RL78/F13, F14, F15

CAN configuration, reception and transmission

R01AN2488EJ0200
Rev. 2.00
Jun. 15, 2018

Introduction

This application note describes the procedures for configuring a control area network (CAN) bus for the RL78/F13, RL78/F14, and RL78/F15 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 devices

This application note is applied to the RL78/F13, RL78/F14, and RL78/F15 microcontrollers.

The table below lists the variables used in this document.

Target devices and their corresponding variables

	Index	Target MCUs	
		RL78/F13 RL78/F14	RL78/F15
CAN channel number	i	i=0	i=0, 1
CAN receive rule entry register number (GAFLIDLj, GAFLIDHj, GAFLMLj, GAFLMHj, GAFLPLj, and GAFLPHj registers)	j	j=0 to 15	j=0 to 39
Transmit/receive FIFO buffer number	k	k=0	k=0, 1
Receive FIFO buffer number	m	m=0, 1	m=0 to 3
Receive buffer number	n	n=0 to 15	n=0 to 31
Transmit buffer number	p	p=0 to 3	p=0 to 7
CAN RAM test register number (RPGACCr register)	r	r=0 to 127	r=0 to 127

Caution: This document mainly describes the CAN module of RL78/F13 and RL78/F14.

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, 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) transition**.

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

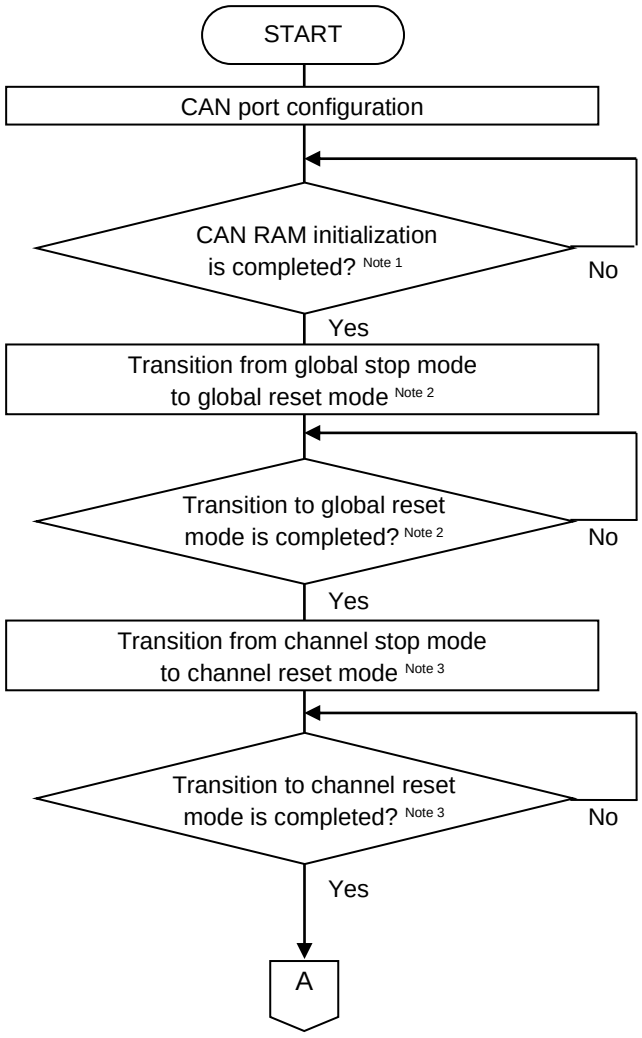
1.1.1 CAN configuration after CAN module is enabled

(1) CAN configuration after CAN module is enabled

Initialize the entire CAN module after the CAN module is enabled.

(2) CAN configuration procedures after CAN module is enabled

Figure 1.1 and **Figure 1.2** show the CAN configuration procedures after the CAN module is enabled.

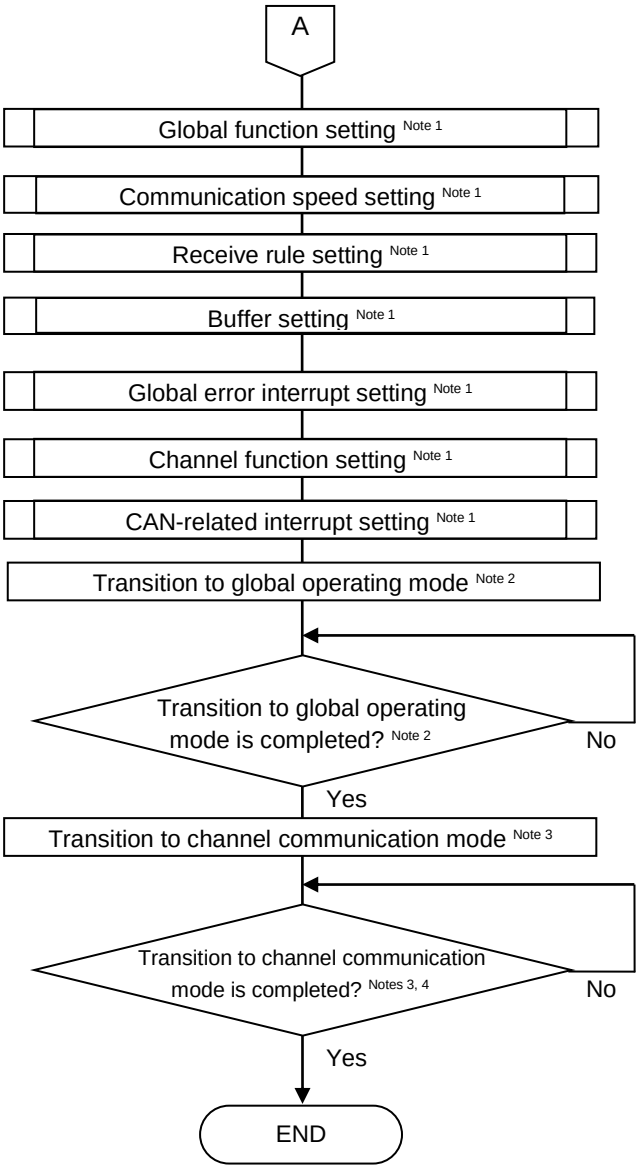


Note 1: After the CAN module is enabled (set the CAN0EN bit in the PER2 register to 1), do not access CAN RAM before the CAN RAM is initialized (set the GRAMINIT bit in the GSTS register to 1). To use the X1 clock, enable the supply of the X1 clock (set the CAN0MCKE bit in the CANCKSEL register to 1).

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 CiCTRL register), check the CiSTSL 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.

Figure 1.1 Configuration procedures after CAN module is enabled (1)



Note 1: For detailed information on the processing for each function, refer to the following sections.

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 CiCTRL reigster), check the CiSTSL reigster 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: When 11 consecutive recessive bits are detected after a transition to channel communication mode is completed, communication will be available (the COMSTS flag in the CiSTSL 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.

Figure 1.2 Configuration procedures after CAN module is enabled (2)

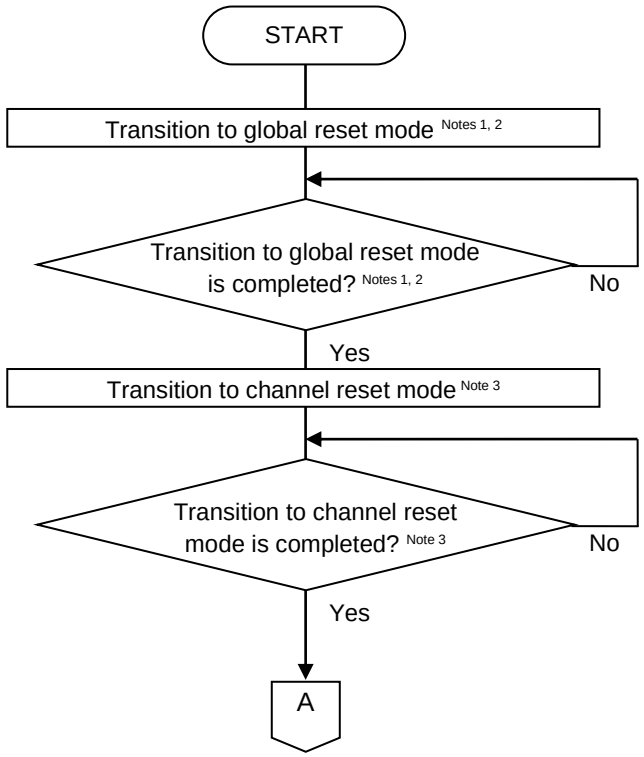
1.1.2 CAN configuration after transition to global reset mode

(1) CAN configuration after transition to global reset mode

Initialize the entire CAN module after the transition to global reset mode is completed.

(2) CAN configuration procedures after transition to global reset mode

Figure 1.3 and **Figure 1.4** show the CAN configuration procedures after the transition to global reset mode is completed.

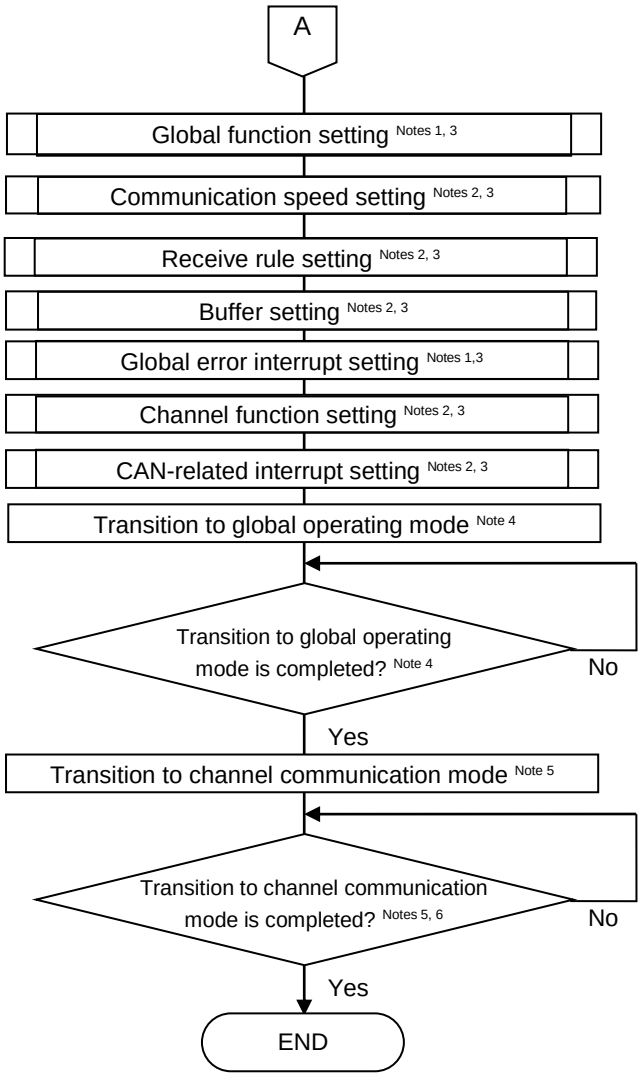


Note 1: When changing a global mode (the GSLPR bit and GMDC[1:0] bits in the GCTRL register), check the GSTS register to see if the transition is completed. Do not modify the value of the GMDC[1:0] bits until the mode transition is completed.

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 CiCTRL register), check the CiSTSL 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.

Figure 1.3 Configuration procedures after transition to global reset mode (1)



Note 1: These settings are not necessarily required because the bit values are reset even after the transition to global reset mode.

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

Note 4: 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 5: When changing a channel mode (the CSLPR bit and CHMDC [1:0] bits in the CiCTRL register), check the CiSTSL 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 6: When 11 consecutive recessive bits are detected after a transition to channel communication mode is completed, communication will be available (the COMSTS flag in the CiSTSL 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.

Figure 1.4 Configuration procedures after transition to global reset mode (2)

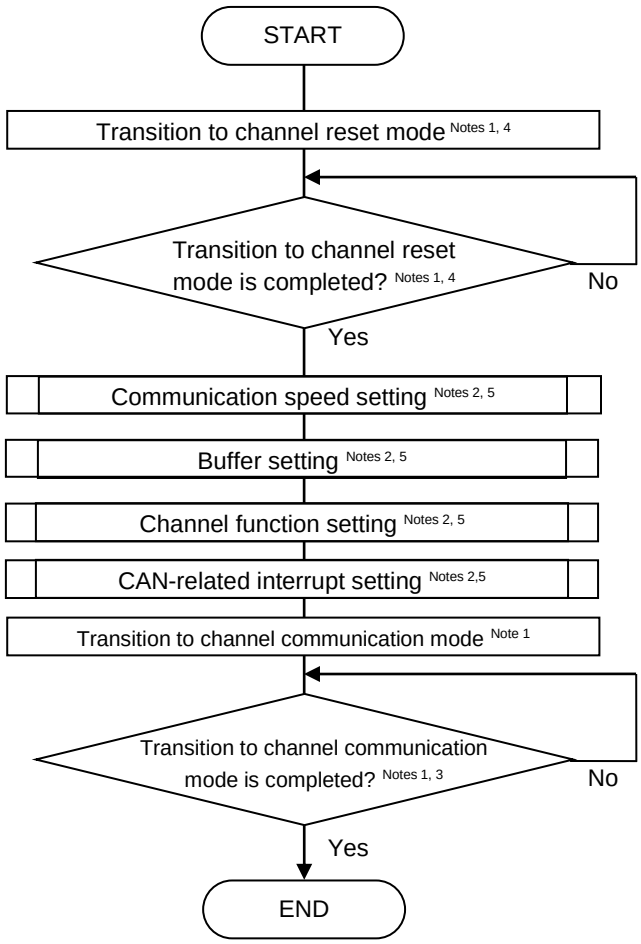
1.1.3 CAN configuration after transition to channel reset mode

(1) CAN configuration after transition to channel reset mode

Initialize the CAN channel(s) after the transition to channel reset mode is completed.

(2) CAN configuration procedures after transition to channel reset mode

Figure 1.5 shows the configuration procedures 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 CiCTRL register), check the CiSTSL 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 communication mode is completed, communication will be available (the COMSTS flag in the CiSTSL 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 1.5 Configuration procedures after transition to channel reset mode

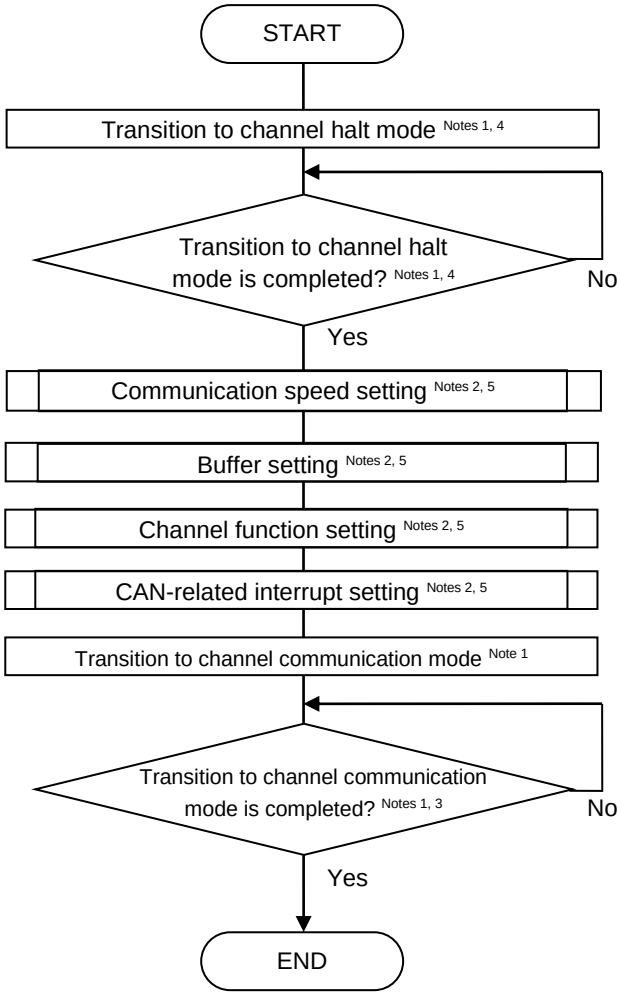
1.1.4 CAN configuration after transition to channel halt mode

(1) CAN configuration after transition to channel halt mode

Initialize the CAN channel(s) after the transition to channel halt mode is completed.

(2) CAN configuration procedures after transition to channel halt mode

Figure 1.6 shows the configuration procedures 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 CiCTRL register), check the CiSTSL 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 communication mode is completed, communication will be available (the COMSTS flag in the CiSTSL 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 CiERFLL 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 1.6 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 stop mode
- Global reset mode
- Global test mode
- Global operating mode

Channel mode
- Channel stop mode
- Channel reset mode
- Channel halt mode
- Channel communication mode

1.2.1 Global modes

These are modes to control the entire CAN module.

Figure 1.7 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**.

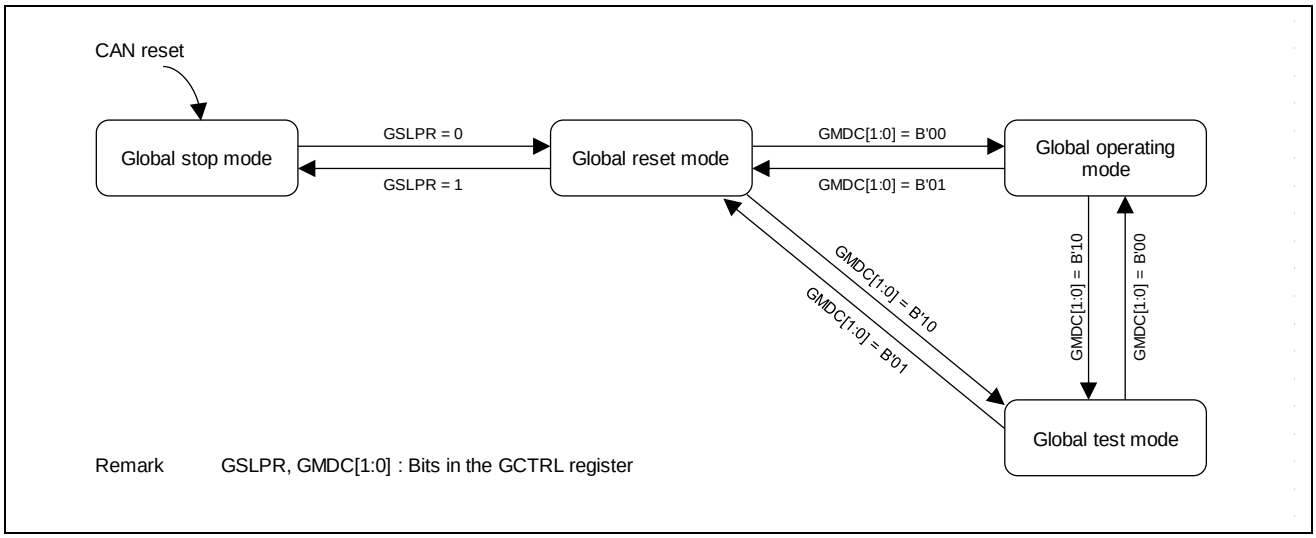


Figure 1.7 Transitions of global modes

(1) Global stop 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 test mode

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

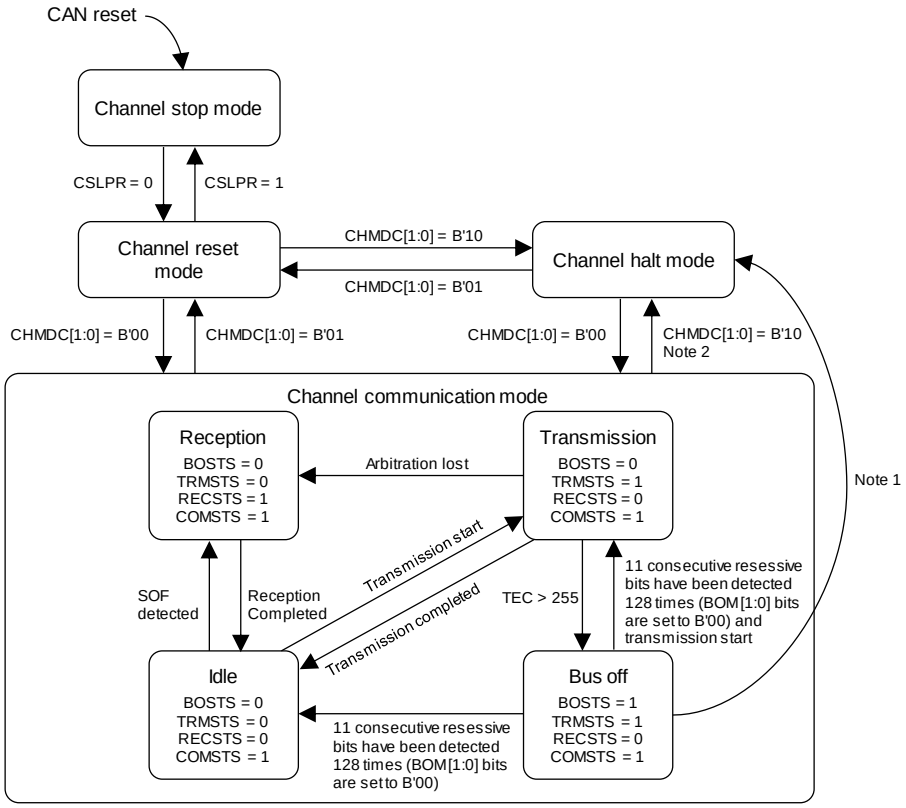
(4) Global operating mode

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 1.8 shows the transitions of channel modes.



Notes 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
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 CiCTRL register
BOM[1:0] : Bits in the CiCTRH register
BOSTS, TRMSTS, RECSTS, COMSTS : Bits in the CiSTSL register

Figure 1.8 Transitions of channel modes

(1) Channel stop 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 communication 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 1.9** 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 test**	**Global reset**	**Global stop**
Channel communication	Channel communication	**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 stop**
Channel stop	Channel stop	Channel stop	Channel stop	Channel stop

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

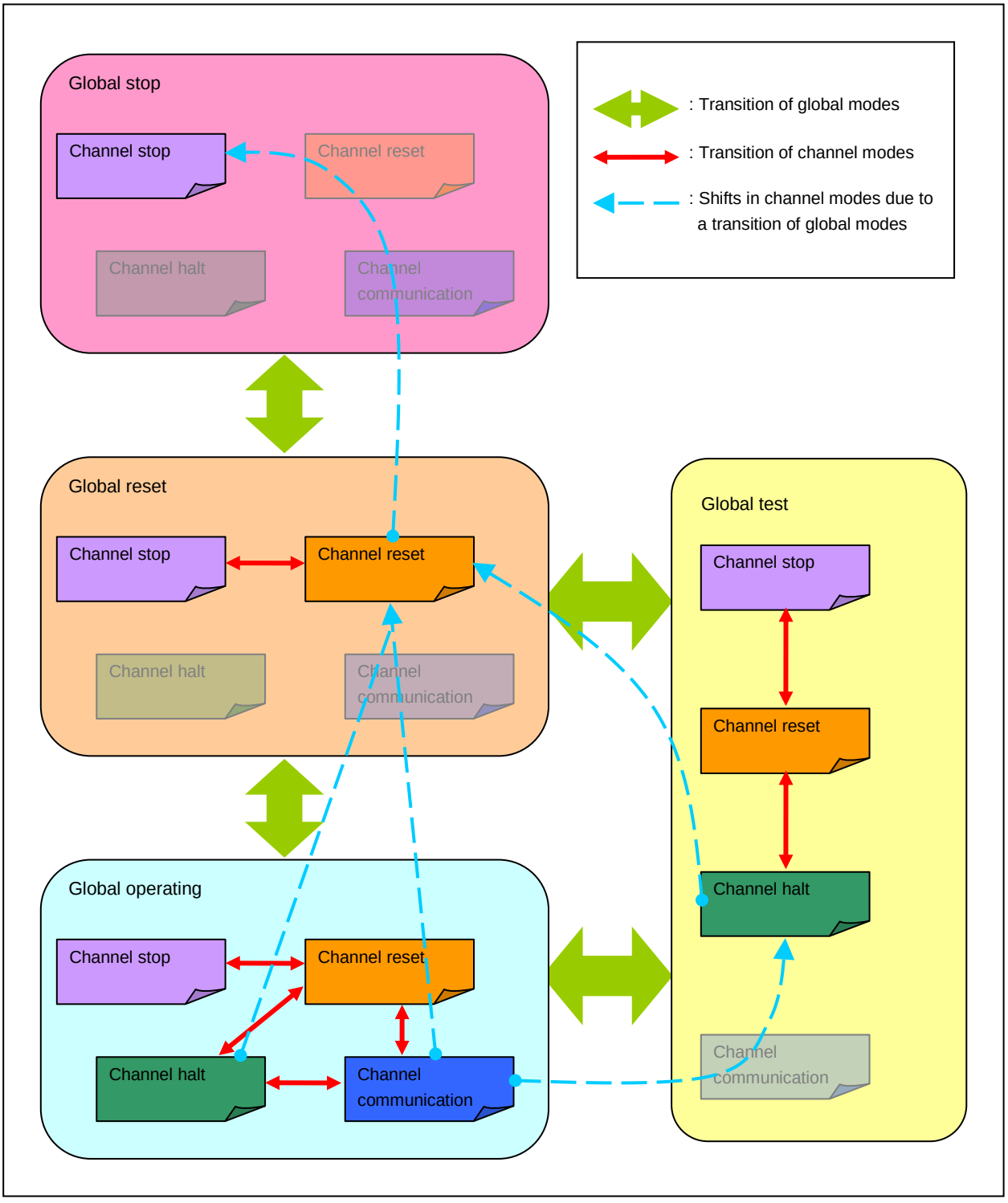


Figure 1.9 Transitions of global modes and channel modes

Table 1.2 Registers to be initialized due to transition to global reset mode and channel reset mode

Registers	Bits/flags
CiCTRL register	CHMDC[1:0]
CiCTRH register	CTMS[1:0], CTME
CiSTSL register	CHLTSTS, EPSTS, BOSTS, TRMSTS, RECSTS, COMSTS
CiSTSH register	REC[7:0], TEC[7:0]
CiERFLL register	ADERR, B0ERR, B1ERR, CERR, AERR, FERR, SERR, ALF, BLF, OVLF, BORF, BOEF, EPF, EWF, BEF
CiERFLH register	CRCREG[14:0]
CFCCLk register	When the transmit/receive FIFO buffer is in transmit mode : CFE
CFSTSk register	When the transmit/receive FIFO buffer is in transmit mode : CFMC[5:0], CFTXIF, CFRXIF, CFMLT, CFFLL, CFEMP
TMCp register	TMOM, TMTAR, TMTR
TMSTSp register	TMTARM, TMTRM, TMTRF[1:0], TMTSTS
TMTRSTS register	TMTRSTSp
TMTCSTS register	TMTCSTSp
TMTASTS register	TMTASTSp
THLCCi register	THLE
THLSTSi register	THLMC[3:0], THLIF, THLELT, THLFLL, THLEMP
GTINTSTS register	TSIFi, TAIFi, CFTIFi, THIFi

Table 1.3 Registers to be initialized due to transition only to global reset mode

Register	Bits/flags
GSTS register	GHLTSTS
GERFLL register	THLES, MES, DEF
GTSC register	TS[15:0]
RMNDi register	RMNSn
RFCCm register	RFE
RFSTSm register	RFMC[5:0], RFIF, RFMLT, RFFLL, RFEMP
CFCCLk register	When the transmit/receive FIFO buffer is in receive mode : CFE
CFSTSk register	When the transmit/receive FIFO buffer is in receive mode : CFMC[5:0], CFFLL, CFEMP, CFTXIF, CFRXIF, CFMLT
RFMSTS register	RFmMLT
CFMSTS register	CFkMLT
RFISTS register	RFmIF
CFISTS register	CFkIF
GTSTCFG register	RTMPS[2:0]
GTSTCTRL 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 1.10 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.

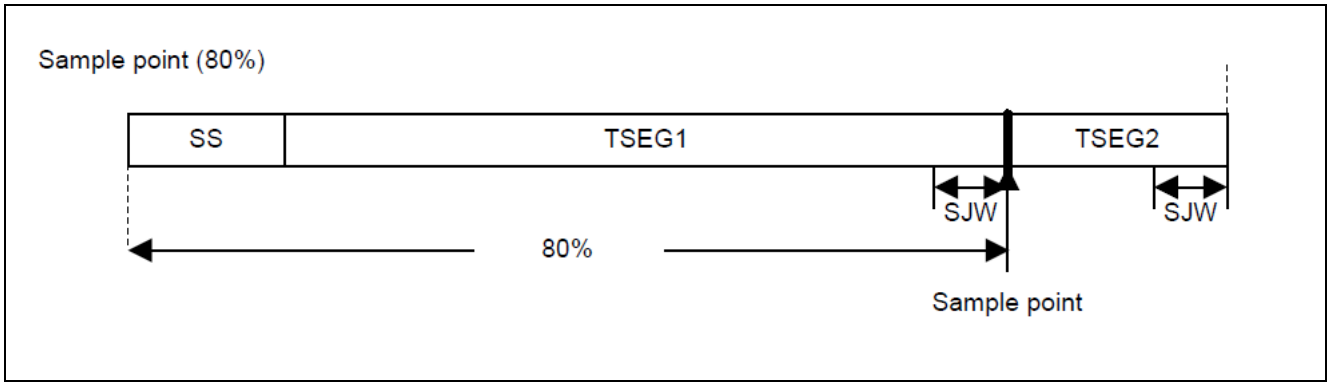


Figure 1.10 Structure of bit segments and sample point

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

SS=1 Tq fixed

Set TSEG1 to a range of 4 Tq to 16 Tq.

Set TSEG2 to a range of 2 Tq to 8 Tq.

Set SJW to a range of 1 Tq to 4 Tq.

Set “SS+TSEG1+TSEG2” to a range of 8 Tq to 25 Tq.

Limitation on TSEG1 and TSEG2

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 clock obtained by dividing the CPU/peripheral hardwre clock by 2 or the X1 clock. Regarding the f_{CAN} settings, see **1.4.5 CAN clock source setting**.

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

Table 1.4 Communication speed calculation and examples of communication speed

Formula for communication speed	$\frac{f_{CAN}}{\text{Baud rate prescaler ratio}^{Note} \times (\text{Tq count of 1-bit time})}$	
f_{CAN} / Communication speed	16MHz	8MHz
1Mbps	8Tq (2) 16Tq (1)	8Tq (1)
500Kbps	8Tq (4) 16Tq (2)	8Tq (2) 16Tq (1)
250Kbps	8Tq (8) 16Tq (4)	8Tq (4) 16Tq (2)
125Kbps	8Tq (16) 16Tq (8)	8Tq (8) 16Tq (4)
83.3Kbps	8Tq (24) 16Tq (12)	8Tq (12) 16Tq (6)
33.3Kbps	8Tq (60) 10Tq (48) 16Tq (30) 20Tq (24)	8Tq (30) 10Tq (24) 16Tq (15) 20Tq (12)

Note: Baud rate prescaler ratio = P+1 (P=0 to 1023)
P: the value set to the BRP bit in the CiCFGL register

Remark: Figures in parentheses indicate baud rate prescaler values.

Table 1.5 Example of bit timing settings

1 bit	**Set value (Tq)**				**Sample point Note (%)**
	SS	TSEG1	TSEG2	SJW	
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
16Tq	1	10	5	1	68.75
	1	11	4	1	75.00
20Tq	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

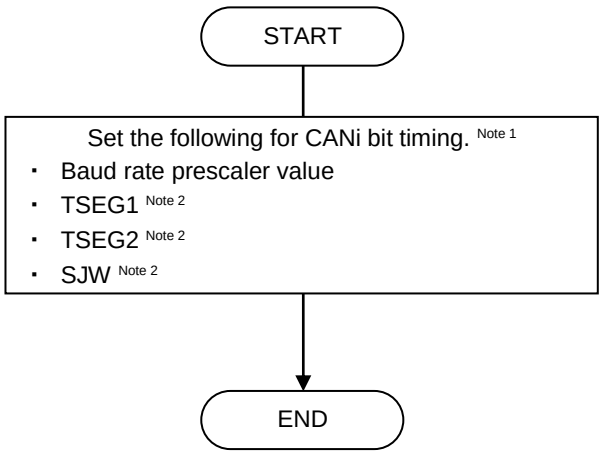
Note: A position determining a level of one bit

1.3.3 Procedure for setting CAN bit timing and communication speed

Figure 1.11 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 CiCFGL and CiCFGH 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 communication mode or channel halt mode.

Note 2: The following conditions need to be satisfied for the settings for TSEG1, TSEG2, and SJW:

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

Figure 1.11 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
Timestamp clock
Interval timer prescaler

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 transmit/receive FIFO buffer (transmit mode).

With the transmit/receive FIFO buffer, the oldest (stored earlier) messages in the transmit/receive FIFO buffer are the targets for priority determination.

When a message is being transmitted from the transmit/receive FIFO buffer, the messages in the same transmit/receive 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 transmit/receive 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 for the messages that have passed through the acceptance filter processing.

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 transmit/receive 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 filter processing is carried out for a message while the DLC replacement function is disabled, the DLC value of the received message is stored in the buffer. In this case, 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 1.6 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 rules 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 1.7 Messages target for data processing based on the mirror function

MME bit in the GCFGL register	GAFLLB bit in the GAFLIDHj 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

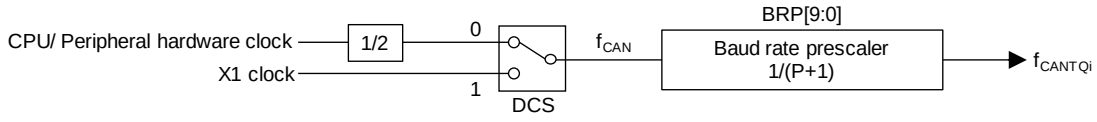
1.4.5 CAN clock source setting

The setting of the CAN clock (f_{CAN}) as a clock source is described below. The following clocks can be used as a CAN clock source.

Clock obtained by frequency-dividing the CPU/peripheral hardware clock (f_{CLK}) by 2
X1 clock (fx) Note

Note: When using the X1 clcok as a clock source, make the X1 clock into the value less than or equal to the half of the CPU/peripheral hardware clock (f_{CAN}). However, if the clock source of f_{CLK} is high-speed on-chip oscillator clock (f_{IH}), make sure that the conditions $f_X < f_{CLK}/2$ is satisfied.

Figure 1.12 illustrates the CAN system clock generator.



$$\text{Communication speed} = \frac{f_{CAN}}{\text{Baud rate prescaler division ratio} \times (\text{Tq count of 1 bit time})}$$

Caution When f_X is to be selected, the following condition must be satisfied.

$$f_X \leq \frac{f_{CLK}}{2}$$ Notes 1,2

Notes 1. If the high-speed on-chip oscillator clock (f_{IH}) or the PLL clock with its sources as the high-speed on-chip oscillator clock is to be selected as the source of the clock signal for f_{CLK}, make sure that the condition $f_X < f_{CLK}/2$ is satisfied.
2. If the high-speed system clock is to be selected as f_{CLK}, do not select f_X as f_{CAN}.

Remark DCS : Bit in the GCFGL register
BRP[9:0] : Bits in the CiCFGL register
f_{CAN} : CAN clock
f_{CANTQi} : CANi Tq clock

Figure 1.12 CAN clock generator

1.4.6 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. The value of the timestamp counter is fetched at the start-of-frame (SOF) [Note] timing of a message and then stored in a receive buffer or a FIFO buffer together with the message ID and its data.

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

Clock obtained by frequency-dividing the CPU/peripheral hardware clock by 2, or
CANi bit time clock

Note: Start of Frame (SOF): A field indicating a start of a frame

When the CANi 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 clock, which is obtained by frequency-dividing the CPU/peripheral hardware clock by 2, is used as a clock source, the timestamp function (counter) is not affected by channel modes.

Figure 1.13 is a block diagram of the timestamp function.

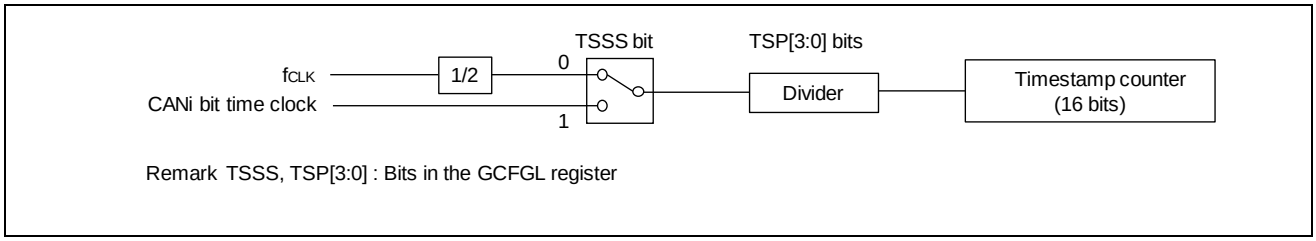


Figure 1.13 Timestamp function

1.4.7 Interval timer prescaler setting

When the TSSS bit is set to 0, f_{CLK} (the clock obtained by frequency-dividing f_{CLK} by 2) is the clock source for the interval timer. The clock source is divided by the ITRCP[15:0] bits in the GCFGH register and the CFITT[7:0] bit in the CFCCHk register.

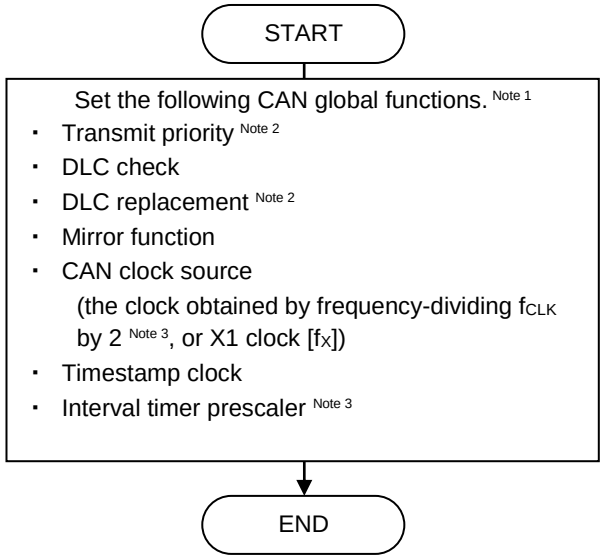
For detailed information on the interval timer function, see **1.6.3(4) Setting of interval timer**.

1.4.8 Setting of global functions

Figure 1.14 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 and GCFGH 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), set the DLC of the receive rule table (the GAFLDLC bit in the GAFLPHj register) to 0.

Note 3: 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 1.14 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.

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

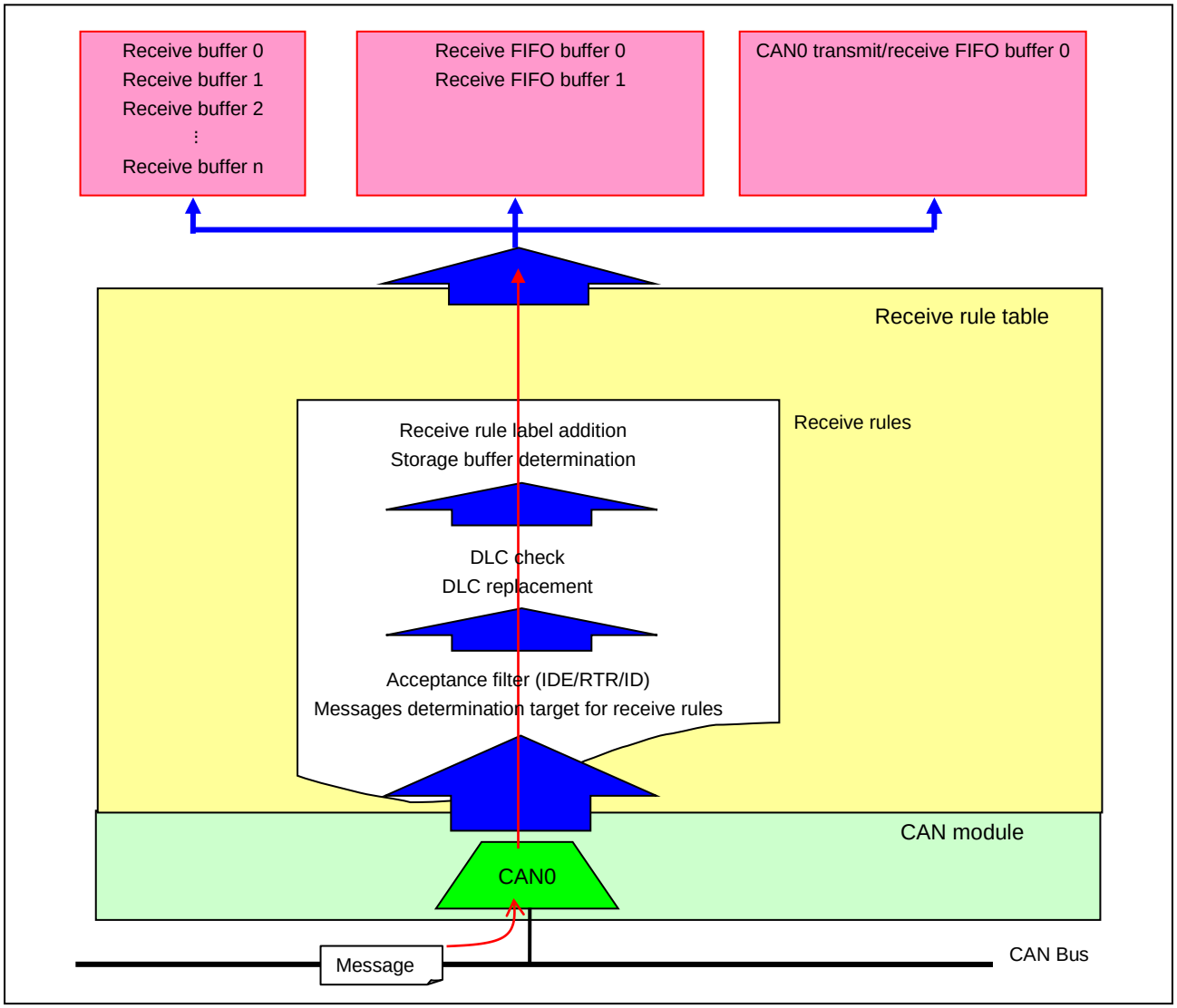


Figure 1.15 Filtering based on receive rule table

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

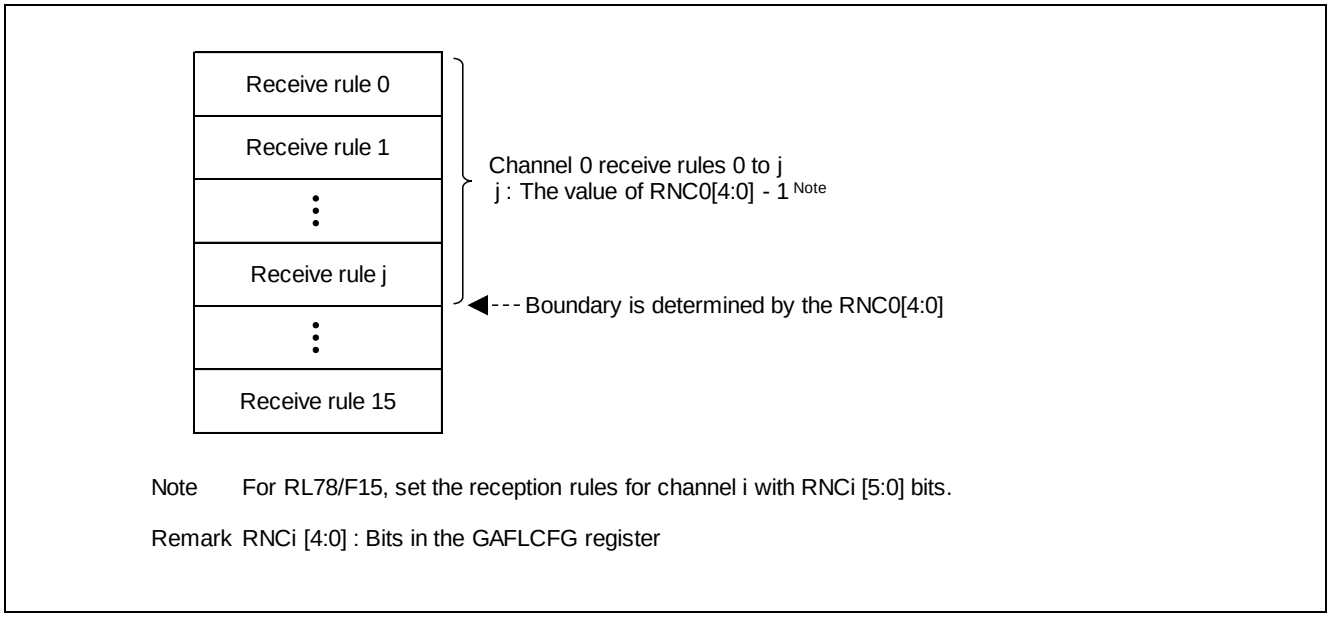


Figure 1.16 Registration of receive rules

1.5.2 Settings of IDE/RTR/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.

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 GAFLIDHj 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 GAFLIDHj 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/ID

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

With this setting, the acceptance filtering is enabled for the bits that have not been masked with the IDE, RTR 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 a 12-bit label to be attached to a message that has passed through the DLC filter. The label can be attached when the message is stored in a buffer.

The 12-bit label 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 label, it becomes possible to identify the 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 m
Transmit/receive FIFO buffer k (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 m plus one receive buffer n
Maximum of two buffers = one receive FIFO buffer m plus one transmit/receive FIFO buffer k

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 that 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 : 010h
Storage buffer: receive buffer 3, receive FIFO buffers 0 and 1

		GAFLIDE/GAFLIDEM	GAFLRTR/GAFLRTRM	GAFLLB	GAFLID/GAFLIDM			
GAFLIDLj, GAFLIDHj		0	0	0	B’00000	B’00000000	B’00000001	B’00100000
GAFLMLj, GAFLMHj		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	GAFLRMV	GAFLRMDP	GAFLFDP
GAFLPHj	6	H’010	-	-	-
GAFLPLj	-	-	1	3	B'00011

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, 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 : 130h
Storage buffer: receive FIFO buffer 0, transmit/receive FIFO buffer 0

		GAFLIDE/GAFLIDEM	GAFLRTR/GAFLRTRM	GAFLLB	GAFLID/GAFLIDM			
GAFLIDLj, GAFLIDHj		0	0	0	B'00000	B'00000000	B'00000001	B'00110000
GAFLMLj, GAFLMHj		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

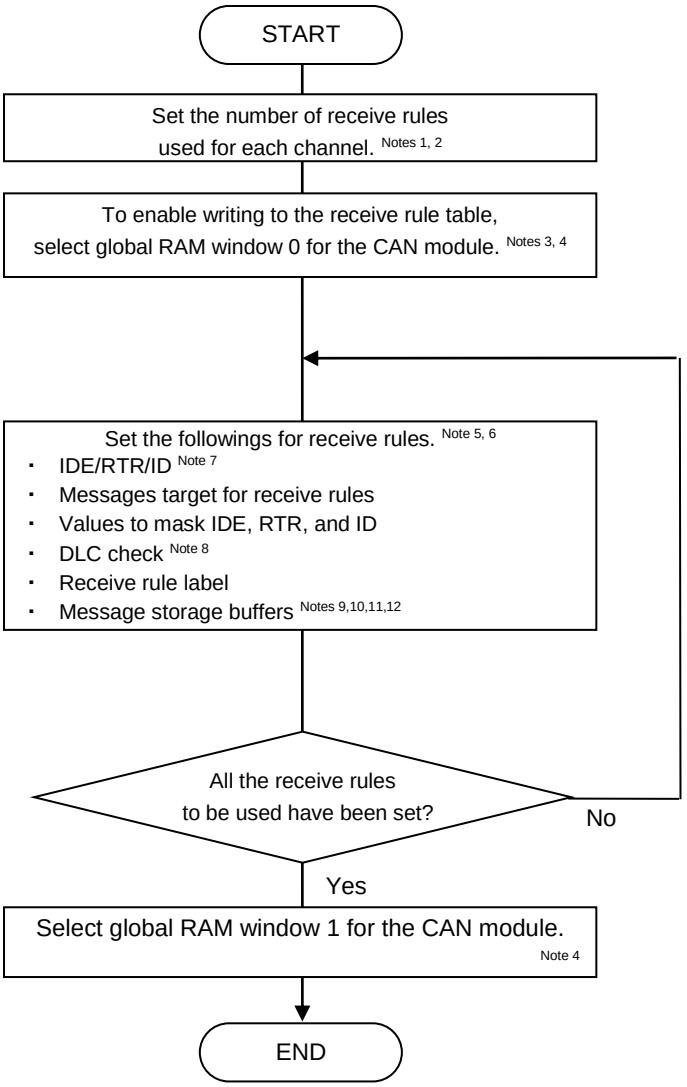
	GAFDLC	GAFLPTR	GAFLRMV	GAFLRMDP	GAFLFDP
GAFLPHj	0	H'130	-	-	-
GAFLPLj	-	-	0	0	B'10001

1.5.9 Procedures for setting receive rule table

Figure 1.17 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 RNCi[4:0] bits in the GAFLCFG register) when the CAN module is in global reset mode.

Note 2: The following conditions need to be satisfied to set the number of receive rules (RNCi[4:0] bits) used for the (each) channel:

*The number of receive rules for one channel must be 16 or less.

*The total number of the receive rules allocated to the (each) channel must not exceed the total number of receive rules that can be registered for the entire module.

Note 3: To write to the receive rule table, global RAM window 0 for the CAN module needs to be selected (set the RPAGE bit in the GRWCR register to 0).

Note 4: After writing to the receive rule table is completed, select global RAM window 1 for the CAN module (set the RPAGE bit in the GRWCR register to 1).

Figure 1.17 Setting procedures for receive rule table (1/2)

Note 5: Modify the receive rules (the GAFLIDLj register, GAFLIDHj register, GAFLMLj register, GAFLMHj register, GAFLPLj register, and GAFLPHj register) when global RAM window 0 for the CAN module is selected (set the RPAGE bit in the GRWCR register to 0) and also the CAN module is in global reset mode.

Note 6: Set the receive rules for the (each) channel in succession. The receive rules cannot be shared with another channel and cannot be set alternately (for each channel).

Note 7: 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 GAFLIDLj register) and also set 0 to bits 15-11 in the GAFLIDLj register and the GAFLID[28:16] bits in the GAFLIDHj register.

Note 8: When the IDE bit is not to be compared (set the GAFLIDEM bit in the GAFLMHj register to 0), set the GAFLIDM[28:16] bits in the GAFLMHj register and the GAFLIDM[15:0] bits in the GAFLMLj register to all 0 to disable the comparision of the ID bits.

Note 9: This setting is valid only when the DLC check is enabled (set the DCE bit in the GCFGL register to 1).

Note 10: A maximum of two FIFO buffers can be selected. However, when storing a message in one receive buffer (set the GAFLRMV bit in the GAFLPLj register to 1), the number of FIFO buffers that can be selected is only one.

Note 11: Select only one receive FIFO buffer and one transmit/receive FIFO buffer which is set to receive mode.

Note 12: 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 1.17 Procedures for setting receive rule table (2/2)

1.6 Buffers and FIFO buffers

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

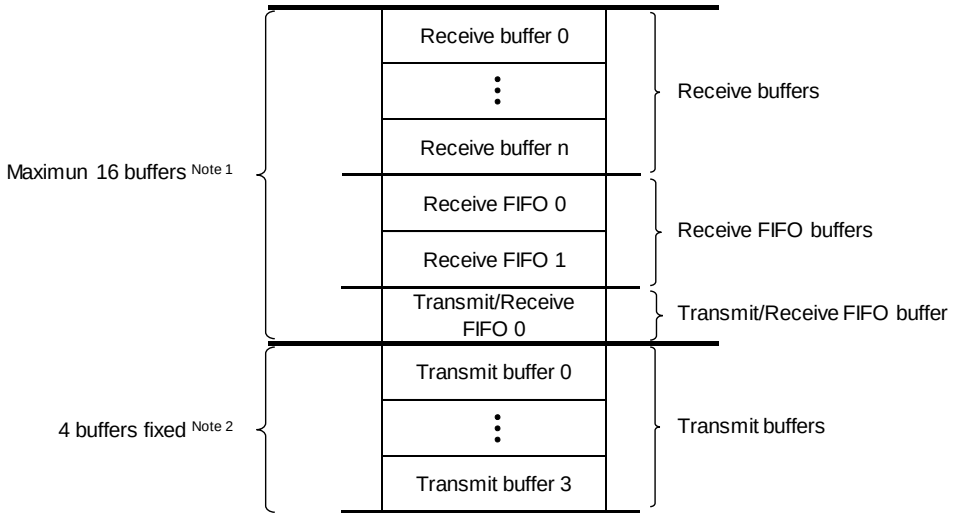
receive buffer
Receive FIFO buffer
Transmit/receive FIFO buffer
Transmit buffer
Transmit history buffer

The below is the limitation on the receive buffer, receive FIFO buffer, and transmit/receive 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 transmit/receive FIFO buffer 0

  (the sum of the buffers above) ≦ 16 buffers

Figure 1.18 illustrates the buffer configuration.



Notes 1. RL78/F15 can have up to 40 buffers.
2. RL78/F15 is fixed to 4 buffers for each channel 0 and 1 (fixed 8 buffers in total).

Caution Recive buffers, receive FIFO buffers, transmit/receive FIFO buffer, and transmit buffers are located in succession.

Figure 1.18 Buffer configuration

1.6.1 Setting of receive 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.

Since there is no receive buffer-related interrupt, there is no need to set interrupts.

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 RFCCm register).

(2) Enable/disable interrupts and set interrupt source

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 RFCCm register is set to 0) when the conditions set by the RFIGCV[2:0] bits in the RFCCm 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 RFCCm 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 transmit/receive FIFO buffer

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

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

(1) Setting of the number of buffers

Set the number of transmit/receive FIFO buffers.

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

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

(2) Enable/disable interrupts and set interrupt sources

Enable/disable interrupts for each transmit/receive FIFO buffer, and set the sources for the interrupts. **Table 1.8** lists the interrupt sources that can be set for each mode of the transmit/receive FIFO buffer.

Table 1.8 Transmit/receiver FIFO buffer interrupt sources

Mode of transmit/receive 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 transmit/receive FIFO receive interrupt request will be generated. Values set to the CFIGCV[2:0] bits: B'000: the transmit/receive FIFO buffer is 1/8 full [Note] B'001: the transmit/receive FIFO buffer is 2/8 full B'010: the transmit/receive FIFO buffer is 3/8 full [Note] B'011: the transmit/receive FIFO buffer is 4/8 full B'100: the transmit/receive FIFO buffer is 5/8 full [Note] B'101: the transmit/receive FIFO buffer is 6/8 full B'110: the transmit/receive FIFO buffer is 7/8 full [Note] B'111: the transmit/receive FIFO buffer is full
	1	Every time one message reception is completed, a transmit/receiver FIFO receive interrupt request is generated.
Transmit mode	0	When the buffer becomes empty (contains no message) upon completion of message transmission, a transmit/receive FIFO transmit completion interrupt request is generated.
	1	Every time one message transmission is completed, a transmit/receiver FIFO transmit interrupt request is generated.

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

The transmit/receive FIFO transmit interrupt triggers the CANi transmit interrupt. The following are the sources for the CANi transmit interrupt.

CANi transmit complete interrupt
CANi transmit abort interrupt
CANi transmit/receive FIFO transmit complete interrutp
CANi transmit history interrupt

(3) Mode setting for transmit/receive FIFO buffer

As a mode of the transmit/receive 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.

(4) Setting of interval timer

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

Table 1.9 lists the count sources for the interval timer and formulas to calculate the interval time.

Table 1.9 Count sources for interval timer and formulas to calculate interval time

CFITR and CFITSS bits in the CFCCHk register	Count source	Formulas Note
B'00	the clock obtained by frequency-dividing the CPU/peripheral hardware clock (f_{CLK})/2 by the value of ITRCP[15:0] bits in the GCFGH register.	$1/f_{CLK}$ x 2 x a x b
B'10	the clock obtained by frequency-dividing the CPU/peripheral hardware clock (f_{CLK})/2 by the value of the ITRCP[15:0] bits in the GCFGH register and multiplying the divided value by 10	$1/f_{CLK}$ x 2 x a x 10 x b
B'x1	CANi bit time clock	$1/f_{CANBIT}$ x b

Remark:
a : a value of the prescaler for the CPU/peripheral hardware clock (a value set to the ITRCP[15:0] bits)
b : a message transmission interval set by the CFITT[7:0] bits in the CFCCHk register
f_{CLK} : CPU/peripheral hardware clock frequency
f_{CANBIT} : CANi bit time clock frequency

(5) Setting of transmit buffer link

Link the transmit/receive 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 transmit/receive FIFO buffer (set to transmit mode).

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

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

- CANi transmit complete interrupt
- CANi transmit abort interrupt
- CANi transmit/receive FIFO transmit complete interrupt
- CANi transmit history interrupt

1.6.5 Setting of transmit history buffers

The settings for transmit history buffers are described below.

Each channel has a single transmit history 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 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.

Transmit/receive FIFO buffers
Transmit buffers, transmit/receive 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 buffer.
Every time history data of one transmission have been stored.

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

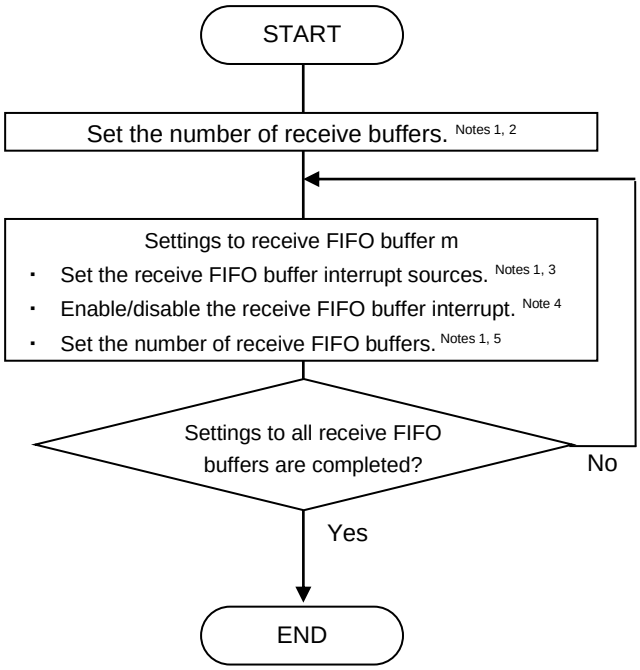
- CANi transmit complete interrupt
- CANi transmit abort interrupt
- CANi transmit/receive FIFO transmit complete interrupt
- CANi transmit history interrupt

1.6.6 Procedures for setting buffers

Figure 1.19 shows the procedures for setting the receive buffer and the receive FIFO buffer. **Figure 1.20** shows the procedures for setting the transmit/receive FIFO buffer, the transmit buffer and the transmit history 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 and the RFDC[2:0] bits in the RFCCm register when the CAN module is in global reset mode.

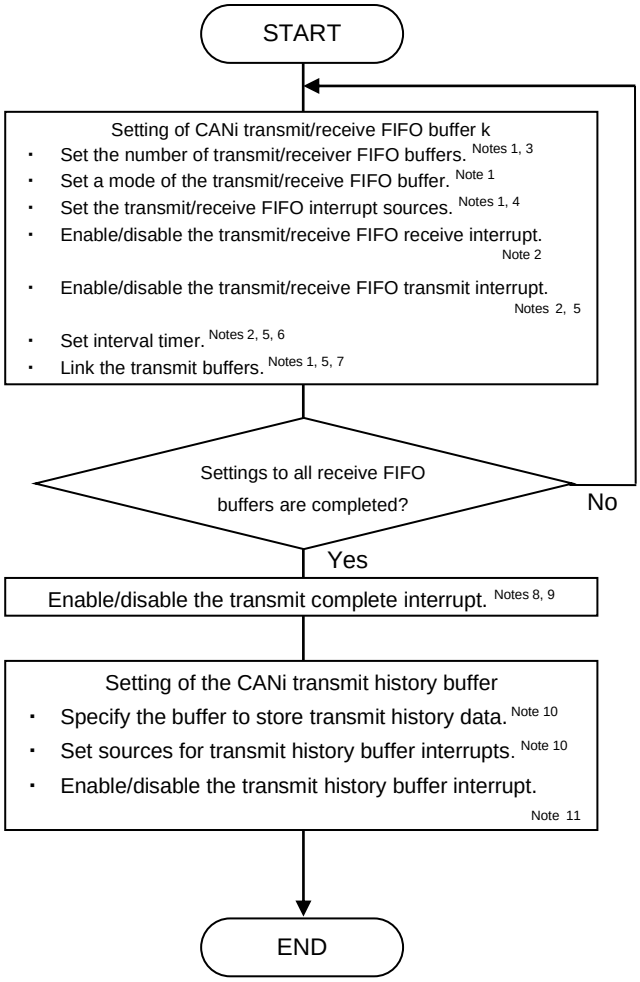
Note 2: Set the number of receive buffers within a range of 0 to 16. Note that when the number (the NRXMB[4:0] bits in the RMNB register) is set to 0, no receive buffer can be used.

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: Enable/disable the receive FIFO buffer interrupt with the RFIE bit in the RFCCm register while the receive FIFO buffer is not in use (when the RFE bit in the RFCCm register is set to 0).

Note 5: 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 1.19 Procedures for setting receive buffers and receive FIFO buffers



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

Note 2: Modify the values of the CFRXIE bit, the CFTXIE bit, the CFITR bit, the CFITSS bit and the CFITT[7:0] bits in the CFCCLk and CFCCHk registers while the transmit/receive FIFO buffer is not in use (when the CFE bit in the CFCCLk register is set to 0).

Note 3: When the transmit/receive 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 transmit/receive 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 transmit/receive 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 CFCCHk register.

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

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

Note 9: Set 0 to the TMIEp 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 THLCCi register when the CAN module is in channel reset mode.

Note 11: Enable/disable the interrupt when the transmit history buffer is not used (when the THLE bit in the THLCCi register is set to 0).

Figure 1.20 Procedures for setting transmit buffers,

transmit/receive FIFO buffers, and transmit history 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
Transmit history buffer 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 transmit/receive 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) Transmit history buffer overflow

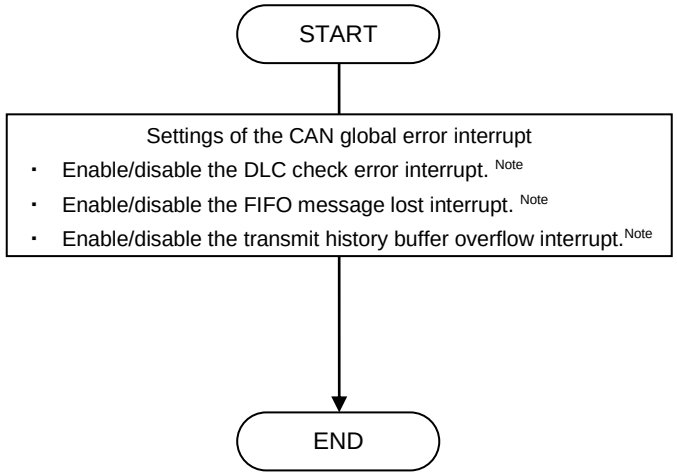
When a transmit history buffer which has been already full attempts to store further transmit history data to the buffer itself, the transmit history buffer overflows.

1.7.2 Procedures for setting global error interrupts

Figure 1.21 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 THLEIE, MEIE and DEIE bits in the GCTRL register when the CAN module is in global reset mode.

Figure 1.21 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

1.8.1 CANi error interrupt

Enable/disable the CANi 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

(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 CiERFLL register is set to 1),
When a recessive bit is detected although a dominant bit has been transmitted (when the B0ERR flag in the CiERFLL register is set to 1),
When a dominant bit is detected although a recessive bit has been transmitted (when the B1ERR flag in the CiERFLL register is set to 1),
When a CRC error is detected (when the CERR flag in the CiERFLL register is set to 1),
When an ACK error is detected (when the AERR flag in the CiERFLL register is set to 1),
When a form error is detected (when the FERR flag in the CiERFLL register is set to 1), or
When a stuff error is detected (when the SERR flag in the CiERFLL 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 CiCTRH 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 communication mode is regarded as the bus lock state.

(8) Arbitration lost

An interrupt is generated when arbitration lost is detected.

1.8.2 CANi 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 CANi transmit interrupts. The following are the sources for the CANi transmit interrupt.

- CANi transmit complete interrupt
- CANi transmit abort interrupt
- CANi transmit/receive FIFO transmit complete interrupt
- CANi 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 1.22** to **Figure 1.25** illustrate the operations at the bus-off recovery.

Table 1.10 Operations at bus off recovery

BOM[1:0] bits in the CiCTRH 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 CiCTRL register is set to B'01).

Note 2: When the CHMDC[1:0] bits in the CiCTRL 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 CiCTRL 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 communication 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.

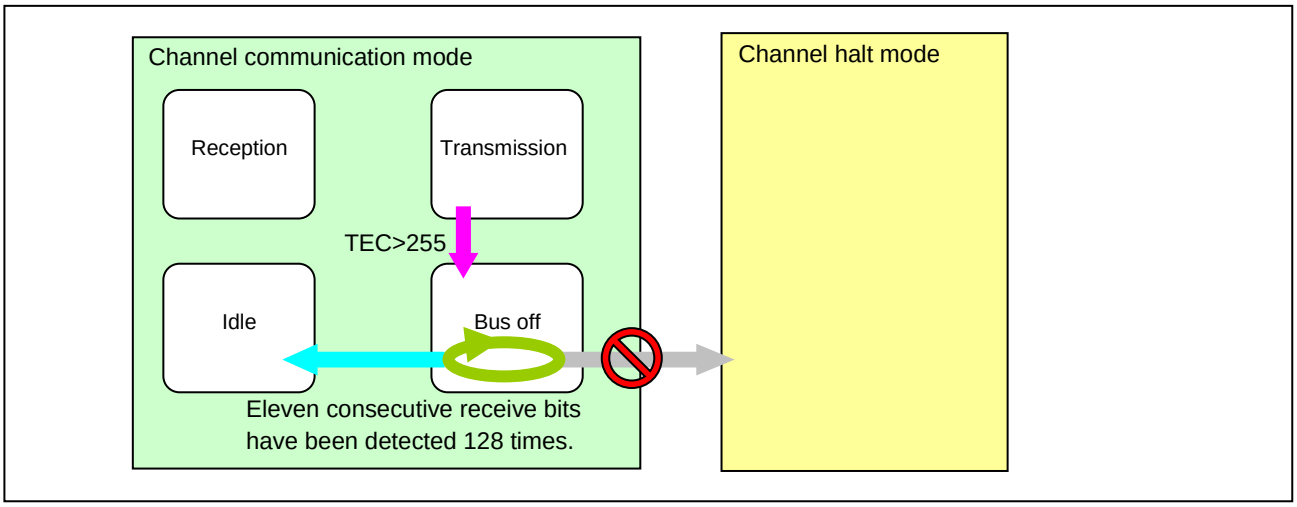


Figure 1.22 ISO11898-1 Specification compliant operation

(when the value of the BOM[1:0] bits is B'00)

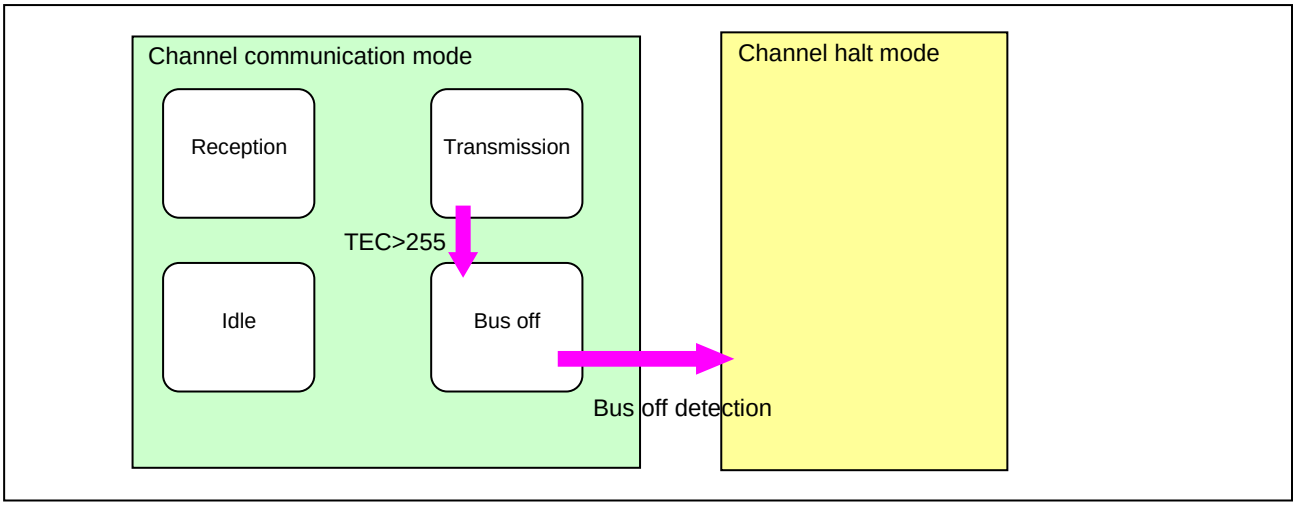


Figure 1.23 Operation at transition to channel halt mode at bus off entry

(when the value of the BOM[1:0] bits is B'01)

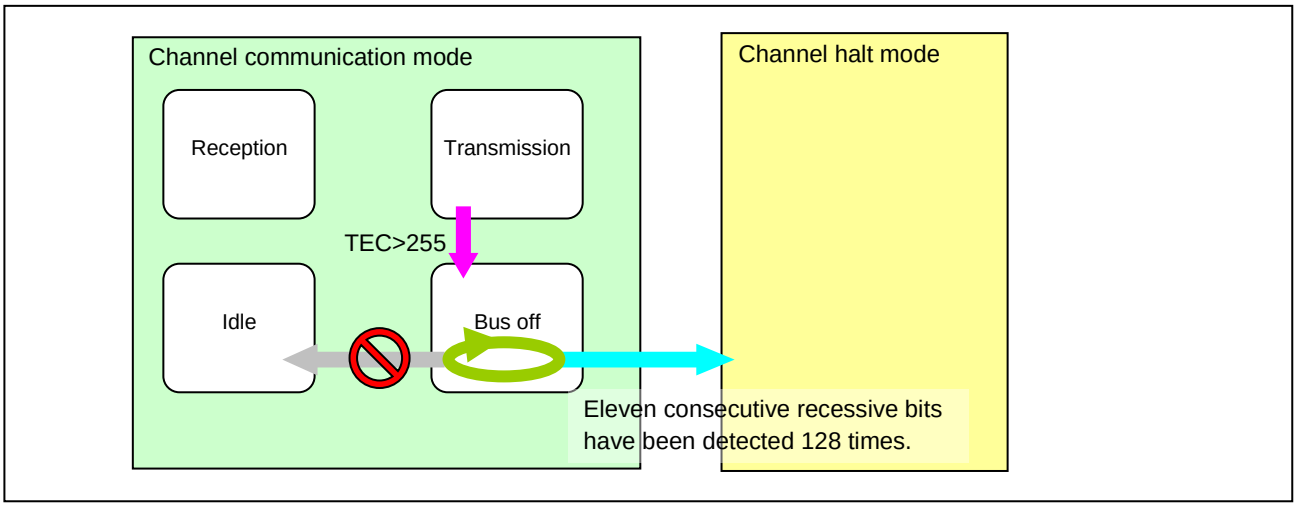


Figure 1.24 Operation at transition to channel halt mode at bus off end
(when the value of the BOM[1:0] bits is B'10)

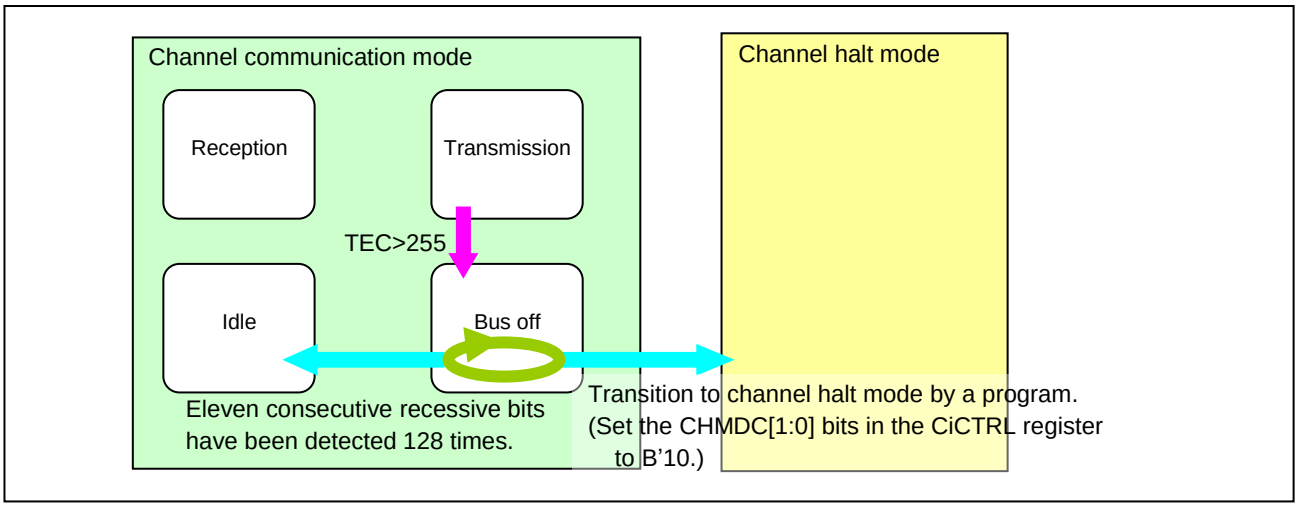


Figure 1.25 Operation at transition to channel halt mode
due to request by the program during bus-off state
(when the value of the BOM[1:0] bits is B'11)

1.8.4 Settings of error display mode

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

Indication of only the first error (Set the ERRD bit in the CiCTRH 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 CiCTRH register to 1.)

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

Figure 1.26 illustrates the operations of the CiERFLL register in each error indication mode.

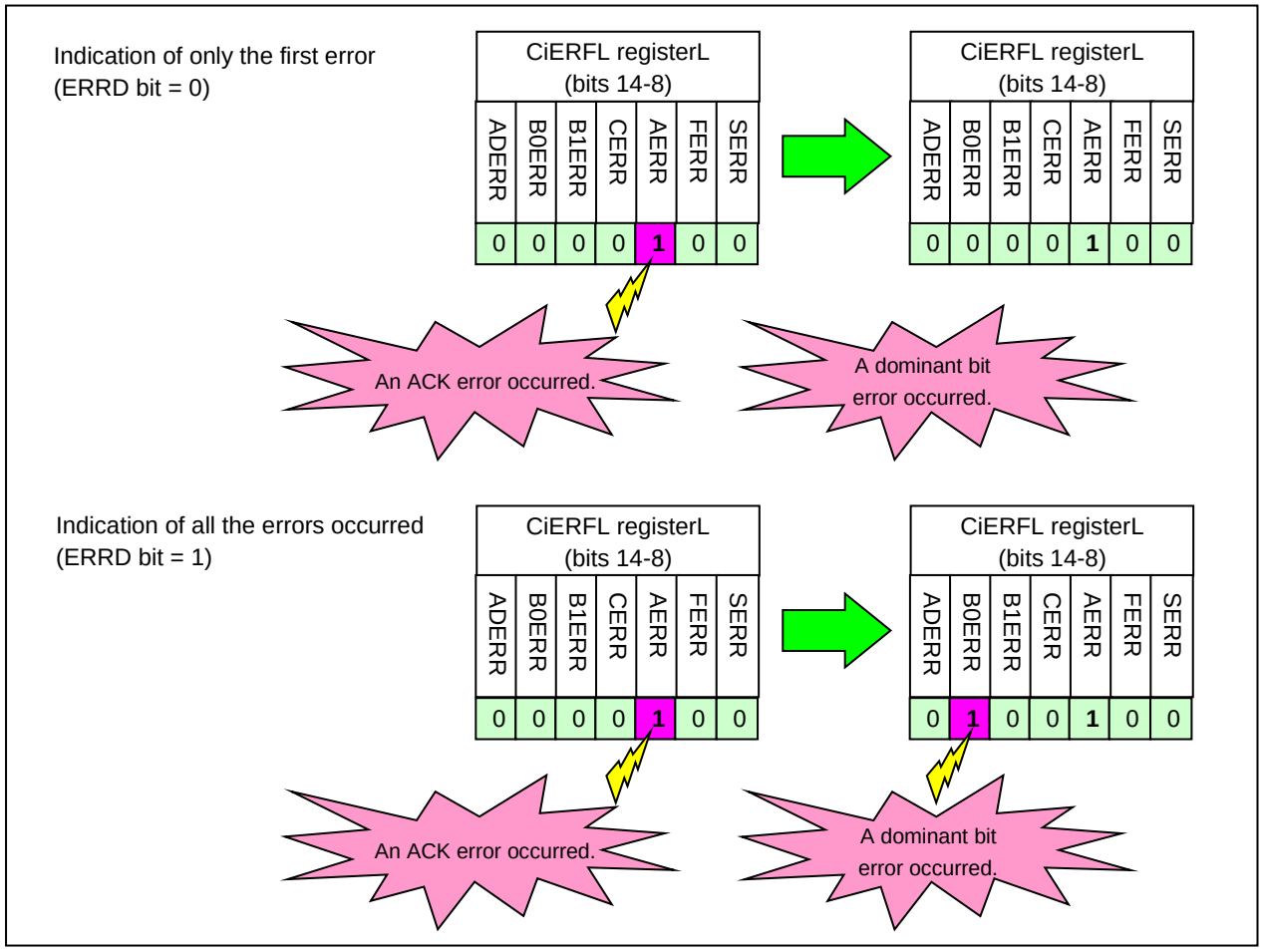


Figure 1.26 Error indication

1.8.5 Settings of communication test mode

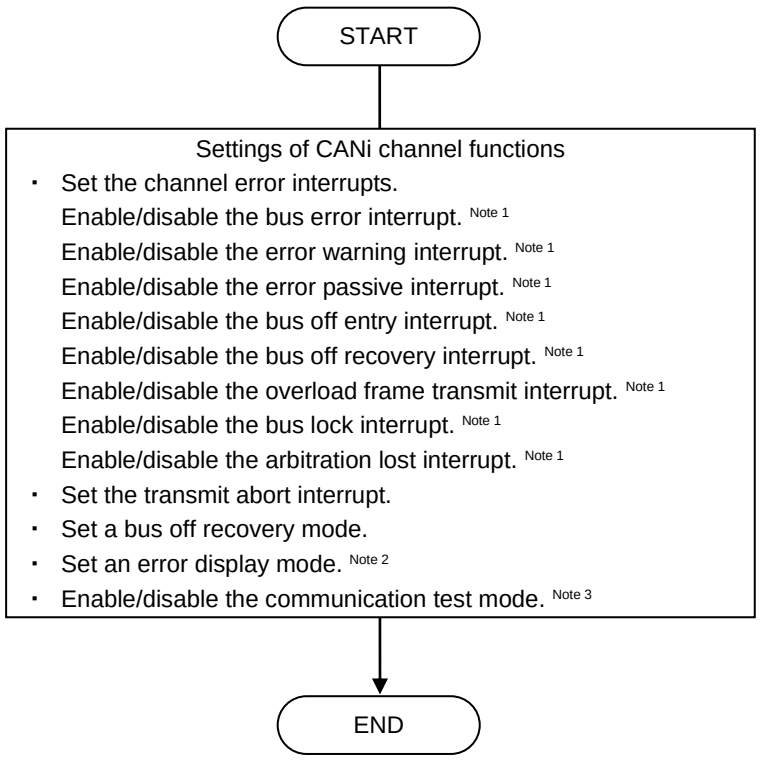
Set the communication test mode. This communication test mode allows self-tests for CAN communication or RAM using the CAN transceiver or MCUs.

1.8.6 Procedures for setting channel functions

Figure 1.27 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, and TAIE bits in the CiCTRL and CiCTRH registers when the CAN module is in channel reset mode.

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

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

Figure 1.27 Channel function setting procedure

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 transmit/receive FIFO buffers

2.2 Reception using receive buffers

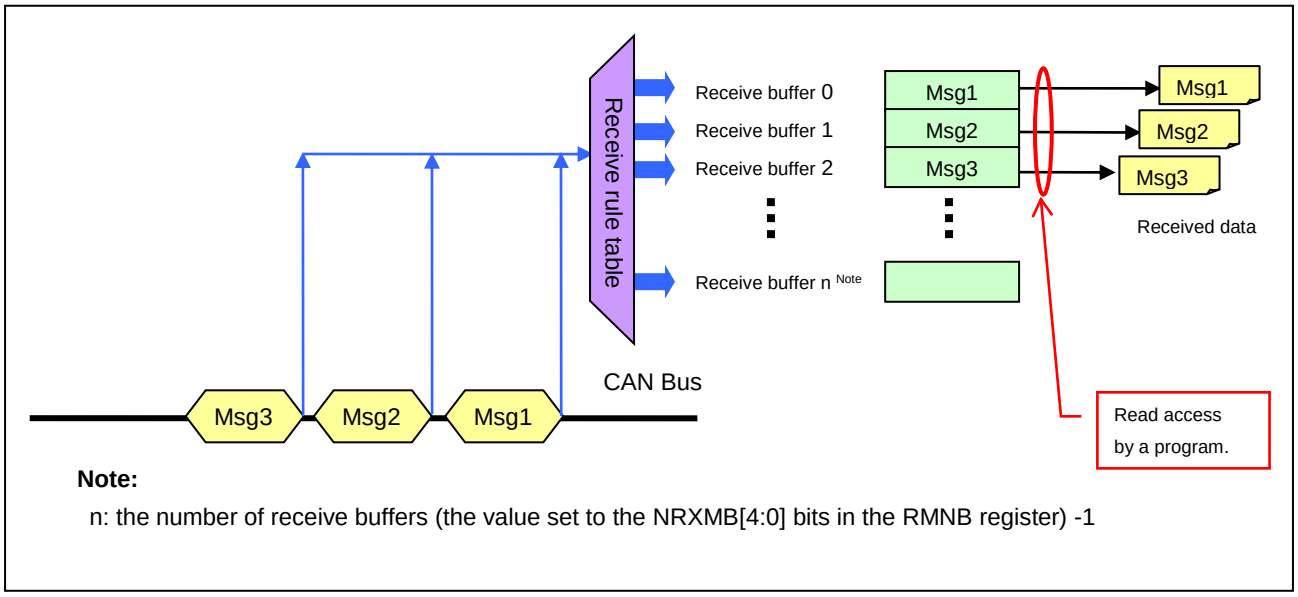
Zero to n+1 receive buffers can be shared by the channel (all channels). Data (messages) in a receive buffer will be overwritten when a new message is stored in the same receive buffer. Thus, the latest receive data can be read.

When a receive buffer receives a message, no interrupt is generated.

When the process of storing a received message in receive buffers starts, the RMNSn flag in the RMNDi register is set to 1, which means receive buffer n contains the new message. Then the data can be read from the RMIDLn and RMIDHn registers, RMTSn register, RMPTRn register, and RMDF0n to RMDF3n registers.

Regarding the configuration to use receive buffers, see **1.1 CAN configuration**.

Figure 2.1 illustrates the operation of receive buffers.

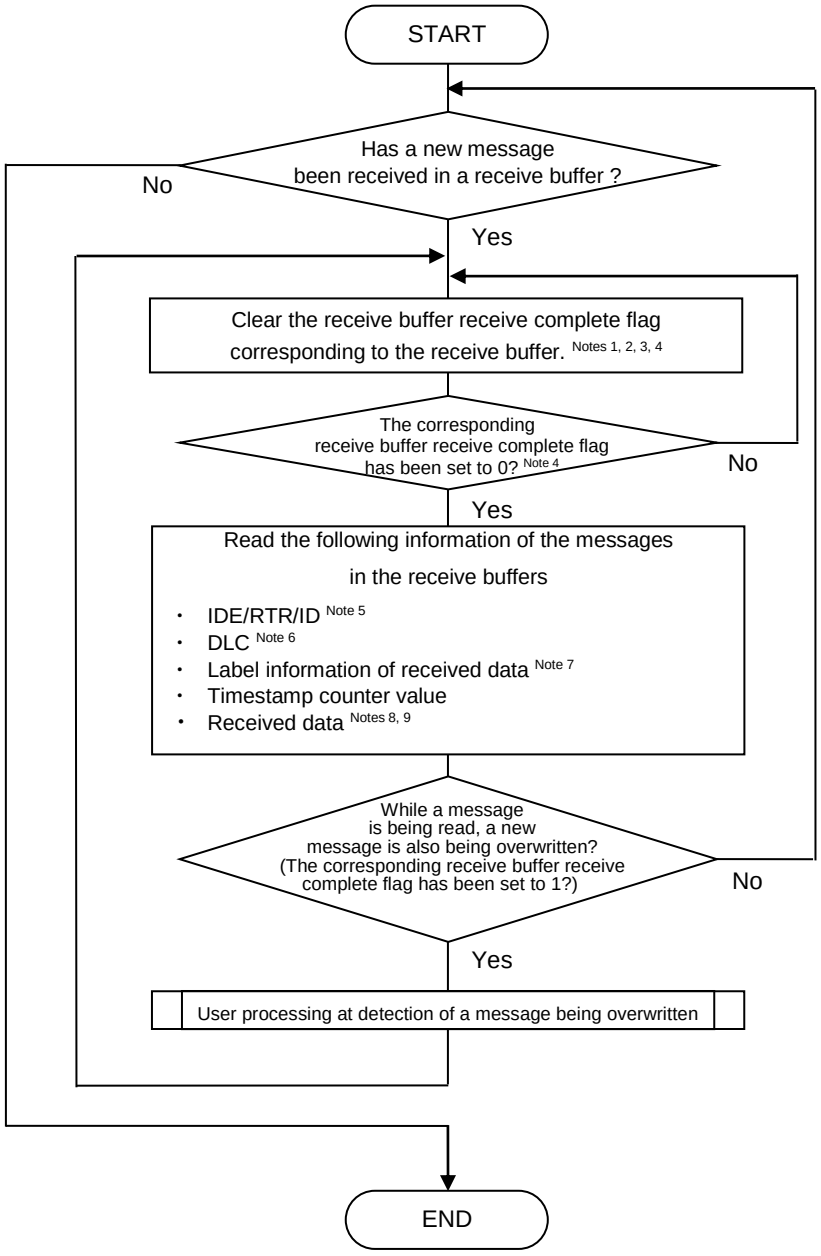


Note:
n: the number of receive buffers (the value set to the NRXMB[4:0] bits in the RMNB register) -1

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: Write 0 to the RMNSn flag in the RMNDi register when the CAN module is in global operating mode or global test mode.

Note 2: To set 0 to the RMNSn flag, write 0 by a program.

Note 3: 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 4: The RMNSn flag cannot be set to 0 while a message is being stored. For storing a message, it takes ten clock cycles of the CPU/peripheral hardware clock.

Figure 2.2 Reading of receive buffers (1/2)

Note 5: When the standard ID is selected, read bits 10-0 of the ID data (the RMID[15:0] bits in the RMIDLn register). Bits 15-11 and the RMID[28:16] bits in the RMIDHn register are read as 0.

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 GAFLPHj 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[11:0] bits in the GAFLPHj register) which has agreed with the value of the received message will be stored.

Note 8: When the DLC value of the received message is smaller than 8 (the value of the RMDLC[3:0] bits in the RMPTRn register is smaller than B'1000), data bytes (the RMDB0[7:0] bits in the RMDF0n register to the RMDB7[7:0] bits in the RMDF3n regiters) where no data have been set are read as H'00.

Note 9: When global RAM window 1 for the CAN module is selected (the RPAGE bit in the GRWCR register is set to 1), the receive buffers (the RMIDLn and RMIDHn registers, RMTSn register, RMPTRn register, RMDF0n to RMDF3n registers) can be read.

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 RFSTSm register) is incremented.

Received messages can be read from the RFIDLm and RFIDHm registers, the RFTSm register, the RFPTRm register, the RFDF0m to RFDF3m 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 RFCCm register), the receive FIFO buffer is full (the RFFLL flag in the RFSTSm 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 RFSTSm 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.

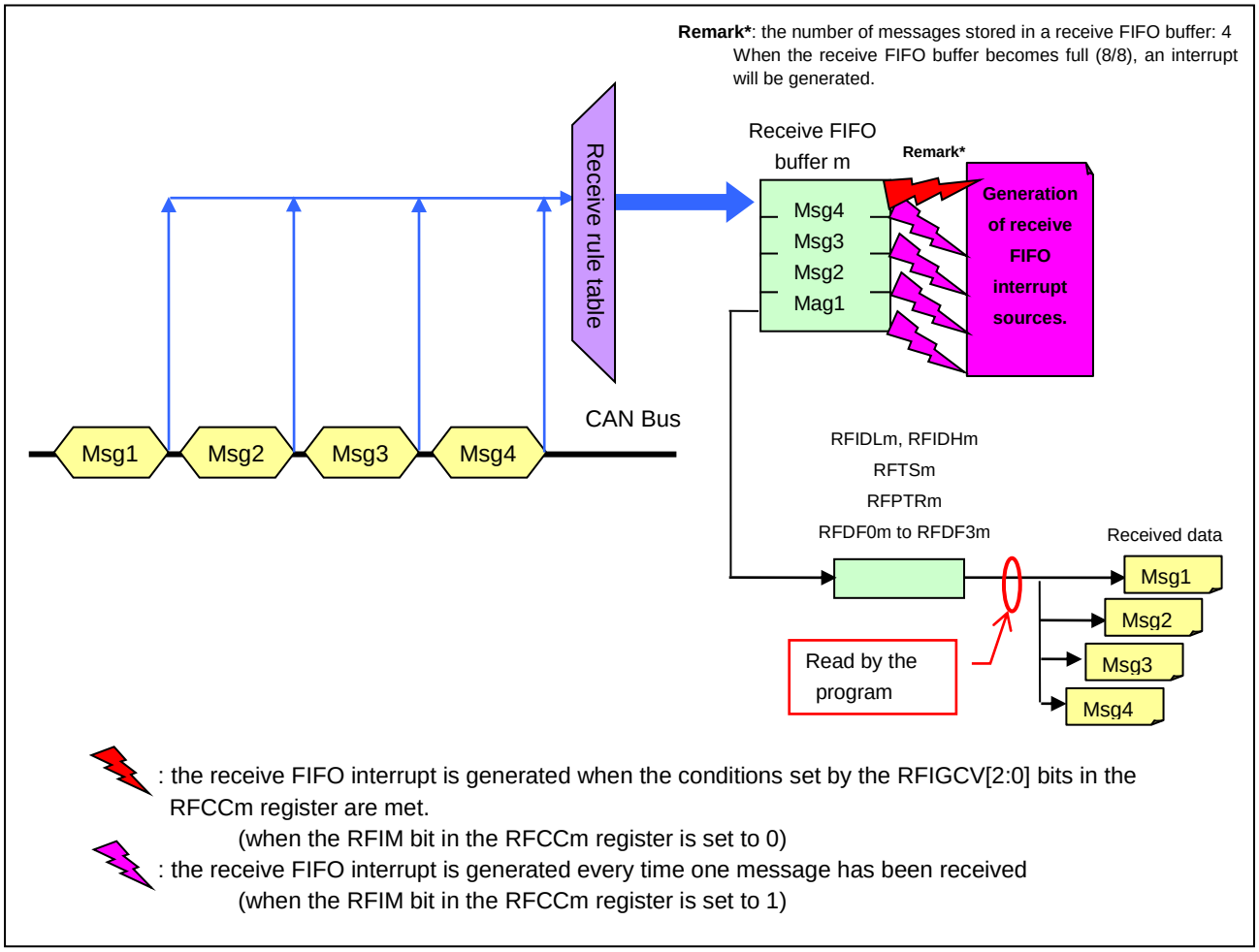
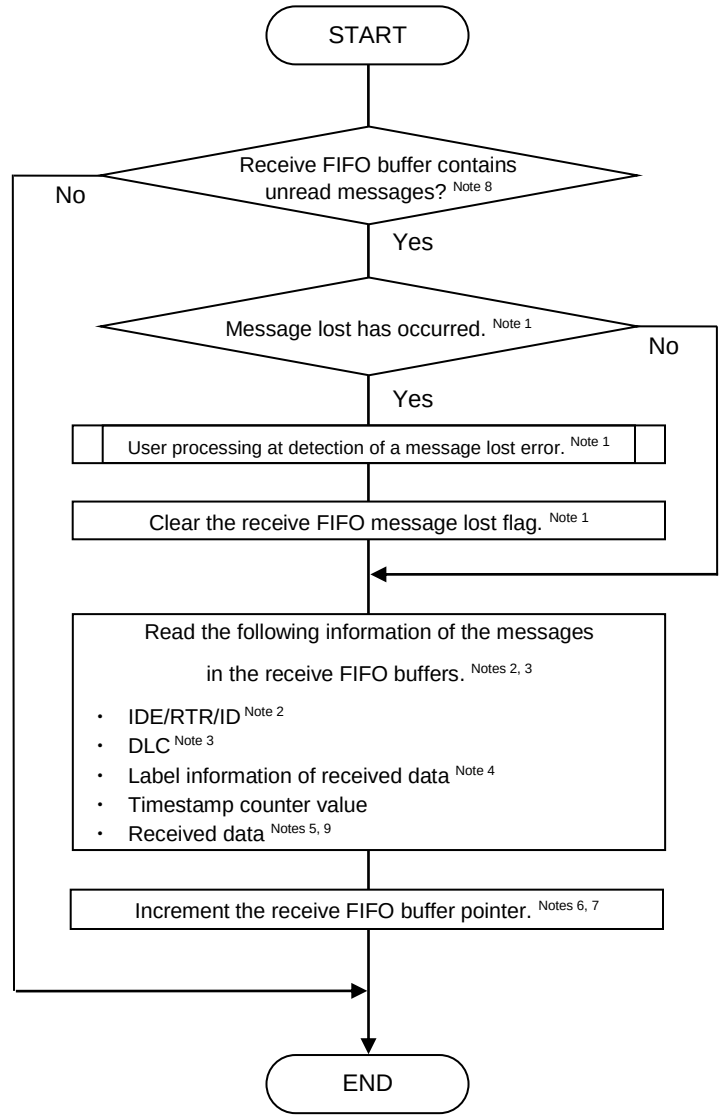


Figure 2.3 Operation of receive FIFO buffer

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

Note 3: 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 GAFLPHj 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.

Figure 2.4 Receive FIFO buffer reading procedure (no interrupt used) (1/2)

Note 4: After the filter processing according to the receive rules, the value set to the label data of the receive rule table (the GAFLPTR[11:0]bits in the GAFLPHj register) which has agreed with the data of the received message will be stored.

Note 5: When the DLC value of the received message is smaller than 8 (when the value of the RFDLC[3:0] bits in the RFPTRm register is smaller than B'1000), data bytes (the RFDB0[7:0] in the RFDF0m register to the RFDB7[7:0] bits in the RFDF3m registers) where no data have been set are read as H'00.

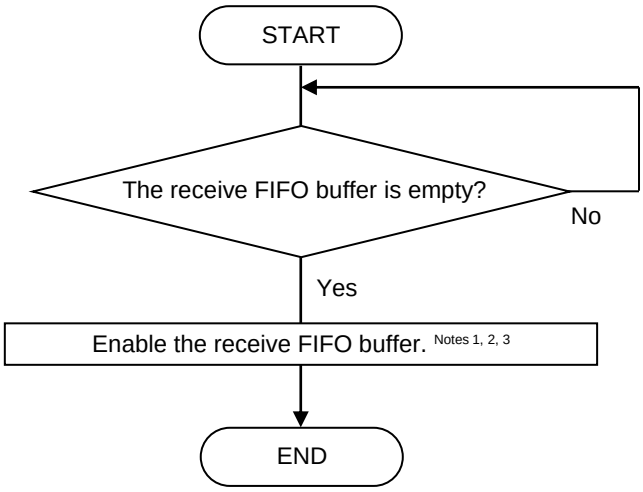
Note 6: After reading the messages in the receive FIFO buffer (the RFIDLm and RFIDHm registers, the RFTSm register, the RFPTRm register, and the RFDF0m to RFDF3m registers), increment the pointer (write H'FF to the RFPC[7:0] bits in the RFPCTRm register).

Note 7: When incrementing the pointer, the receive FIFO buffers must be used (the RFE bit in the RFCCm register is set to 1) and also the receive FIFO buffer needs to contain any unread message (when the RFEMP flag in the RFSTSm register is set to 0).

Note 8: 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).

Note 9: When global RAM window 1 for the CAN module is selected (the RPAGE bit in the GRWCR register is set to 1), the receive buffers (the RFIDLm and RFIDHm registers, the RFTSm register, the RFPTRm register, and the RFDF0m to RFDF3m registers) can be read.

Figure 2.4 Receive FIFO buffer reading procedure (no interrupt used) (2/2)

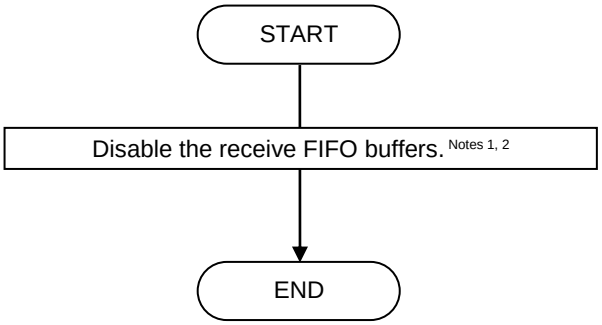


Note 1: To enable/disable the receive FIFO buffer, modify the value of the RFE bit in the RFCCm register when the CAN module is in global operating mode or global test mode.

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 RFCCm 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 RFCCm register when the CAN module is in global operating mode or global test 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 RFSTSm 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 RFCCm 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 RFSTSm 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 RFCCm 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 RFCCm register are met, the receive FIFO interrupt request will be issued (the RFIM bit in the RFCCm 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 RFCCm 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 RFCCm 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 4.3 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 transmit/receive FIFO buffers

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

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

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

The received messages can be read out from the CFIDLk, CFIDHk, CFTSk, CFPTRk, and CFDF0k to CFDF3k registers. Messages in the transmit/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 to be stored in the transmit/receive FIFO buffer (a value set by the CFDC[2:0] bits in the CFCCLk register), the transmit/receive FIFO buffer is full (the CFFLL flag in the CFSTSk register is set to 1).

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

Regarding the configuration to use the transmit/receive FIFO buffer, see **1.1 CAN configuration**.

Figure 2.7 illustrates the receiving operation of the transmit/receive FIFO buffer.

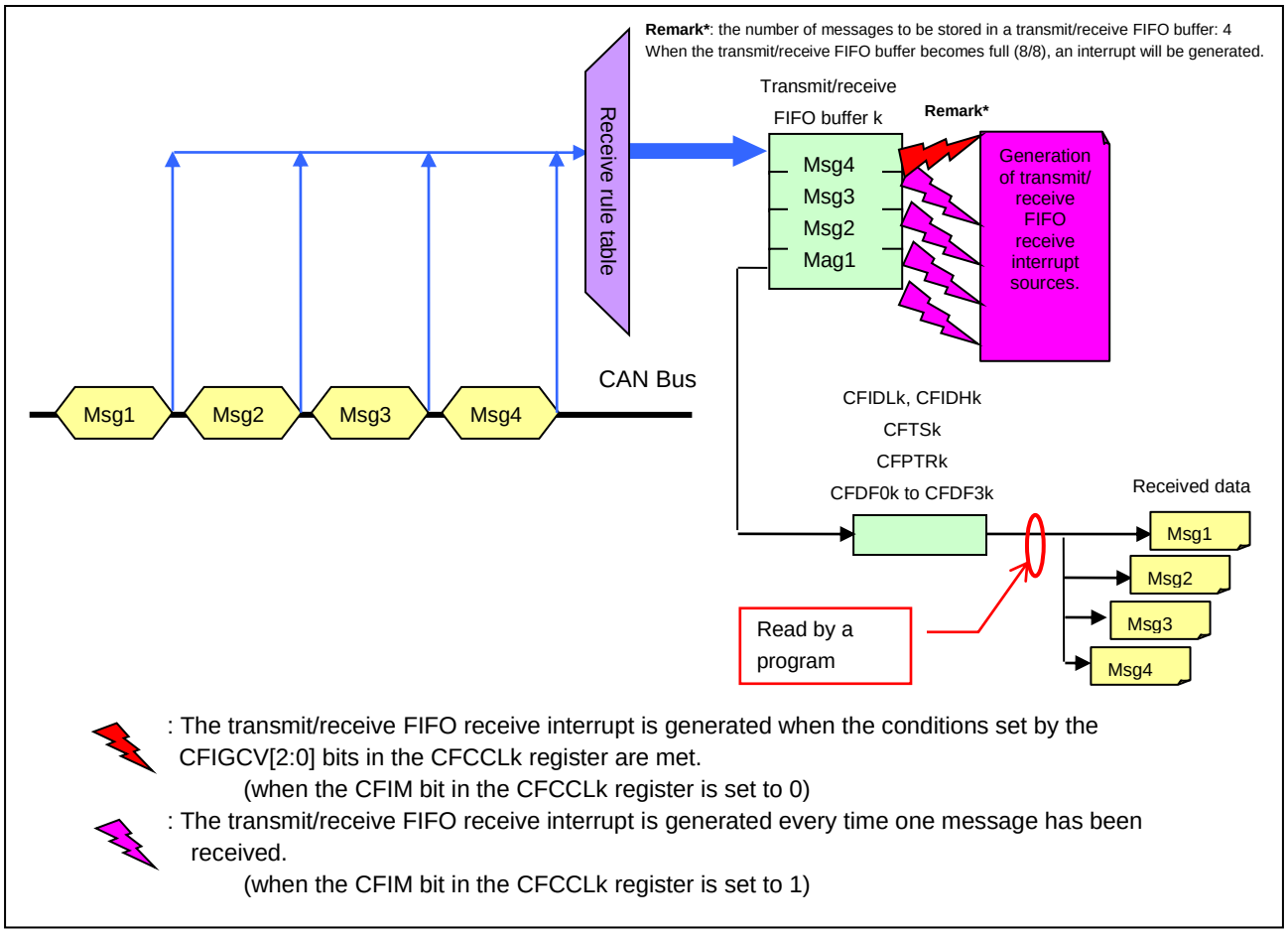
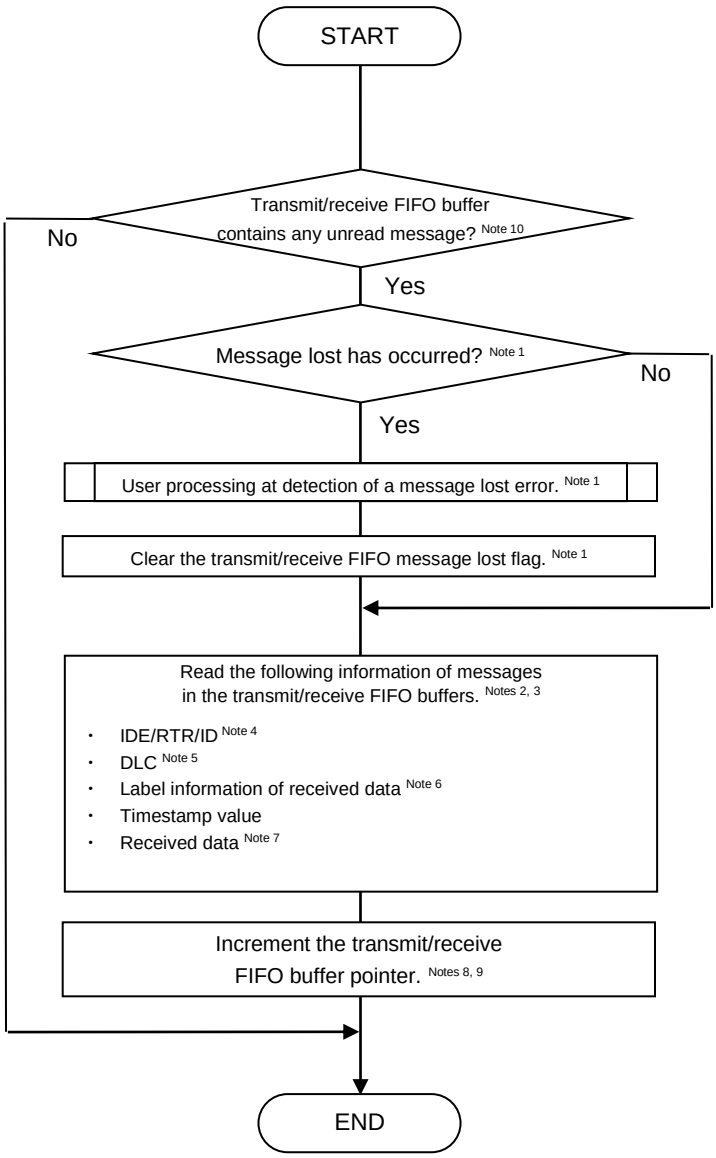


Figure 2.7 Operation of transmit/receive FIFO buffer (in receive mode)

2.4.1 Procedures for reading transmit/receive FIFO buffers

Figure 2.8 shows the procedures for reading the transmit/receive FIFO buffers. **Figure 2.9** and **Figure 2.10** respectively show the procedures for enabling and disabling the transmit/receive 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 transmit/receive FIFO buffers (the CFIDLk and CFIDHk registers, the CFTSk register, the CFPTRk register, and the CFDF0k to CFDF3k registers) can be read only when receive mode is selected (the value of the CFM[1:0] bits in the CFCCHk register is B'00) and also global RAM window 1 for the CAN module is selected (the RPAGE bit in the GRWCR register is set to 1).

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

Figure 2.8 Procedures for reading transmit/receive FIFO buffer

(in receive mode) (no interrupt used) (1/2)

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

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 GAFLPHj 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[11:0]bits in the GAFLPHj register) which has agreed with the value of the received message will be stored.

Note 7: When a DLC value of a received message is smaller than 8 (the value of the CFDLC[3:0] bits in the CFPTRk register is smaller than B'1000), data bytes where data has not been set (the CFDB0[7:0] bits in the CFDF0k register to the CFDB7[7:0] bits in the CFDF3k registers) are read as H'00.

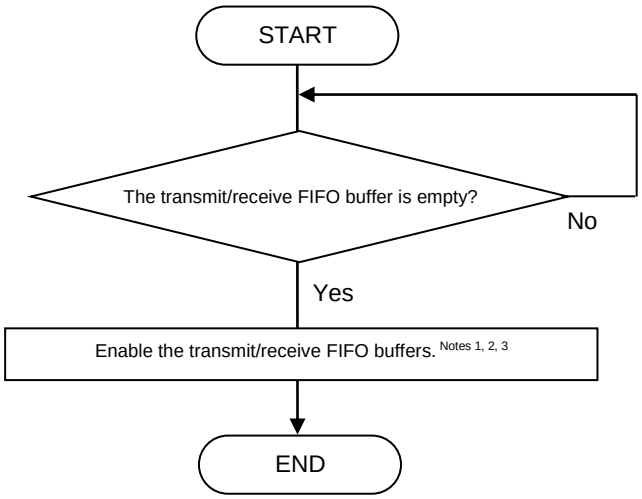
Note 8: After reading out the messages of the transmit/receive FIFO buffer (the CFIDLk and CFIDHk registers, the CFTSk register, the CFPTRk register, the CFDF0k to CFDF3k registers), increment the pointer (write H'FF to the CFPC[7:0] bits in the CFPCTRk register).

Note 9: When incrementing the pointer, the transmit/receive FIFO buffer must be used (the CFE bit in the CFCCLk register is set to 1) and also the transmit/receive FIFO buffer needs to contain an unread message (the CFEMP flag in the CFSTSk register is set to 0).

Note 10: To read all the unread messages stored in the transmit/receive 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 transmit/receive FIFO buffer

(in receive mode) (no interrupt used) (2/2)

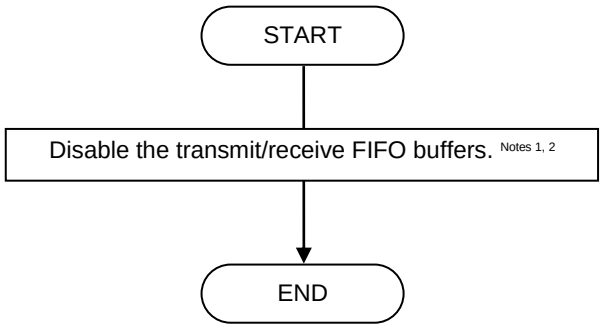


Note 1: In receive mode, enable/disable the transmit/receive FIFO buffer (the CFE bit in the CFCCLk register) when the CAN module is in global operating mode or global test mode.

Note 2: Before using the transmit/receive FIFO buffer (set the CFE bit to 1), perform the CAN configuration.

Note 3: When the number of transmit/receive FIFO buffers is set to 0 (when the value of the CFDC[2:0] bits in the CFCCLk register is B'000), do not enable the transmit/receive FIFO buffers.

Figure 2.9 Procedures for enabling transmit/receive FIFO buffers



Note 1: In receive mode, enable/disable the transmit/receive FIFO buffer (the CFE bit in the CFCCLk register) when the CAN module is in global operating mode or global test mode.

Note 2: Even if the transmit/receive FIFO buffer is disabled (set the CFE bit to 0) while an interrupt request is present (the CFRXIF flag in the CFSTSk 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 transmit/receive FIFO buffers

2.4.2 Interrupt handling for transmit/receive FIFO buffers (in receive mode)

(1) Transmit/receive FIFO receive interrupt handling

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

Even if the transmit/receive FIFO buffers are disabled (set the CFE bit to 0) while an interest request is present (the CFRXIF flag in the CFSTSk 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 transmit/receive FIFO receive interrupt can be enabled/disabled for each transmit/receive FIFO buffer with the CFRXIE bit in the CFCCLk register.

The following are the generation sources for the transmit/receive 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 CFCCLk register, the transmit/receive FIFO receive interrupt request is generated (the CFIM bit in the CFCCLk register is set 0).
  Values set to the CFIGCV[2:0] bits:
  - B’000: the transmit/receive FIFO buffer is 1/8 full Note
  - B’001: the transmit/receive FIFO buffer is 2/8 full
  - B’010: the transmit/receive FIFO buffer is 3/8 full Note
  - B’011: the transmit/receive FIFO buffer is 4/8 full
  - B’100: the transmit/receive FIFO buffer is 5/8 full Note
  - B’101: the transmit/receive FIFO buffer is 6/8 full
  - B’110: the transmit/receive FIFO buffer is 7/8 full Note
  - B’111: the transmit/receive FIFO buffer is full
– Every time one message is received, the transmit/receive FIFO receive interrupt request is generated (the CFIM bit in the CFCCLk register is set to 1).

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

To enable the generation of the transmit/receive 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 transmit/receive 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 4.3 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 transmit/receive 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 transmit/receive FIFO buffers
Transmit history 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 transmit/receive FIFO buffer (set to transmit mode).

When the transmit buffers are linked to the transmit/receive FIFO buffer (set to transmit mode), write H'00 to the corresponding TMCp register and set the TMIEp bit in the TMIEC register to 0 (interrupt disabled). In this case, the corresponding flags of the corresponding TMSTSp, TMTRSTS, 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 TMCp register to 1), message transmission is enabled.

The transmission result can be confirmed by the TMTRF[1:0] flag in the corresponding TMSTSp 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 TMIEp bit in the TMIEC register.

Figure 3.1 illustrates the operation of transmit buffers.

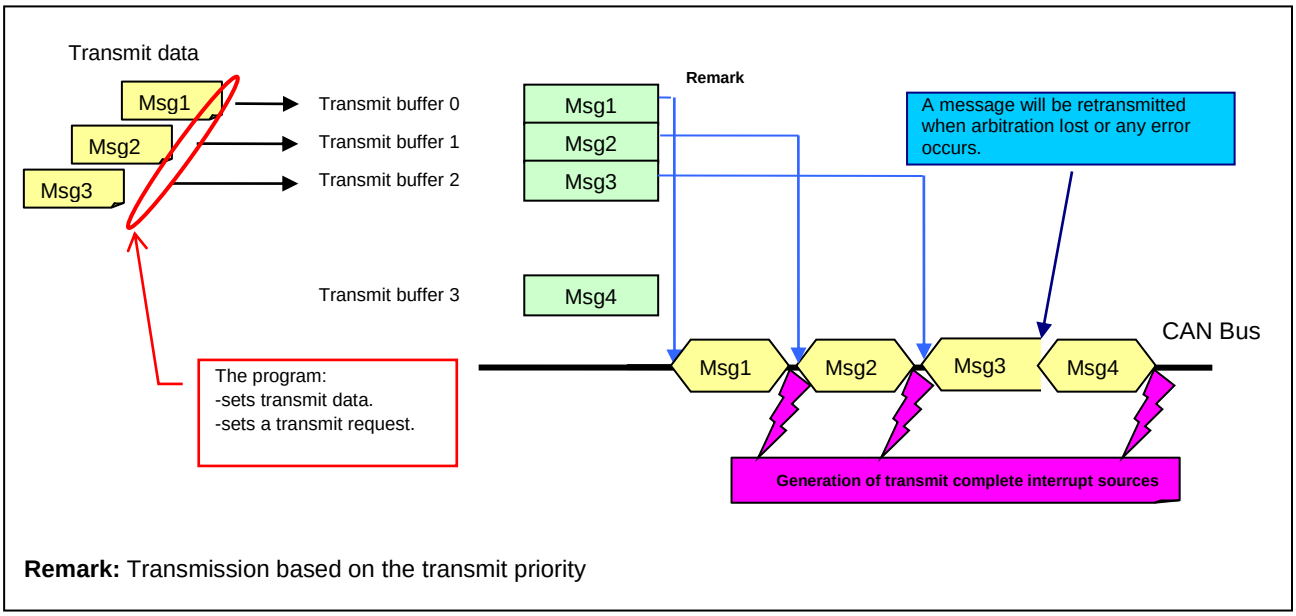
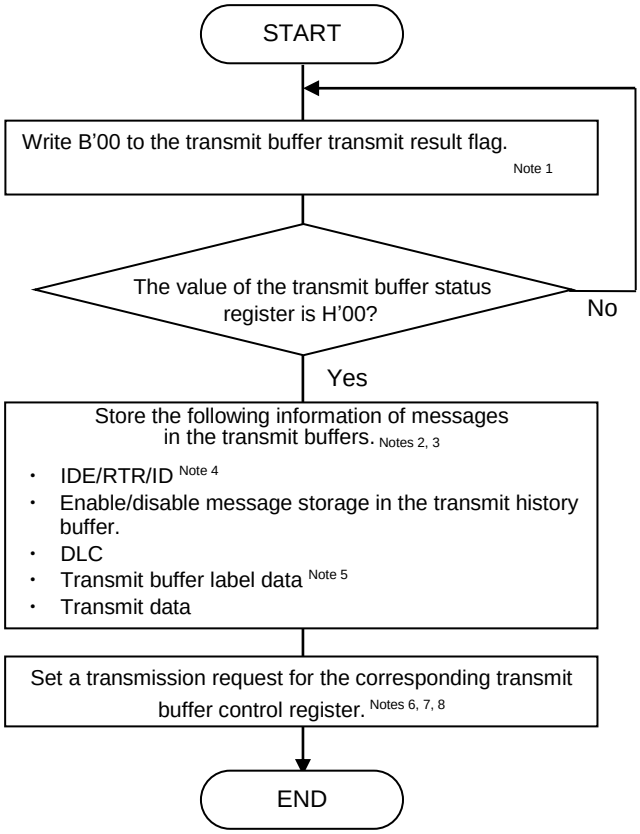


Figure 3.1 Operation of transmit buffers (transmission from channel 0)

(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 TMSTSp register when the CAN module is in channel communication mode or channel halt mode. Do not write any values other than B'00.

Note 2: Modify the values of the TMIDLp and TMIDHp registers, the TMPTRp register, the TMDF0p to TMDF3p registers when a transmission request is not issued for the corresponding transmit buffers (the TMTRM bit in the TMSTSp register is set to 0) and also global RAM window 1 for the CAN moduel is selected (the RPAGE bit in the GRWCR register is set to 1).

Note 3: Do not write data to the TMIDLp and TMIDHp registers, the TMPTRp register, and the TMDF0p to TMDF3p registers, each of which is linked to the transmit/receive FIFO buffer.

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

Note 5: This can be stored only when the messages are to be stored in the transmit history buffers (the THLDTE bit and the THLE bit in the THLCCi register are both set to 1 and the THLEN bit in the TMIDHp register is set to 1).

Note 6: Modify the values of the TMCp register when the CAN module is in channel communication mode or channel halt mode.

Note 7: Write H'00 to the TMCp registers in the following case:

When a transmit buffer is linked to the transmit/receive FIFO buffer

Note 8: Set a transmission request (set the TMTR bit in the TMCp register to 1) while a transmission request is not present (the value of the TMTRF[1:0] flag in the TMSTSp 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.

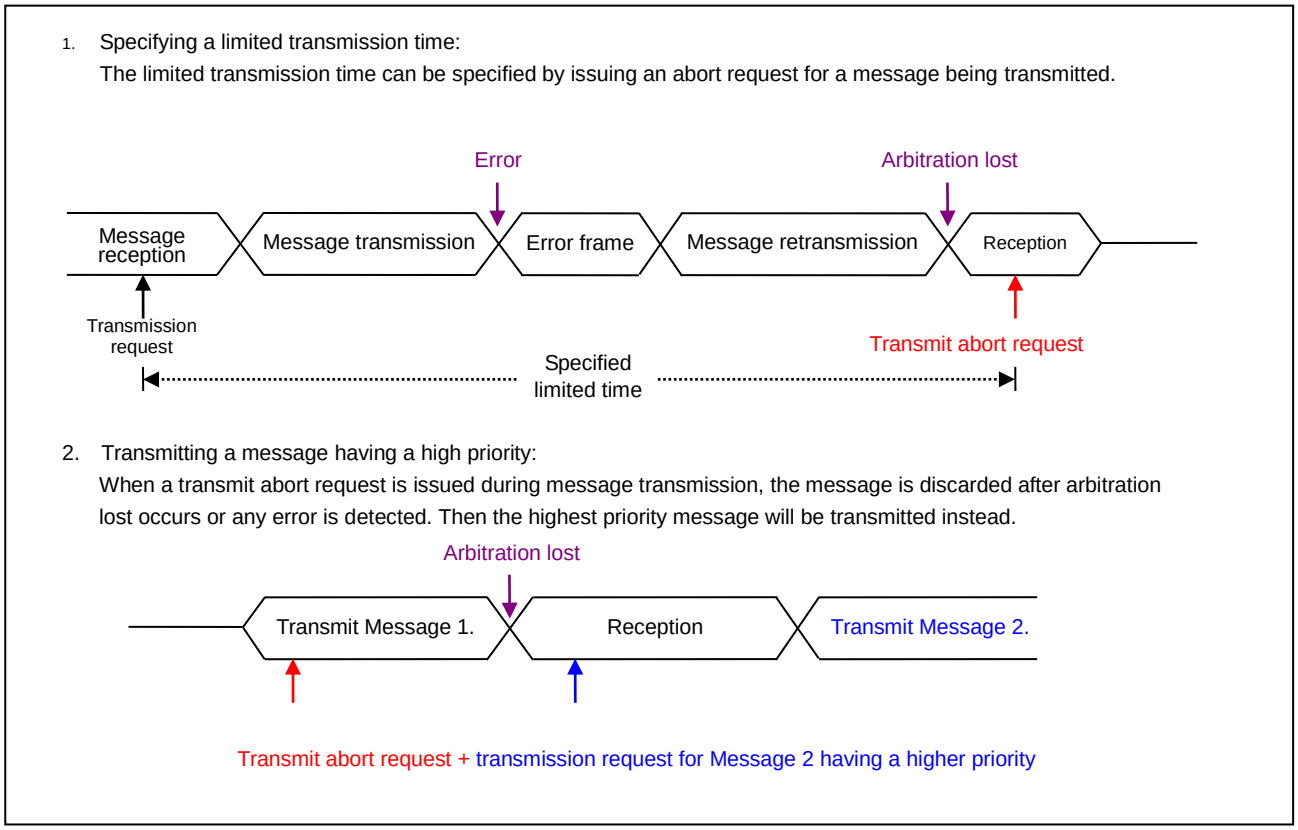


Figure 3.3 Application example of transmit abort function

When a transmit abort request is issued for the transmit buffer (the TMTAR bit in the TMCp register is se to 1) having a transmission request (the TMTRM bit in the TMSTSp 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 TMSTSp 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.

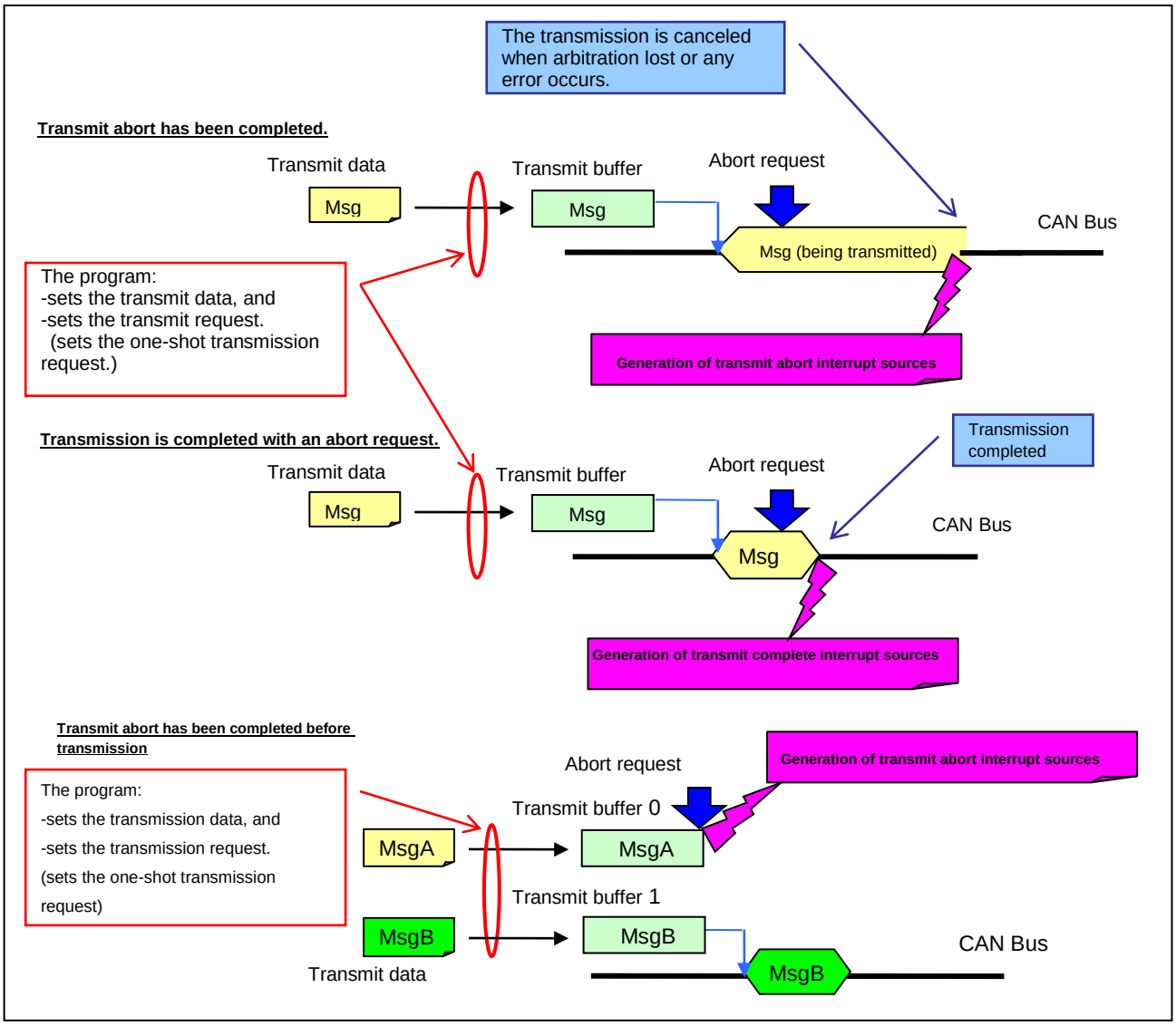
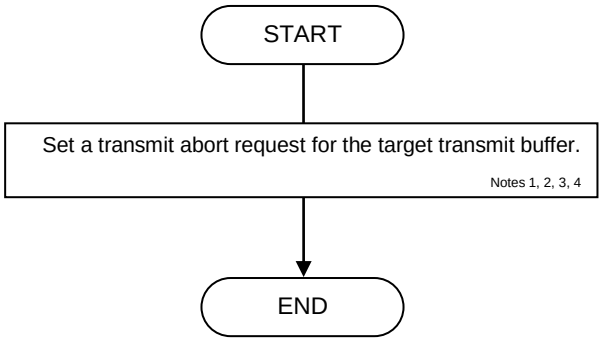


Figure 3.4 Operation when transmission is aborted

(1) Transmit abort procedure

Figure 3.5 shows the procedures for aborting message transmission.



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

Note 2: Write H'00 to the TMCp register in the following case:

When a transmit buffer is linked to transmit/receive FIFO buffer

Note 3: A transmit abort request can be issued (the TMTAR bit of the TMCp register is set to 1) while a transmit request for the transmit buffer is present (the TMTR bit in the TMCp 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 TMSTSp 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(3) Processing after transmission/transmission abort are completed.**

Figure 3.5 Transmission abort procedure

3.2.3 One-shot transmission function

When one-shot transmission is enabled (the TMOM bit in the TMCp 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 TMSTSp 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.

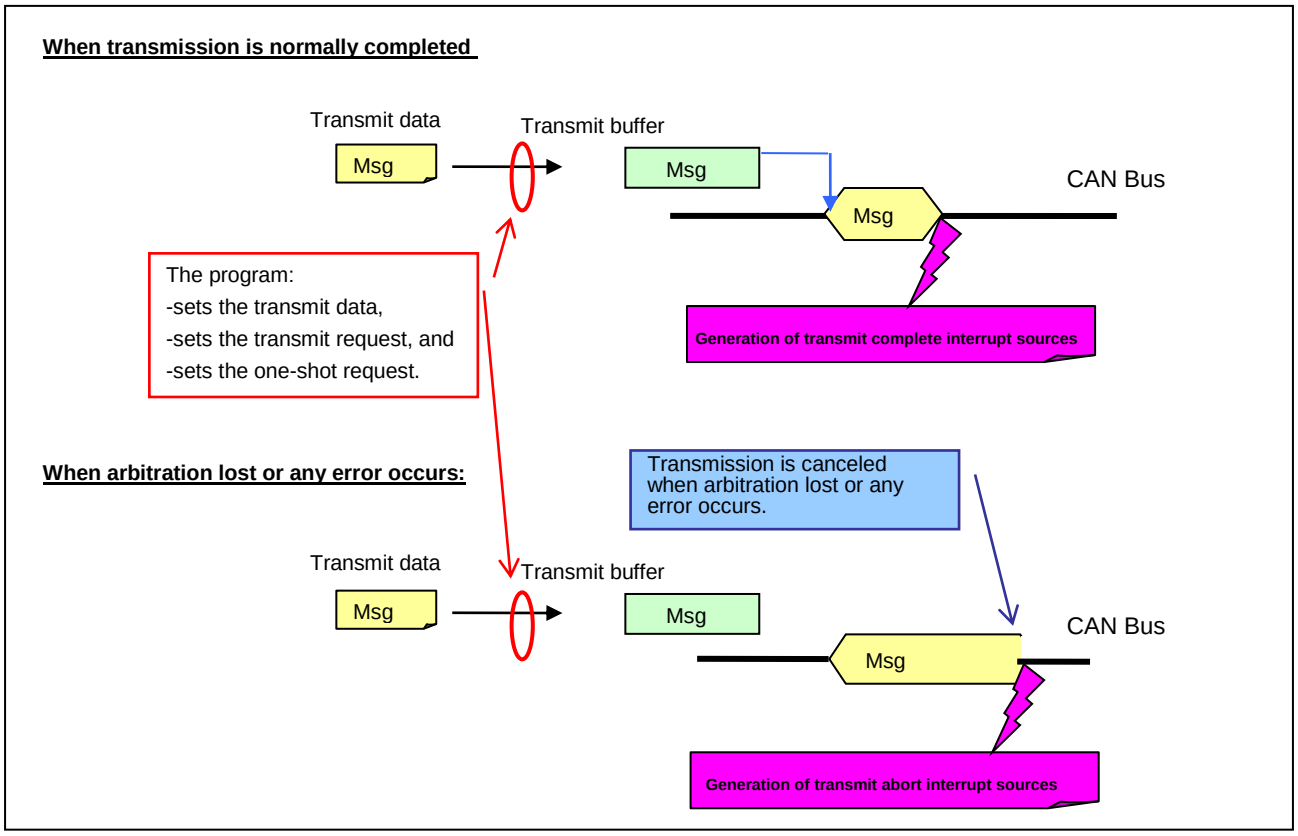
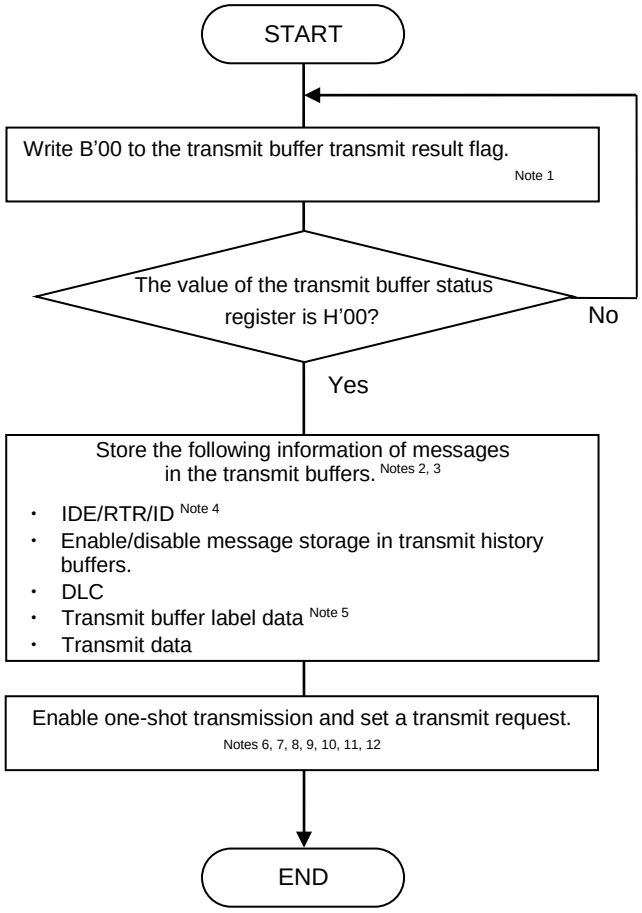


Figure 3.6 Operation of one-shot transmission

(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 TMSTSp register when the CAN module is in channel communication mode or channel halt mode. Do not write any values other than B'00.

Note 2: Modify the values of the TMIDLp and TMIDHp registers, the TMPTRp register, and the TMDF0p to TMDF3p registers when no transmission request is issued for the corresponding transmit buffer(s) (the TMTRM bit in the TMSTSp register is set to 0) and also global RAM window 1 for the CAN moduel is selected (the RPAGE bit in the GRWCR register is set to 1).

Note 3: Do not write data to the TMIDLp and TMIDHp registers, the TMPTRp register and TMDF0p to TMDF3p registers, each of which is linked to the tranmit/receive FIFO buffer.

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

Note 5: This can be stored only when transmit history data is stored in the transmit history buffer (the THLDTE and THLE bits in the THLCCi register are both set to 1 and the THLEN bit in the TMIDHp register is set to 1).

Note 6: Modify the value of the TMCp register when the CAN module is in channel communication mode or channel halt mode.

Note 7: Write H'00 to the TMCp register in the following case:

When a transmit buffer is linked to a transmit/receive FIFO buffer

Note 8: A transmit request can be set (set the TMTR bit in the TMCp register to 1) when no transmit request is present (the value of the TMTRF[1:0] flag of the TMSTSp register is B'00).

Note 9: 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 10: Enable the one-shot transmission (set the TMOM bit in the TMCp register to 1) when no transmit request is present for the transmit buffer (set the TMTRM bit in the TMSTSp register to 0).
Note 11: To enable the one-shot transmission, simultaneously set the transmit request (set the TMTR and TMOM bits to 1).
Note 12: 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 TMSTSp 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 CANi transmit interrupt will be generated when the transmission is completed. The transmit complete interrupt is enabled/disabled with the TMIEp bit in the TMIEC register for each transmit buffer.

The following are the sources for the CANi 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 CANi transmit interrupt can also be confirmed by the GTINTSTS register.

CANi transmit complete interrupt
CANi transmit abort interrupt
CANi transmit/receive FIFO transmit complete interrupt
CANi transmit history interrupt

To generate the CANi 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 CANi 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 4.3 CAN-related interrupt processing**".

(2) Transmit abort completion interrupt handling

Once the transmit abort interrupt is enabled, the CANi 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 CiCTRH 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 CANi 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 CANi transmit interrupt can also be confirmed by the GTINTSTS register.

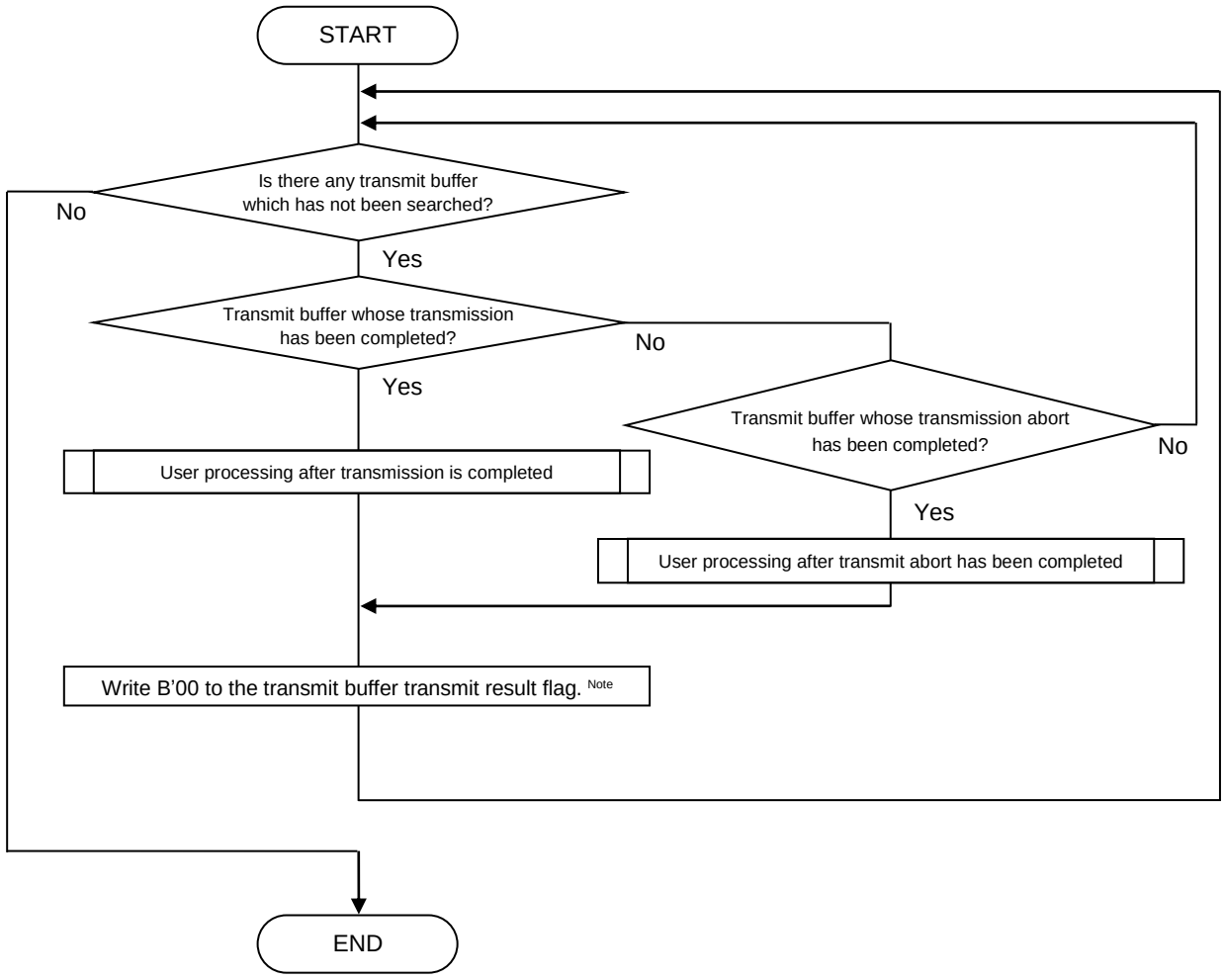
CANi transmit complete interrupt
CANi transmit abort interrupt
CANi transmit/receive FIFO transmit complete interrupt
CANi transmit history interrupt

To generate the CANi 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 CANi 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 4.3 CAN-related interrupt processing**".

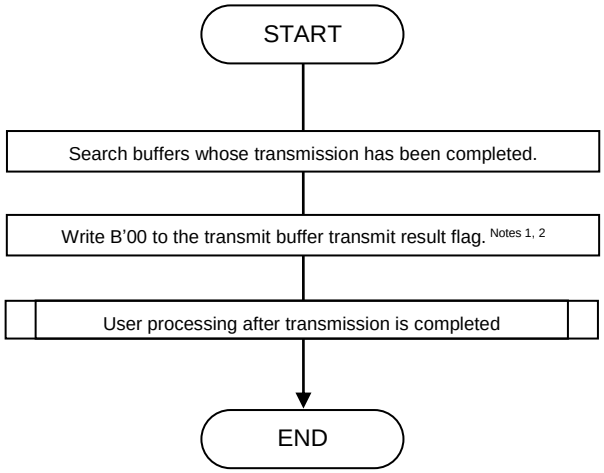
(3) Processing after transmission/transmission abort are completed

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



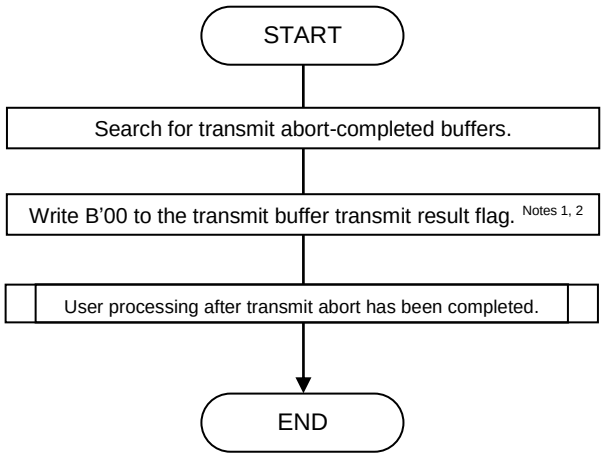
Note: Write B'00 to the TMTRF[1:0] flag in the TMSTSp register when the CAN module is in channel communication mode or channel halt mode. Do not write any values other than B'00.

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 TMSTSp register when the CAN module is in channel communication 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 TMSTSp register when the CAN module is in channel communication 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 transmit/receive FIFO buffer

Data frames or remote frames will be transmitted using a transmit/receive FIFO buffer.

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

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

The transmit/receive FIFO buffer is linked to transmit buffers (set by the CFTML[1:0] bits in the CFCCHk register). When the transmit/receive FIFO buffer is used (the CFE bit in the CFCCLk register is set to 1), messages stored in the transmit/receive 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 transmit/receive FIFO buffer has the following transmission functions. Regarding the configuration using the transmit/receive 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 transmit/receive FIFO buffer are transmitted sequentially on a first-in, first-out basis.

Figure 3.11 illustrates the transmission operation of the transmit/receive FIFO buffer.

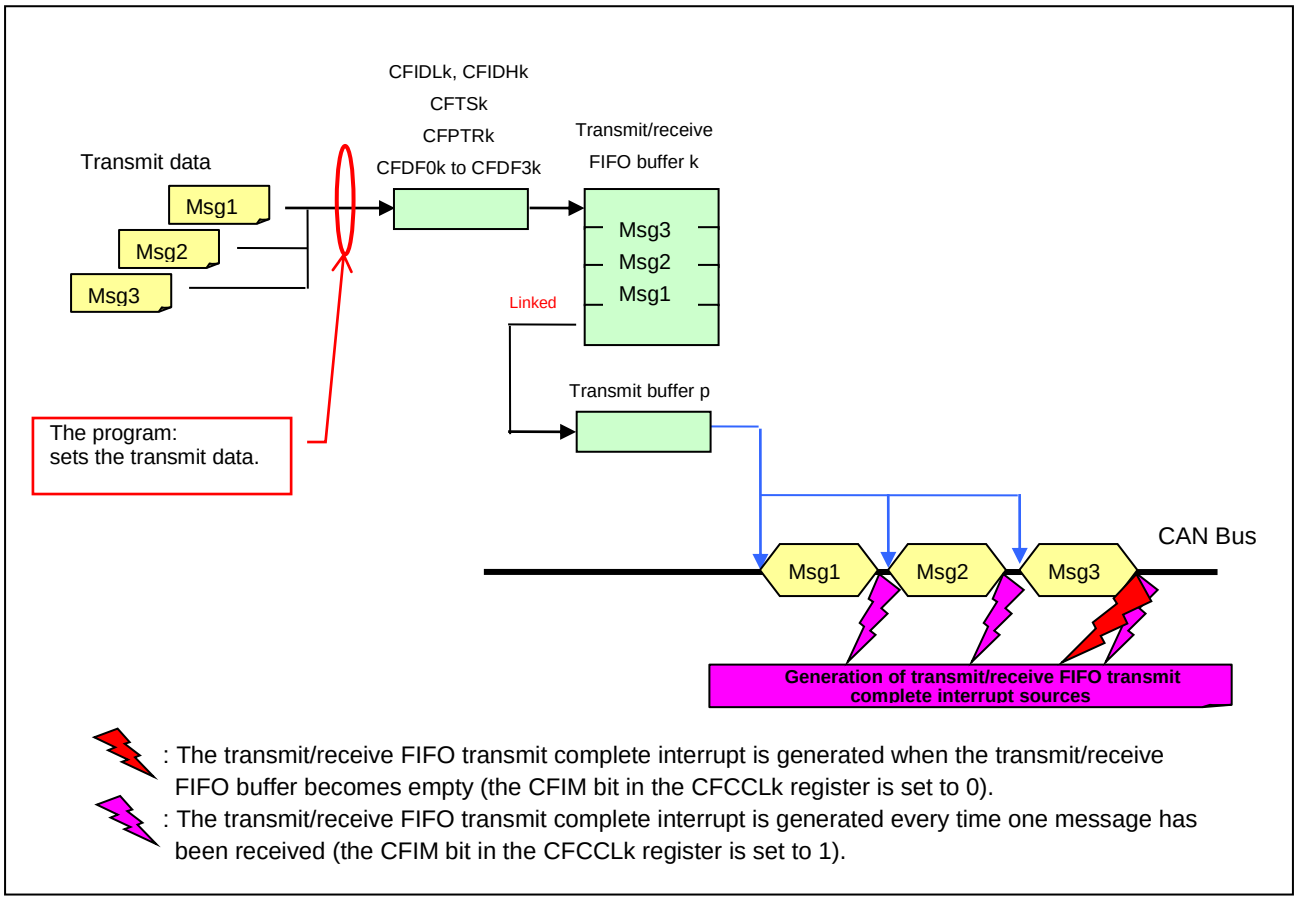
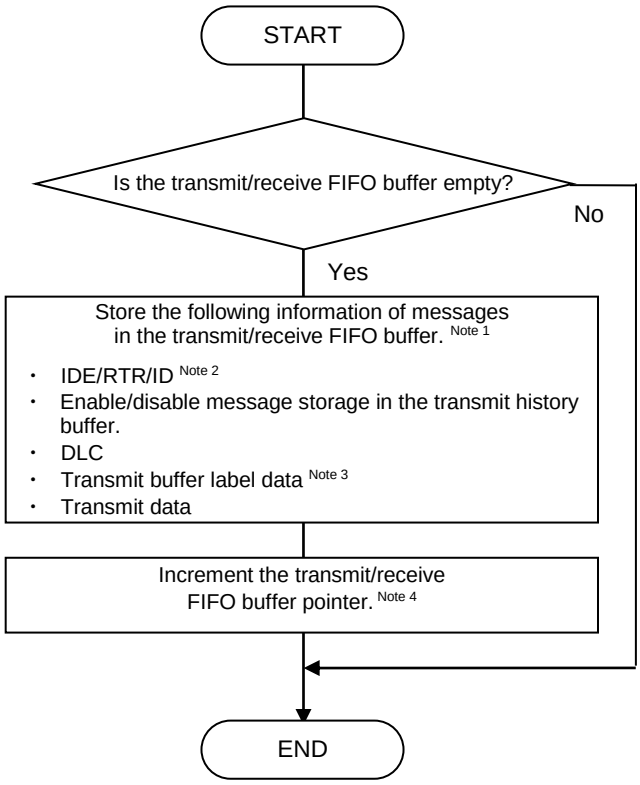


Figure 3.11 Operation of transmit/receive FIFO buffer (in transmit mode)

(1) Procedures for transmitting messages from transmit/receive FIFO buffers

Figure 3.12 shows the procedures for transmitting messages from the transmit/receive FIFO buffer. **Figure 3.13** and **Figure 3.14** show respectively the procedures for enabling and disabling the transmit/receive FIFO buffer.



Note 1: The values of the CFIDLk and CFIDHk registers, the CFPTRk register, the CFDF0k to CFDF3k regsiters can be modified only when the transmit/receive FIFO buffer is in transmit mode (the value of the CFM[1:0] bits in the CFCCHk register is B'01) and also global RAM window 1 for the CAN module is selected (the RPAGE bit in the GRWCR register is set to 1).

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

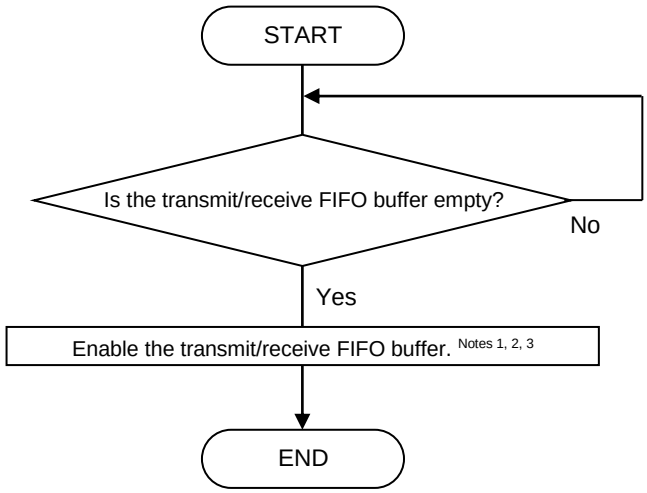
Note 3: This can be stored only when the transmit/receive FIFO buffer is in transmit mode and messages are to be stored in the transmit history buffer (the THLE bit in the THLCCi is set to 1 and the THLEN bit in the CFIDHk register is set to 1).

Note 4: Increment the transmit/receive FIFO buffer pointer when the conditions below are satisfied.

(Write H'FF to the CFPC bit in the CFPCTRk register).

- the transmit/receive FIFO buffer is in transmit mode
- the transmit/receive FIFO buffer is used (the CFE bit in the CFCCLk register is set to 1),
- after a message is stored in the transmit/receive FIFO buffer
- the transmit/receive FIFO buffer is not full (the CFFLL flag in the CFSTSk register is set to 0)

Figure 3.12 Procedures for transmitting messages from transmit/receive FIFO buffer



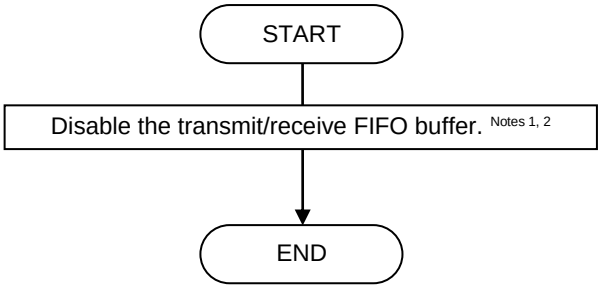
Note 1: In transmit mode, if the transmit/receive FIFO buffer is disabled while a message is being transmitted or when a message to be transmitted next has been determined, the transmit/receive FIFO buffer will be empty after:

- the transmission is completed,
- a CAN bus error is detected, or
- arbitration lost occurs.

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

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

Figure 3.13 Procedures for enabling transmit/receive FIFO buffer



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

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

Figure 3.14 Procedures for disabling transmit/receive FIFO buffer

3.3.2 Transmit abort function

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

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

Regarding the transmit abort procedures of the transmit/receive FIFO buffer, refer to **Figure 3.14**.

3.3.3 Interval transmission function

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

When the transmit/receive FIFO buffer is enabled (the CFE bit in the CFCCLk 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 transmit/receive 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 transmit/receive 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 CFCCHk register	Count sources	Formulas Note
B’00	the clock obtained by frequency-dividing the CPU/peripheral hardware clock/2 by the value of the ITRCP[15:0] bit in the GCFGH register	$1/f_{CLK} \times 2 \times a \times b$
B’10	the clock obtained by frequency-dividing CPU/peripheral hardware clock/2 by the value of the ITRCP[15:0] bits in the GCFGH register and multiplying the divided value by 10	$1/f_{CLK} \times 2 \times a \times 10 \times b$
B’x1	CANi bit time clock	$1/f_{CANBIT} \times b$

Note:

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 CFCCHk register)
f_{CLK} : CPU/peripheral hardware clock frequency
f_{CANBIT} : CANi bit time clock frequency

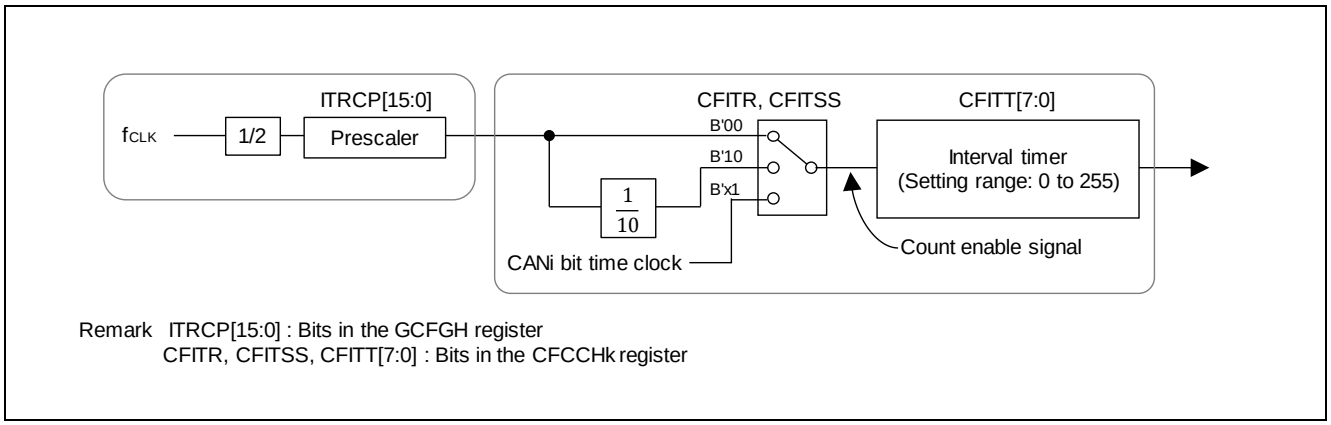
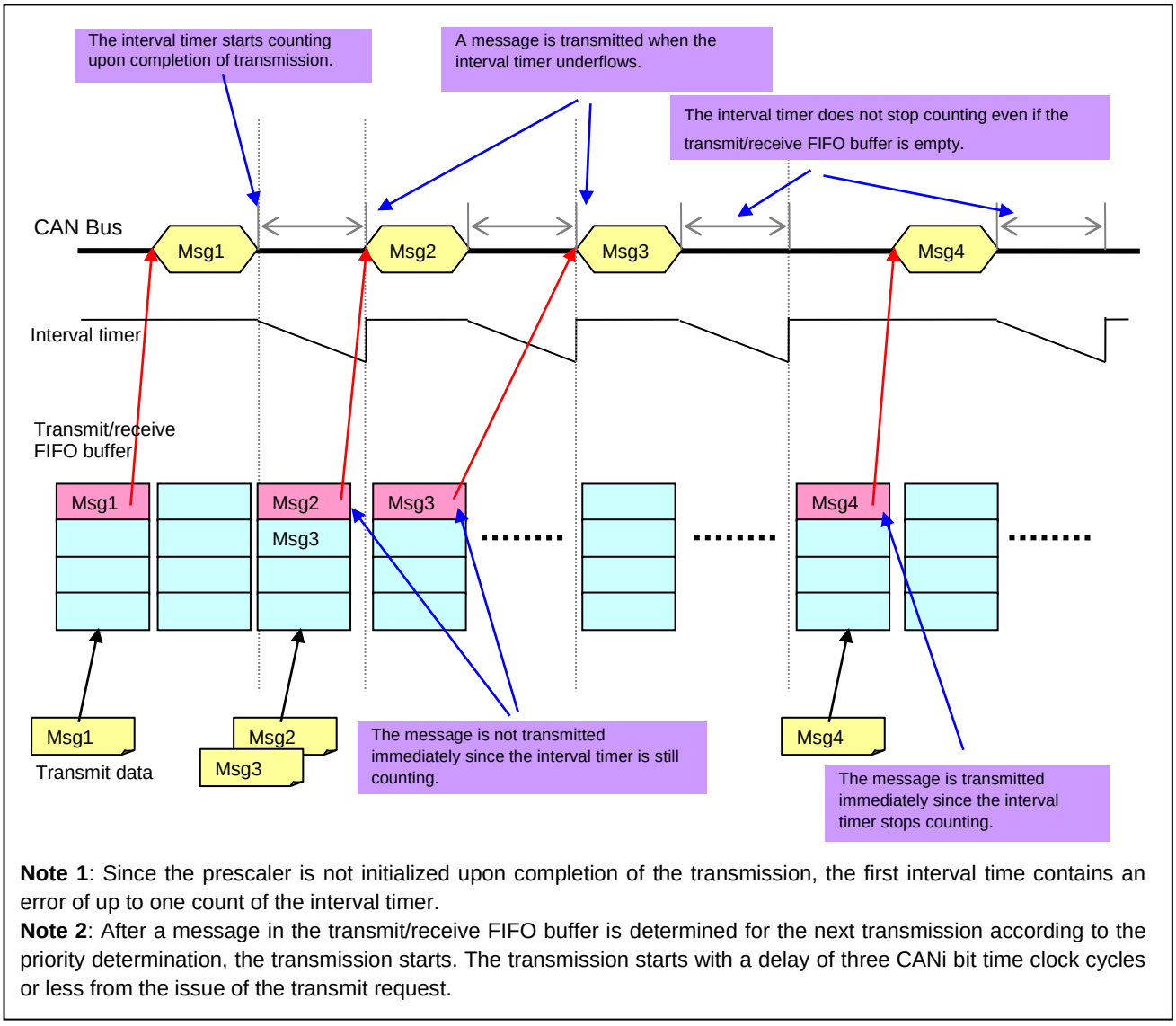


Figure 3.15 Block diagram of interval timer



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 CANi bit time clock cycles or less from the issue of the transmit request.

Figure 3.16 Interval transmission (in transmit mode)

3.3.4 Interrupt handling of transmit/receive FIFO buffer (in transmit mode)

(1) Transmit/receive FIFO transmit interrupt handling

Once the transmit/receive FIFO transmit complete interrupt is enabled, the CANi transmit interrupt will be generated when the conditions set by the CFIM bit in the CFCCLk register are satisfied.

The following are the sources for the CANi 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 CANi transmit interrupt can also be confirmed by the GTINTSTS register.

CANi transmit complete interrupt
CANi transmit abort interrupt
CANi transmit/receive FIFO transmit complete interrupt
CANi transmit history interrupt

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

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

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

When the buffer becomes empty upon completion of message transmission, a transmit/receive FIFO transmit complete interrupt request will be generated.
Every time one message transmission is completed, a transmit/receive 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 CANi 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 4.3 CAN-related interrupt processing**".

3.4 Transmit history buffer function

Data of the message that has been transmitted (transmission history data) can be stored in the transmit history buffer. One channel has one transmit history 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 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 buffer as transit history data.

Buffer type

A type of buffer (transmit buffer or transmit/receive FIFO buffer) that has transmitted the stored messages

Buffer number

The number (No.) of the transmit buffer or transmit/receive FIFO buffer that has transmitted the message.
(Refer to **Table 3.2**)

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 type (the value of the BT flag in the THLACCi register) / **the value of the BN flag in the THLACCi register**	**B'01**	**B'10**
	Transmit buffer No.	**Transmit/receive FIFO buffer**
B'00	Transmit buffer 0	Numbers (No.) of the transmit buffers linked to transmit/receive FIFO buffer with the CFTML[1:0] bits in the CFCCHk 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 buffer.

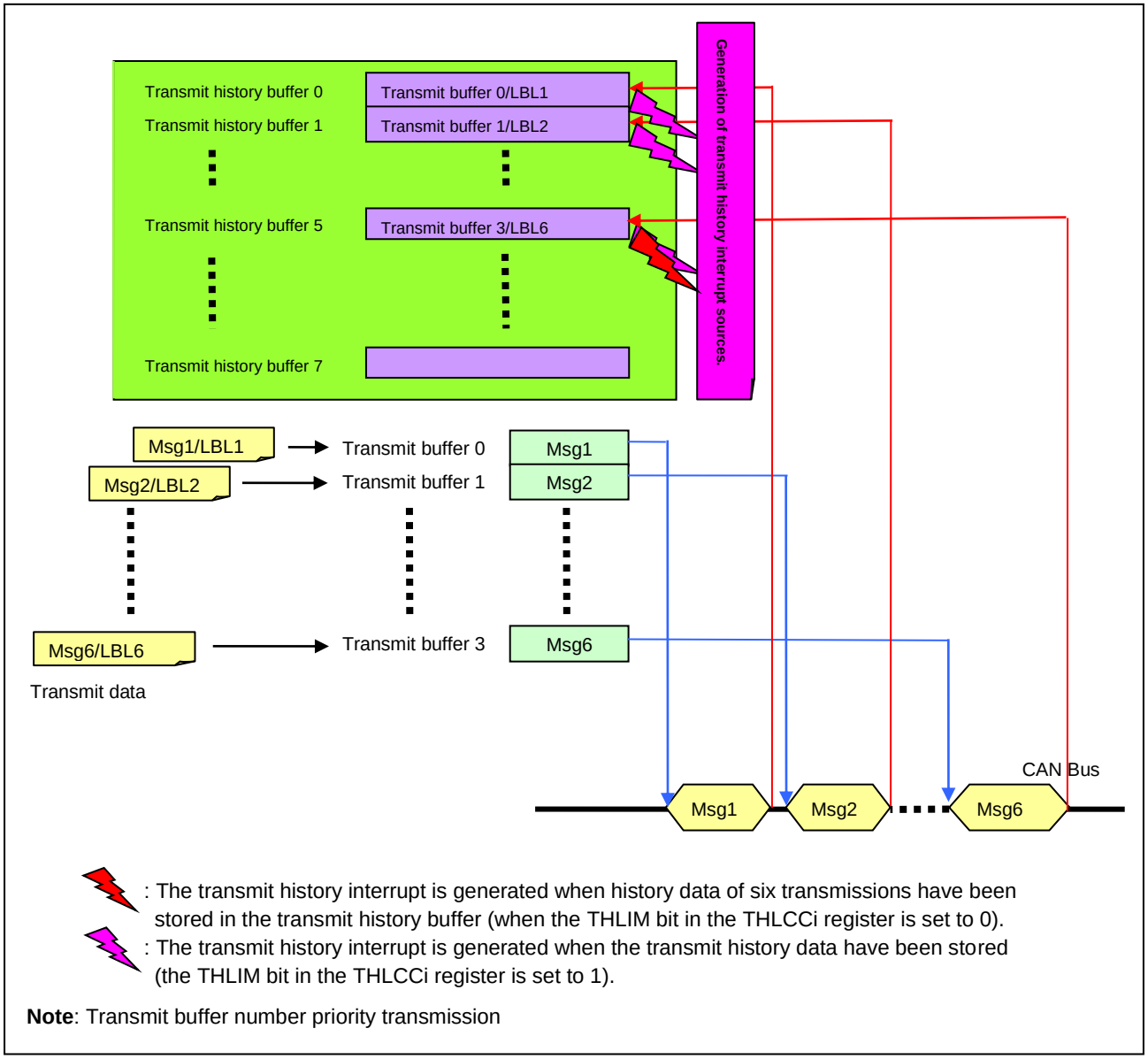
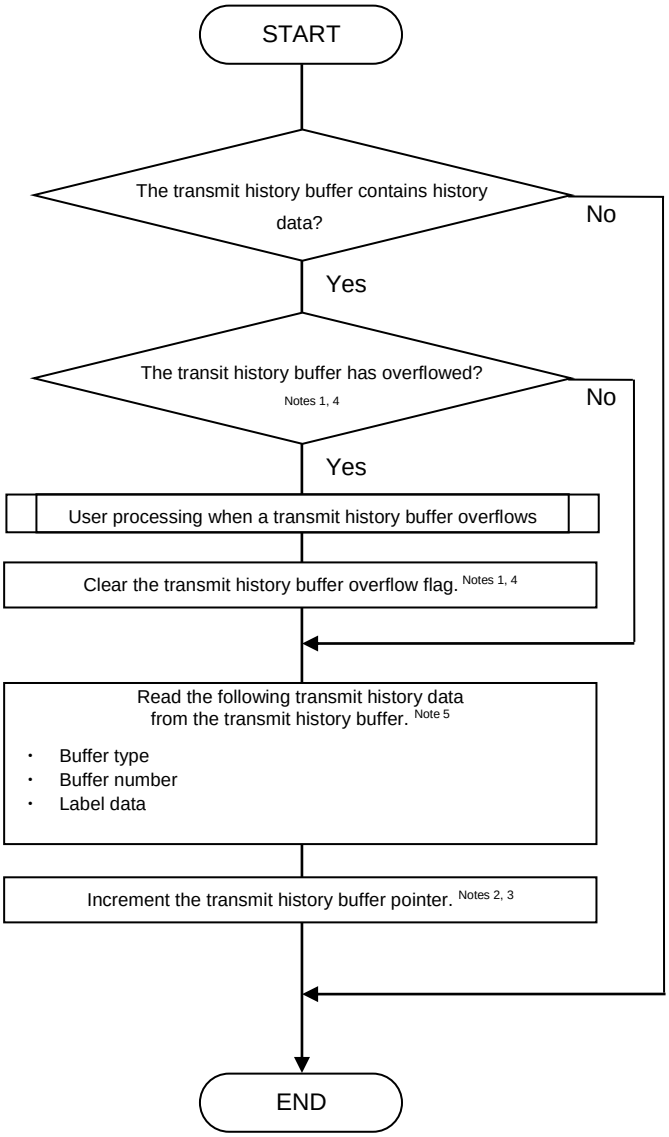


Figure 3.17 Operations of transmit history buffer

(1) Procedures for reading transmit history buffers

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



Note 1: Write 0 to the THLELT bit in the THLSTSi register by a program.

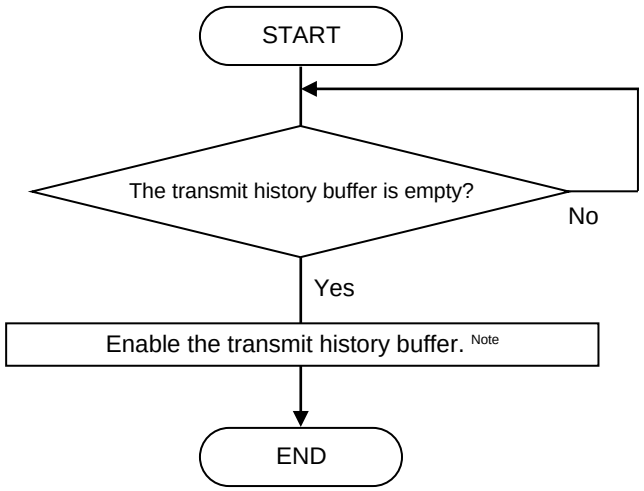
Note 2: After the transmit history buffer (THLACCi register) is read, increment the transmit history buffer pointer (write H'FF to the THLPC[7:0] bits in the THLPCTRi register).

Note 3: When incrementing the transmit history buffer pointer, the transmit history buffer must be used (the THLE bit in the THLCCi register is set to 1) and also the transmit history buffer should contain any message (the THLEMP bit in the THLSTSi register is set to 0).

Note 4: When the transmit history buffer overflow interrupt is enabled, perform these procedures while the interrupt is being handled.

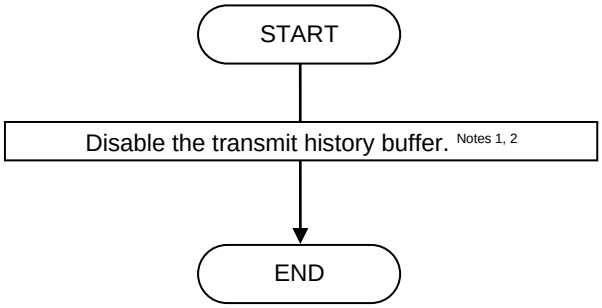
Note 5: After successful completion of message transmission, the processing may be delayed by up to 38 clocks of f_{CLK} before the transmit history data has been stored.

Figure 3.18 Procedure for reading transmit history buffer



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

Figure 3.19 Procedure for enabling transmit history buffer



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

Note 2: Even if the transmit history buffer is disabled (set the THLE bit to 0) while an interrupt request is present (the THLIF flag in the THLSTSi 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 buffer

3.4.2 Handling of transmit history buffer interrupt

(1) Transmit history interrupt handling

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

The following are the generation sources for the CANi 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 CANi transmit interrupt can also be confirmed by the GTINTSTS register.

CANi transmit complete interrupt
CANi transmit abort interrupt
CANi transmit/receive FIFO transmit complete interrupt
CANi transmit history interrupt

Even if the transmit history buffer is disabled (set the THLE bit to 0) while an interrupt request is present (the THLIF flag in the THLSTSi register is set to 1), 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 THLCCi register for each transmit history 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 buffer
An interrupt request which is generated every time history data of one transmission is stored.

To generate the CANi 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 CANi 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 4.3 CAN-related interrupt processing**".

(2) Global error interrupt handling

Once the transmit history buffer overflow interrupt is enabled, a global error interrupt will be generated when an overflow of the transmit history buffer is detected. The transmit history buffer overflow 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 and IF2H)

Interrupt mask flag registers (MK2L and MK2H)

Priority specification flag registers (PR02L, PR02H, PR12L and PR12H)

The following CAN-related interrupts can be used:

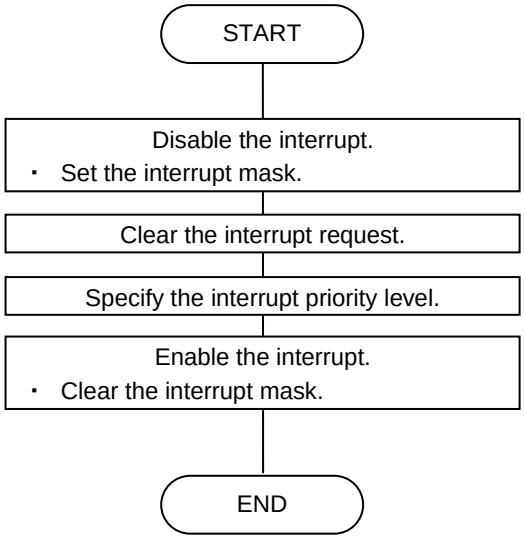
CAN global receive FIFO interrupt
CAN global error interrupt
CANi channel transmit interrupt
CANi transmit/receive FIFO receive interrtup
CANi channel error interrupt
CANi wakeup interrupt

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
Global error interrupt	DLC check error
	FIFO message lost
	transmit history buffer overflow
CANi transmit interrupt	CANi transmit complete interrupt request
	CANi transmit abort interrupt request
	CANi transmit/receive FIFO transmit complete interrupt request
	CANi transmit history interrupt request
CANi transmit/receive FIFO receive interrupt	When CANi transmit/receive FIFO receive interrupt request is issued
CANi error interrupt	bus error
	error warning
	error passive state
	bus off entry
	bus off recovery
	overload frame transmission
	bus lock
	arbitration lost
CANi wakeup interrupt	When a falling edge of a signal from the CAN bus is detected

4.1.1 Procedures for setting CAN-related interrupts

Figure 4.1 shows the procedures for setting interrupts.



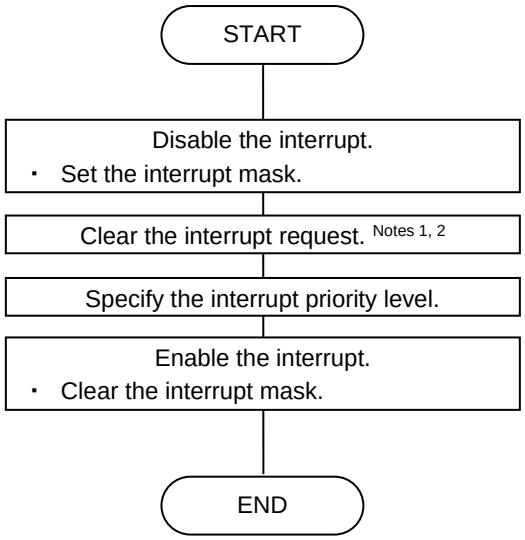
Note : When the CANi wakeup interrupt is used, bit 0 in the peripheral enable register 2 (the CAN0EN bit in the PER2 register) needs to be set to 1.

Figure 4.1 Interrupt setting procedures

4.1.2 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**.

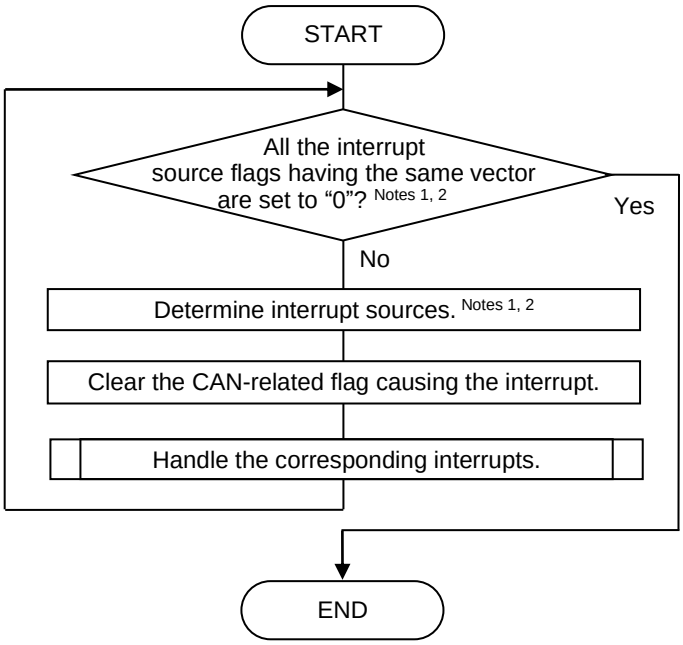
Figure 4.2 shows how to clear the interrupt source flags during interrupt handling.



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).

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

Figure 4.2 CAN-related 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 CANi wakeup interrupt request flag does not exist in the CAN module, because the interrupt function controls CANi wakeup interrupt.

Figure 4.3 CAN-related interrupt processing

5. Cautions regarding processing flow

5.1.1 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.

5.1.2 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.

5.1.3 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 5.2** shows the processing including the specified loop time. **Table 5.1** and **Table 5.2** show the maximum transition time of each mode.

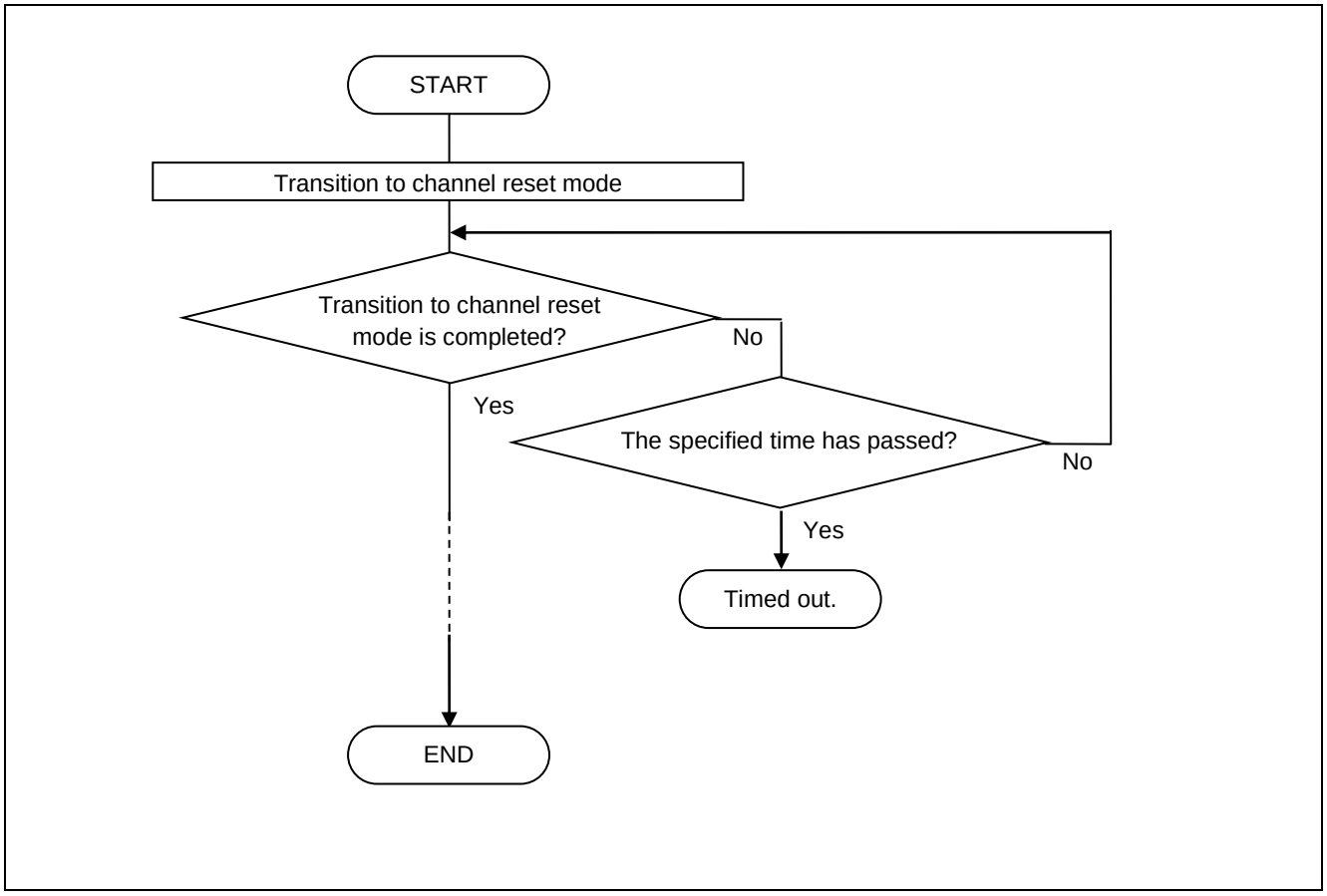


Figure 5.1 Processing having specified loop time

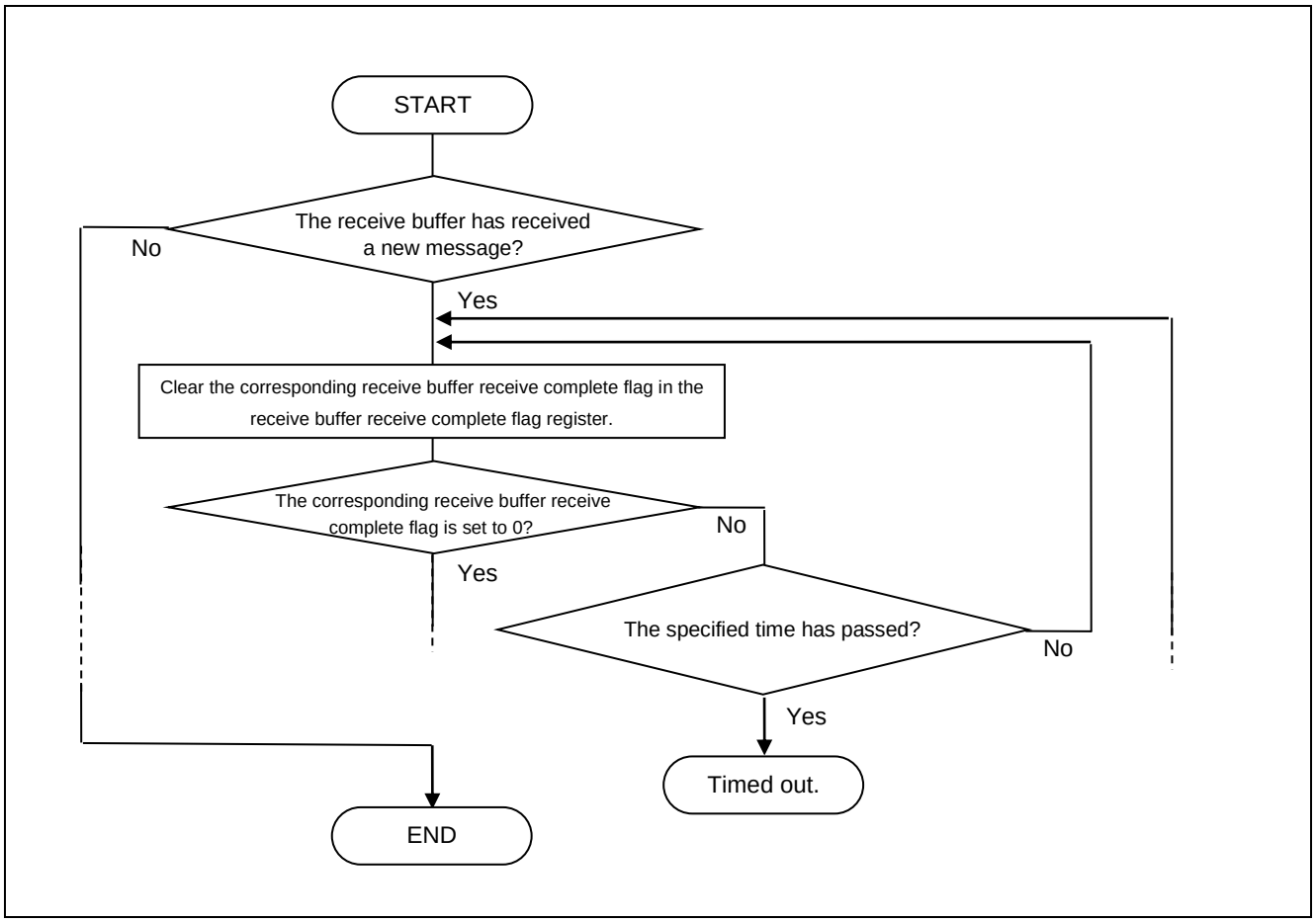


Figure 5.2 Operation with specified loop time

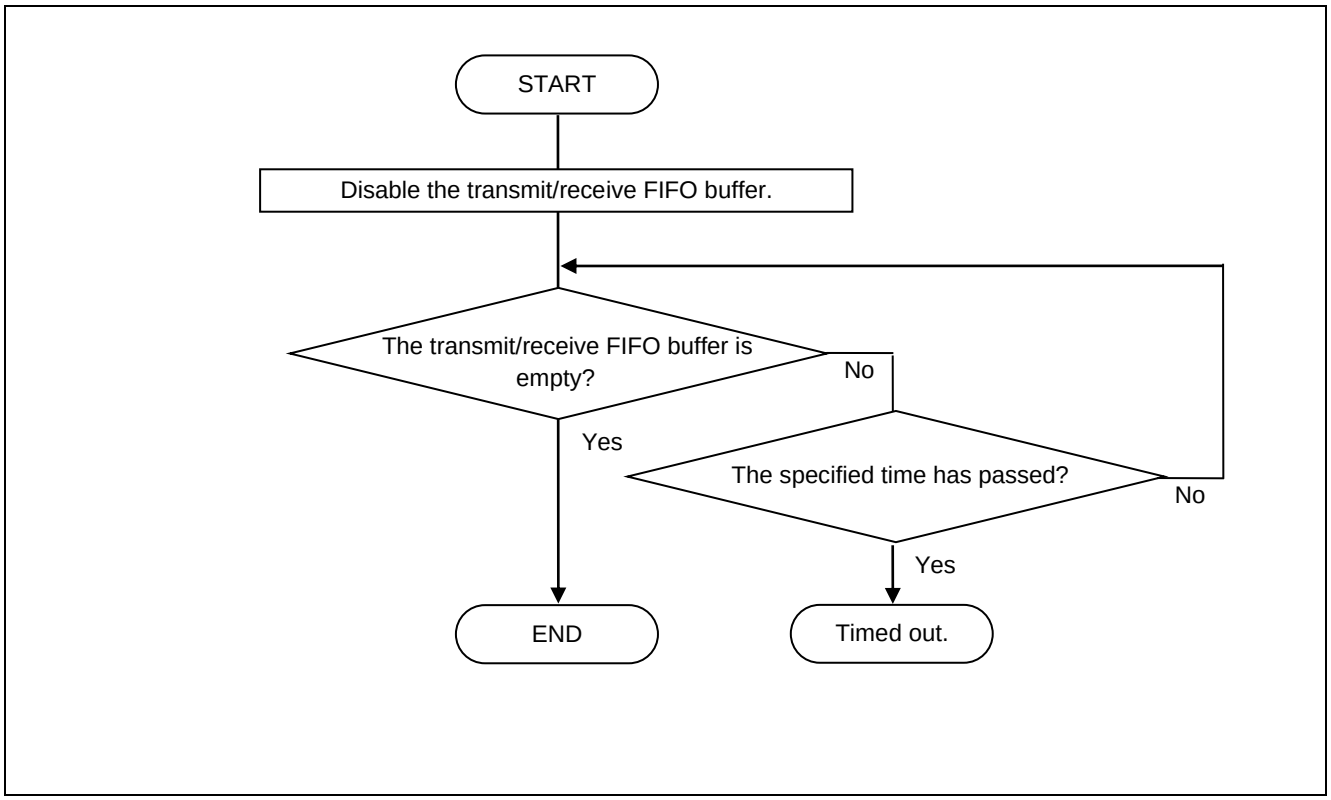


Figure 5.3 Operation with limited loop time

Table 5.1 Maximum transition time of global modes

Mode before transition	Mode after transition	Maximum transition time
Global stop	Global reset	3 f_{CLK} cycles
Global reset	Global stop	3 f_{CLK} cycles
Global reset	Global test	3 f_{CLK} cycles
Global reset	Global operating	3 f_{CLK} cycles
Global test	Global reset	3 f_{CLK} cycles
Global test	Global operating	3 f_{CLK} cycles
Global operating	Global reset	3 f_{CLK} cycles
Global operating	Global test	2 CAN frames

Table 5.2 Maximum transition time of channel modes

Mode before transition	Mode after transition	Maximum transition time
Channel stop	Channel reset	3 f_{CLK} cycles
Channel reset	Channel stop	3 f_{CLK} cycles
Channel reset	Channel halt	3 CAN bit times
Channel reset	Channel communication	2 CAN bit times
Channel halt	Channel reset	3 f_{CLK} cycles
Channel halt	Channel communication	3 CAN bit times
Channel communication	Channel reset	3 f_{CLK} cycles
Channel communication	Channel halt	2 CAN frames

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	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				
	Timestamp clock				
	Interval timer prescaler				
Communication speed setting	Bit timing	✓	△	△	△
	Communication speed				
Receive rule table setting		✓	△	-	-
Buffer setting	Receive buffer	✓	△	-	-
	Receive FIFO buffer				
	Transmit/receive FIFO buffer			△ Note 2	△ Note 2
	Transmit buffer			△	△
	Transmit history 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:
CFTML[1:0] bits, CFM[1:0] bits, CFIGCV[2:0] bits, CFIM bit, and CFE bit in the CFCCLk and CFCCHk registers

6.2 CAN-related interrupt sources

Table 6.2 lists the CAN-related interrupt sources.

Table 6.2 CAN-related interrupt sources

Interrupt	Generation source	Interrupt enable bit Note 1	Conditions	How to clear Note 1
CAN global receive FIFO interrupt	Recive FIFOm interrupt request	RFIE bit in the RFCCm register	When the conditions set by the RFIGCV[2:0] bits in the RFCCm register are satisfied. Note 2	Set the RFIF flag in the RFSTSm register to 0.
			Every time one message is received.	
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 transmit/receive FIFO buffer is detected.	(all channels) Set the CFMLT flag in the CFSTSk register to 0. Set the RFMLT flag in the RFSTSm register to 0
			When a message lost error of the receive FIFO buffer is detected.	
	Transmit history buffer overflow	THLEIE bit in the GCTRL register	When the transmit history buffer attempts to store further transmit history data although the buffer is already full.	(all channels) Set the THLELT flag in the THLSTSi register to 0.
CANi channel transmit interrupt	CANi transmit complete interrupt request	TMIEp bit in the TMIEC register	When the buffer becomes empty upon completion of message transmission	Set the TMTRF[1:0] flag in the TMSTSp register to B'00.
	CANi transmit abort interrupt request	TAIE bit in the CiCTRH register	Every time transmission of one message is completed.	
	CANi transmit/receive FIFO transmit complete interrupt request	CFTXIE bit in the CFCCLk register	When the buffer becomes empty upon completion of message transmission	Set the CFTXIF flag in the CFSTSk register to 0.
			Every time transmission of one message is completed.	
	CANi transmit history interrupt request	THLIE bit in the THLCCi register	When history data of six transmissions have been stored in the transmit history buffer.	Set the THLIF flag in the THLSTSi register to 0.
			Every time history data of one transmission are stored.	
CANi transmit/ receive FIFO receive interrupt	CANi transmit/receive FIFO receiv interrupt request	CFRXIE in the CFCCLk register	When the conditions set by the CFIGCV[2:0] bits in the CFCCLk register are satisfied. Note 3	Set the CFRXIF flagin the CFSTSk register to 0.
			Every time reception of one message is completed.	
CANi channel error itnerrupt	Bus error	BEIE bit in the CiCTRL register	When any one of the ADERR, B0ERR, B1ERR, CERR, AERR, FERR, and SERR flags of the TRFRIT register is set to 1. Note 4	Set the BEF flag in the CiERFLL register to 0.
	Error warning	EWIE bit in the CiCTRL register	When the value of the REC[7:0] bits or TEC[7:0] bits in the CiSTSH register exceeds 95.	Set the EWF flag in the CiERFLL register to 0.
	Error passive	EPIE bit in the CiCTRL 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 CiERFLL register to 0.
	Bus off entry	BOEIE bit in the CiCTRL register	When the CAN module has enterred the bus off state (TEC[7:0] bits > 255)	Set the BOEF flag in the CiERFLL register to 0.
	Bus off recovery	BORIE bit in the CiCTRL 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 CiERFLL register to 0.
	Overload frame transmit	OLIE bit in the CiCTRL	When the overload frame transmit condition has been detected when performing reception or transmission.	Set the OVLF flag in the CiERFLL register to 0.
	Bus lock	BLIE bit in the CiCTRL register	When 32 consectutive dominant bits have been detected on the CAN bus in channel communication mode.	Set the BLF flag in the CiERFLL register to 0.
	Arbitration lost	ALIE bit in the CiCTRL register	When arbitration lost is detected	Set the ALF flag in the CiERFLL register to 0.
CANi wakeup interrupt	Detection of a CAN bus falling edge	--	When a falling edge is detected in the CRXDi pin.	--

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 each User's Manual for Hardware.

Note 2: Values set to the RFIGCV[2:0] bits in the RFCCm 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 RFCCm register is B'001), do not perform this settings.

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

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

Remark *: When the number of messages to be stored in the transmit/receive FIFO buffer is set to 4 (the value of the CFDC[2:0] bit of the CFCCLk 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 CiERFLL register is set to 1 and also a form error has been detected in the ACK delimiter.
- The B0ERR flag in the CiERFLL register is set to 1 and also a recessive bit has been detected though a dominant bit was transmitted.
- The B1DRR flag in the CiERFLL register is set to 1 and also a dominate bit has been detected though a recessive bit was transmitted.
- The CERR flag in the CiERFLL register is set to 1 and also a CRC error has been detected.
- The AERR flag in the CiERFLL register is set to 1 and also an ACK error has been detected.
- The FERR flag in the CiERFLL register is set to 1 and also a form error has been detected.
- The SERR flag in the CiERFLL 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 CiCTRL register is set to B'01 (channel reset mode)
- When the RTBO bit in the CiCTRL register is set to 1 (forcible return from the bus-off state is made)
- When the BOM[1:0] bit in the CiCTRH 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.

6.3 Operations when a receive buffer has received a message and operations when the receive (transmit/receive) 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 transmit/receive 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 (transmit/receive) 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)
Transmit/receive FIFO buffer (in receive mode)	discarded	Global error interrupt (message lost error in the transmit/receive FIFO buffer)

Note:

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

TMCp register			Event	Transmission result (TMTRF[1:0] flag in the TMSTSp 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		

Website and Support

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

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

All trademarks and registered trademarks are the property of their respective owners.

Revision History

		Description	
Rev.	Date	Page	Summary
1.00	2014.11.27	-	1st Edition
2.00	2018.06.15	1	Added RL78/F15 to the target product. Deleted a row of "Each status register number (xx)" in table.
		6	Corrected GSLPR bit name in Note 1 of Figure 1.3.
		8	Corrected CSLPR bit name in Note 1 of Figuer 1.5.
		15	Corrected BOEF and CFRXIF bits name in Table 1.2.
		15	Corrected RMNDi register name, RFFLL and CFRXIF bits name in Table 1.3.
		23	Added description of Note in chapter 1.4.5. Added Note 1 and Note 2 in Figure 1.12.
		25	Corrected GAFLPHj register name in Note 2 of Figure 1.14.
		29, 30	Corrected GAFLPHj and GAFLPLj registers name in Example 1 and 2 of chapter 1.5.8.
		42	Corrected CiCTRH register name in chapter 1.8.1 “(4)Bus off entry”.
		52	Corrected RMPTRn and RMDLC[3:0] register/bits name, and CFDB0[7:0] to CDFB7[7:0] bits in Note 8 of Figure 2.2.
		57	Added explanation when using interrupt in 2.3.2 (1) Receive FIFO interrupt processing. Corrected bit name of RFCCm register.
		60	Corrected CFDB0[7:0] to CDFB7[7:0] bits in Note 7 of Figure 2.8.
		62	Added description of transmit/receive FIFO receive interrupt in chapter 2.4.2 “(1) Transmit/receive FIFO receive interrupt handling”.
		62	Corrected GCTRL register name in chapter 2.4.2 ”(2) Global error interrupt handling”.
		65, 70	Corrected TMIDHp and TMID[28:16] register/bits name in Figure 3.2 and Fiigure 3.7.
		71, 72	Added description of CANi transmit interrupt in chapter 3.2.4 “(1) Transmit complete interrupt handling”, and “(2) Transmit abort completion interrupt handling”.
		72	Corrected CiCTRH register name in chapter 3.2.4 “(2) Transmit abort completion interrupt handling”.
		80	Added description of CANi transmit interrupt in chapter 3.3.4 “(1) Transmit/receive FIFO transmit interrupt handling”.
		85	Added description of CANi transmit interrupt in chapter 3.4.2 “(1) Transmit history interrupt handling”.
		88, 89	Divided Note in Figure 4.2 and Note 1 in Figure 4.3 into Note 1 and Note 2.
		94	Corrected Table 6.2 register and bit name. ・ From CFSTSkL to CFSTSk register. ・ From MES to MEIE bit. ・ From THOF to THLELT bit. ・ From CiCTR to CiCTRH register. ・ From CFSTSLk to CFSTSk register. ・ From TRFRIT bit in the CFCCkL register to CFIGCV[2:0] bits in the CFCCLk register. ・ From CiERFLL to CiSTSH register.
		95	Corrected bit name of CFCCLk register in Note 3 of Table 6.2.

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. 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 LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

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

   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

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

4. Clock Signals

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

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

5. Differences between Products

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

   — The characteristics of Microprocessing unit or Microcontroller unit products in the same group but having a different part number may differ in terms of the 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 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.
5. 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.
6. 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.
7. 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.
8. 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.
9. 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.
10. 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.
11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

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

(Rev.4.0-1 November 2017)

RENESAS

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

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

Renesas Electronics America Inc.
1001 Murphy Ranch Road, Milpitas, CA 95035, U.S.A.
Tel: +1-408-432-8888, Fax: +1-408-434-5351

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700, Fax: +44-1628-651-804

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709 Quantum Plaza, No.27 ZhichunLu, Haidian District, Beijing, 100191 P. R. China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, 200333 P. R. China
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore 560 038, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
17F, KAMCO Yangjae Tower, 262, Gangnam-daero, Gangnam-gu, Seoul, 06265 Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5338

© 2018 Renesas Electronics Corporation. All rights reserved.

Colophon 7.0