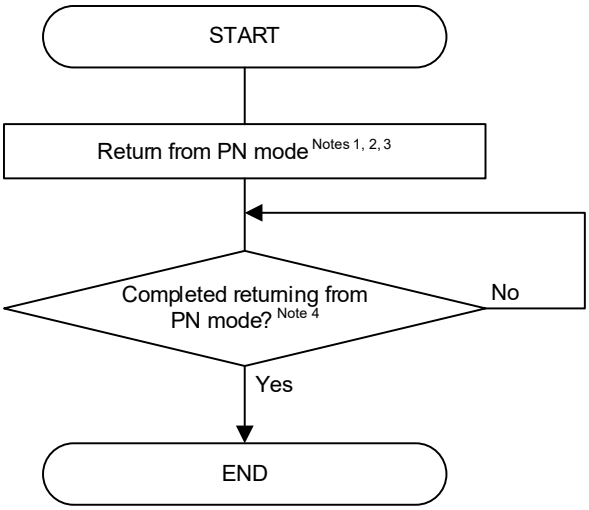
(2) Procedure for recovering from PN mode

When the received message is buffered by a match to the PNF receive rule, PNF recovers to its previously configured operating state from PN mode. For details of setting the return destination, see “1.8.7 Settings of PN Mode Recovery Operation”.

If automatic recovery by storing the received message is not performed, the PNF mode control bit (the PNMDC[1:0] bits in the C0FDCTROL register) is rewritten to perform the reconfiguration process.

PN mode recovery processing sets the PNF mode control bits to the same value as the PNF waver mode select bit (RPNMD[1:0] bits in the C0FDCFGL register). When setting to continue PN mode (RPNMD[1:0] bits is B'11), PN mode can be removed by a channel reset mode transition.

Figure 1.31 shows the procedure for recovering from PN mode.



Note 1: The PNF mode control bit (the PNMDC[1:0] bits in the C0FDCTRH register) should be rewritten in channel operation mode or channel halt mode.

Note 2: Set the PNF mode control bit and key code (the KEY[7:0] bit in the C0FDCTRH register) at the same time. Set the key code to H'C4.

Note 3: Set the PNF mode control bit to the same value as the PNF recover mode select bit (RPNMD[1:0] bits in the C0FDCFGL register).

Note 4: After changing the PNF mode control bits, ensure that the PNF mode status flag (PNSTS[1:0] bits in the C0FDSTSL register) switches the mode. During receiving, the value of the PNF mode control bit is reflected in the PNF mode status flag after the receive is complete, otherwise the value is reflected immediately.

Figure 1.31 Procedure for Recovering from PN Mode

2. Reception

2.1 Reception Function

There are the following types of reception to receive CAN messages. For details, refer to the following sections:

- Reception using receive buffers
- Reception using receive FIFO buffers
- Reception using common FIFO buffers

2.2 Reception Using Receive Buffers

0 to 16 receive buffers can be shared by the channel (all channels). Data (messages) in a receive buffer will be overwritten when a new message is stored in the same receive buffer. Thus, the latest receive data can be read.

When the process of storing a received message in receive buffers starts, the RMNSn flag in the RMND register is set to 1, which means receive buffer n contains the new message. Then the data can be read from the RMIDnL and RMIDnH registers, the RMPTRnL and RMPTRnH registers, the RMFDSTSnL and RMFDSTSnH registers, RMDFn_0L to RMDFn_15L and RMDFn_0H to RMDFn_15H registers.

Regarding the configuration to use receive buffers, see “1.1 CAN Configuration”.

Figure 2.1 illustrates the operation of receive buffers.

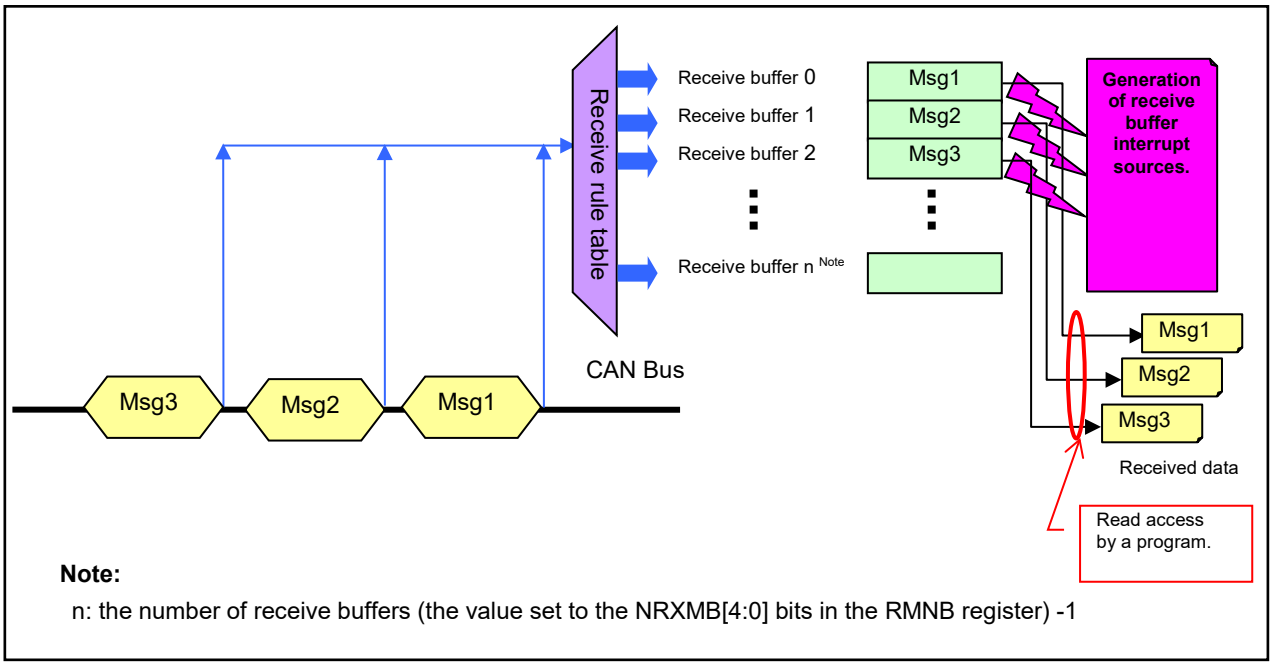
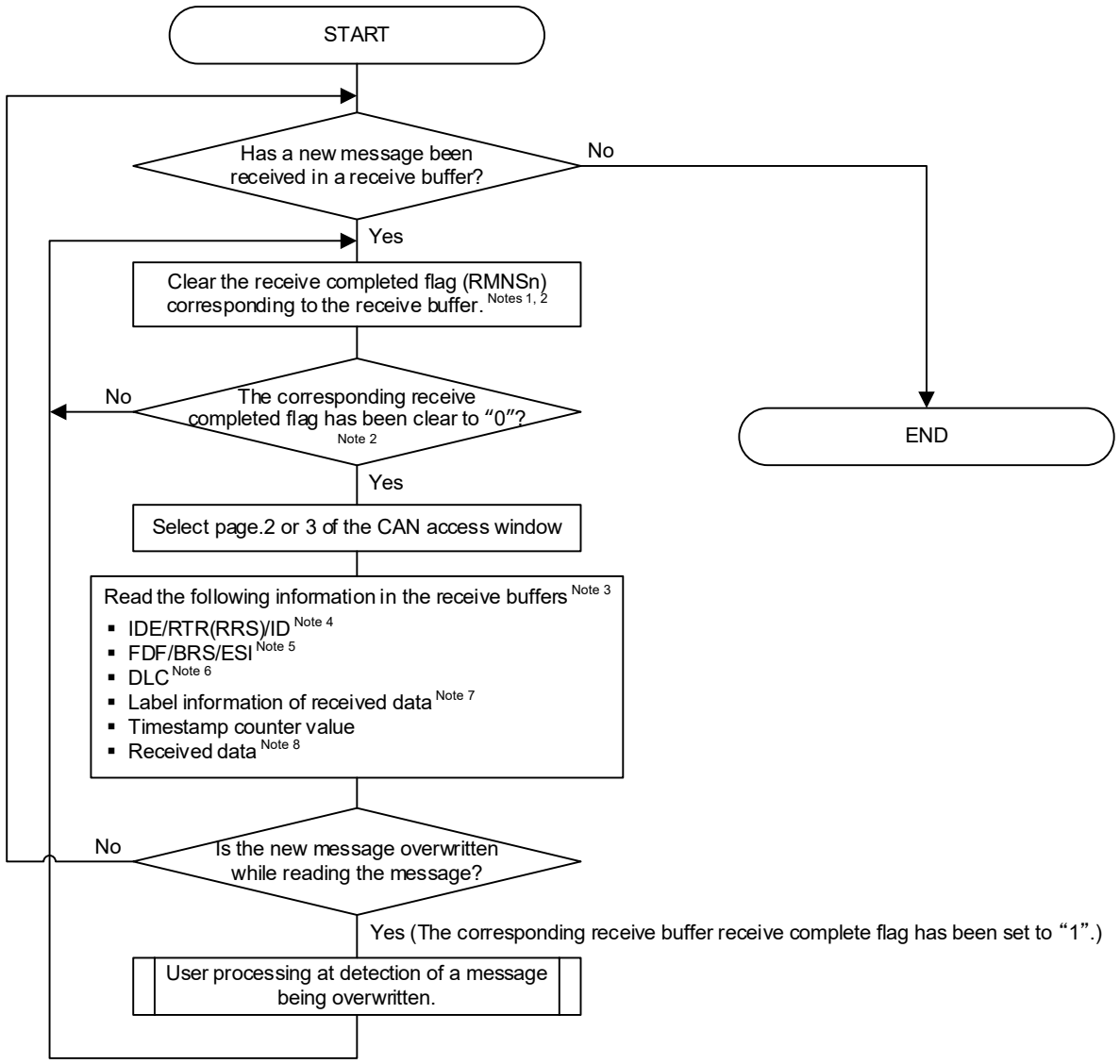


Figure 2.1 Operation of Receive Buffers

2.2.1 Procedures for Reading Receive Buffers

Figure 2.2 shows the procedures for reading the receive buffers.



Note 1: The RMNSn flag in the RMND register must be written when the CAN module is in global operating mode or global halt mode. When writing 0 to the RMNSn flag, respectively write 0 and 1 to the bits which will be cleared and to the bits which will not be cleared using a MOV instruction.

Note 2: The RMNSn flag cannot be set to 0 while a message is being stored. The time to store the message depends on the payload size of the receive buffer. When the payload size is 8 bytes (the RMPLS[2:0] bits in the RMNB register is B'000), it is 6 clock cycles of the CPU/peripheral hardware clock. For every 4 bytes of stored payload size, 1 clock is added to 20 clock for 64 bytes (the RMPLS[2:0] bits in the RMNB register is B'111).

Note 3: For pages 2 or 3 of the CAN access window (the RPAGE[1:0] bits in the CFDGRWC register is B'10 or B'11), the receive buffer (the RMIDnL/H register, the RMPTRnL/H register, the RMFDSTSnL/H register and the RMDFn_0L/H register to the RMDFn_15L/H register) can be read.

Note 4: When the standard ID is selected, read bits 10-0 of the ID data (the RMID[15:0] bits in the RMIDnL register). Bits 15-11 and the RMID[28:16] bits in the RMIDnH register are read as 0.

Note 5: For Classical CAN frames (the RMFDF bit in the RMFDSTSnL register is 0), the RMBRS and RMESI bits in the RMFDSTSkL register are always 0 read.

Figure 2.2 Reading of Receive Buffers (1/2)

Note 6: When the DLC replacement is enabled (the DCE and DRE bits in the GCFGL register are both set to 1) after the filter processing according to the receive rules, the DLC value specified in the receive rule table (the GAFLDLC[3:0] bits in the GAFLP0iL register) which has agreed with a DLC value of the received message will be stored in place of the DLC value of the received message. In other cases, the DLC value of the received message will be stored without the replacement of the DLC value.

Note 7: After the filter processing according to the receive rules, the value set to the label data of the receive rule table (the GAFLPTR[15:0] bits in the GAFLP0iH register, the GAFLIFL0 bit in the GAFLP0iL register and the GAFLIFL1 bit in the GAFLMiH register) which has agreed with the value of the received message will be stored.

Note 8: When the DLC value (the RMDLC[3:0] bits in the RMPTRnH register) of the received message is smaller than the payload size of the receive buffer (the RMPLS[2:0] bits in the RMNB register), data bytes where no data have been set are read as H'00. Do not read the RMDFn_pL/H register that correspond to areas that exceed the size specified in the payload size setting (the RMPLS[2:0] bits in the RMNB register).

Figure 2.2 Reading of Receive Buffers (2/2)

2.3 Reception Using Receive FIFO Buffers

There are two receive FIFO buffers which can be shared by the channel (all channels). Each receive FIFO buffer can retain messages up to the number equal to the number of receive buffers that each receive FIFO buffer has.

Once the received message has been stored in the receive FIFO buffer, the value of the corresponding message count display counter (the RFMC[5:0] bits in the RFSTSk register) is incremented.

Received messages can be read from the RFIDkL and RFIDkH registers, the RFPTRkL and RFPTRkH register, the RFFDSTSkL and RFFDSTSkH register, the RFDFk_0L to RFDFk_15H registers. Messages in the receive FIFO buffers can be read sequentially on a first-in, first-out basis.

When the value of the message count display counter matches the number of messages that can be stored in a single receive FIFO buffer (a value set by the RFDC[2:0] bits in the RFCCk register), the receive FIFO buffer is full (the RFFLL flag in the RFSTSk register is set to 1).

When all the messages have been read out from the receive FIFO buffer, the receive FIFO buffer is empty (contains no message) (the RFEMP flag in the RFSTSk register is set to 1).

Regarding the configuration to use the receive FIFO buffer, see “1.1 CAN Configuration”.

Figure 2.3 illustrates the operation of the receive FIFO buffers.

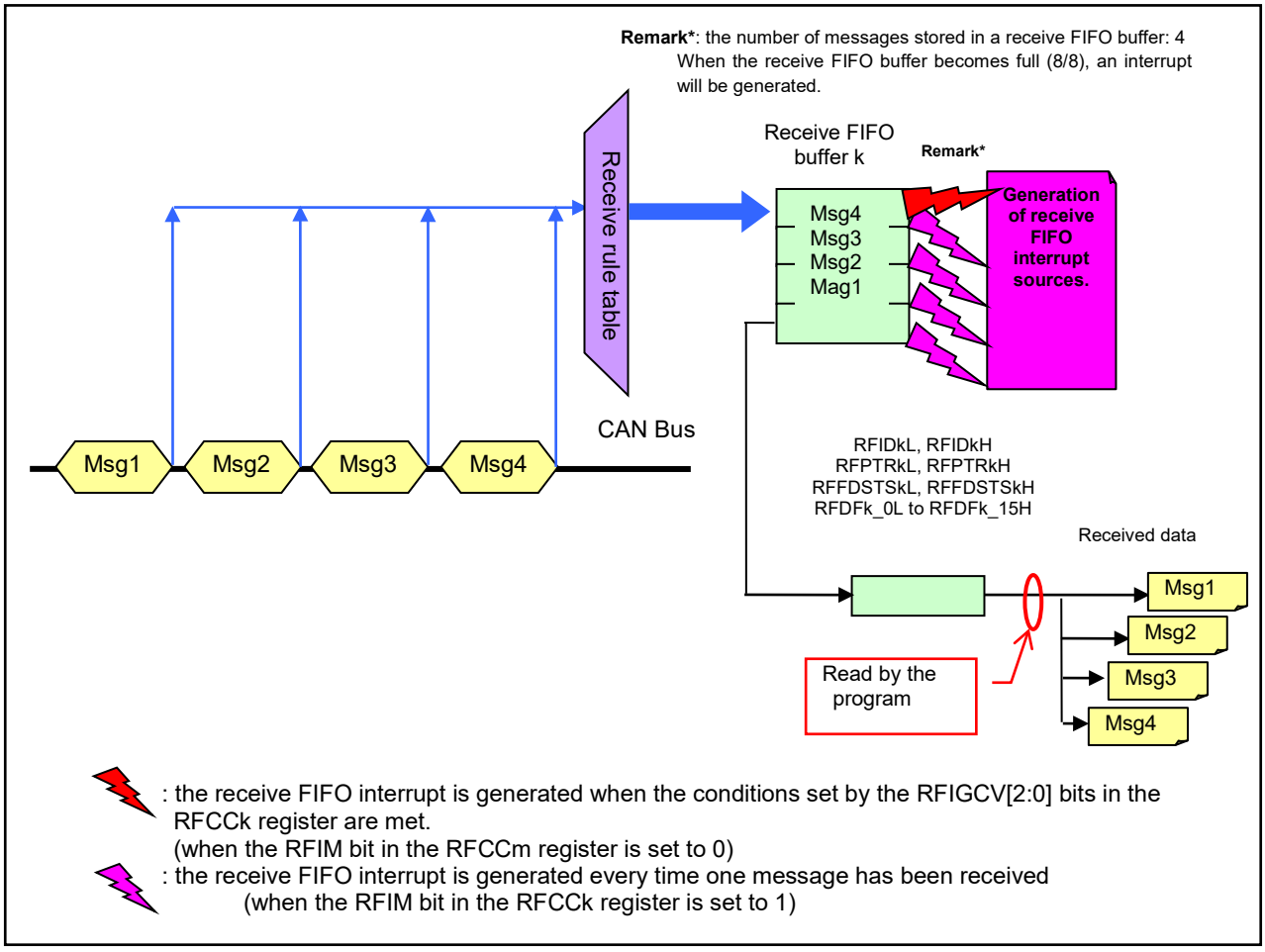
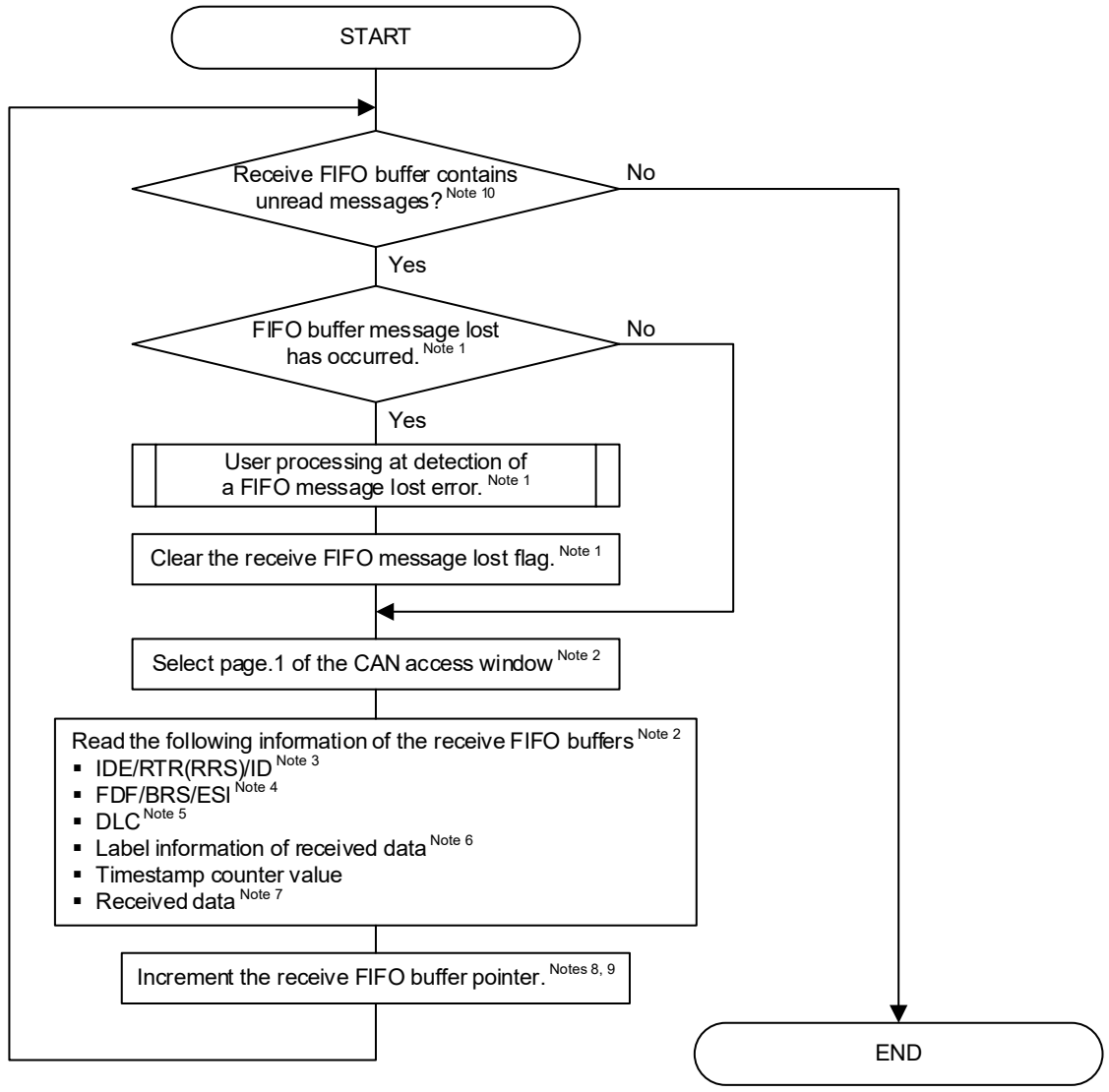


Figure 2.3 Operation of Receive FIFO Buffer

2.3.1 Procedures for Reading Receive FIFO Buffers

Figure 2.4 shows the procedures for reading receive FIFO buffers. Figure 2.5 and Figure 2.6 show the procedures for enabling and disabling the receive FIFO buffers, respectively.



Note 1: When the FIFO message lost interrupt is enabled, these steps need to be done while the global error interrupt is being handled.

Note 2: For page 1 of the CAN access window (the RPAGE[1:0] bits in the CFDGRWC register is B'01), the FIFO receive buffer (the RFIDkL/H register, the RFPTRkL/H register, the RFFDSTSkL/H register and the RFDFk_0L/H register to the RFDFk_15L/H register) can be read.

Note 3: When the standard ID is selected, read bits 10-0 of the ID data (the RFID[15:0] bits in the RFIDkL register). The bits 15-11 of the RFIDkL register and the RFID[28:16] bits in the RFIDkH register are read as 0.

Note 4: For the Classical CAN frames (the RFFDF bit in the RFFDSTSkL register is 0), the RFBRS bit and the RFESI bit in the RFFDSTSkL register can be read 0.

Figure 2.4 Receive FIFO Buffer Reading Procedure (no interrupt used) (1/2)

Note 5: When the DLC replacement is enabled (the DCE and DRE bits in the GCFGL register are both set to 1) after the filter processing according to the receive rules, the DLC value specified in the receive rule table (the GAFLDLC[3:0] bits in the GAFLP0iL register) which has agreed with a DLC value of the received message will be stored in place of the DLC value of the received message. In other cases, the DLC value of the received message will be stored without the replacement of the DLC value.

Note 6: After the filter processing according to the receive rules, the value set to the label data of the receive rule table (the GAFLPTR[15:0]bits in the GAFLP0iH register, the GAFLIFL0 bit in the GAFLP0iL register and the GAFLIFL1 bit in the GAFLMiH register) which has agreed with the data of the received message will be stored.

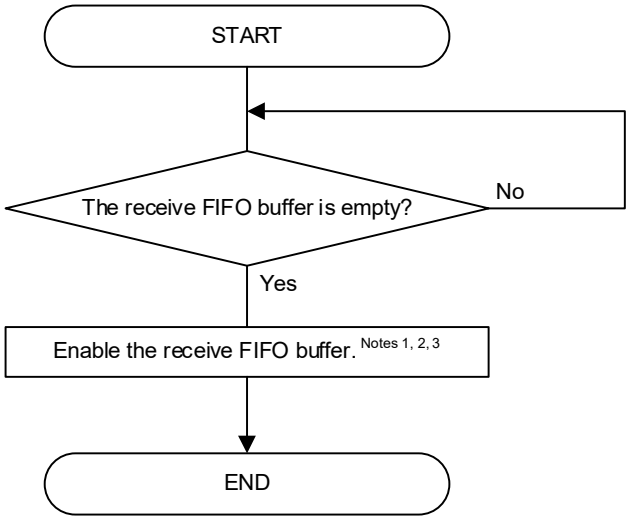
Note 7: When the DLC value (the RFDLC[3:0] bits in the RFPTRkH register) of the received message is smaller than the payload size of the receive FIFO buffer (the RFPLS[2:0] bits in the RFCCk register), data bytes where no data have been set are read as H'00. Do not read the RFDFk_pL register and the RFDFk_pH register that correspond to areas that exceed the size specified in the payload size setting.

Note 8: After reading the messages in the receive FIFO buffer (the RFIDkL/H registers, the RFPTRkL/H registers, the RFFDSTSkL/H registers, and the RFDFk_0L/H to RFDFk_15L/H registers), update the pointer (write H'FF to the RFPC[7:0] bits in the RFPCTRk register).

Note 9: When updating the pointer, the receive FIFO buffers must be used (the RFE bit in the RFCCk register is set to 1) and also the receive FIFO buffer needs to contain any unread message (when the RFEMP flag in the RFSTSk register is set to 0).

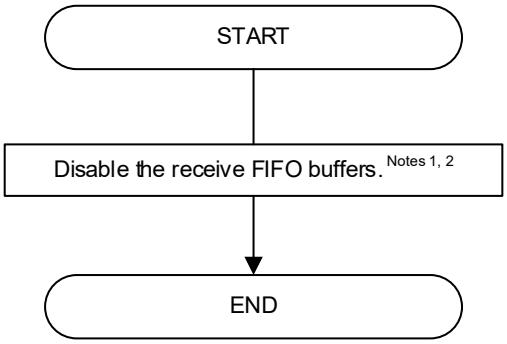
Note 10: To read all the unread messages of the receive FIFO buffer, repeat reading the messages using e.g. a loop statement until the buffer becomes empty (contains no message).

Figure 2.4 Receive FIFO Buffer Reading Procedure (no interrupt used) (2/2)



Note 1: To enable/disable the receive FIFO buffer, modify the value of the RFE bit in the RFCCk register when the CAN module is in global operating mode or global halt mode.
Note 2: Before enabling the receive FIFO buffer (set the RFE bit to 1), perform the configuration settings to use the receive FIFO buffer.
Note 3: When the number of receive FIFO buffers is set to 0 (write B'000 to the RFDC[2:0] bits in the RFCCk register), do not enable the receive FIFO buffers.

Figure 2.5 Procedures for Using Receive FIFO Buffers



Note 1: To enable/disable the receive FIFO buffer, modify the value of the RFE bit in the RFCCk register when the CAN module is in global operating mode or global halt mode.
Note 2: Even if the receive FIFO buffer is disabled (set the RFE bit to 0) while an interrupt request is present (the RFIF flag in the RFSTSk register is set to 1), the interrupt request flag (the RFIF flag) is not automatically set to 0. Clear the interrupt request flag by a program.

Figure 2.6 Proceeding for Disabling Receive FIFO Buffers

2.3.2 Processing for Receive FIFO-related Interrupts

(1) Receive FIFO interrupt processing

Once the receive FIFO interrupt is enabled, a receive FIFO interrupt will be generated when the conditions set by the RFIM bit in the RFCCk registers are met.

Even if the receive FIFO buffers are disabled (set the RFE bit to 0) while an interrupt request is present (the RFIF flag in the RFSTSk register is set to 1), the interrupt request flag (the RFIF flag) is not automatically set to 0. Clear the interrupt request flag by a program.

The receive FIFO interrupt can be enabled/disabled by the RFIE bit in the RFCCk register for each receive FIFO buffer. The following are the generation sources for the receive FIFO interrupt.

- When the conditions set by the RFIGCV[2:0] bits in the RFCCk register are met, the receive FIFO interrupt request will be issued (the RFIM bit in the RFCCk register is set to 0).
  Values set to the RFIGCV[2:0] bits:
  - B’000: the receive FIFO buffer is 1/8 full Note
  - B’001: the receive FIFO buffer is 2/8 full
  - B’010: the receive FIFO buffer is 3/8 full Note
  - B’011: the receive FIFO buffer is 4/8 full
  - B’100: the receive FIFO buffer is 5/8 full Note
  - B’101: the receive FIFO buffer is 6/8 full
  - B’110: the receive FIFO buffer is 7/8 full Note
  - B’111: the receive FIFO buffer is full.
- Every time one message is received, a receive FIFO interrupt request will be issued (the RFIM bit in the RFCCk register is set to 1).

Note: Do not set these values when the number of messages to be received in the receive FIFO buffers is set to 4 (when the value of the RFDC[2:0] bits in the RFCCk register is B’001).

To generate the receive FIFO interrupt, all the interrupt enable bits corresponding to the bits which have been set to 1 (listed in Table 6.2) need to be set to 0.

When the receive FIFO interrupt is used, confirm that all corresponding interrupt request flags have been set to 0 within interrupt servicing before ending the interrupt processing, refer to “Figure 4.2 CAN-related Interrupt Processing”.

(2) Global error interrupt handling

Once the FIFO message lost interrupt is enabled, a global error interrupt will be generated when a receive FIFO buffer message lost error is detected. The FIFO message lost interrupt can be enabled/disabled with the MEIE bit in the GCTRL register for the entier CAN module.

2.4 Reception Using Common FIFO Buffers

The common FIFO buffer can be used either in receive mode or transmit mode. (This section describes only the common FIFO buffer operating in receive mode.)

Each channel has one dedicated common FIFO buffer. Like the receive FIFO buffer, a single common FIFO buffer (set to receive mode) can retain messages up to the number equal to the number of receive buffers that the common FIFO buffer has.

When a received message has been stored in the common FIFO buffer set to receive mode, the value of the corresponding message count display counter (the CFMC[5:0] bits in the CFSTS register) is incremented.

The received messages can be read out from the CFIDL and CFIDH registers, the CFPTRL and CFPTRH registers, the CFFDCSTSL and CFFDCSTSH registers, and CFDF0L/H to CFDF15L/H registers. Messages in the common FIFO buffers can be read sequentially on a first-in, first-out basis.

When the value of the message count display counter matches the number of messages to be stored in the common FIFO buffer (a value set by the CFDC[2:0] bits in the CFCCH register), the common FIFO buffer is full (the CFFLL flag in the CFSTS register is set to 1).

When all the messages have been read out from the common FIFO buffer, the common FIFO buffer becomes empty (contains no message) (the CFEMP flag in the CFSTS register is set to 1).

Regarding the configuration to use the common FIFO buffer, see “1.1 CAN Configuration”.

Figure 2.7 illustrates the receiving operation of the common FIFO buffer.

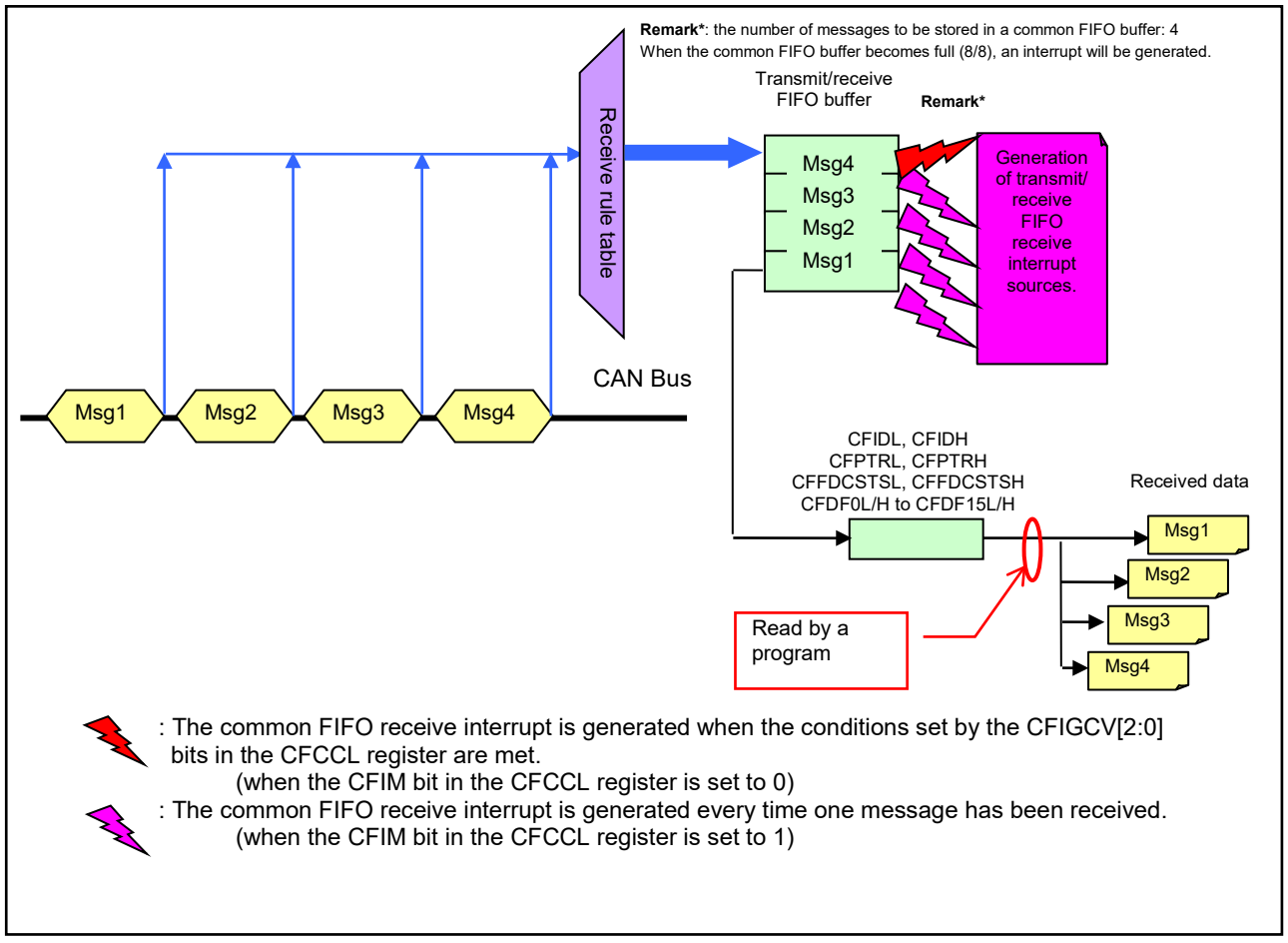
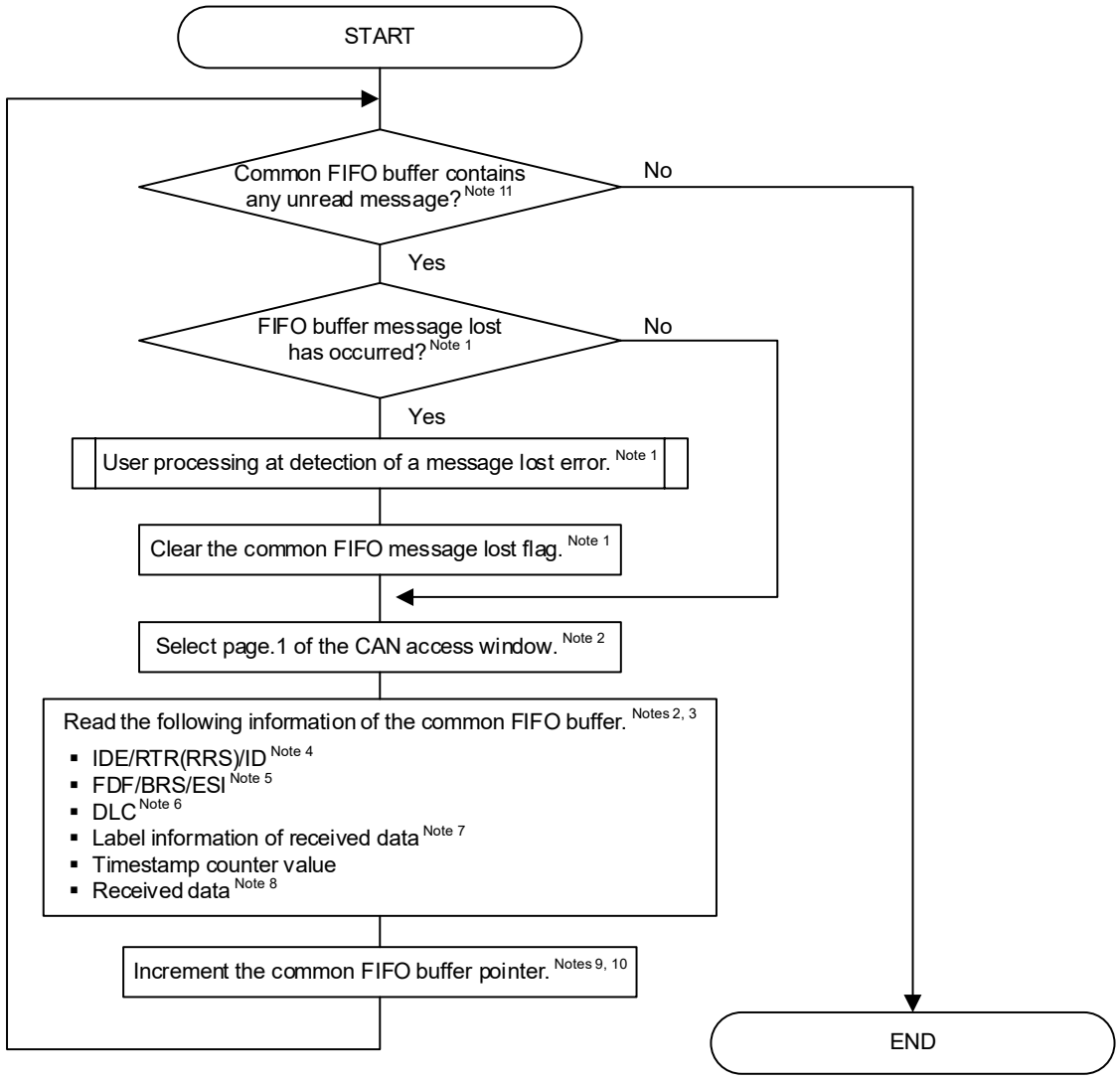


Figure 2.7 Operation of Common FIFO Buffer (in receive mode)

2.4.1 Procedures for Reading Common FIFO Buffers

Figure 2.8 shows the procedures for reading the common FIFO buffers. Figure 2.9 and Figure 2.10 respectively show the procedures for enabling and disabling the common FIFO buffers.



Note 1: When the FIFO message lost interrupt is enabled, these steps need to be done while the global error interrupt is being handled.

Note 2: The common FIFO buffers (the CFIDL/H registers, the CFPTRL/H register, the CFFDCSTSL/H register and the CFDF0L/H to CFDF15L/H registers) can be read only when receive mode is selected (the value of the CFM bit in the CFCCL register is 0) and also the window 1 for the CAN module is selected (the RPAGE[1:0] bits in the CFDGRWC register is set to B'01).

Note 3: In receive mode, it is impossible to enable/disable (the THLEN bit in the CFIDH register) the storage of transmit history data.

Note 4: When the standard ID is selected, read bits 10 to 0 of the ID data (the CFID[15:0] bits in the CFIDL register). The bits 15 to 11 of the ID data (the CFID[15:0] bits in the CFIDL register) and the CFID[28:16] bits in the CFIDH register can be read as 0.

Note 5: For the Classical CAN frames (the CFFDF bit in the CFFDSTSkL register is 0), the CFBRS bit and the CFESI bit in the CFFDSTSkL register can be read 0.

Figure 2.8 Procedures for Reading Common FIFO Buffer (in receive mode) (no interrupt used) (1/2)

Note 6: When the DLC replacement is enabled (the DCE and DRE bits in the GCFGL register are both set to 1) after the filter processing according to the receive rules, the DLC value specified in the receive rule table (the GAFLDLC[3:0] bits in the GAFLP0iL register) which has agreed with a DLC value of the received message will be stored in place of the DLC value of the received message. In other cases, the DLC value of the received message will be stored without the replacement of the DLC value.

Note 7: After the filter processing according to the receive rules, the value set to the label data of the receive rule table (the GAFLPTR[15:0]bits in the GAFLP0iH register, the GAFLIFL0 bit int the GAFLP0iL register and the GAFLIFL1 bit in the GAFLMiH register) which has agreed with the value of the received message will be stored.

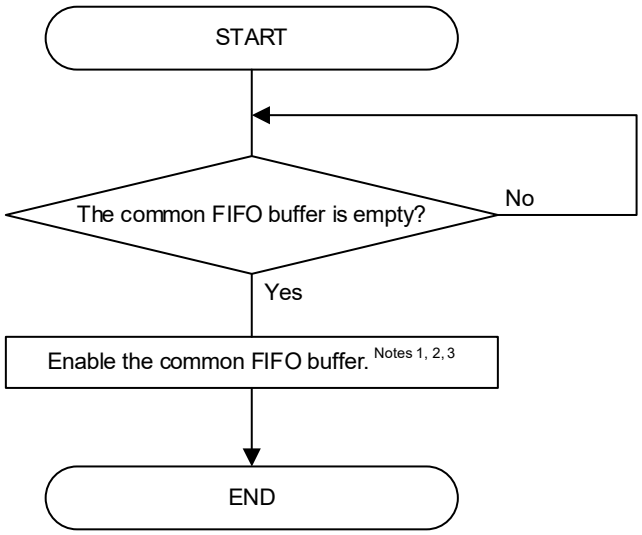
Note 8: When the DLC value (the CFDLC[3:0] bits in the CFPTRkH register) of the received message is smaller than the payload size of the common FIFO buffer (the CFPLS[2:0] bits in the CFCCL register), data bytes where no data have been set are read as H'00. Do not read the CFDFpL register and the CFDFpH register that correspond to areas that exceed the size specified in the payload size setting.

Note 9: After reading out the messages of the common FIFO buffer (the CFIDL/H registers, the CFPTRL/H registers, and the CFDF0L/H to CFDF15L/H registers), update the pointer (write H'FF to the CFPC[7:0] bits in the CFPCTR register).

Note 10: When updating the pointer, the common FIFO buffer must be used (the CFE bit in the CFCCL register is set to 1) and also the common FIFO buffer needs to contain an unread message (the CFEMP flag in the CFSTS register is set to 0).

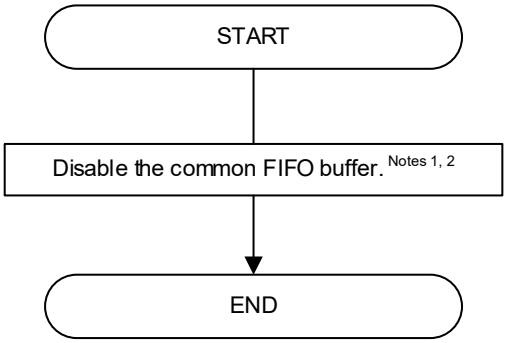
Note 11: To read all the unread messages stored in the common FIFO buffer, repeat reading all the messages using e.g. a loop statement until the buffer becomes empty (contains no message).

Figure 2.8 Procedures for Reading Common FIFO Buffer (in receive mode) (no interrupt used) (2/2)



Note 1: In receive mode, enable/disable the common FIFO buffer (the CFE bit in the CFCCL register) when the CAN module is in global operating mode or global halt mode.
Note 2: Before using the common FIFO buffer (set the CFE bit to 1), perform the CAN configuration.
Note 3: When the number of common FIFO buffers is set to 0 (when the value of the CFDC[2:0] bits in the CFCCH register is B'000), do not enable the common FIFO buffers.

Figure 2.9 Procedures for Enabling Common FIFO Buffers



Note 1: In receive mode, enable/disable the common FIFO buffer (the CFE bit in the CFCCL register) when the CAN module is in global operating mode or global halt mode.
Note 2: Even if the common FIFO buffer is disabled (set the CFE bit to 0) while an interrupt request is present (the CFRXIF flag in the CFSTS register is set to 1), the interrupt request flag (CFRXIF flag) is not automatically set to 0. Clear the interrupt request flag by a program.

Figure 2.10 Procedures for Disabling Common FIFO Buffers

2.4.2 Interrupt Handling for Common FIFO Buffers (in receive mode)

(1) Common FIFO receive interrupt handling

Once the common FIFO receive interrupt is enabled, the common FIFO receive interrupt is generated when the conditions set by the CFIM bit in the CFCCL register are met.

Even if the common FIFO buffers are disabled (set the CFE bit in the CFCCL register to 0) while an interest request is present (the CFRXIF flag in the CFSTS register is set to 1), the interrupt request flag (CFRXIF) is not automatically set to 0. Clear the interrupt request flag by a program.

The common FIFO receive interrupt can be enabled/disabled for each common FIFO buffer with the CFRXIE bit in the CFCCL register.

The following are the generation sources for the common FIFO receive interrupt in receive mode:

- When the number of received messages amounts to the number specified by the CFIGCV[2:0] bits in the CFCCL register, the common FIFO receive interrupt request is generated (the CFIM bit in the CFCCL register is set 0).
  Values set to the CFIGCV[2:0] bits:
  - B’000: the common FIFO buffer is 1/8 full Note
  - B’001: the common FIFO buffer is 2/8 full
  - B’010: the common FIFO buffer is 3/8 full Note
  - B’011: the common FIFO buffer is 4/8 full
  - B’100: the common FIFO buffer is 5/8 full Note
  - B’101: the common FIFO buffer is 6/8 full
  - B’110: the common FIFO buffer is 7/8 full Note
  - B’111: the common FIFO buffer is full
- Every time one message is received, the common FIFO receive interrupt request is generated (the CFIM bit in the CFCCL register is set to 1).

Note: Do not set these values when the number of messages to be received in the common FIFO buffer is set to 4 (the value of the DFDC [2:0] bits in the CFCCH register is B’001).

To enable the generation of the common FIFO receive interrupt, all the corresponding interrupt request bits which have been set to 1 (listed in Table 6.2) need to be set to 0.

When the common FIFO receive interrupt is used, confirm that all corresponding interrupt request flags have been set to 0 within interrupt servicing before ending the interrupt processing, refer to “Figure 4.2 CAN-related Interrupt Processing”.

(2) Global error interrupt handling

Once the FIFO message lost interrupt is enabled, a global error interrupt will be generated when a message lost error of the common FIFO buffer is detected. The FIFO message lost interrupt can be enabled/ disabled collectively for the entire CAN module using the MEIE bit in the GCTRL register.

3. Transmission

3.1 Transmission Function

There are the following functions to transmit CAN messages. For details, refer to the following sections:

- Transmission using transmit buffers
- Transmission using common FIFO buffers
- Transmit history list buffer

3.2 Transmission Using Transmit Buffers

Data frames or remote frames are transmitted using transmit buffers.

One channel has four transmit buffers which can be used as a transmit buffer itself or be linked to the common FIFO buffer (set to transmit mode).

When the transmit buffers are linked to the common FIFO buffer (set to transmit mode), write H'00 to the corresponding TMCm register and set the TMIEm bit in the TMIEC register to 0 (interrupt disabled). In this case, the corresponding flags of the corresponding TMSTSm, TMTRSTS, TMTARSTS, TMTCSTS, and TMTASTS registers will not be overwritten.

The transmit buffers have the following functions. Regarding the configuration to use the transmit buffers, see "1.1 CAN Configuration":

- Message transmission
- Transmit abort function
- One-shot transmission function (retransmission-disabling function)

3.2.1 Message transmission function

This is a function to transmit data frames or remote frames.

By setting a transmit request to the transmit buffer (set the TMTR bit in the TMCm register to 1), message transmission is enabled.

The transmission result can be confirmed by the TMTRF[1:0] flag in the corresponding TMSTSm register. When the transmission completes successfully, the value of the TMTRF[1:0] flag is B'10 (transmission has been completed [without a transmit abort request]), or the value of the TMTRF[1:0] flag is B'11 (transmission has been completed [with a transmit abort request]). For the case in which the value of the TMTRF[1:0] flag is B'11 (transmission has been completed [with a transmit abort request]), see "3.2.2 Transmit Abort Function".

The interrupt for the transmit completion can be enabled/disabled for each buffer with the TMIEm bit in the TMIEC register.

Figure 3.1 illustrates the operation of transmit buffers.

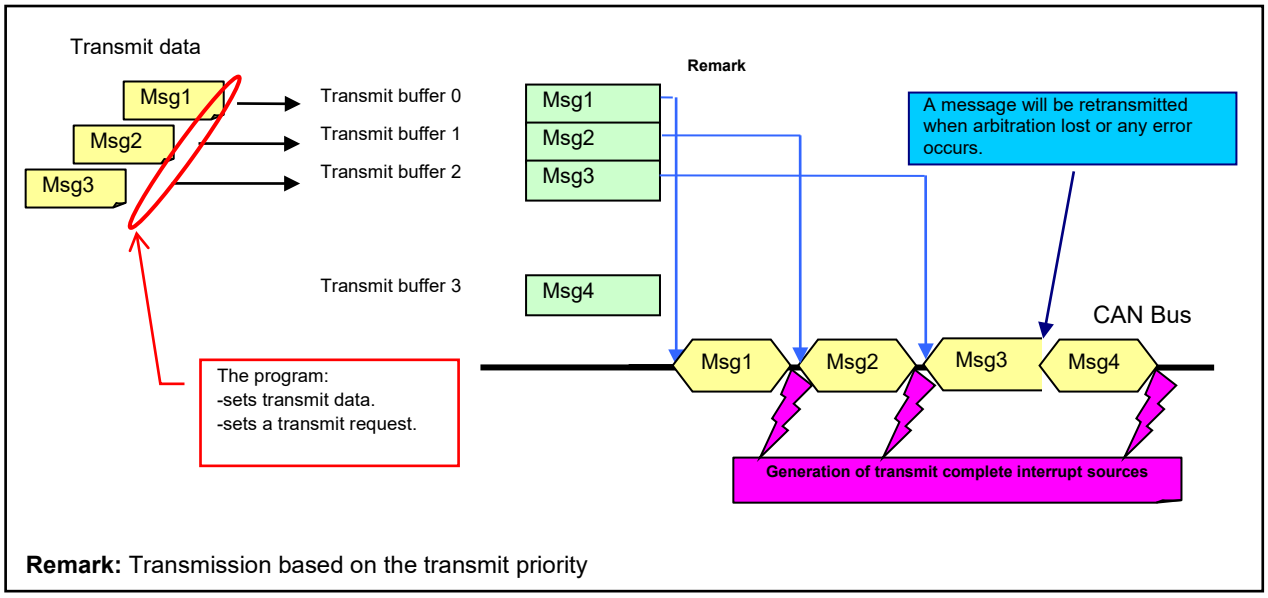
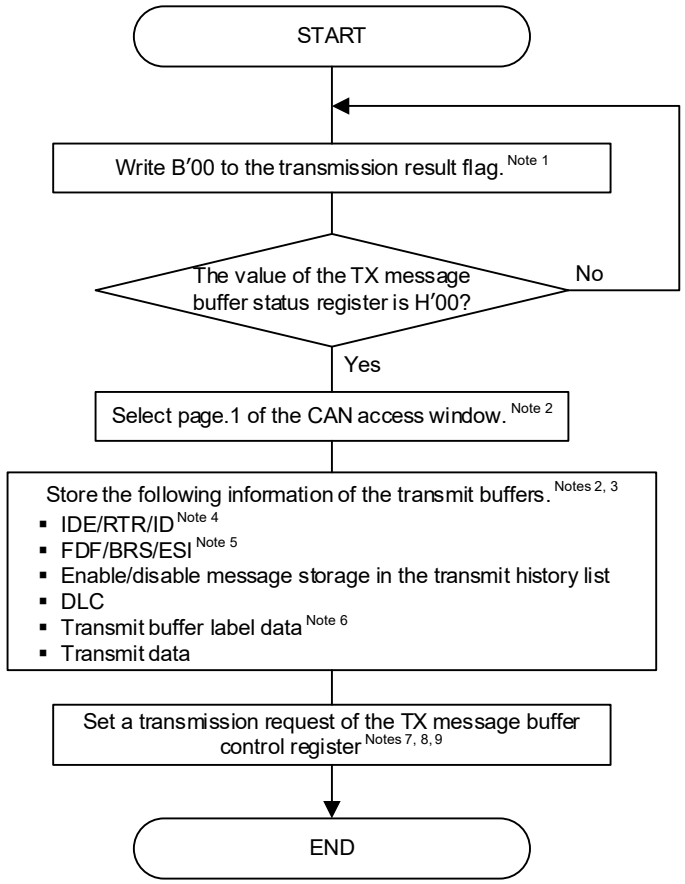


Figure 3.1 Operation of Transmit Buffers (transmission from channel 0)

(1) Procedures for transmitting messages from transmit buffers

Figure 3.2 shows the procedures for transmitting messages from transmit buffers.



Note 1: Write B'00 to the TMTRF[1:0] flag in the TMSTSm register when the CAN module is in channel operation mode or channel halt mode. Do not write any values other than B'00.

Note 2: Modify the values of the TMIDmL/H registers, the TMPTRmH register, the TMFDCTRmL/H registers, and the TMDFm_0L/H to TMDFm_15L/H registers when a transmission request is not issued for the corresponding transmit buffers (the TMTRM bit in the TMSTSm register is set to 0) and also page 1 of the CAN access window is selected (the RPAGE[1:0] bits in the CFDGRWC register is set to B'01).

Note 3: Do not write data to the TMIDmL/H registers, the TMPTRmH register, the TMFDCTRmL/H registers, and the TMDFm_0L/H to TMDFm_15L/H registers, each of which is linked to the common FIFO buffer.

Note 4: To set the standard ID to the transmit ID (the TMID[15:0] bits in the TMIDmL register), set the ID data to bits 10-0. Set 0 to bits 15-11 in the TMIDmL register and the TMID[28:16] bits in the TMIDmH register.

Note 5: When set to the CAN-FD frame (the TMFDF bit in the TMFDCTRmL register is set to 1), the RRS bit is sent dominant regardless of the value of the TMRTR bit in the TMIDmH register.

Note 6: When set to the Classical CAN frame (the TMFDF bit in the TMFDCTRmL register is set to 0), the TMBRS and TMESI bits in the TMFDCTRmL register must be set to 0.

Note 7: This can be stored only when the messages are to be stored in the transmit history list buffers (the THLDTE bit and the THLE bit in the THLCC register are both set to 1 and the THLEN bit in the TMIDmH register is set to 1).

Note 8: Modify the values of the TMCm register when the CAN module is in channel operation mode or channel halt mode.

Note 9: Write H'00 to the TMCm registers in the following case:
When a transmit buffer is linked to the common FIFO buffer

Note 10: Set a transmission request (set the TMTR bit in the TMCm register to 1) while a transmission request is not present (the value of the TMTRF[1:0] flag in the TMSTSm register is B'00).

Figure 3.2 Procedures for transmitting messages from transmit buffers

3.2.2 Transmit Abort Function

When two or more nodes start transmission simultaneously, arbitration lost occurs in a node transmitting the lowest CAN ID priority message. [In one-shot transmission, the message transmission is aborted. Meanwhile, in normal transmission, the message transmission is hold (retransmitted)]. Unless the arbitration result is a “win” or a message is transmitted while the CAN Bus is idle, message transmission will not be completed successfully.

To deal with the cases in which the arbitration result is not a “win” or a message is transmitted while the CAN Bus is not idle, a transmission abort function to discard the message which is being retransmitted is provided. This transmission abort function can be used to specify a limited transmission time for one message transmission or to preferentially transmit a message having a higher priority due to its urgent need.

Figure 3.3 shows an application example of the transmit abort function.

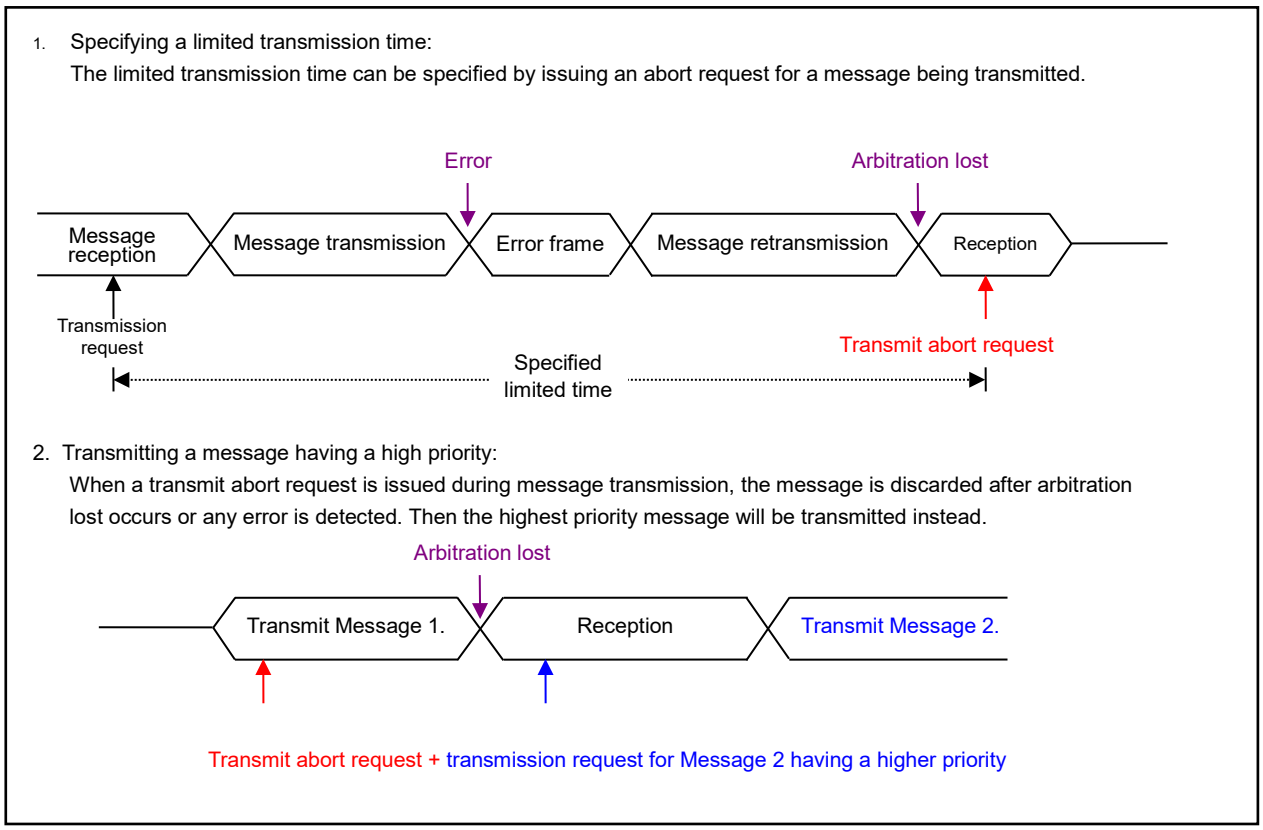


Figure 3.3 Application Example of Transmit Abort Function

When a transmit abort request is issued for the transmit buffer (the TMTAR bit in the TMCm register is set to 1) having a transmission request (the TMTRM bit in the TMSTSm register is set 1), the transmission request is canceled.

After the transmit abort request is issued, the transmission is canceled as described below:

- A message which is being transmitted or a message which will be transmitted next based on the transmit priority determination
  - When arbitration lost occurs
  - When any error occurs.
- Messages other than the above-mentioned
  - When a transmit abort request is issued.

When transmit abort has been completed, the value of the TMTRF[1:0] flag in the TMSTSm register is B'01. Then the transmit request is canceled (the TMTRM bit is set to 0).

When transmission is completed without arbitration lost or any errors after a transmit abort request had been issued for a message which is being transmitted or will be transmitted next based on the transmit priority determination, the transmission has been completed successfully (with a transmit abort request: the value of the TMTRF[1:0] flag is B'11).

Figure 3.4 illustrates the operation at transmit abort.

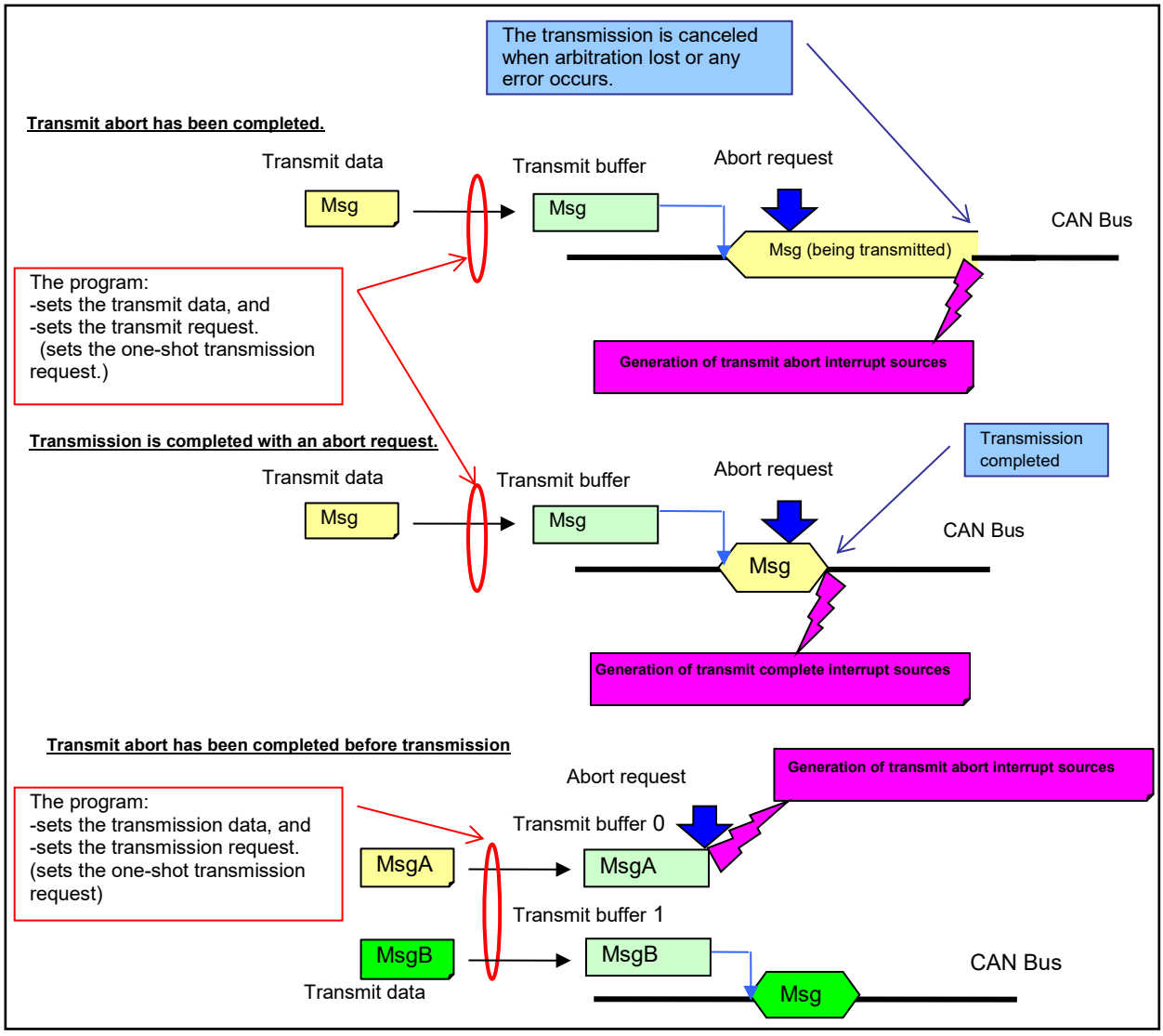
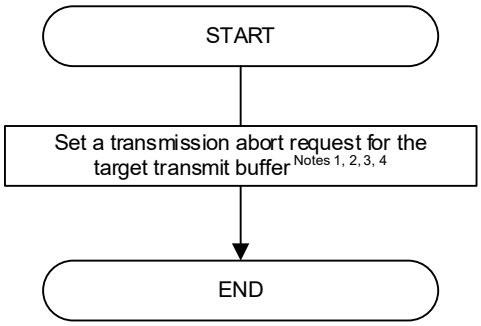


Figure 3.4 Operation when Transmission is Aborted

(1) Transmit abort procedure

Figure 3.5 shows the procedures for aborting message transmission.



Note 1: Modify the value of the TMCm register when the CAN module is in channel operation mode or channel halt mode.

Note 2: Write H'00 to the TMCm register in the following case:
When a transmit buffer is linked to common FIFO buffer

Note 3: A transmit abort request can be issued (the TMTAR bit of the TMCm register is set to 1) while a transmit request for the transmit buffer is present (the TMTR bit in the TMCm register is set to 1).

Note 4: Depending on the timing of this setting, the transmit result may be that the transmission has been successfully completed (with a transmit abort request [the value of the TMTRF[1:0] flag in the TMSTSm register is B'11]). Therefore, to determine the transmit result, check whether the value of the TMTRF[1:0] flag is B'01 or B'11. Regarding the procedures for confirming transmission completion and transmission abort completion, refer to "3.2.4 (1) Transmit complete interrupt handling".

Figure 3.5 Transmission Abort Procedure

3.2.3 One-shot Transmission Function

When one-shot transmission is enabled (the TMOM bit in the TMCm register is set to 1) while a message transmit request is present, transmission is carried out only once. Even if arbitration lost or any error occurs, retransmission will not be performed.

The result of one-shot transmission can be confirmed with the TMTRF[1:0] flag in the TMSTSm register. When one-shot transmission has been successfully completed, the transmission result is that the transmission has been completed (without a transmit abort request [the value of the TMTRF[1:0] flag is B'10]) or that the transmission has been completed (with a transmit abort request [the value of the TMTRF[1:0] flag is B'11]). When arbitration lost or any error occurs, the transmit abort has been completed (the value of the TMTRF[1:0] flag is B'01). (Regarding the transmission result "the transmission has been completed (with a transmit abort request (the value of the TMTRF[1:0] flag is B'11)", see "3.2.2 Transmit Abort Function".

Figure 3.6 illustrates the operation of one-shot transmission.

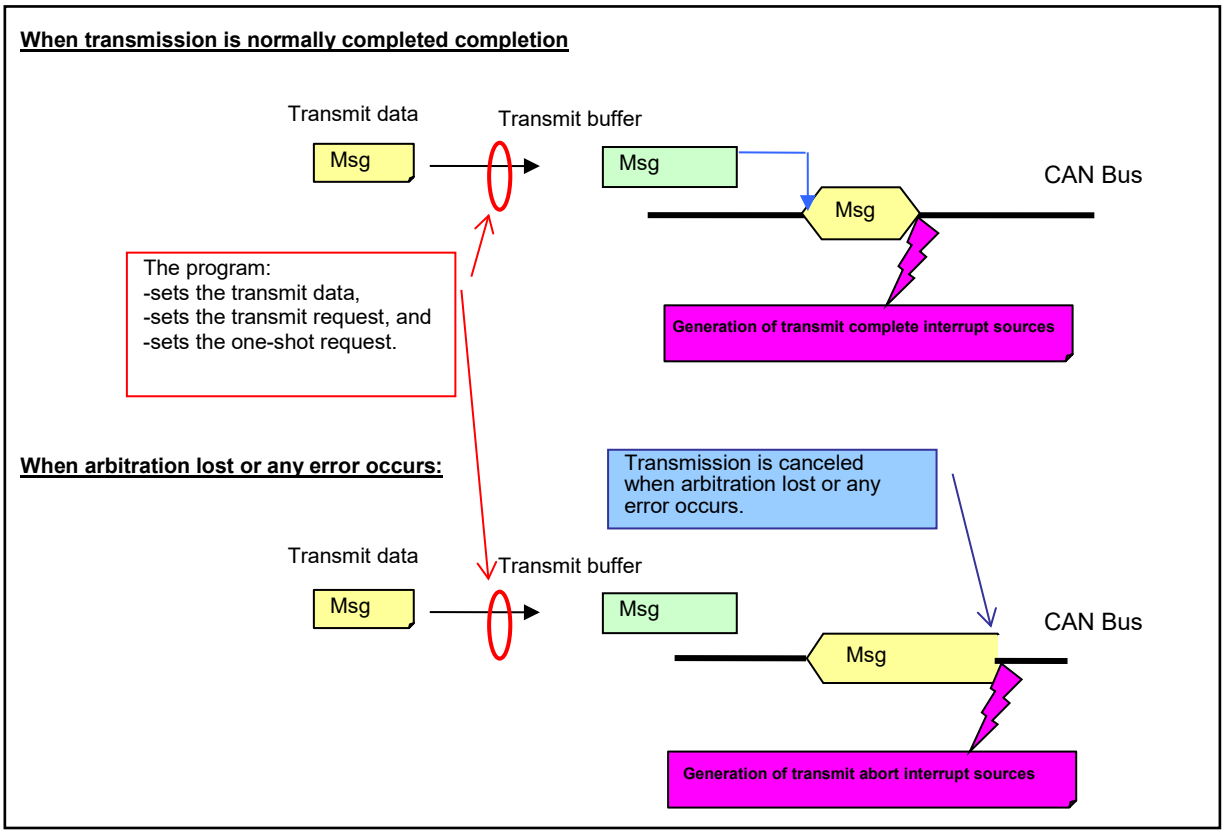
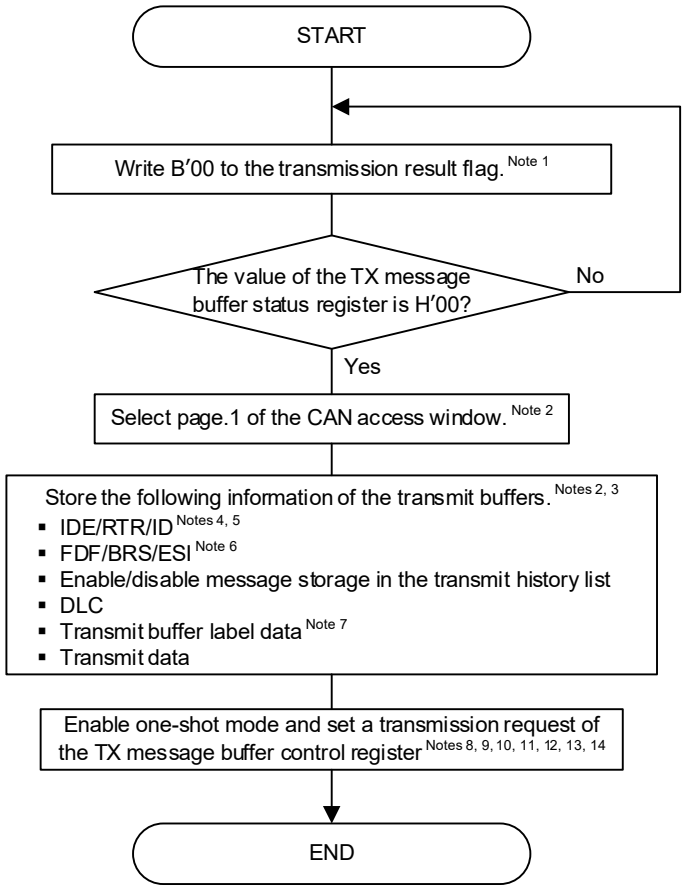


Figure 3.6 Operation of One-shot Transmission

(1) One-shot transmission procedures

Figure 3.7 shows the procedures for one-shot transmission.



Note 1: Write B'00 to the TMTRF[1:0] flag in the TMSTSm register when the CAN module is in channel operation mode or channel halt mode. Do not write any values other than B'00.

Note 2: Modify the values of the TMIDmL/H registers, the TMPTRmH register, and the TMDFm_0L/H to TMDFm_15 L/H registers when no transmission request is issued for the corresponding transmit buffer(s) (the TMTRM bit in the TMSTSm register is set to 0) and also page 1 of the CAN access window is selected (the RPAGE[1:0] bits in the CFDGRWC register is set to B'01).

Note 3: Do not write data to the TMIDmL and TMIDmH registers, the TMPTRmH register and TMDFm_0 to TMDFm_15 registers, each of which is linked to the common FIFO buffer.

Note 4: To set the standard ID to the transmit ID (the TMID[15:0] bits in the TMIDmL register), set the ID data to bits 10-0. Set 0 to bits 15-11 in the TMIDmL register and the TMID[28:16] bits in the TMIDmH register.

Note 5: When set to the CAN-FD frame (the TMFDF bit in the TMFDCTRmL register is set to 1), the RRS bit is sent dominant regardless of the value of the TMRTR bit in the TMIDmH register.

Note 6: When set to the Classical CAN frame (the TMFDF bit in the TMFDCTRkL register is set to 0), the TMBRS and TMESI bits in the TMFDCTRkL register must be set to 0.

Note 7: This can be stored only when transmit history data is stored in the transmit history list buffer (the THLDTE and THLE bits in the THLCC register are both set to 1 and the THLEN bit in the TMIDmH register is set to 1).

Note 8: Modify the value of the TMCm register when the CAN module is in channel operation mode or channel halt mode.

Note 9: Write H'00 to the TMCm register in the following case:
When a transmit buffer is linked to a common FIFO buffer

Note 10: A transmit request can be set (set the TMTR bit in the TMCm register to 1) when no transmit request is present (the value of the TMTRF[1:0] flag of the TMSTSm register is B'00).

Note 11: Even if the transmission fails, the retransmission specified in the CAN protocol will not be performed.

Figure 3.7 One-shot Transmission Procedures (1/2)

Note 12: Enable the one-shot transmission (set the TMOM bit in the TMCm register to 1) when no transmit request is present for the transmit buffer (set the TMTRM bit in the TMSTSm register to 0).

Note 13: To enable the one-shot transmission, simultaneously set the transmit request (set the TMTR and TMOM bits to 1).

Note 14: Depending on the timing of these settings, the transmit result may be that the transmission has been successfully completed (with a transmit abort request [the value of the TMTRF[1:0] flag in the TMSTSm register is B'11]). Therefore, to determine the transmit result, check whether the value of the TMTRF[1:0] flag is B'01 or B'11. Regarding the procedures for confirming transmission completion and transmit abort completion, see "3.2.4 (3) Processing after transmission/transmission abort are completed".

Figure 3.7 One-shot Transmission Procedures (2/2)

3.2.4 Interrupt Handling for Transmit Buffers

(1) Transmit complete interrupt handling

Once the transmit complete interrupt is enabled, the CAN0 transmit interrupt will be generated when the transmission is completed. The transmit complete interrupt is enabled/disabled with the TMIEm bit in the TMIEC register for each transmit buffer.

The following are the sources for the CAN0 transmit interrupt. When two or more sources are used for the interrupt, identify each source while the interrupt is being handled as needed.

The generation sources for the CAN0 transmit interrupt can also be confirmed by the GTINTSTS register.

- CAN0 transmit complete interrupt
- CAN0 transmit abort interrupt
- CAN0 common FIFO transmit complete interrupt
- CAN0 transmit history interrupt

To generate the CAN0 transmit interrupt, all the interrupt enable bits corresponding to the bits which have been set to 1 (listed in Table 6.2) need to be set to 0.

When the CAN0 transmit interrupt is used, confirm that all corresponding interrupt request flags have been set to 0 within interrupt servicing before ending the interrupt processing, refer to “Figure 4.2 CAN-related Interrupt Processing”.

(2) Transmit abort completion interrupt handling

Once the transmit abort interrupt is enabled, the CAN0 transmit interrupt will be generated when the transmit abort has been completed. The transmit abort interrupt can be enabled/disabled with the TAIE bit in the C0CTRH register for each channel. However, when the transmission has been completed (with an abort request [the value of the TMTRF[1:0] flag is B’11), a transmit abort interrupt will not be generated but a transmit complete interrupt will be generated.

The following are the sources for the CAN0 transmit interrupt. When two or more generation sources are used for the interrupt, identify each source while the interrupt is being handled as needed.

The generation sources for the CAN0 transmit interrupt can also be confirmed by the GTINTSTS register.

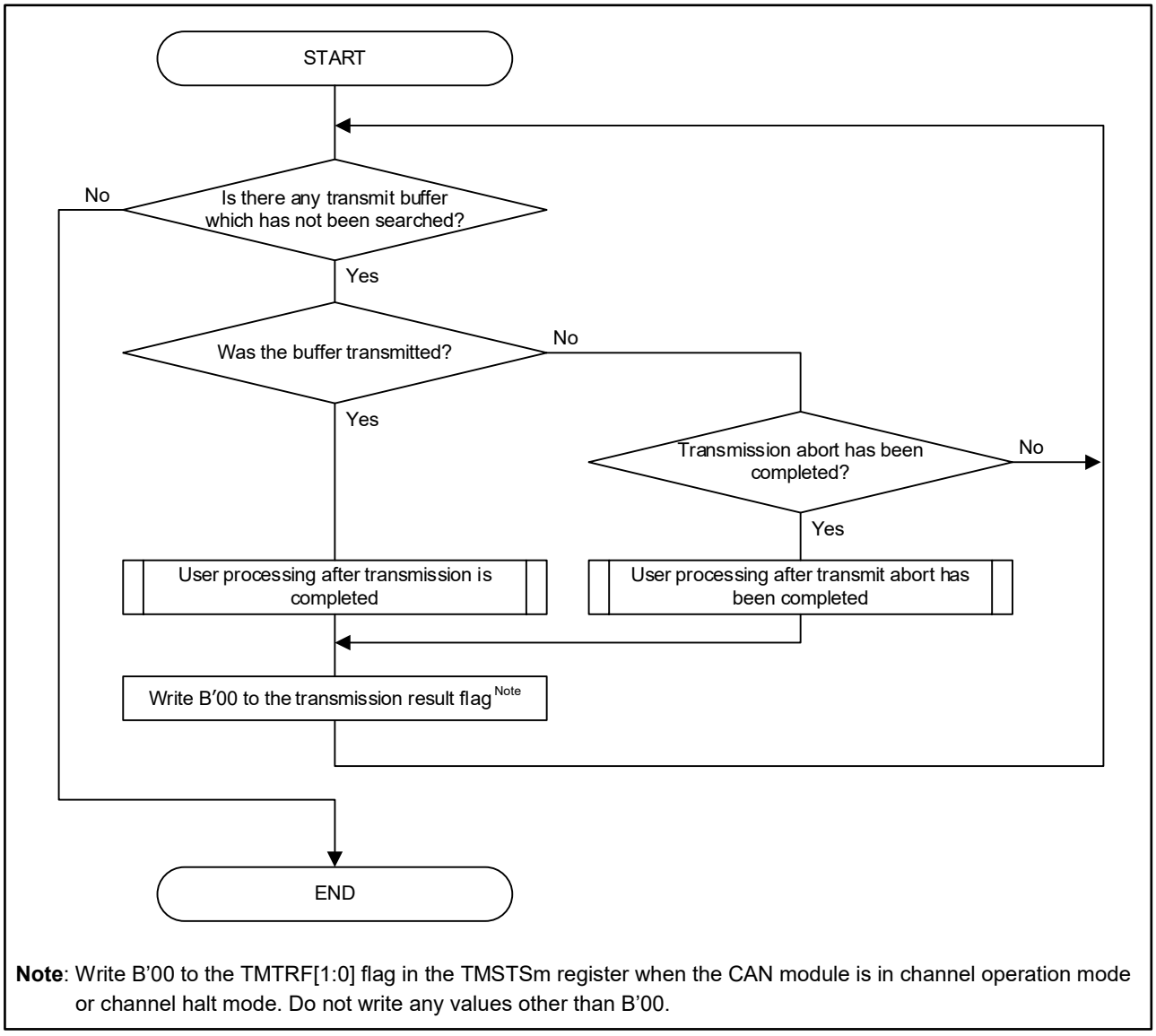
- CAN0 transmit complete interrupt
- CAN0 transmit abort interrupt
- CAN0 common FIFO transmit complete interrupt
- CAN0 transmit history interrupt

To generate the CAN0 transmit interrupt, all the interrupt enable bits corresponding to the bits which have been set to 1 (listed in Table 6.2) need to be set to 0.

When the CAN0 transmit interrupt is used, confirm that all corresponding interrupt request flags have been set to 0 within interrupt servicing before ending the interrupt processing, refer to “Figure 4.2 CAN-related Interrupt Processing”.

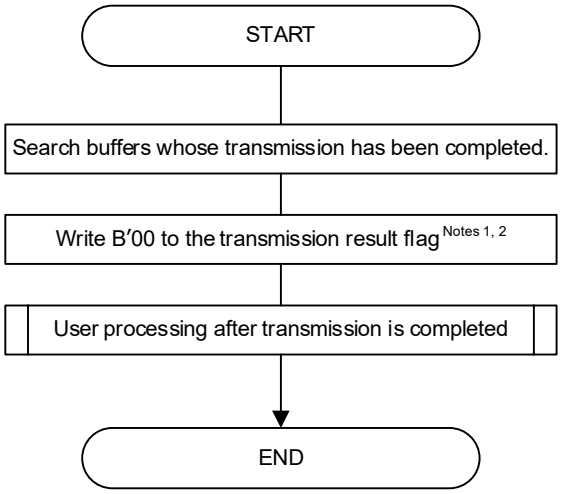
(3) Processing after transmission/transmission abort are completed

Figure 3.8, Figure 3.9 and Figure 3.10 show the procedures after transmission and transmit abort are completed.



Note: Write B'00 to the TMTRF[1:0] flag in the TMSTSm register when the CAN module is in channel operation mode or channel halt mode. Do not write any values other than B'00.

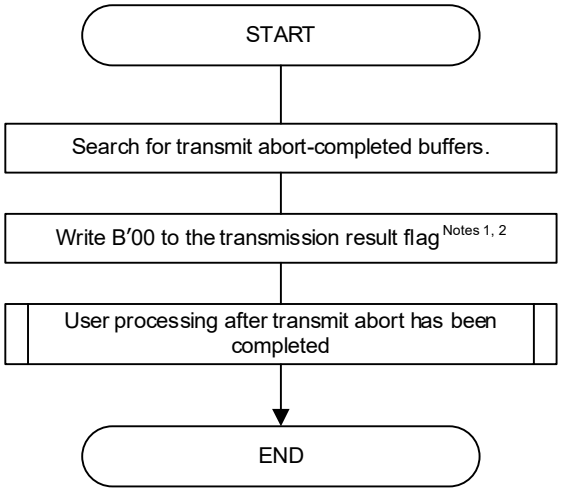
Figure 3.8 Processing After Transmit/Transmit Abort is Completed (no interrupt used)



Note 1: Write B'00 to the TMTRF[1:0] flag in the TMSTSm register when the CAN module is in channel operation mode or channel halt mode. Do not write any values other than B'00 to this flag.

Note 2: Refer to Figure 4.2 regarding the procedures for handling interrupt source-related flags while interrupts are used.

Figure 3.9 Processing After Transmit is Completed (when interrupts are used)



Note 1: Write B'00 to the TMTRF[1:0] flag in the TMSTSm register when the CAN module is in channel operation mode or channel halt mode. Do not write any values other than B'00 to this flag.

Note 2: Refer to Figure 4.2 regarding the procedures for handling interrupt-related flags while interrupts are used.

Figure 3.10 Processing After Transmit Abort is Completed (when interrupts are used)

3.3 Transmission Using Common FIFO Buffer

Data frames or remote frames will be transmitted using a common FIFO buffer.

One channel has one common FIFO buffer that can store a maximum of 16 messages. The messages are transmitted sequentially on a first-in, first-out basis.

The common FIFO buffer can be used in either receive mode or transmit mode. (This section describes the common FIFO buffer in transmit mode only).

The common FIFO buffer is linked to transmit buffers (set by the CFTML[1:0] bits in the CFCCH register). When the common FIFO buffer is used (the CFE bit in the CFCCL register is set to 1), messages stored in the common FIFO buffer are to be checked for transmit priority. The transmit priority determination processing is carried out only for the messages to be transmitted next.

The common FIFO buffer has the following transmission functions. Regarding the configuration using the common FIFO buffer, see “1.1 CAN Configuration”.

- Message transmission function
- Transmit abort function
- Interval transmission function

3.3.1 Message Transmission Function

This is a function to transmit data frames or remote frames. Messages stored in the common FIFO buffer are transmitted sequentially on a first-in, first-out basis.

Figure 3.11 illustrates the transmission operation of the common FIFO buffer.

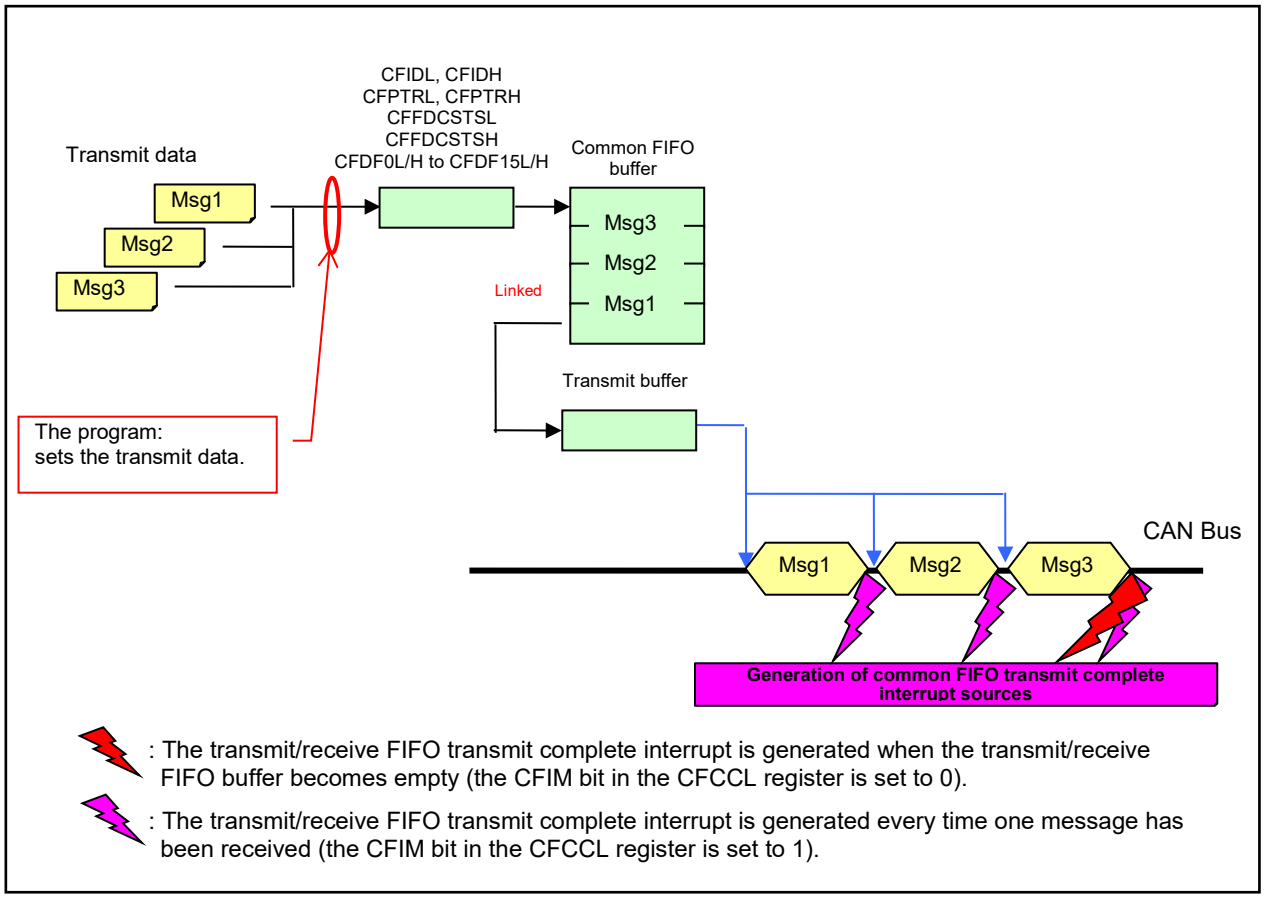
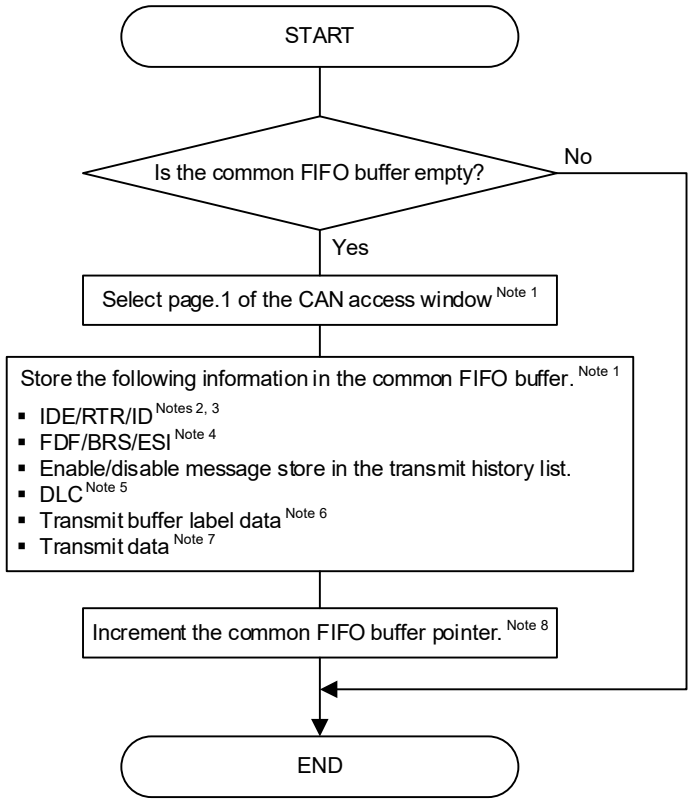


Figure 3.11 Operation of Common FIFO Buffer (in transmit mode)

(1) Procedures for transmitting messages from common FIFO buffers

Figure 3.12 shows the procedures for transmitting messages from the common FIFO buffer. Figure 3.13 and Figure 3.14 show respectively the procedures for enabling and disabling the common FIFO buffer.



Note 1: The values of the CFIDL/H registers, the CFPTRL/H register, and the CFDF0L/H to CFDF15L/H regsiters can be modified only when the common FIFO buffer is in transmit mode (the value of the CFM bit in the CFCCL register is 1) and also page 1 of the CAN access window is selected (the RPAGE[1:0] bits in the CFDGRWC register is set to B'01).

Note 2: In transmit mode, to set the standard ID to the transmit ID (the CFID[15:0] bits in the CFIDL register), set the ID data to bits 10-0. Set 0 to bits 15-11 and the CFID[28:16] bits in the CFIDH register.

Note 3: When set to the CAN-FD frame (the CFFDF bit in the CFFDSTSL register is set to 1), the RRS bit is sent dominant regardless of the value of the CFRTR bit in the CFIDH register.

Note 4: When set to the Classical CAN frame (the CFFDF bit in the CFFDSTSL register is set to 0), the CFBRS and CFESI bits in the CFFDCSTSL register must be set to 0.

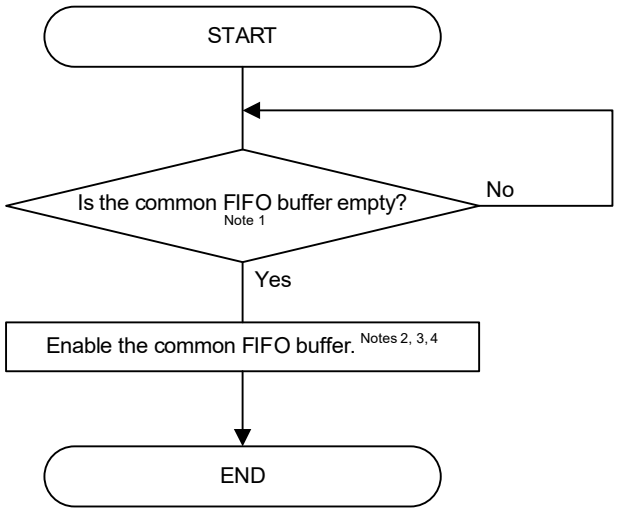
Note 5: When the DLC (the CFDLC[3:0] bits in the CFPTRH register) of the transmitted message exceeds the payload size of the common FIFO buffer (the CFPLS[2:0] bits of the CFCCL register), the excess payload is padded with H'CC.

Note 6: This can be stored only when the common FIFO buffer is in transmit mode and messages are to be stored in the transmit history list buffer (the THLE bit in the THLCC is set to 1 and the THLEN bit in the CFIDH register is set to 1).

Note 7: Do not write the CFDFpL register or the CFDFpH register that correspond to areas exceeding the size specified in the payload size setting (the CFPLS[2:0] bits in the CFCCL register).

Note 8: Increment the common FIFO buffer pointer when the conditions below are satisfied.
(Write H'FF to the CFPC bit in the CFPCTR register).
- the common FIFO buffer is in transmit mode
- the common FIFO buffer is used (the CFE bit in the CFCCL register is set to 1),
- after a message is stored in the common FIFO buffer
- the common FIFO buffer is not full (the CFFLL flag in the CFSTS register is set to 0)

Figure 3.12 Procedures for Transmitting Messages from Common FIFO Buffer



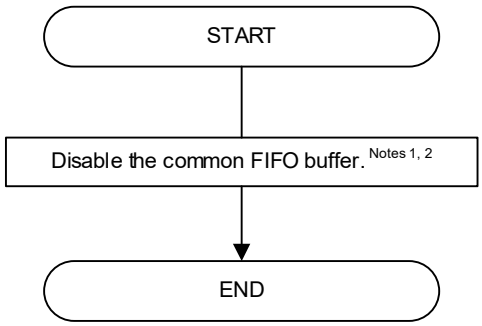
Note 1: In transmit mode, if the common FIFO buffer is disabled while a message is being transmitted or when a message to be transmitted next has been determined, the common FIFO buffer will be empty after:
- the transmission is completed,
- a CAN Bus error is detected, or
- arbitration lost occurs.

Note 2: In transmit mode, enable/disable the common FIFO buffer with the CFE bit in the CFCCL register while the CAN module is in channel operation mode or channel halt mode.

Note 3: Before enabling the common FIFO buffer (set the CFE bit to 1), perform the configuration to use the common FIFO buffer.

Note 4: When the number of common FIFO buffers is set to 0 (write B'000 to the CFDC[2:0] bits in the CFCCH register), do not enable the receive FIFO buffers.

Figure 3.13 Procedures for Enabling Common FIFO Buffer



Note 1: In transmit mode, enable/disable the common FIFO buffer with the CFE bit in the CFCCL register while the CAN module is in channel operation mode or channel halt mode.

Note 2: Even if the common FIFO buffer is disabled (set the CFE bit to 0) while an interrupt request is present (the CFTXIF flag in the CFSTS register is set to 1), the interrupt request flag (the CFTXIF flag) is not automatically set to 0. Clear the interrupt request flag by a program.

Figure 3.14 Procedures for Disabling Common FIFO Buffer

3.3.2 Transmit Abort Function

By disabling the common FIFO buffer, transmissions of the messages in the common FIFO buffer can be aborted. In this case, transmission of a message which is being transmitted, and transmissions of all the messages in the common FIFO buffer can be aborted (the common FIFO buffer will be empty (the CFEMP flag in the CFSTS register is set to 1). The completion of the transmit abort can be confirmed by checking whether the common FIFO buffer is empty.

An interrupt will not be generated upon completion of the transmit abort of the common FIFO buffer. However, if transmit abort is executed while a message is being transmitted, a common FIFO transmit complete interrupt may be generated. For details, refer to Figure 3.3.

Regarding the transmit abort procedures of the common FIFO buffer, refer to Figure 3.14.

3.3.3 Interval Transmission Function

To consecutively transmit messages from the common FIFO buffer which is set to transmit mode, message transmission interval time can be set.

When the common FIFO buffer is enabled (the CFE bit in the CFCCL register is set to 1), the interval timer starts counting (after bit 7 of EOF in the CAN protocol) after the first message has been successfully transmitted from the common FIFO buffer. After a specified interval time has passed, the next message will be transmitted. Then the interval time will be reset.

The interval timer stops when:

- the common FIFO buffer is disabled (the CFE bit is set to 0)
- the CAN module transitions to channel reset mode.

Table 3.1 shows the count sources for the interval timer and formulas to calculate the interval time. Figure 3.15 is a block diagram of the interval timer. Figure 3.16 illustrates the interval timer operation.

Table 3.1 Count Sources for Interval Timer and Formulas to Calculate Interval Time

CFITR and CFITSS bits in the CFCCH register	Count sources	Formulas Note
B'00	the clock obtained by frequency-dividing the CPU/peripheral hardware clock by the value of the ITRCP[15:0] bit in the GCFGH register	$1/f_{CLK} \times a \times b$
B'10	the clock obtained by frequency-dividing CPU/peripheral hardware clock by the value of the ITRCP[15:0] bits in the GCFGH register and multiplying the divided value by 10	$1/f_{CLK} \times a \times 10 \times b$
B'x1	・ CAN-FD mode or CAN-FD only mode – CAN0 data bit time clock ・ Classical CAN only mode – CAN0 nominal bit time clock	$1/f_{CANBIT} \times b$

Remark: Each section used in the formula is described below.

a : a prescaler value of the CPU/peripheral hardware clock (a value set to the ITRCP[15:0] bits)

b : a transmission interval of messages (the CFITT[7:0] bits in the CFCCH register)

f_{CLK}: CPU/peripheral hardware clock frequency

f_{CANBIT}:

- CAN-FD mode or CAN-FD only mode
  – CAN0 data bit time clock frequency
- Classical CAN only mode
  – CAN0 nominal bit time clock frequency

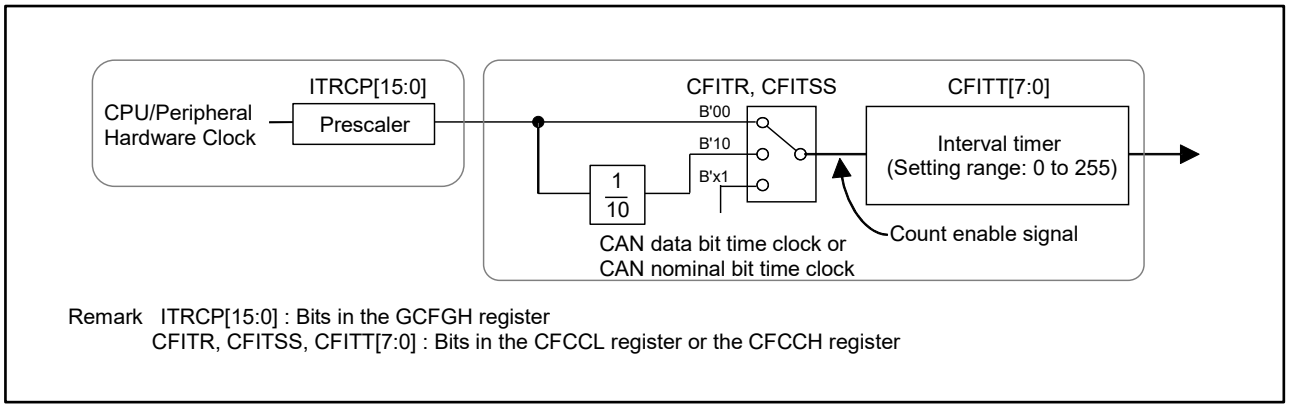
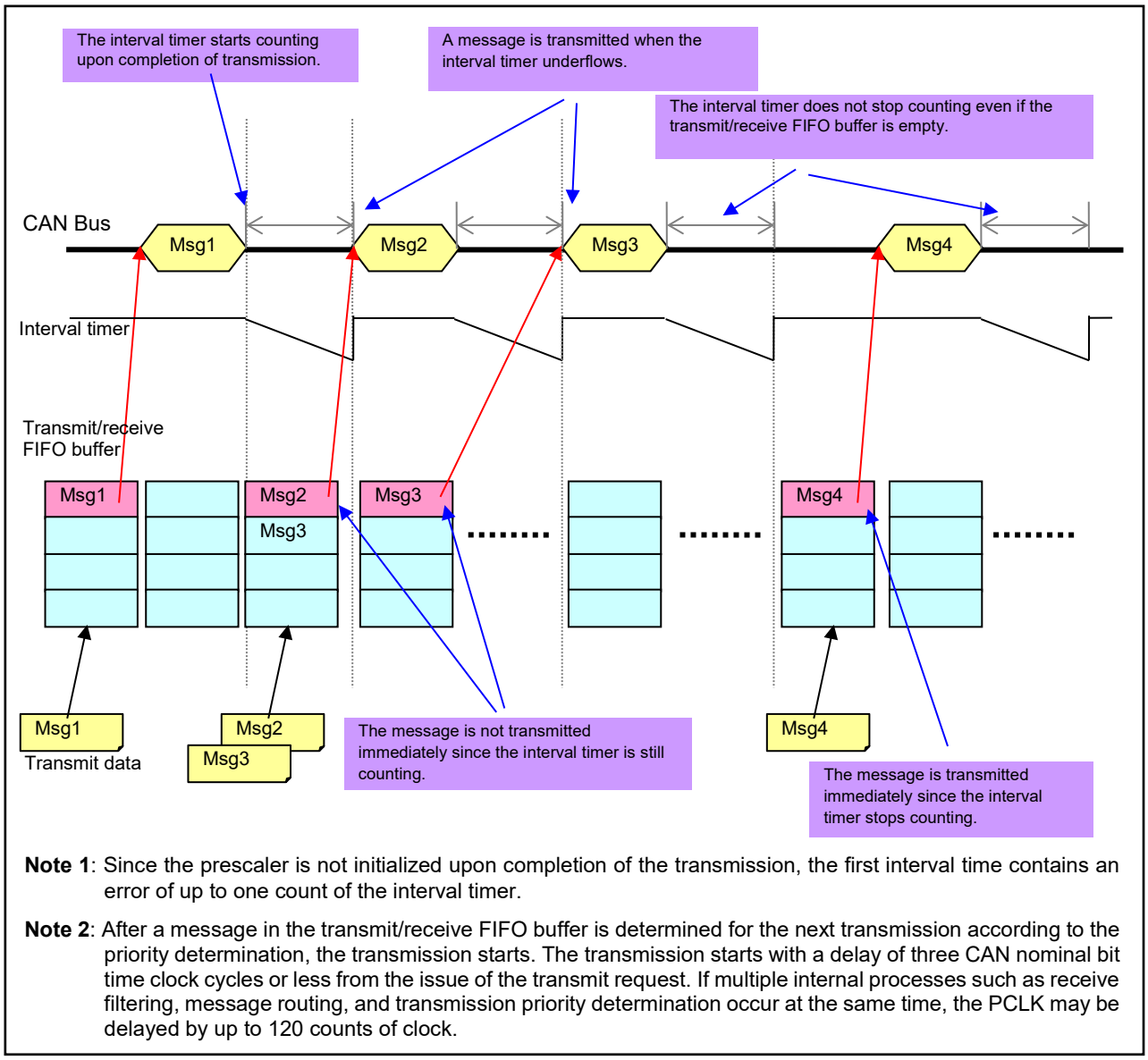


Figure 3.15 Block Diagram of Interval Timer



Note 1: Since the prescaler is not initialized upon completion of the transmission, the first interval time contains an error of up to one count of the interval timer.

Note 2: After a message in the transmit/receive FIFO buffer is determined for the next transmission according to the priority determination, the transmission starts. The transmission starts with a delay of three CAN nominal bit time clock cycles or less from the issue of the transmit request. If multiple internal processes such as receive filtering, message routing, and transmission priority determination occur at the same time, the PCLK may be delayed by up to 120 counts of clock.

Figure 3.16 Interval Transmission (in transmit mode)

3.3.4 Interrupt Handling of Common FIFO Buffer (in transmit mode)

(1) Common FIFO transmit interrupt handling

Once the common FIFO transmit complete interrupt is enabled, the CAN0 transmit interrupt will be generated when the conditions set by the CFIM bit in the CFCCL register are satisfied.

The following are the sources for the CAN0 transmit interrupt. When two or more sources are used for the interrupt generation, identify each source as needed while the interrupt is being handled.

The generation sources for the CAN0 transmit interrupt can also be confirmed by the GTINTSTS register.

- CAN0 transmit complete interrupt
- CAN0 transmit abort interrupt
- CAN0 common FIFO transmit complete interrupt
- CAN0 transmit history interrupt

Even if the common FIFO buffer is disabled (set the CFE bit in the CFCCL register to 0) while an interrupt request is present (the CFTXIF flag in the CFSTS register is 1), the interrupt request flag (the CFTXIF flag) is not automatically set to 0. Clear the interrupt request flag by a program.

The common FIFO transmit interrupt can be enabled/disabled with the CFTXIE bit in the CFCCL register for each common FIFO buffer.

The following are the sources for the common FIFO transmit complete interrupt when the common FIFO buffer is in transmit mode.

- When the buffer becomes empty upon completion of message transmission, a common FIFO transmit complete interrupt request will be generated.
- Every time one message transmission is completed, a common FIFO transmit complete interrupt request will be generated.

To generate a transmit interrupt, all the interrupt enable bits corresponding to the bits which have been set to 1 (listed in Table 6.2) need to be cleared (set to 0).

When the CAN0 transmit interrupt is used, confirm that all corresponding interrupt request flags have been set to 0 within interrupt servicing before ending the interrupt processing, refer to “Figure 4.2 CAN-related Interrupt Processing”.

3.4 Transmit History List Buffer Function

Data of the message that has been transmitted (transmission history data) can be stored in the transmit history list buffer. One channel has one transmit history list buffer which can store history data up to eight transmissions.

3.4.1 Function to Store Transmit History Data

The following can be set:

– A type of buffer that transmits a message
– Whether or not to store the transmit history data can be set for each message.

The type of buffer that transmits a message can be set when the CAN configuration is performed. Regarding the configuration to use the transmit history list buffer, refer to “1.1 CAN Configuration”.

Whether to store transmit history data and settings of label data can be set for each message transmission.

Regarding the setting procedures, see Figure 3.2 and Figure 3.12.

After message transmission has been successfully completed, the following information are stored in the transmit history list buffer as transit history data.

- Buffer type
  A type of buffer (transmit buffer or common FIFO buffer) that has transmitted the stored messages.
- Buffer number
  The number (No.) of the transmit buffer or common FIFO buffer that has transmitted the message.
  (Refer to Table 3.2)
- Timestamp
  Timestamp value of the transmitted message.
- Label data
  Label information of transmitted messages: the label information can be set for each storage of transmitted message.

Table 3.2 Buffer Number (No.) Having Transmission History Data

Buffer number (the BN[1:0] flag in the THALACC0L register)	**Buffer type (the BT[1:0] flag in the THLACC0L register)**	
	B’01	**B’10**
	Transmit buffer No.	**Common FIFO buffer**
B’00	Transmit buffer 0	Numbers (No.) of the transmit buffers linked to common FIFO buffer with the CFTML[1:0] bits in the CFCCH register
B’01	Transmit buffer 1	
B’10	Transmit buffer 2	
B’11	Transmit buffer 3	

Figure 3.17 illustrates the operations of the transmit history list buffer.

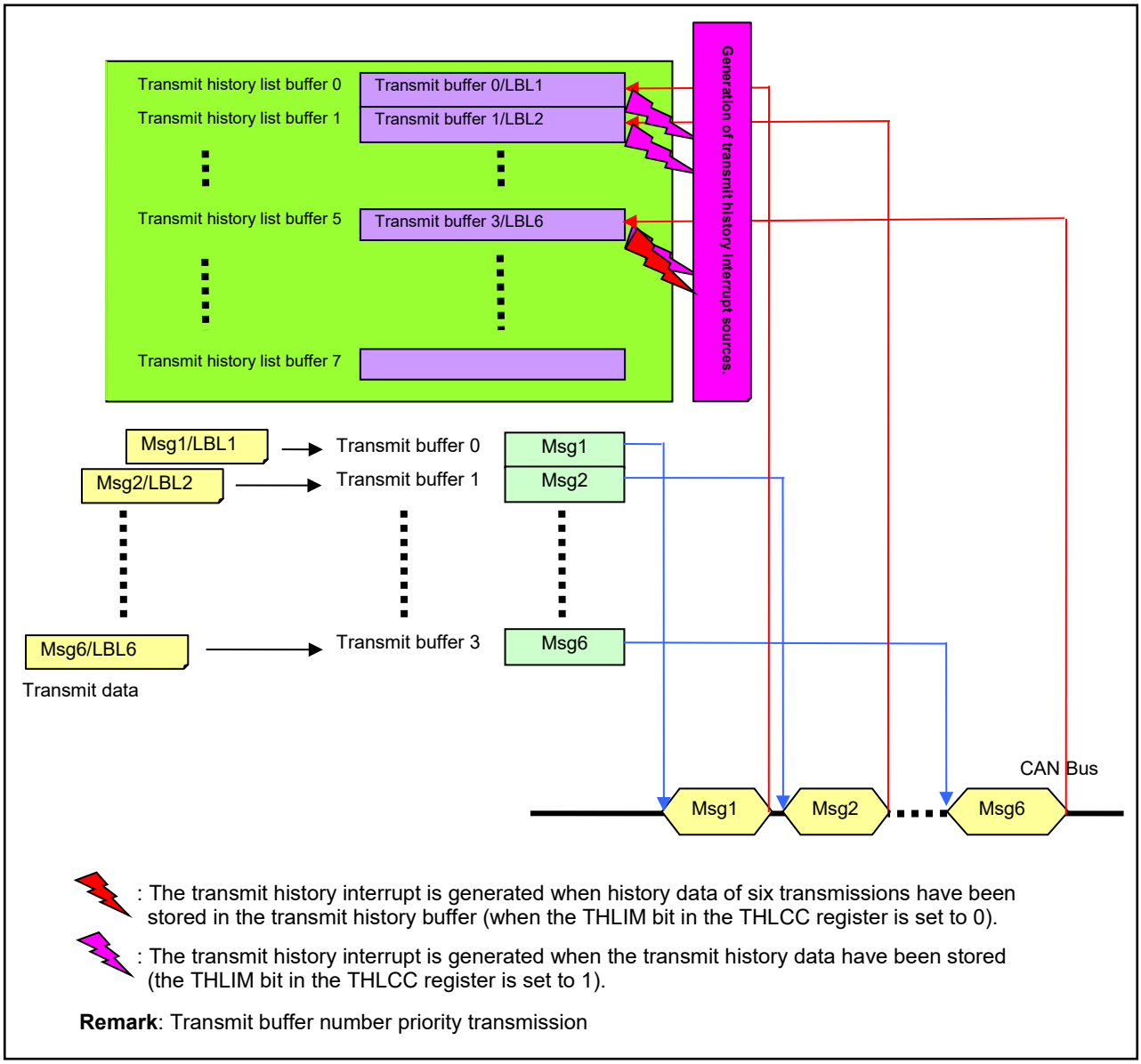
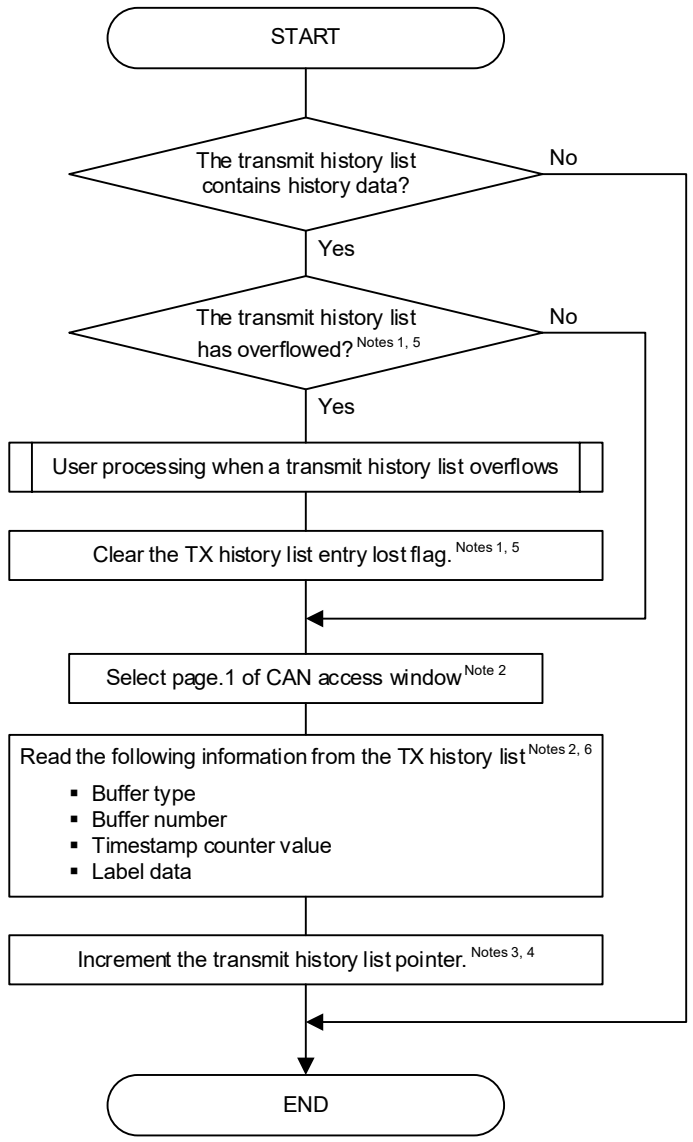


Figure 3.17 Operations of Transmit History List Buffer

(1) Procedures for reading transmit history list buffers

Figure 3.18 shows the procedures for reading transmit history data from transmit history list buffers. Figure 3.19 and Figure 3.20 show respectively the procedures for enabling and disabling the transmit history list buffers.



Note 1: The THLELT bit in the THLSTS register should be cleared by writing "0" by a program.

Note 2: The transmission history data (the THLACC0H register, the THLACC0L register, the THLACC1H register and the THLACC1L register) can be read only when page 1 of the CAN access window (the RPAGE[1:0] bits in the CFDGRWC register is B'01).

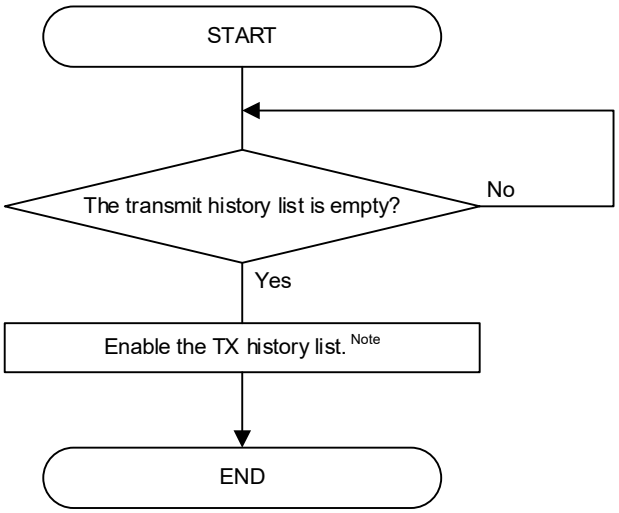
Note 3: After the transmit history list buffer (THLACC register) is read, update the transmit history list buffer pointer (write H'FF to the THLPC[7:0] bits in the THLPCTR register).

Note 4: When updating the transmit history list buffer pointer, the transmit history list buffer must be used (the THLE bit in the THLCC register is set to 1) and also the transmit history list buffer should contain any message (the THLEMP bit in the THLSTS register is set to 0).

Note 5: When the TX history list entry lost interrupt is enabled, perform these procedures while the interrupt is being handled.

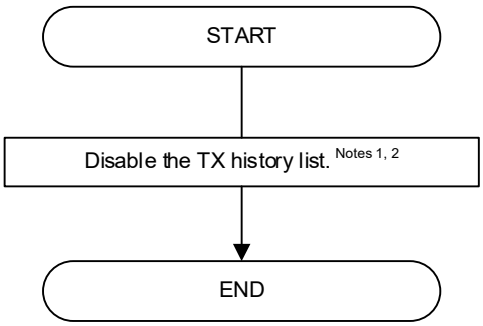
Note 6: After successful completion of message transmission, the processing may be delayed by up to 70 clocks of f_{CLK} before the transmit history data has been stored.

Figure 3.18 Procedure for Reading Transmit History List Buffer



Note: Enable/disable the transmit history list buffer (the THLE bit in the THLCC register) when the CAN module is in channel operation mode or channel halt mode.

Figure 3.19 Procedure for Enabling Transmit History List Buffer



Note 1: Enable/disable the transmit history list buffer (the THLE bit in the THLCC register) when the CAN module is in channel operation mode or channel halt mode.

Note 2: Even if the transmit history list buffer is disabled (set the THLE bit to 0) while an interrupt request is present (the THLIF flag in the THLSTS register is set to 1), the interrupt request flag (the THLIF) is not automatically set to 0. Clear the flag by a program.

Figure 3.20 Disabling the Transmit History List Buffer

3.4.2 Handling of Transmit History List Buffer Interrupt

(1) Transmit history interrupt handling

Once the transmit history interrupt is enabled, the CAN0 transmit interrupt will be generated when the conditions set by the THLIM bit in the THLCC register are satisfied.

The following are the generation sources for the CAN0 transmit interrupt. When two or more generation sources are used for the interrupt, identify each source while the interrupt is being handled.

The generation sources for the CAN0 transmit interrupt can also be confirmed by the GTINTSTS register.

- CAN0 transmit complete interrupt
- CAN0 transmit abort interrupt
- CAN0 common FIFO transmit complete interrupt
- CAN0 transmit history interrupt

Even if the transmit history list buffer is disabled (set the THLE bit in the THLCC register to 0) while an interrupt request is present (the THLIF flag in the THLSTS register is set to 1), the interrupt flag (the THLIF flag) is not automatically set to 0. Clear the interrupt flag by a program.

The transmit history interrupt can be enabled/disabled with the THLIE bit in the THLCC register for each transmit history list buffer.

The following are the generation sources for a transmit history interrupt:

- An interrupt request which is generated when history data of six transmissions have been stored in the transmit history list buffer
- An interrupt request which is generated every time history data of one transmission is stored.

To generate the CAN0 transmit interrupt, all the interrupt enable bits corresponding to the bits which have been set to 1 (listed in Table 6.2) need to be set to 0.

When the CAN0 transmit interrupt is used, confirm that all corresponding interrupt request flags have been set to 0 within interrupt servicing before ending the interrupt processing, refer to “Figure 4.2 CAN-related Interrupt Processing”.

(2) Global error interrupt handling

Once the TX history list entry lost interrupt is enabled, a global error interrupt will be generated when an overflow of the transmit history list buffer is detected. The TX history list entry lost interrupt can be collectively enabled/disabled for the entire CAN module with the THLEIE bit in the GCTRL register.

4. CAN-related Interrupt

To enable/disable CAN-related interrupts, the corresponding registers below need to be set:

Interrupt request flag registers (IF2L, IF2H and IF3H)
Interrupt mask flag registers (MK2L, MK2H and MK3H)
Priority specification flag registers (PR02L, PR12L, PR12H, PR03H and PR13H)

The following CAN-related interrupts can be used:

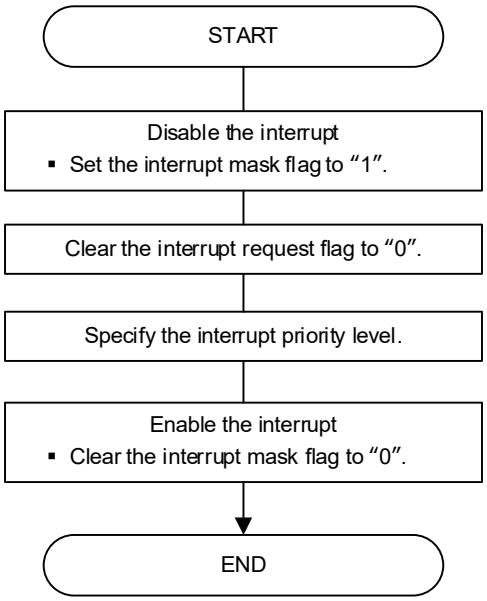
- CAN global receive FIFO interrupt
- CAN global receive buffer interrupt
- CAN global error interrupt
- CAN0 channel transmit interrupt
- CAN0 common FIFO receive interrtup
- CAN0 channel error interrupt
- CAN0 wakeup interrupt
- CAN RAM ECC interrupt

Table 4.1 CAN-related Interrupts and Generation Sources

Interrupts	Generation sources
Global receive FIFO interrupt	When a receive FIFO buffer interrupt request is issued
CAN global receive buffer interrupt	When a receive buffer interrupt request is issued
Global error interrupt	DLC check error
	FIFO message lost
	TX history list entry lost
	CAN-FD payload overflow
CAN0 transmit interrupt	CAN0 transmit complete interrupt request
	CAN0 transmit abort interrupt request
	CAN0 common FIFO transmit complete interrupt request
	CAN0 transmit history interrupt request
CAN0 common FIFO receive interrupt	When CAN0 common FIFO receive interrupt request is issued
CAN0 error interrupt	bus error
	error warning
	error passive state
	bus off entry
	bus off recovery
	overload frame transmission
	bus lock
	arbitration lost
	communication error occurrence counter overflow
	successful communication occurrence counter overflow
	transceiver delay compensation violation
CAN0 wakeup interrupt	When a falling edge of a signal from the CAN Bus is detected
CAN RAM ECC interrupt	When CAN RAM ECC 1 bit error is corrected/2 bit error is detected

4.1 Procedures for Setting CAN-related Interrupts

Figure 4.1 shows the procedures for setting interrupts.



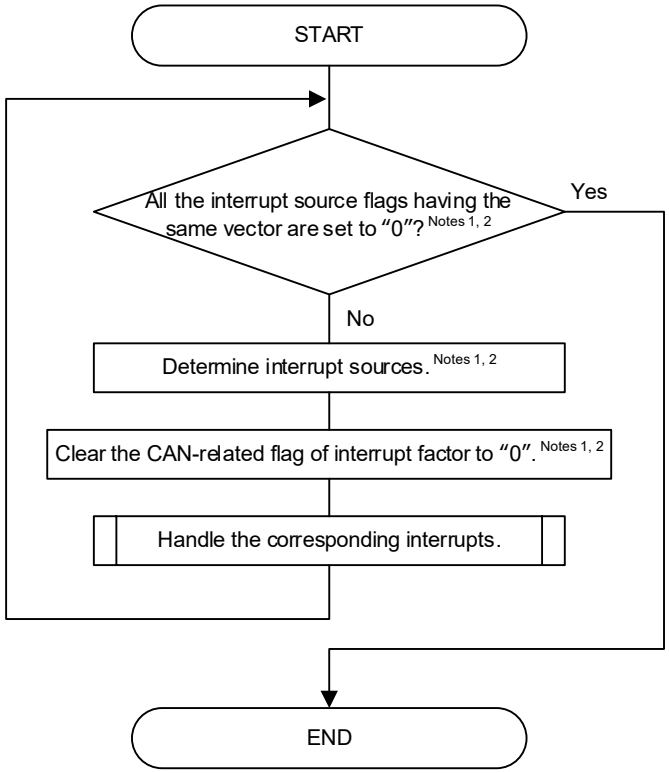
Remark: When the CAN0 wakeup interrupt is used, bit 0 in the peripheral enable register 2 (the CAN0EN bit in the PER2 register) needs to be set to 1.

Figure 4.1 Interrupt Setting Procedures

4.2 CAN-related Interrupt Handling

To use interrupts, interrupt source flags need to be cleared (set to 0). Regarding CAN-related flags corresponding to each interrupt source flags of the interrupt functions, see “6.2 CAN-related Interrupt Sources”.

Figure 4.2 shows the procedure for clear the interrupt source flags during interrupt handling.



Note 1: Before finishing the interrupt handling, make sure that interrupt request flags of the CAN module corresponding to the interrupt source have all been cleared (set to 0).
eg) If only the receiving FIFO buffer 0 is enabled (the RFIE flag of the RFCC0 register is 1) and the receive FIFO interrupt is used, set the RFIF flag of the RFSTS0 register to 0 within the interrupt handling.
After that, confirm that the RFIF flag of the RFSTS0 register has reached “0” and end the interrupt handling.

Note 2: The CAN0 wakeup interrupt request flag does not exist in the CAN module, because the interrupt function controls CAN0 wakeup interrupt.

Figure 4.2 CAN-related Interrupt Processing

5. Cautions Regarding Processing flow

5.1 Functions Used in this Application Note

For the purpose of clarifying the processing specific to each feature (function), this application note describes the processing, even if it is one line statement, by using functions. The function processing is not necessarily required to write a program.

5.2 Settings for Every Channel

This application note describes the processing only for one channel even the processing needs to be individually performed for every channel. When writing a program, be sure to perform the processing for each channel as needed.

5.3 Infinite Loop

In order to simplify the descriptions of this application note, an infinite loop is used in some processing flows. It is recommended that a program should be written so as to exit from the loop after a specified time has passed. Figure 5.1 to Figure 5.3 shows the processing including the specified loop time. Table 5.1 to Table 5.2 show the maximum transition time of each mode.

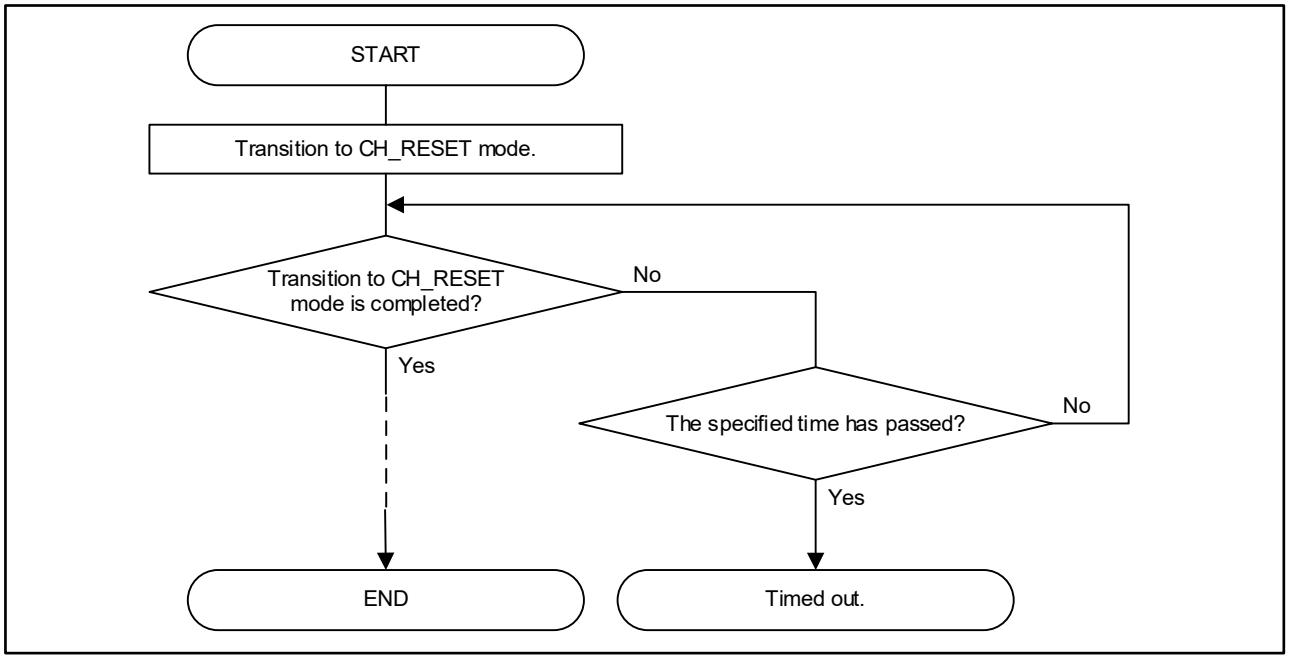


Figure 5.1 Processing Having Specified Loop Time (at mode transition)

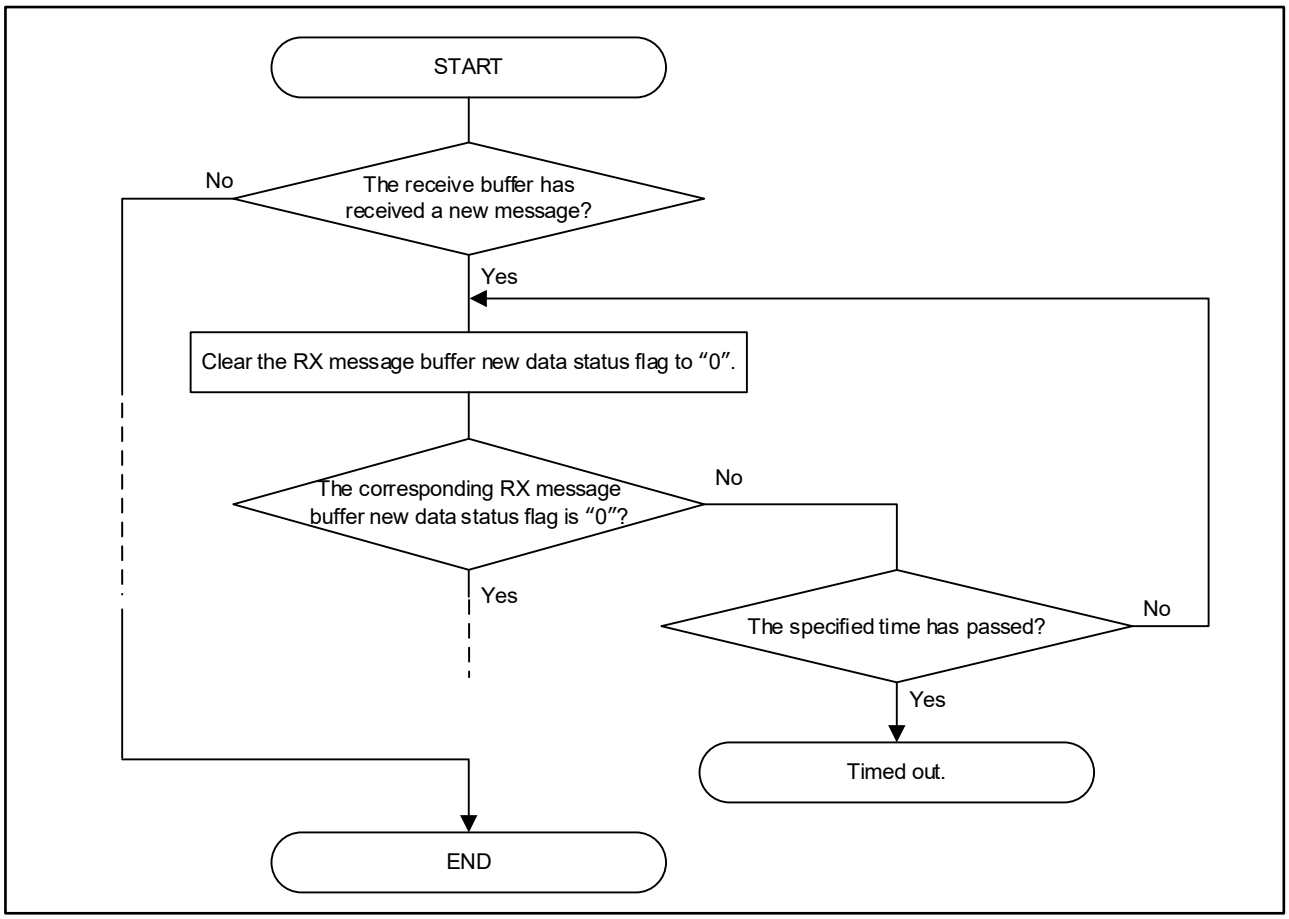


Figure 5.2 Operation with Specified Loop Time (when receive completion flag cleared)

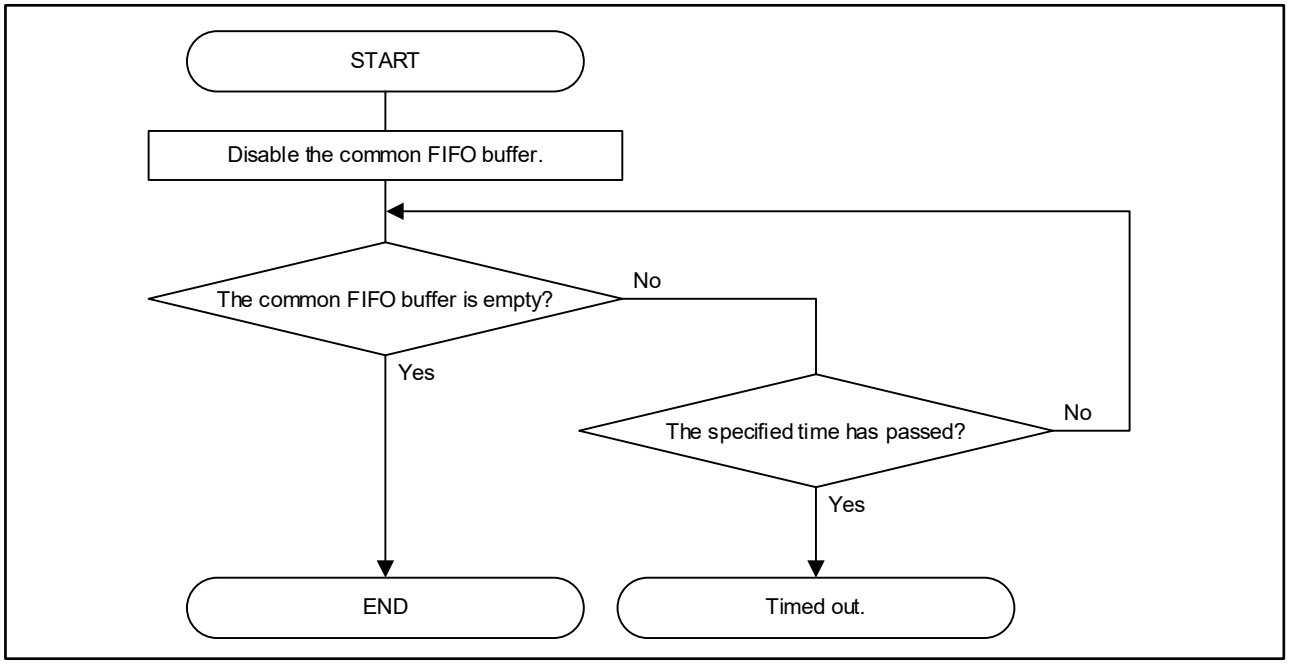


Figure 5.3 Operation with Limited Loop Time (when confirming buffer emptying)

Table 5.1 Maximum Transition Time of Global Modes

Mode before transition	Mode after transition	Maximum transition time
Global sleep	Global reset	3 f_{CLK} cycles
Global reset	Global sleep	3 f_{CLK} cycles
Global reset	Global halt	10 f_{CLK} cycles
Global reset	Global operating	10 f_{CLK} cycles
Global halt	Global reset	2 CAN bit times Note 1
Global halt	Global operating	3 f_{CLK} cycles
Global operating	Global reset	2 CAN bit times Note 1
Global operating	Global halt	3 CAN frames Note 1, 2

Note 1: The nominal bit rate of channel 0 is the CAN bit time and the CAN frame time.
Note 2: The maximum transition time if no errors have occurred on the bus.

Table 5.2 Maximum Transition Time of Channel Modes

Mode before transition	Mode after transition	Maximum transition time
Channel sleep	Channel reset	3 f_{CLK} cycles
Channel reset	Channel sleep	3 f_{CLK} cycles
Channel reset	Channel halt	3 CAN bit times Note 1
Channel reset	Channel operation	4 CAN bit times Note 1
Channel halt	Channel reset	2 CAN bit times Note 1
Channel halt	Channel operation	4 CAN bit times Note 1, 2
Channel operation	Channel reset	2 CAN bit times Note 1
Channel operation	Channel halt	2 CAN frames Note 3

Note 1: The CAN bit time of the nominal bit rate.
Note 2: When the value of the baud rate prescaler (the NBRP[9:0] bits of the C0NCFGL register) is changed in channel HALT mode before the transition to channel operation mode, the maximum transition time may deviate.
Note 3: The maximum transition time if no errors have occurred on the bus.

6. Appendix

6.1 Configuration Processing for Each Status

Table 6.1 lists the configuration processing for each status.

Table 6.1 Configuration for Each Status

Processing		CAN configuration Note 1			
		After MCU reset or SW reset	After transition to global reset mode	After transition to channel reset mode	After transition to channel halt mode
CAN status	Transition of global modes	✓	✓	-	-
	Transition of channel modes	✓	✓	✓	✓
Global function setting	Transmit priority	✓	△	-	-
	DLC check				
	DLC replacement function				
	Mirror function				
	Clock				
	CAN-FD payload overflow				
	Timestamp clock				
	Interval timer prescaler				
	RES bit protocol exception event detection				
	Timestamp Capture				
Communication speed setting	Bit timing	✓	△	△	△
	Communication speed				
Receive rule table setting		✓	△	-	-
Buffer setting	Receive buffer	✓	△	-	-
	Receive FIFO buffer				
	Common FIFO buffer			△ Note 2	△ Note 2
	Transmit buffer			△	△
	Transmit history list buffer				
Global error function setting		✓	△	-	-
Channel function setting		✓	△	△	△

Note 1: ✓: Settings are required
-: Settings are prohibited
△: Settings are not required

Note 2: The following bits need to be modified in global reset mode:
CFDC[2:0] bits, CFTML[1:0] bits, CFIGCV[2:0] bits, CFIM bit, CFM bit and CFPLS[2:0] bits in the CFCCL and CFCCH registers

6.2 CAN-related Interrupt Sources

Table 6.2 lists the CAN-related interrupt sources.

Table 6.2 CAN-related Interrupt Sources (1/2)

Interrupt	Generation source	Interrupt enable bit Note 1	Conditions	How to clear Note 1
CAN global receive FIFO interrupt	Recive FIFOk interrupt request	RFIE bit in the RFCCk register	When the conditions set by the RFIGCV[2:0] bits in the RFCCk register are satisfied. Note 2	Set the RFIF flag in the RFSTSk register to 0.
			Every time one message is received.	
CAN global receive buffer interrupt	Receive buffer interrupt request	RMIEn bit in the RMIEC register	Every time one message is received.	Set the RMNSn flag in the RMND register to 0.
CAN global error interrupt	DLC check error	DEIE bit in the GCTRL register	When a DLC check error is detected	Set the DEF flag in the GERFLL register to 0.
	FIFO message lost	MEIE bit in the GCTRL register	When a messge lost error of the common FIFO buffer is detected.	(all channels) Set the CFMLT flag in the CFSTS register to 0. Set the RFMLT flag in the RFSTSk register to 0
			When a message lost error of the receive FIFO buffer is detected.	
	TX history list entry lost	THLEIE bit in the GCTRL register	When the transmit history list buffer attempts to store further transmit history data although the buffer is already full.	(all channels) Set the THLELT flag in the THLSTS register to 0.
	CAN-FD payload overflow	CMPOFIE bit in the GCTRL register	When a CAN-FD payload overflow is detected.	Set the CMPOF flag in the GERFLL register to 0.
CAN0 channel transmit interrupt	CAN0 transmit complete interrupt request	TMIEm bit in the TMIEC register	When the buffer becomes empty upon completion of message transmission	Set the TMTRF[1:0] flag in the TMSTSm register to B'00.
	CAN0 transmit abort interrupt request	TAIE bit in the C0CTRH register	Every time transmission of one message is completed.	
	CAN0 common FIFO transmit complete interrupt request	CFTXIE bit in the CFCCL register	When the buffer becomes empty upon completion of message transmission	Set the CFTXIF flag in the CFSTS register to 0.
			Every time transmission of one message is completed.	
	CAN0 transmit history interrupt request	THLIE bit in the THLCC register	When history data of six transmissions have been stored in the transmit history list buffer.	Set the THLIF flag in the THLSTS register to 0.
			Every time history data of one transmission are stored.	
CAN0 transmit/ receive FIFO receive interrupt	CAN0 common FIFO receive interrupt request	CFRXIE in the CFCCL register	When the conditions set by the CFIGCV[2:0] bits in the CFCCL register are satisfied. Note 3	Set the CFRXIF flagin the CFSTS register to 0.
			Every time reception of one message is completed.	
CAN0 channel error itnerrupt	Bus error	BEIE bit in the C0CTRL register	When any one of the ADERR, B0ERR, B1ERR, CERR, AERR, FERR, and SERR flags of the C0ERFLL register is set to 1. Note 4	Set the BEF flag in the C0ERFLL register to 0.
	Error warning	EWIE bit in the C0CTRL register	When the value of the REC[7:0] bits or TEC[7:0] bits in the C0STSH register exceeds 95.	Set the EWF flag in the C0ERFLL register to 0.
	Error passive	EPIE bit in the C0CTRL register	When the CAN module has entered the error passive state (REC[7:0] or TEC[7:0] bits > 127)	Set the EPF flag in the C0ERFLL register to 0.
	Bus off entry	BOEIE bit in the C0CTRL register	When the CAN module has enterred the bus off state (TEC[7:0] bits > 255)	Set the BOEF flag in the C0ERFLL register to 0.
	Bus off recovery	BORIE bit in the C0CTRL reigster	When 11 consecutive recessive bits have been detected 128 times and the CAN module returns from the bus off state. Note 5	Set the BORF flag in the C0ERFLL register to 0.
	Overload frame transmit	OLIE bit in the C0CTRL	When the overload frame transmit condition has been detected when performing reception or transmission.	Set the OVLF flag in the C0ERFLL register to 0.

(Notes are provided at the end of this table.)

Table 6.2 CAN-related Interrupt Sources (2/2)

Interrupt	Generation source	Interrupt enable bit Note 1	Conditions	How to clear Note 1
CAN0 channel error itnerrupt	Bus lock	BLIE bit in the C0CTRL register	When 32 consectutive dominant bits have been detected on the CAN Bus in channel operation mode.	Set the BLF flag in the C0ERFLL register to 0.
	Arbitration lost	ALIE bit in the C0CTRL register	When arbitration lost is detected	Set the ALF flag in the C0ERFLL register to 0.
	Communication error occurrence counter overflow	EOCOIE bit in the C0CTRL register	When communication error occurrence counter overflow is detcted.	Set the EOCO flag in the C0FDSTS register to 0.
	Successful communication occurrence counter overflow	SOCOIE bit in the C0CTRL register	When successful communication occurrence counter overflow is detected.	Set the SOCO flag in the C0FDSTS register to 0.
	Transceiver delay compensation violation	TDCVFIE bit in the C0CTRL register	When transceiver delay compensation violation.	Set the TDCVF flag in the C0FDSTS register to 0.
CAN0 wakeup interrupt	Detection of a CAN Bus falling edge	--	When a falling edge is detected in the CRXDi pin.	--
CAN RAM ECC interrupt	Detection of a CAN RAM ECC 1 bit error/2 bit eeror	Note 6	When a ECC 1 bit error is corrected/2 bit error is detected in CAN RAM.	Note 6

Note 1: Note that interrupt request flags and interrupt enable bits of the interrupt functions are not included in this list. For details, refer to interrupt-related sections of RL78/F24 User's Manual for Hardware.

Note 2: Values set to the RFIGCV[2:0] bits in the RFCCk register

- B'000: the receive FIFO buffer is 1/8 full *
- B'001: the receive FIFO buffer is 2/8 full
- B'010: the receive FIFO buffer is 3/8 full *
- B'011: the receive FIFO buffer is 4/8 full
- B'100: the receive FIFO buffer is 5/8 full *
- B'101: the receive FIFO buffer is 6/8 full
- B'110: the receive FIFO buffer is 7/8 full *
- B'111: the receive FIFO buffer is full.

Remark *: When the number of messages to be stored in the receive FIFO buffer is 4 (the value of the RFDC[2:0] bits in the CFCCH register is B'001), do not perform this settings.

Note 3: Settings to the CFIGCV[2:0] bits in the CFCCL register

- B'000: the common FIFO buffer is 1/8 full *
- B'001: the common FIFO buffer is 2/8 full
- B'010: the common FIFO buffer is 3/8 full *
- B'011: the common FIFO buffer is 4/8 full
- B'100: the common FIFO buffer is 5/8 full *
- B'101: the common FIFO buffer is 6/8 full
- B'110: the common FIFO buffer is 7/8 full *
- B'111: the common FIFO buffer is full.

Remark *: When the number of messages to be stored in the common FIFO buffer is set to 4 (the value of the CFDC[2:0] bit of the CFCCL register is B'001), do not perform this setting.

Note 4: When any one of the following is detected, an interrupt will be generated:

- The ADERR flag in the C0ERFLL register is set to 1 and also a form error has been detected in the ACK delimiter.
- The B0ERR flag in the C0ERFLL register is set to 1 and also a recessive bit has been detected though a dominant bit was transmitted.
- The B1DRR flag in the C0ERFLL register is set to 1 and also a dominate bit has been detected though a recessive bit was transmitted.
- The CERR flag in the C0ERFLL register is set to 1 and also a CRC error has been detected.
- The AERR flag in the C0ERFLL register is set to 1 and also an ACK error has been detected.
- The FERR flag in the C0ERFLL register is set to 1 and also a form error has been detected.
- The SERR flag in the C0ERFLL register is set to 1 and also a stuff error has been detected.

Note 5: An interrupt will not be generated when the CAN module returns from the bus-off state due to the following conditions before 11 consecutive recessive bits have been detected 128 times (the BORF flag will not be set to 1):

- When the value of the CHMDC[1:0] bits in the C0CTRL register is set to B'01 (channel reset mode)
- When the RTBO bit in the C0CTRL register is set to 1 (forcible return from the bus-off state is made)
- When the BOM[1:0] bit in the C0CTRH register is set to B'01 (transition to channel halt mode at bus off entry)
- When the value of the CHMDC[1:0] bits is B’10 when the value of the BOM[1:0] bits is B’11 (transition to channel halt mode during the bus off state due to a request from a program) and also before 11 consecutive recessive bits have been detected 128 times.

Note 6: The interrupt request flags and interrupt permission bits in the CAN RAM ECC function are not listed. For more information, see the User's Manual Hardware Edition of the target product.

6.3 Operations When a Receive Buffer Has Received a Message and Operations When the Receive (common) FIFO Buffer is Full

Table 6.3 shows the operation in the following cases: when a receive buffer has received a message or when the receive FIFO buffer or the common FIFO buffer (in receive mode) attempts to receive further messages although the buffers are already full.

Table 6.3 Operations When a Receive Buffer Has Received a Message or When the Receive (Common) FIFO Buffer is Full

FIFO/Buffer	When a next message is received Note	Interrupt request
Receive buffer	overwritten	None
Receive FIFO buffer	discarded	Global error interrupt (message lost error in the receive FIFO buffer)
Common FIFO buffer (in receive mode)	discarded	Global error interrupt (message lost error in the common FIFO buffer)

Remark: Each section is described below.
Overwritten: The next message will be overwritten in the receive buffer.
Discarded: The next message will be discarded (the message is not stored in FIFO buffer), which means a message lost error occurs.

6.4 Requests to Transmit Buffers

The interrupt sources vary according to a request issued to the transmit buffer and the conditions for stopping transmission.
Table 6.4 lists the requests to the transmit buffer and interrupt sources.

Table 6.4 Requests to Transmit Buffers and Interrupt Sources

TMCm register			Event	Transmission result (TMTRF[1:0] flag in the TMSTSm register)	Interrupt sources
Transmit request (TMTR)	Transmit abort request (TMTAR)	One-shot transmit request (TMOM)			
1	0	0	Transmission is completed.	B’10: Transmission has been completed without an abort request	Transmit compete interrupt
			Arbitration lost or any error occurs.	B’00: Transmission is in progress.	None
1	1	0	Transmission is completed.	B’11: Transmission has been completed with an abort request.	Transmit compete interrupt
			Arbitration lost or any error occurs.	B’01: Transmission has been aborted.	Transmit abort interrupt
1	0	1	Transmission is completed.	B’10: Transmission has been completed without an abort request	Transmit compete interrupt
			Arbitration lost or any error occurs.	B’01: Transmission has been aborted.	Transmit abort interrupt
1	1	1	Transmission is completed.	B’11: Transmission has been completed with an abort request.	Transmit compete interrupt
			Arbitration lost or any error occurs.	B’01: Transmission has been aborted.	Transmit abort interrupt
0	x	x	Setting prohibited		

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	2022.09.30	-	1st Edition

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)
   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on
   The state of the product is undefined at the time 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 time 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 time 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 time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state
   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins
   Handle unused pins in accordance 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 the 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.
5. Clock signals
   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is 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. Additionally, 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.
6. Voltage application waveform at input pin
   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses
   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products
   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Notice

1. 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 or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving 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, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   “Standard”: Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.
   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, 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 and impractical, you are responsible for evaluating the safety of the final products or systems 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. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall 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. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.
(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.