RH850/U2C Group

FlexRay Application Note

Summary

This application note summarizes operation examples using FlexRay. The FlexRay program is assumed to be located in the user area.

This document and program are intended to help users understand the functions implemented in the RH850/U2C and are not intended for use in mass production design..

In addition, it does not reflect the latest manuals, errata, technical updates, or updates to the development environment. When using the relevant functions, treat this program as a reference only and use the latest documentation and development environment at your own risk.

Applicable device

This document applies to RH850/U2Cx.

【Note】When downloading to the Configuration Setting Area, set the desired option bytes in “set_csa.c”, enable downloading, and then rewrite the option bytes.
For details, refer to the RH850/U2C Series Startup Application Note.

(1) Select "*****(Debug Tool)" from the project tree.
(2) Select the “Download file settings” tab.
(3) “Allow downloads to the Configuration Setting Area” = set to “Yes”

Contents

1. Introduction

This application note provides information on how to use the FlexRay module on the RH850/U2Cx and examples of software development.

1.1 Functions Used

The hardware functions of RH850/U2Cx used in this application note are shown below。

- FlexRay

2. Initial Settings

To use the FlexRay module on the RH850/U2Cx, carry out the following initial settings.

- Port Settings
- Interrupt Settings
- Initialization of the FlexRay Module
- FlexRay Clock Setting
- Setting of FlexRay Communication Parameters and Others
- Setting of Message RAM

The following sections describe the initialization procedures for each item.

2.1 Port Settings

In this operation example, FlexRay signals are assigned to Port 24. The FlexRay pin settings are shown in Table 2-1, and the functions assigned to each FlexRay terminal are shown in Table 2-2.

Table 2-1 FlexRay Module Pin Settings

Port Name	Pin Name	Port Control Register (PCRn_m) Settings
P24_12	FLX0TXDA	PCR24_12 = 0x00000046
P24_11	FLX0RXDA	PCR24_11 = 0x00000056
P24_10	FLX0TXENA	PCR24_10 = 0x00000045
P24_7	FLX0TXDB	PCR24_7 = 0x00000045
P24_8	FLX0RXDB	PCR24_8 = 0x00000056
P24_9	FLX0TXENB	PCR24_9 = 0x00000045

Table 2-2 FlexRay Module Pin Description

Pin Name	Functions
FLX0TXDA	Channel A Transmit Data Output Pin
FLX0RXDA	Channel A Transmit Data Input Pin
FLX0TXENA	Channel A Transmit Data Enable Pin “H” : Transmission Disabled “L” : Transmission Enabled
FLX0TXDB	Channel B Transmit Data Output Pin
FLX0RXDB	Channel B Transmit Data Input Pin
FLX0TXENB	Channel B Transmit Data Enable Pin “H” : Transmission Disabled “L” : Transmission Enabled

2.2 Initialization of the FlexRay Module

After power on reset and immediately after executing the CLEAR_RAMS command by the FLXAnFRSUCC1 register, the internal RAM of the FlexRay module is initialized (all bits are set to “0”).

Since register settings of the FlexRay module cannot be performed while the internal RAM is being initialized, confirm that the internal RAM initialization process has been completed.

2.3 FlexRay Clock

The sampling clock supplied to the FlexRay module is the high-speed peripheral clock (CLKC_HSB: 80 MHz).

2.3.1 Setting of FlexRay Sampling Clock and FlexRay Communication Time Unit

This section describes the settings of each clock used in the FlexRay module. These clocks are generated by dividing the high-speed peripheral clock (CLKC_HSB) inside the FlexRay module.

Before performing communication on the FlexRay bus, the clocks used to sample the bus, the Macrotick (MT)—which serves as the common time unit in the FlexRay network—and the Microtick (uT), which is a local time unit within the node, must be set individually.

The relationship between the high-speed peripheral clock (CLKC_HSB) and the FlexRay communication speed is shown in Table 2-3.

Table 2-3 The relationship between the high-speed peripheral clock (CLKC_HSB) and the FlexRay communication speed

High-speed Peripheral Clock（CLKC_HSB）	FLXAnFRPRTC1 Register BRP[1:0] Setting Value	Bit Clock（Bit/s）	Communication Speed
80MHz	00	100ns	10Mbps
80MHz	01	200ns	5Mbps
80MHz	1x	400ns	2.5Mbps

■Setting Example

Table 2-4 shows example setting values for each time unit when the high-speed peripheral clock (CLKC_HSB) is 80 MHz and the communication bit rate is 10 Mbps. Figure 2-1 shows an example of the setting procedure.

Table 2-2 Example Setting Values for Each Time Unit

Name	Setting Values
Sampling Period	12.5ns
"Communication Speed	10Mbps
Microtick	25ns
Macrotick	1us
Communication Cycle	1ms

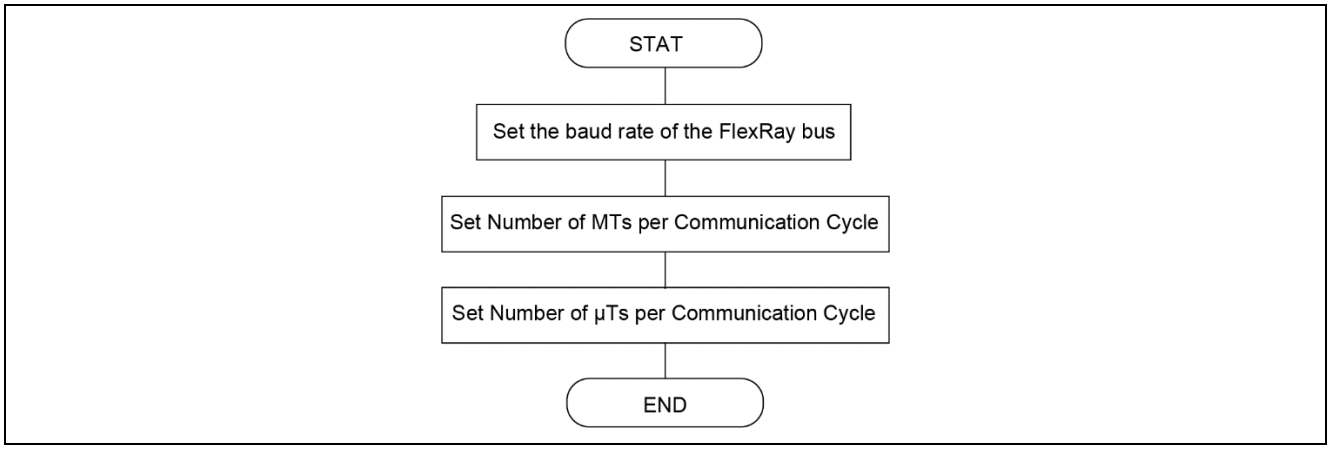


Figure 2-1 Example of the Setting Procedure for Each Time Unit

Explanation of the Configuration Procedure Example

When the high-speed peripheral clock (CLKC_HSB) is 80 MHz and the BRP bits of the FLXAnFRPRTC1 register are set to “00”, the sampling period becomes 80 MHz (12.5 ns) with no division.
The communication bit rate is 1/8 of the sampling period (fixed according to the FlexRay specification), resulting in 80 MHz / 8 ⇒ 10 Mbps.

For the uT, when the BRP bits of the FLXAnFRPRTC1 register are set to “00”, it is divided by 2 from the high-speed peripheral clock (CLKC_HSB) (fixed), giving a 1 uT length of 12.5 ns × 2 = 25 ns.

The communication cycle length is uT length (25 ns) × 400,000 ⇒ 1 ms.

The Macrotick period is 1 communication cycle = 1,000 MT = 40,000 uT, so 1 MT ⇒ 40 uT ⇒ 1 µs.

【Note】 1. There is no register to set the number of uTs per MT (pMicroPerMacroNom).
The communication controller automatically calculates pMicroPerMacroNom from the number of MTs and the number of uTs per communication cycle.

2. Configuration must be performed in the CONFIG state (FLXAnFRCCSV register, POCS bits = “001111”). Writing to the FLXAnFRPRTC1, FLXAnFRGTUC1, and FLXAnFRGTUC2 registers is not allowed in any other state.

2.4 Setting of FlexRay Communication Parameters and Others

2.4.1 Setting of Communication Cycle

The FlexRay module determines the communication cycle configuration by setting the Static segment, Dynamic segment, and the NIT start position. The communication cycle configuration is shown in Figure 2-2.

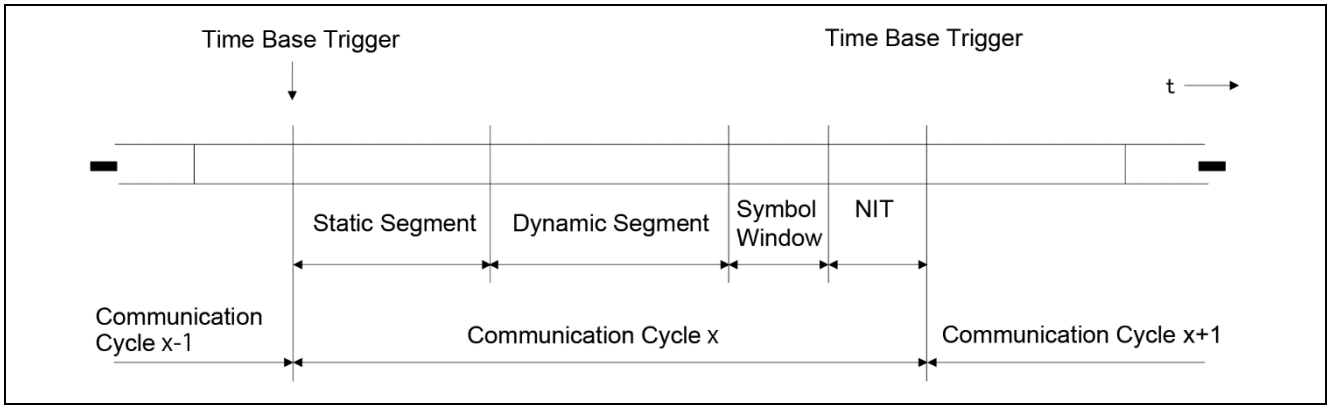


Figure 2-2 Communication Cycle Configuration

An example of the communication cycle setting procedure is shown below. Example setting values for the communication cycle are shown in Table 2-5, and the setting procedure example is shown in Figure 2-3.

Table 2-3 Example Setting Values for the Communication Cycle

Name	Setting Values
1 Communication Cycle Length	1000 MT
Static Slot Length	54MT
Number of Static Slots	6 slot
ActionPoint Position	5MT
Mini Slot Length	24MT
Number of Mini Slots	6slot
Mini-Slot ActionPoint Position	2MT
NIT Start Position	990MT

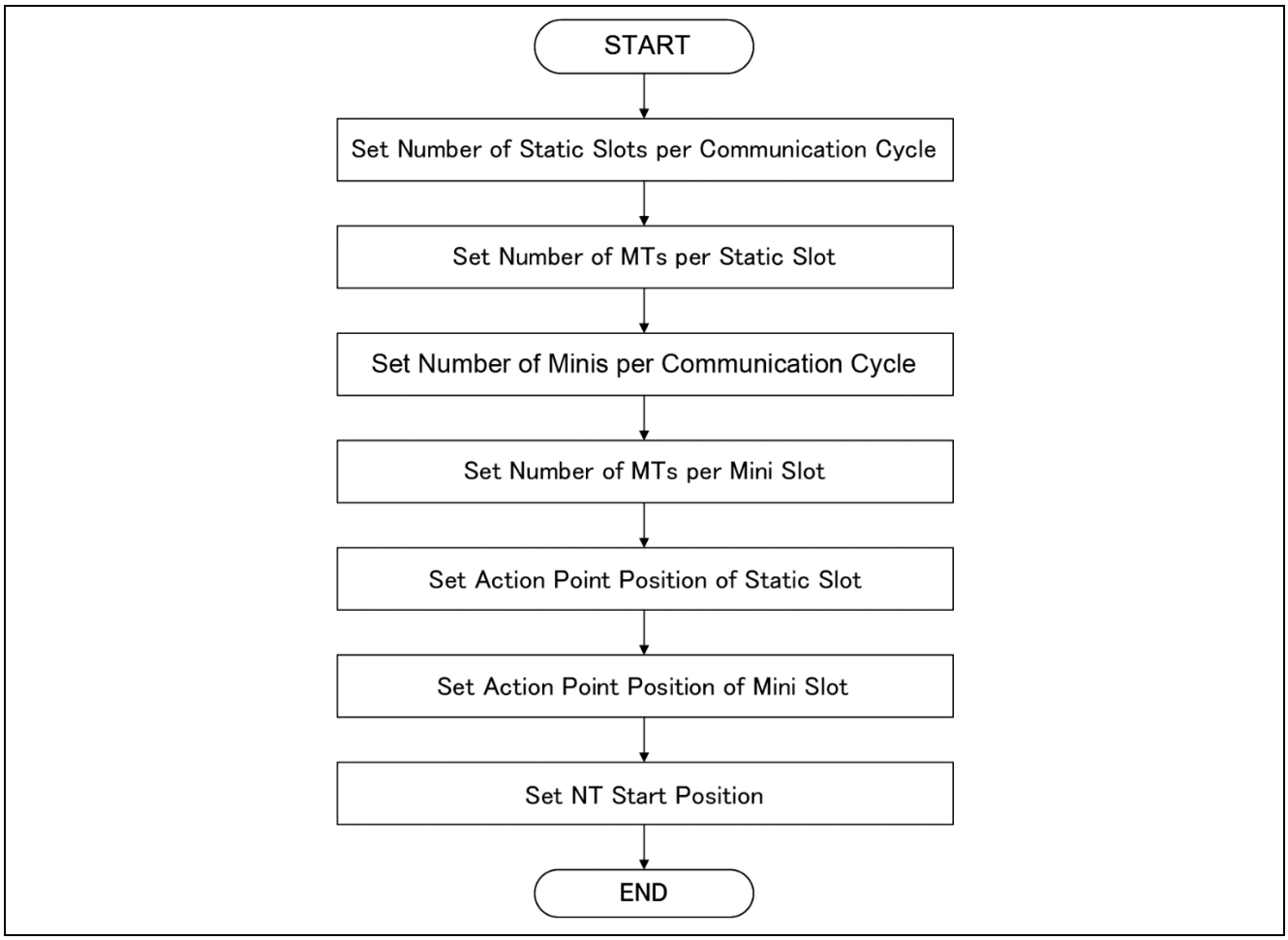


Figure 2-3 Example of Communication Cycle Setting Procedure

【Note】1. There is no register for setting the start position or length of the Symbol window.
The Symbol window extends from the end of the Dynamic segment to the start of the NIT.
2. The communication cycle setting should be performed in the CONFIG state (POCS bits in the FLXAnFRCCSV register = “001111”).
It is not possible to write to the FLXAnFRGTUC2, FLXAnFRGTUC4, or FLXAnFRGTUC7 to FLXAnFRGTUC9 registers in any other state.

2.4.2 Node Role (Setting of Startup Node)

In FlexRay, the node role for Startup frame transmission must be determined during the initial configuration. This configuration affects the node's state transition path during startup.

The Startup frame must always also serve as a Sync frame.

The state transition paths for the various FLXAnFRSUCC1 flag settings are shown in Figure 2-4.

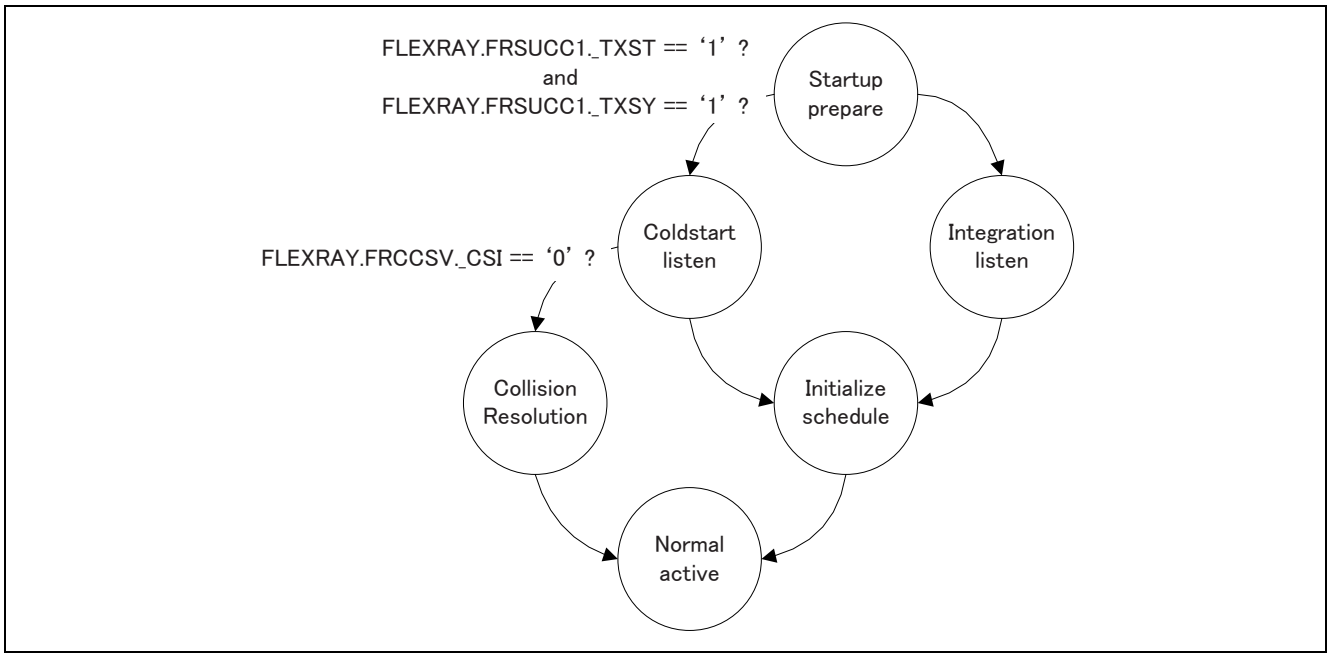


Figure 2-4 Node Role Setting

An example setting procedure and sample program for configuring a Startup node are shown below.。

Table 2-4 Example Configuration of a Startup Node

Name	Setting Values
Startup Frame Transmission	yes
Sync Frame Transmission	yes
Coldstart Inhibit Flag	yes
Startup Retry Count	31 times

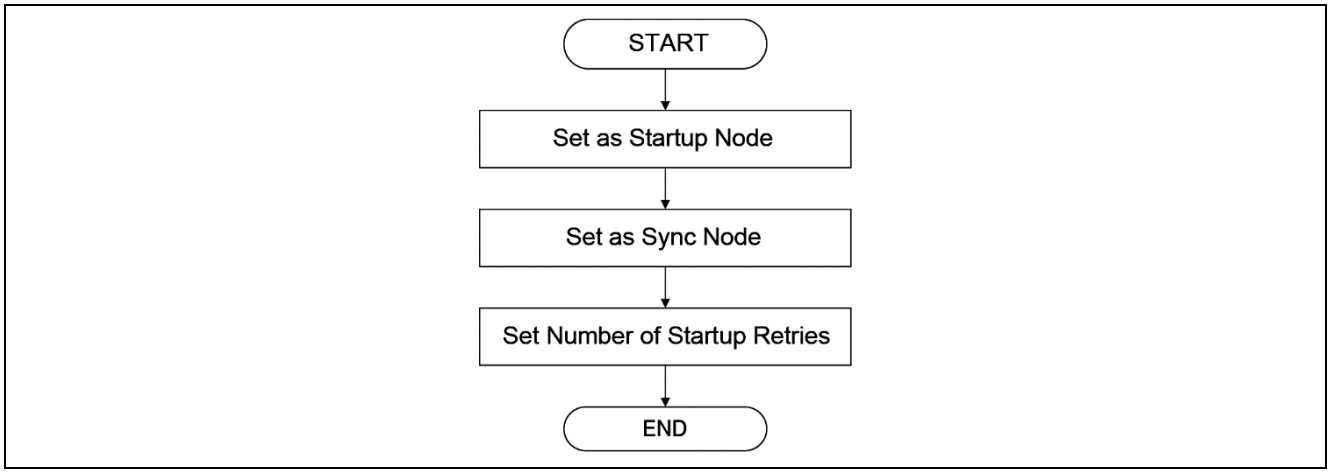


Figure 2-5 Example of Control Procedure

Explanation of the Setting Procedure Example

The TXST and TXSY bits in the FLXAnFRSUCC1 register enable operation as a Startup node and as a Sync node, respectively.

To configure as a Startup node, set the TXST bit in the FLXAnFRSUCC1 register to “1”.
To configure as a Sync node, set the TXSY bit to “1”.

Startup and Sync frames are transmitted only from Message Buffer 0 and Message Buffer 1.

Set the ID (pKeySlotId) of the frame to be used as a Startup or Sync frame in the header of Message Buffer 0 and Message Buffer 1.

If the SPLM bit in the FLXAnFRMRC register is set to “0”, only Message Buffer 0 is used.
If set to “1”, both Message Buffer 0 and Message Buffer 1 are used.
In this case, Startup and Sync frames with different payload data can be transmitted on Channel A and Channel B.

【Note】

1. Perform the setting in the CONFIG state (FLXAnFRCCSV register POCS bits = "001111"). In any other state, writing to the TXST, TXSY, or CSA bits of the FLXAnFRSUCC1 register is not allowed.
2. When the CSI (Coldstart Inhibit) bit of the FLXAnFRCCSV register is “1”, Cold_start is inhibited. This flag is automatically set to “1” upon transition to the READY state.
   To allow Cold_start, clear the CSI bit of the FLXAnFRCCSV register to “0” by writing “1001” (ALLOW_COLDSTART command) to the CMD bits of the FLXAnFRSUCC1 register.
3. Setting TXSY = 0 and TXST = 1 in the FLXAnFRSUCC1 register is invalid.
4. Check the consistency between the SPLM bit of the FLXAnFRMRC register and the CH bit (channel filter) of the FLXAnFRWRHS1 register.

2.5 Setting of Message RAM

2.5.1 Message RAM Buffer Configuration

The FlexRay module contains an 8 KB Message RAM. The number of buffers can be set arbitrarily from 0 to 128.

By setting the SEC bit of the FLXAnFRMRC register, it is possible to enable or disable writing to specific message buffers and transmission in the Static segment.

The buffer configuration within the Message RAM is shown in Figure 2-6.

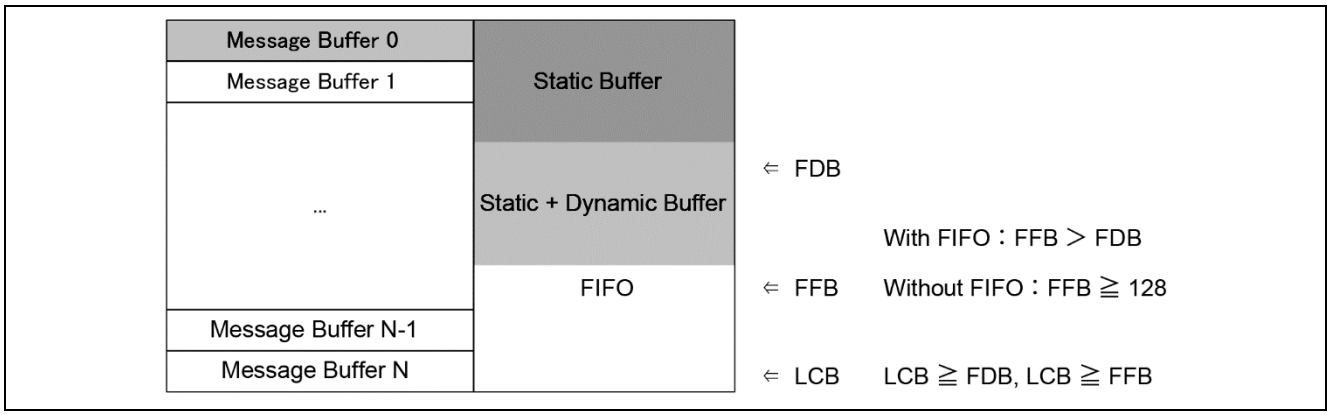


Figure 2-6 Message Buffer Configuration

【Note】

1. When configuring the FLXAnFRMRC register, note the following.
   The size of the FIFO buffer is determined by the FFB and LCB bits. When using both Dynamic buffers and FIFO buffers, the value of the FFB bit must be larger than the value of the FDB bit. When not using Dynamic buffers, set the FDB bit to 128 or a value greater than 128. When not using FIFO buffers, set the FFB bit to 128 or a value greater than 128. The value of the LCB bit must be larger than the FFB bit value when using FIFO buffers, or larger than the FDB bit value when also using Dynamic buffers.
2. The FlexRay module does not check whether the Message RAM configuration is correct.
3. Message RAM configuration must be performed in the CONFIG state (FLXAnFRCCSV register, POCS bits = “001111”).
   Writing to the FLXAnFRMHDC or FLXAnFRMRC registers is not allowed in any other state.。
4. After a hardware reset, the module enters the Default_config state (FLXAnFRCCSV register, POCS bits = “000000”).
   Before initial configuration, transition to the Config state by issuing the CHI CONFIG command (FLXAnFRSUCC1 register, CMD bits = “0001”).

■Example of Header Configuration (1) (When Only Static Buffers Are Configured in the Message RAM)

An example of the configuration when only Static buffers are configured in the Message RAM is shown in Table 2-7, and an example of the configuration procedure is shown in Figure 2-7.

Table 2-7 Example of the Message RAM Configuration

RAM（4byte）	Buffer Number	Configuration	Contents
0...3	Message Buffer 0	↓Static Buffer	Header Section
4...7	Message Buffer 1		
	・・・・		
52...55	Message Buffer 6		←LCB
56...59	Message Buffer 7		
60			Data Section
	・・・・		
2047			

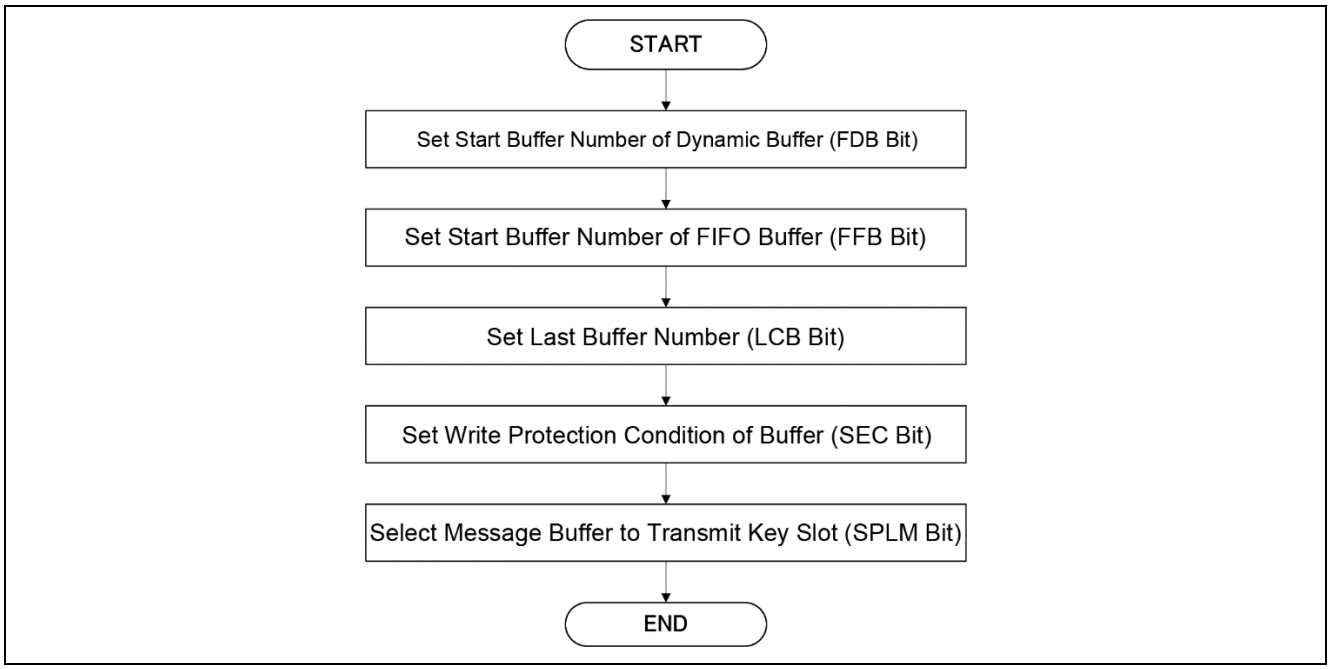


Figure 2-7 Example of configuration procedure

Explanation of the Setting Procedure Example

In this example, only static buffers are configured, and dynamic buffers and FIFO buffers are not configured. Set the FDB bit of the FLXAnFRMRC register to “128” and the FFB bit of the FLXAnFRMRC register to “128.”

Since the number of message buffers is eight, set the LCB bit of the FLXAnFRMRC register to “7.”

■Example of Header Section Configuration（2）（When configuring Static buffers and FIFO buffers in the Message RAM）

An example configuration when configuring Static buffers and FIFO buffers in the Message RAM is shown in Table 2-8.

Table 2-8 Example of Message RAM Configuration

RAM（4byte）	Buffer Number	Configuration	Contents
0...3	Message Buffer 0	↓Static Buffer	Header Section
4...7	Message Buffer 1		
8...11	Message Buffer 2		←FFB
12...15	Message Buffer 3		←LCB
16			Data Sction
	....		
2047			

Explanation of the Setting Procedure Example and Program Example

Since there are 4 message buffers, set the LCB bit of the FLXAnFRMRC register to "3".

In this example, 2 Static buffers and 2 FIFO buffers are configured. Since no Dynamic buffers are used, set the FDB bit of the FLXAnFRMRC register to “128”.

For 2 FIFO buffers, set the FFB bit of the FLXAnFRMRC register to “2” and the LCB bit to “3”.

Change the macro definition value MRAMC_SET in the flexray.c file as required.

■Example of Header Section Configuration（3）（When configuring Static buffers and Dynamic Buffer）

An example configuration when configuring Static buffers and Dynamic buffers in the Message RAM is shown in Table 2-9.

Table 2-9 Example of Message RAM Configuration

RAM（4byte）	Buffer Number	Configuration	Contents
0...3	Message Buffer 0	↓Static Buffer	Header Section
4...7	Message Buffer 1		
8...11	Message Buffer 2		
12...15	Message Buffer 3		
16...19	Message Buffer 4		
20...23	Message Buffer 5		
24...27	Message Buffer 6	↓Static+Dynamic Buffer	
28...31	Message Buffer 7		←FDB
32...35	Message Buffer 8		
36...39	Message Buffer 9		
40...43	Message Buffer 10		
44...47	Message Buffer 11		←LCB
48			Data Section
	....		
2047			

Explanation of the Setting Procedure Example and Program Example

Since there are 12 message buffers, set the LCB bit of the FLXAnFRMRC register to "11".

In this example, 6 Static buffers and 6 Dynamic buffers are configured. Since no Dynamic buffers are used, set the FFB bit of the FLXAnFRMRC register to “128”.

Since the starting number of the Dynamic buffer header is “6”, set the FDB bit of the FLXAnFRMRC register to “6”. Since the number of Dynamic buffers is 6, set the LCB bit of the FLXAnFRMRC register to “11”.。

Change the macro definition value MRAMC_SET in the flexray.c file as required.

2.5.2 Structure of Data Field and Header Field

The FlexRay module allows the data length of each buffer to be set arbitrarily in the range of 0 to 254 bytes in 2-byte increments.

Within the Message RAM, frame information is divided into a header section and a data section.

Each message buffer requires a header section (fixed size: 16 bytes) and a data section (variable size).

When configuring transmit and receive frames, the header section is set using the FLXAnFRWRHS1–FLXAnFRWRHS3 registers, and the data section is set using the FLXAnFRWRDS1–FLXAnFRWRDS64 registers.

The data set in each register is transferred to the message buffer in the Message RAM via the input buffer (IBF). For the transfer method, refer to "3. Data Setting to and Reading from Message RAM." This section describes the configuration only.

An example of the internal structure of the Message RAM is shown in Figure 2-8, and Table 2-10 shows the data length per buffer and the number of buffers that can be allocated.

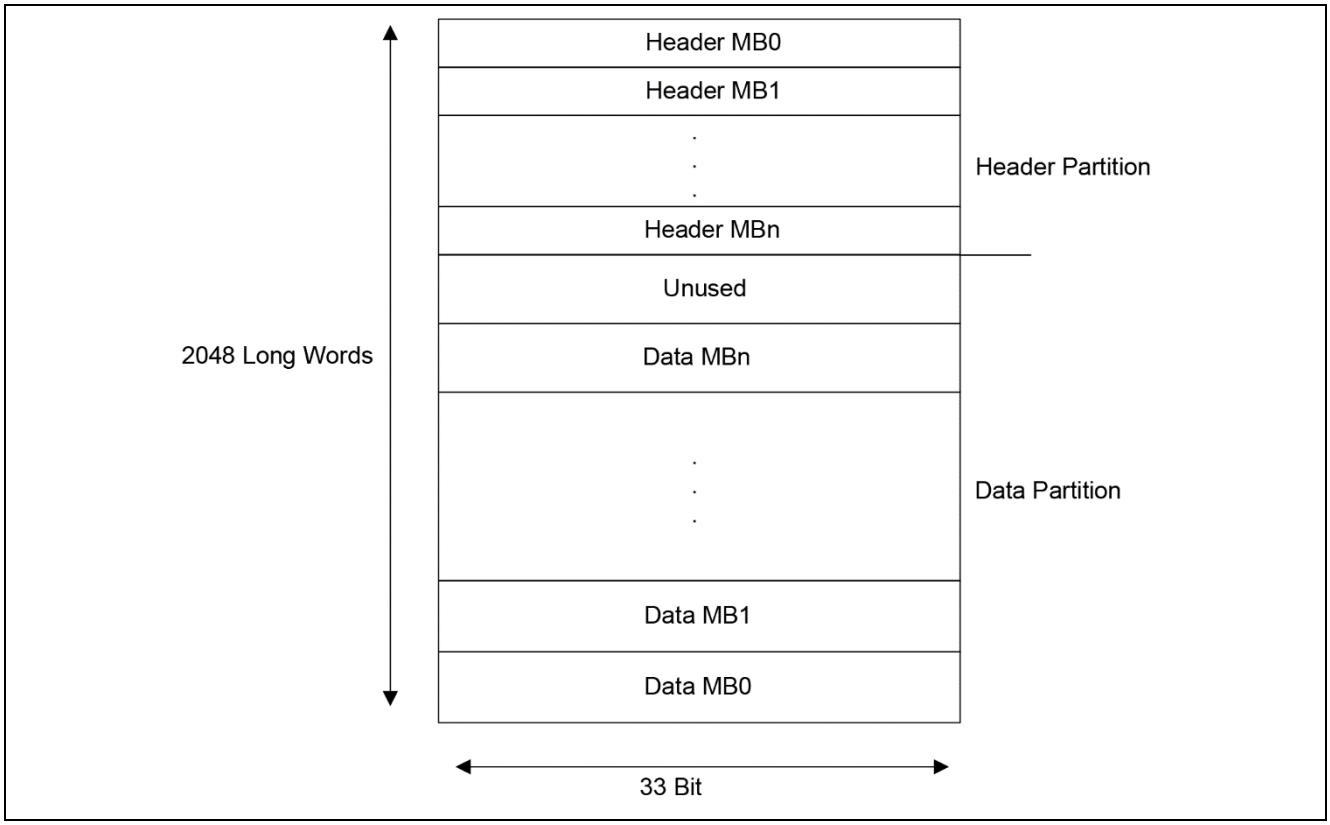


Figure 2-8 Message RAM Internal Structure

Table 2-10 Data Length per Buffer and Number of Allocatable Buffers

Data Length per Buffer	Number of Allocatable Buffers	Description
254 bytes	30 units	
128 bytes	56 units	
48 bytes	128 units	

【Note】

1. Set the data length of each buffer so that the total of the "header section + data section" of all buffers fits within 8 KB. The table shows the number of buffers that can be allocated when all buffers have the same data length.
2. The payload length set in the header section of FIFO buffers and the length of the data section must be the same for all buffers. These can be configured using the PLC0–PLC6 bits of the FLXAnFRWRHS2 register and the DP0–DP10 bits of the FLXAnFRWRHS3 register.

2.5.3 Data Pointer

The FlexRay module calculates the start address of the data field from the data pointer in the header field when reading or writing frame information to or from the data field of a message buffer.

The data pointer is set to the value of “start address of data field ÷ 4 bytes,” with the start address of Message RAM (address of message buffer 0) defined as “H’0.”
Set the data pointer value to the DP0 to DP10 bits of the FLXAnFRWRHS3 register.

【Note】

1. According to the above specification, the start address of the data field must be set to a multiple of 4.
2. Do not change the value of the DP0 to DP10 bits of the FLXAnFRWRHS3 register during operation.
3. Even if an incorrect value is set, such as specifying the same address in multiple buffers whose data pointers refer to header fields, the FlexRay module still performs data storage into message buffers.
   In such cases, payload data in the data field may be lost or may be overwritten with invalid data.
   When frame data is attempted to be written to a header field due to incorrect configuration, the WAHP bit of the FLXAnFRMHDF register is set to “1.” In this case, the header field is protected against overwriting, and the data will be lost.。

■Example of data pointer calculation（1）

An example of the Message RAM configuration and the data pointer values for each buffer are shown in Table 2-11.

Table 2-11 Example of Message RAM Configuration

RAM（4byte）	Buffer Number	Configuration		Contents
0...3	Message Buffer 0	Static Buffer、DP = 12		Header Section
4...7	Message Buffer 1	Static Buffer、DP = 14		
8...11	Message Buffer 2	Static Buffer、DP = 16		←LCB
12		MB0 Data1	MB0 Data 0	Data Section
13		Not used*	MB0 Data 2	←DP MB0
14		MB1 Data 1	MB1 Data 0	
15		Not used*	MB1 Data 2	←DP MB1
16		MB2 Data 1	MB2 Data 0	
17		Not used*	MB2 Data 2	←DP MB2
18...2046		Not used		
2047		Not used		

【Note】 * This section cannot be accessed.

Explanation of the Control Procedure Example

Since the number of message buffers is three, the LCB bit of the FLXAnFRMRC register is set to “2”.
In this case, the data pointer must be “12” or greater, which is calculated as (LCB + 1) × 4 = “12”.

Because the data length of the static frame is 6 bytes, set the SFDL bit of the FLXAnFRMHDC register to “3”.

The data pointers for each message buffer are as follows:

The data pointer of Message Buffer 0: DP bit of the FLXAnFRWRHS3 register = “12”

The data pointer of Message Buffer 1: DP bit of the FLXAnFRWRHS3 register = “14”

The data pointer of Message Buffer 2: DP bit of the FLXAnFRWRHS3 register = “16”

In this example, since neither the FIFO buffer nor the dynamic buffer is used, the FDB bit of the FLXAnFRMRC register is set to “128”, and the FFB bit of the FLXAnFRMRC register is set to “128”.

■Example of data pointer calculation（2）

An example of the Message RAM configuration and the data pointer values for each buffer are shown in Table 2-12.

Table 2-12 Example of Message RAM Configuration

RAM（4byte）	Buffer Number	Configuration		Contents
0...3	Message Buffer 0	Static Buffer、DP = 2046		Header Section
4...7	Message Buffer 1	Static Buffer、DP = 2044		
8...11	Message Buffer 2	Dynamic Buffer、DP = 2042		←FDB
12...15	Message Buffer 3	Dynamic Buffer、DP = 2040		←LCB
16...2039		Not used		Data Section
2040		MB3 Data1	MB3 Data 0	←DP MB3
2041		MB3 Data 3	MB3 Data 2	
2042		MB2 Data 1	MB2 Data 0	←DP MB2
2043		MB2 Data 3	MB2 Data 2	
2044		MB1 Data 1	MB1 Data 0	←DP MB1
2045		Not used *	MB1 Data 2	
2046		MB0 Data 1	MB0 Data 0	←DP MB0
2047		Not used *	MB0 Data 2	

【Note】 * This section cannot be accessed.

Explanation of the Control Procedure Example

Since the number of message buffers is four, the LCB bit of the FLXAnFRMRC register is set to “3”. In this case, the data pointer must be equal to or greater than (FLXAnFRMRC register LCB + 1) × 4 = “16”.

In this example, two static buffers and two dynamic buffers are set. The FIFO buffer is not used. The FDB bit of the FLXAnFRMRC register is set to “2”, the LCB bit of the FLXAnFRMRC register is set to “3”, and the FFB bit of the FLXAnFRMRC register is set to “128”.

Since the data length of the static frame is 6 bytes, the SFDL bit of the FLXAnFRMHDC register is set to “3”.

Since the payload length of the dynamic buffer is 8 bytes, the PLC bit of the FLXAnFRWRHS2 register of message buffers 2 and 3 is set to “4”.

The data pointer of each message buffer is as follows:

The DP bit of the FLXAnFRWRHS3 register for message buffer 0 is set to “2046”.
The DP bit of the FLXAnFRWRHS3 register for message buffer 1 is set to “2044”.
The DP bit of the FLXAnFRWRHS3 register for message buffer 2 is set to “2042”.
The DP bit of the FLXAnFRWRHS3 register for message buffer 3 is set to “2040”.

■Example of data pointer calculation（3）

An example of the Message RAM configuration and the data pointer values for each buffer are shown in Table 2-13.

Table 2-13 Example of Message RAM Configuration

RAM（4byte）	Buffer Number	Configuration		Contents
0...3	Message Buffer 0	Static Buffer、DP = 1100		Header Section
4...7	Message Buffer 1	Static Buffer、DP = 1101		
8...11	Message Buffer 2	Dynamic Buffer、DP = 1985		←FDB
12...15	Message Buffer 3	Dynamic Buffer、DP = 1987		
16...19	Message Buffer 4	FIFO Buffer、DP = 79		←FFB
20...23	Message Buffer 5	FIFO Buffer、DP = 81		
24...27	Message Buffer 6	FIFO Buffer、DP = 83		
28...31	Message Buffer 7	FIFO Buffer、DP = 85		←LCB
32		Not used		Data Section
33...78		Not used		
79		MB4 Data 1	MB4 Data 0	←DP MB4
80		MB4 Data 3	MB4 Data 2	
81		MB5 Data 1	MB5 Data 0	←DP MB5
82		MB5 Data 3	MB5 Data 2	
83		MB6 Data 1	MB6 Data 0	←DP MB6
84		MB6 Data 3	MB6 Data 2	
85		MB7 Data 1	MB7 Data 0	←DP MB7
86		MB7 Data 3	MB7 Data 2	
87...1099		Not used		
1100		MB0 Data 1	MB0 Data 0	←DP MB0
1101		MB1 Data 1	MB1 Data 0	←DP MB1
1102...1984		Not used		
1985		MB2 Data 1	MB2 Data 0	
1986		Not used *	MB2 Data 2	←DP MB2
1987		MB3 Data 1	MB3 Data 0	
1988		Not used *	MB3 Data 2	←DP MB3
1989...2046		Not used *		
2047		Not used *		

【Note】 * This section cannot be accessed.

Explanation of the Control Procedure Example

Since the number of message buffers is eight, the LCB bit of the FLXAnFRMRC register is set to “7”. In this case, the data pointer must be equal to or greater than (FLXAnFRMRC register LCB bit + 1) × 4 = “32”.

In this example, two static buffers, two dynamic buffers, and four FIFO buffers are used. The FDB bit of the FLXAnFRMRC register is set to “2”, the FFB bit of the FLXAnFRMRC register is set to “4”, the LCB bit of the FLXAnFRMRC register is set to “7”, and the FFB bit of the FLXAnFRMRC register is set to “128”.

Since the data length of the static frame is 4 bytes, the SFDL bit of the FLXAnFRMHDC register is set to “2”.

Since the payload length of the dynamic buffer is 6 bytes, the PLC bit of the FLXAnFRWRHS2 register of message buffers 2 and 3 is set to “3”.

Since the payload length of the FIFO buffer is 8 bytes, the PLC bit of the FLXAnFRWRHS2 register of message buffers 4 to 7 is set to “4”.

The data pointer of each message buffer is as follows:

The DP bit of the FLXAnFRWRHS3 register for message buffer 0 is set to “1100”.

The DP bit of the FLXAnFRWRHS3 register for message buffer 1 is set to “1101”.

The DP bit of the FLXAnFRWRHS3 register for message buffer 2 is set to “1985”.

The DP bit of the FLXAnFRWRHS3 register for message buffer 3 is set to “1987”.

The DP bit of the FLXAnFRWRHS3 register for message buffer 4 is set to “79”.

The DP bit of the FLXAnFRWRHS3 register for message buffer 5 is set to “81”.

The DP bit of the FLXAnFRWRHS3 register for message buffer 6 is set to “83”.

The DP bit of the FLXAnFRWRHS3 register for message buffer 7 is set to “85”.

3. Communication Controller Status

3.1 Communication Controller State Transition Diagram

Transitions between states are made by writing commands to the CMD bit of the FLXAnFRSUCC1 register.

The sequence from power-on to the start of network communication is shown in Figure 3-1.

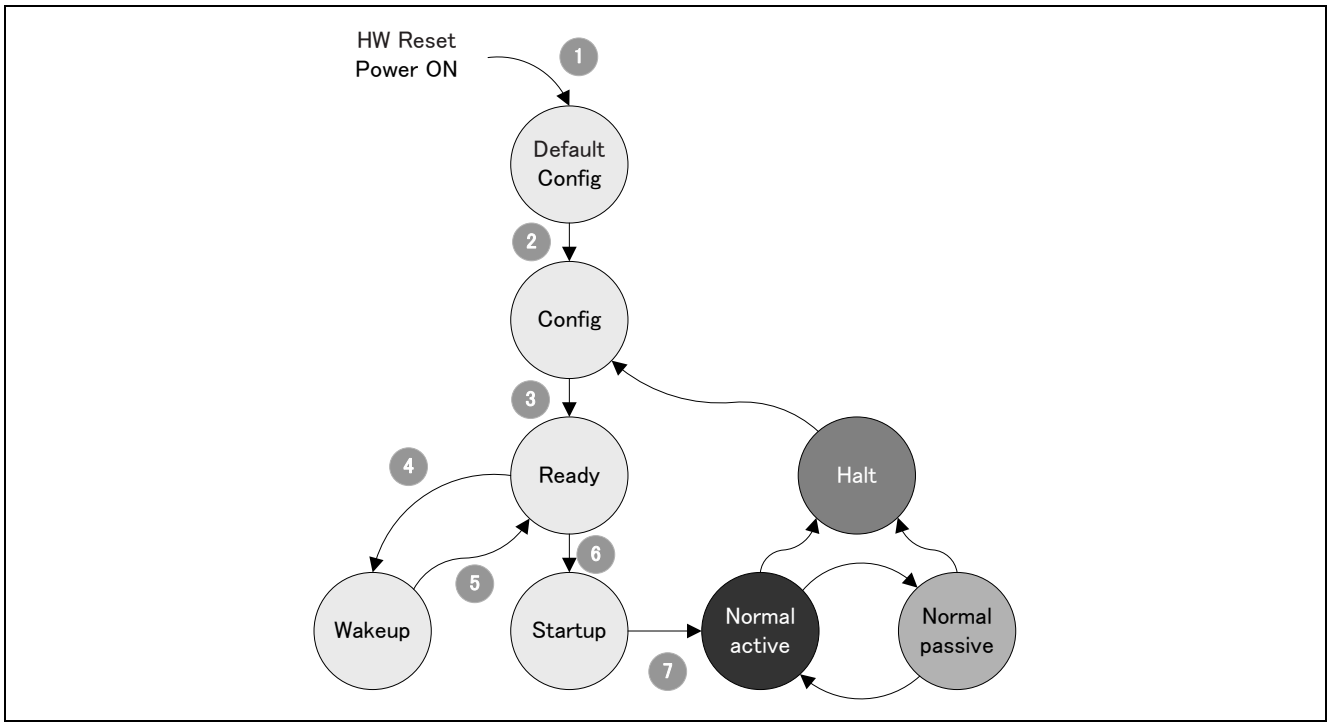


Figure 3-1 Flow from power-on to the start of network communication

The operations required to transition between each state are shown in Table 3-1.

Table 3-1 The Operations Required to Transition Between Each State

State Before Transition	State After Transition	Required Operations
①All States	Default_config	—
②Default Config	Config	Write “0001” (Config) to the CMD bit of the FLXAnFRSUCC1 register
③Config	Ready	After unlocking the CLK bit of the FLXAnFRLCK register, write “0010” (Ready) to the CMD bit of the FLXAnFRSUCC1 register
④Ready	Wakeup	Write “0011” (Wakeup) to the CMD bit of the FLXAnFRSUCC1 register
⑤Wakeup	Ready	Write “0010” (Ready) to the CMD bit of the FLXAnFRSUCC1 register (automatically transitions upon Wakeup completion or interruption)
⑥Ready	Startup	Write “0100” (Run) to the CMD bit of the FLXAnFRSUCC1 register
⑦Startup	Normal active	- (Automatically transitions upon Startup completion)
Startup Normal active	Ready	Write “0010” (Run) to the CMD bit of the FLXAnFRSUCC1 register

3.2 Default Config to Ready States

After reset, the device is in the Default Config state.

After transitioning to the Config state by the CHI CONFIG command (CMD bit of the FLXAnFRSUCC1 register), perform the initial settings for the various configuration registers. Before transitioning from the CONFIG state to any other state, the CLK bit of the FLXAnFRLCK register must be unlocked prior to writing a command to the CMD bit. This function prevents unintended operations of the communication controller caused by software runaway or other unintended actions.

An example of the operation procedure is shown in Figure 3-2.

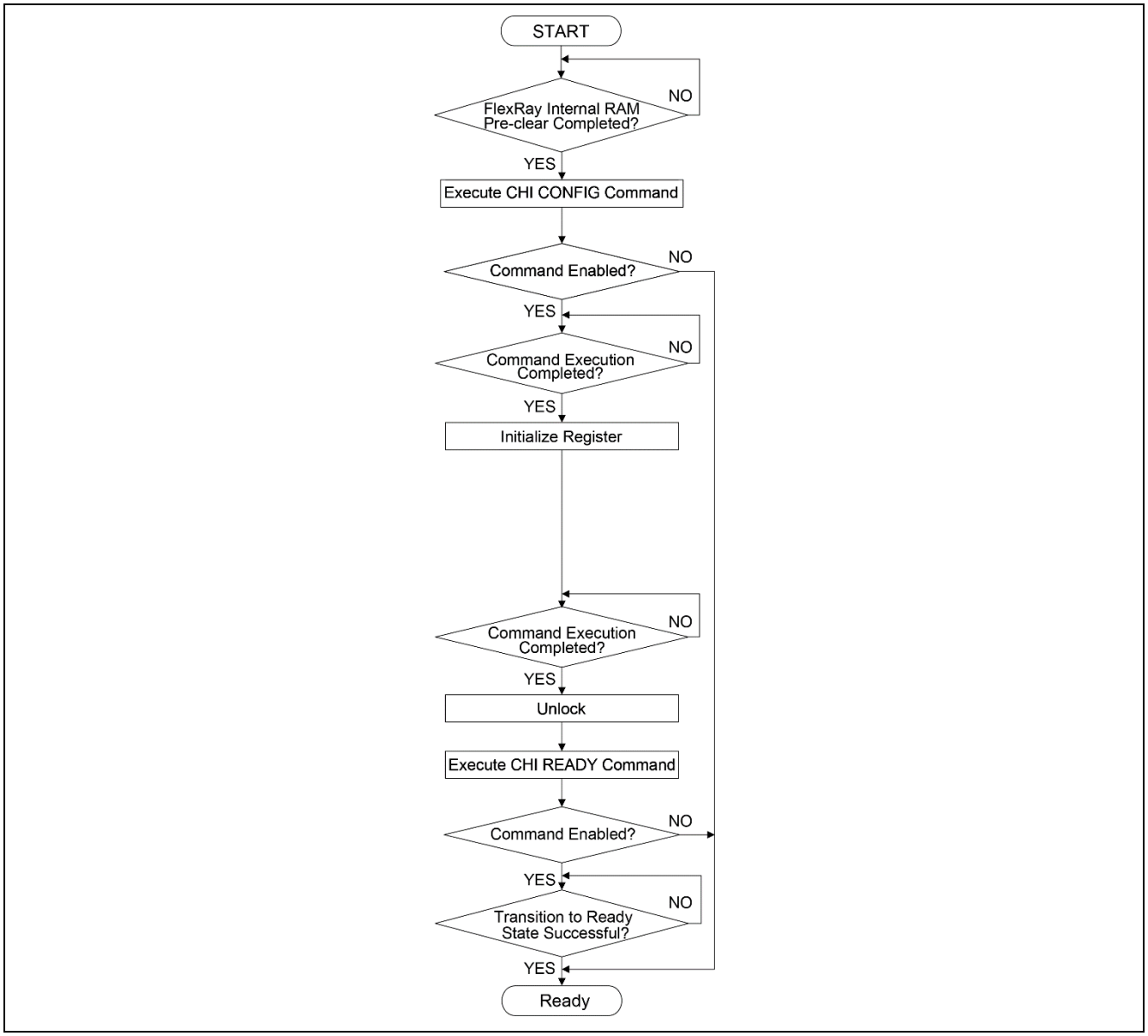


Figure 3-2 Example Setting Procedure

【Note】

1. After reset, do not configure any registers while the Message RAM is being cleared. Configure each register only after confirming that the CRAM bit of the FLXAnFRMHDS register and the PBSY bit of the FLXAnFRSUCC1 register are “0”.
2. To transition from the Config state to another state, the lock must be released via the FLXAnFRLCK register. When releasing the lock, writes to the FLXAnFRLCK register must be performed consecutively. If any other operation is performed between the two writes, the procedure must be restarted from the first write.

3.3 Wakeup State

In FlexRay, all channels must be in the active state before communication starts.

In the Wakeup state, the Wakeup master node activates the nodes connected to the channels.

If the connected channels are not active and the node itself is the Wakeup master node, it transitions to the Wakeup state by issuing the CHI WAKEUP command (CMD bit of the FLXAnFRSUCC1 register) and executes the Wakeup sequence.

If only one channel is active and the other channel needs to be woken up, the node returns from the Ready state to the Config state, changes the channel for sending the Wakeup pattern, and then transitions back to the Wakeup state.

Changes in the Wakeup status can be confirmed via the WST bit of the FLXAnFRSIR register and the WSV0–WSV2 bits of the FLXAnFRCCSV register. When a valid Wakeup pattern is received, the WUPA and WUPB bits of the FLXAnFRSIR register are set.

Figure 3-3 shows the state transition diagram during Wakeup.

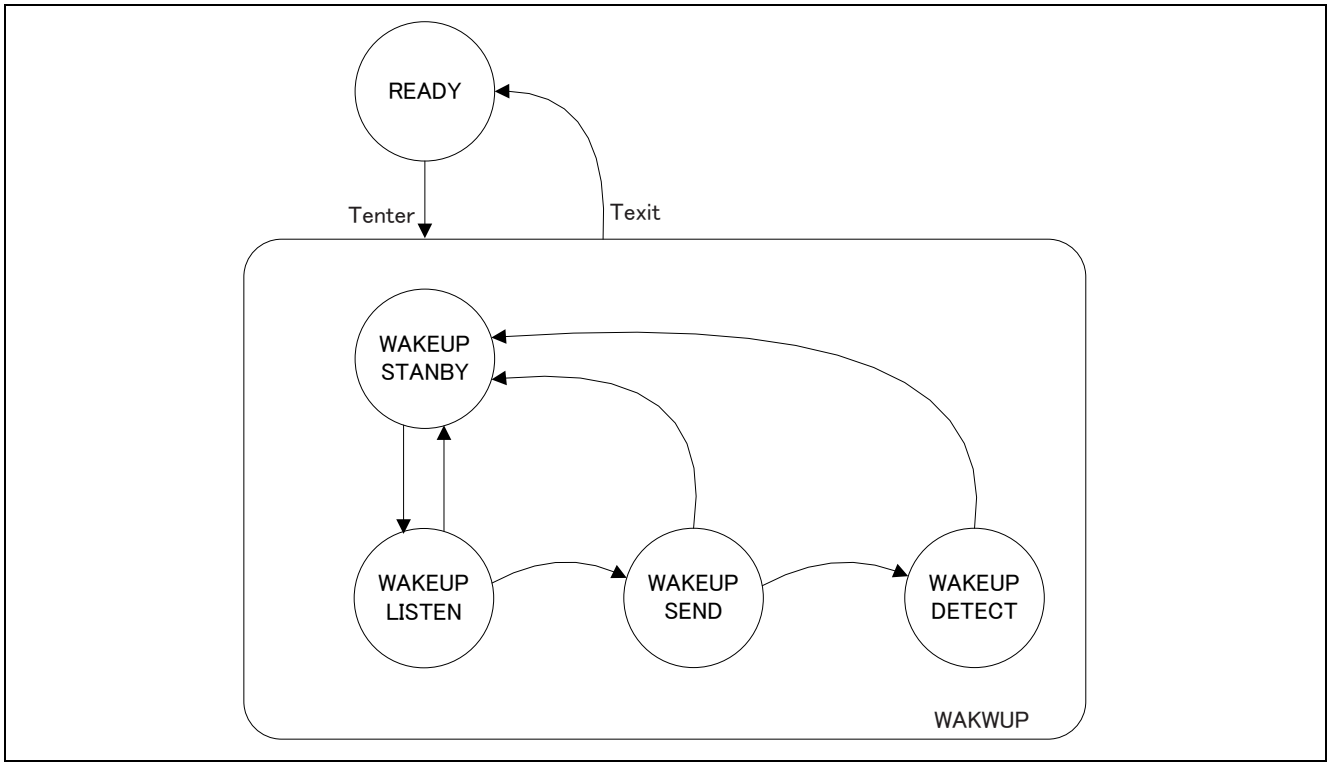


Figure 3-3 State Transition Diagram during Wakeup

【Note】

1. During the Wakeup state, the bus driver must receive the minimum required power for Wakeup.
2. In FlexRay, it is not possible to confirm whether all nodes are active in the Wakeup state. Cold-start nodes must wait for the time required for all nodes to complete Wakeup before starting Startup.
3. The Wakeup channel must be selected using the WUCS bit of the FLXAnFRSUCC1 register. In FlexRay, a node is not allowed to wake up two channels simultaneously.

Figures 3-4 through 3-7 show examples of the Wakeup procedure when operating as a Wakeup master node or a Cold-start node.

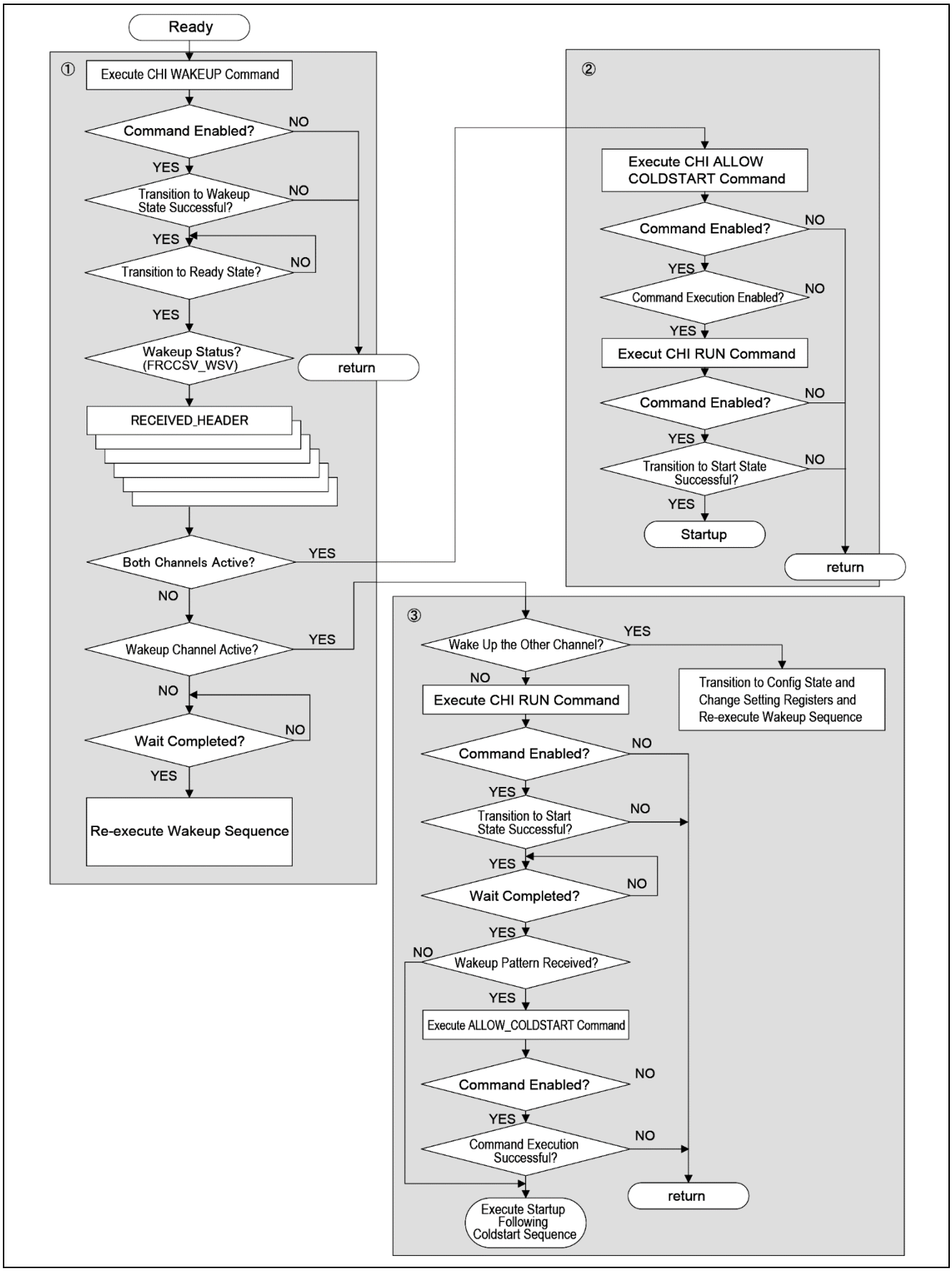


Figure 3-3 Example Wakeup Sequence（1）

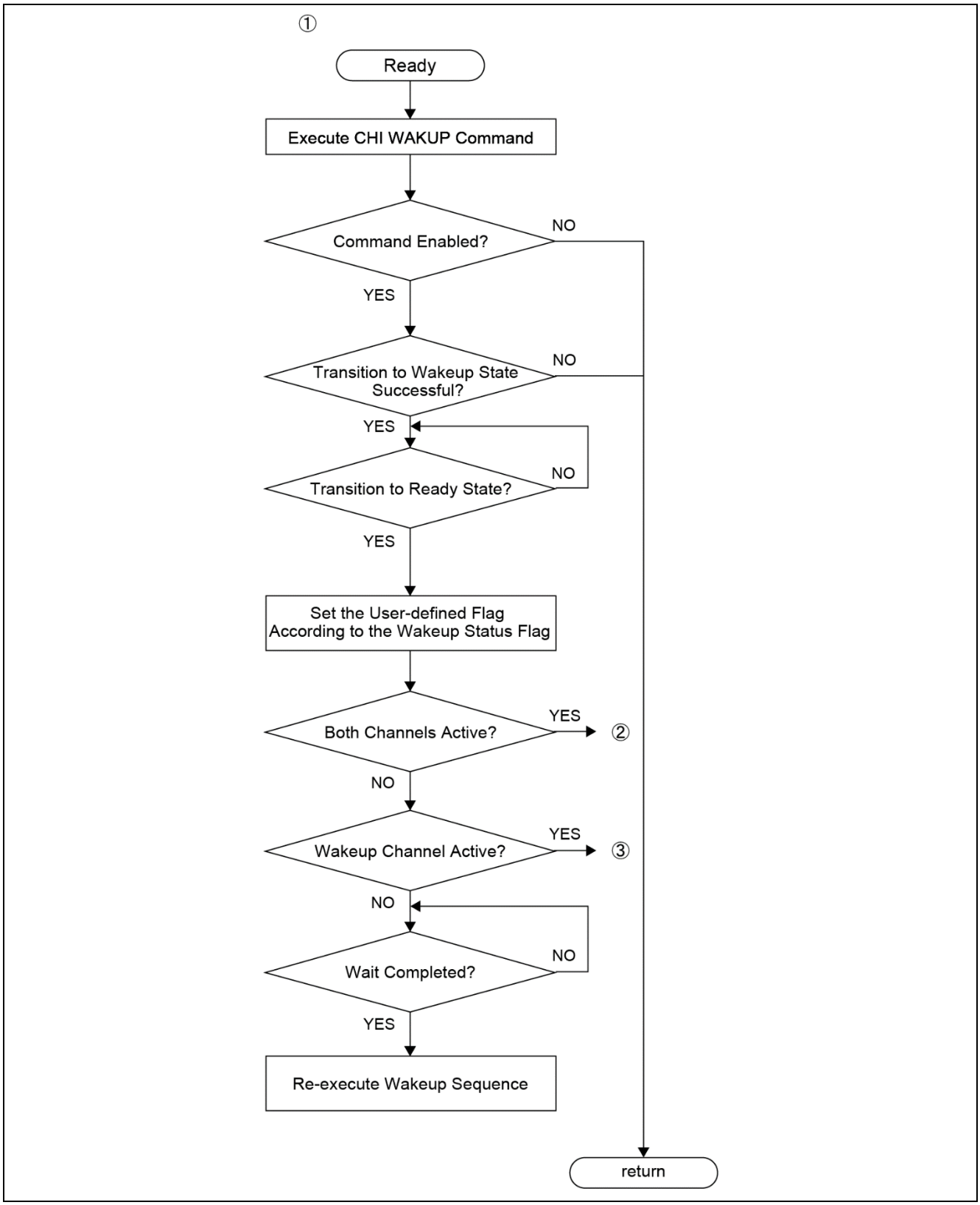


Figure 3-4 Example Wakeup Sequence（2）

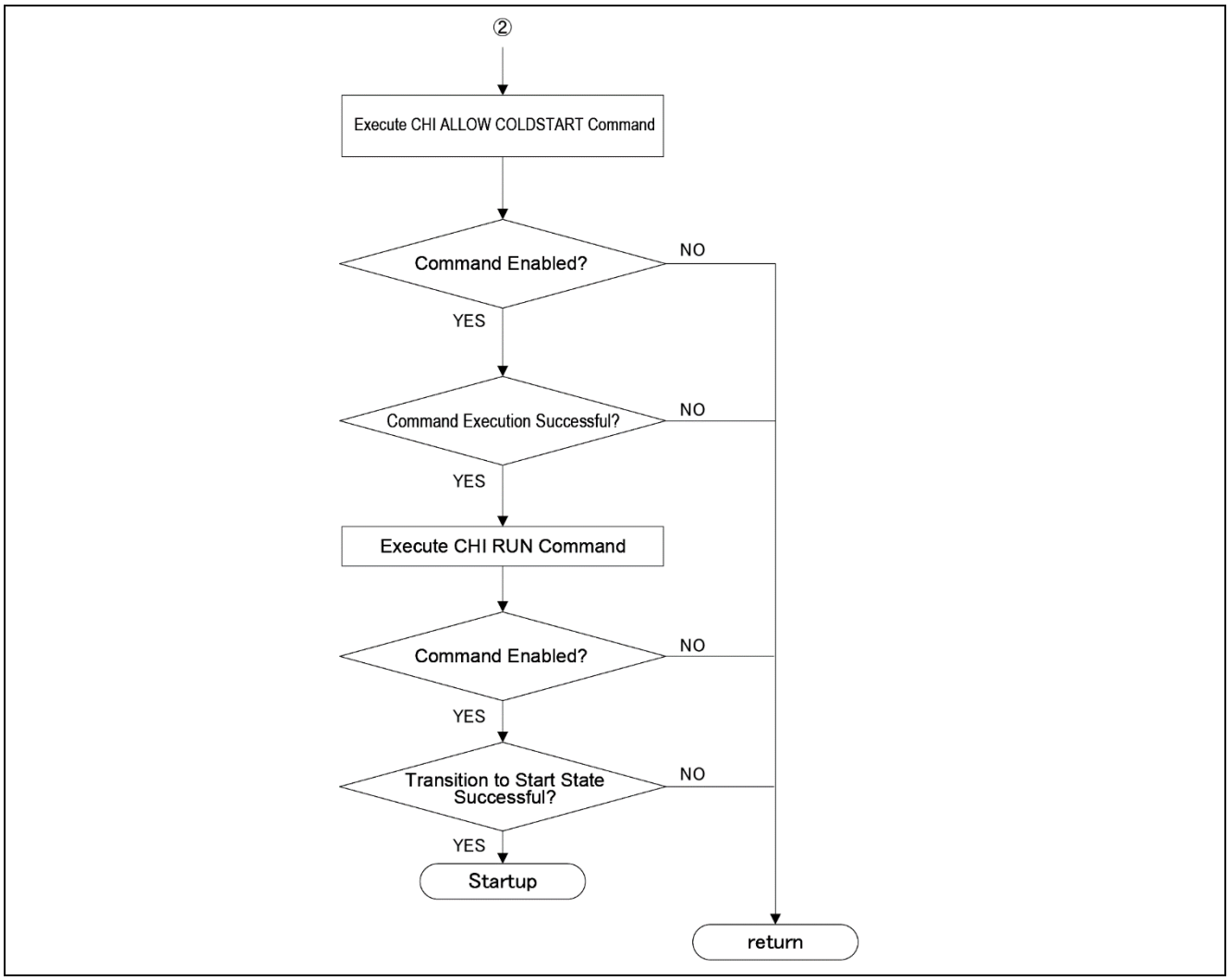


Figure 3-5 Example Wakeup Sequence（3）

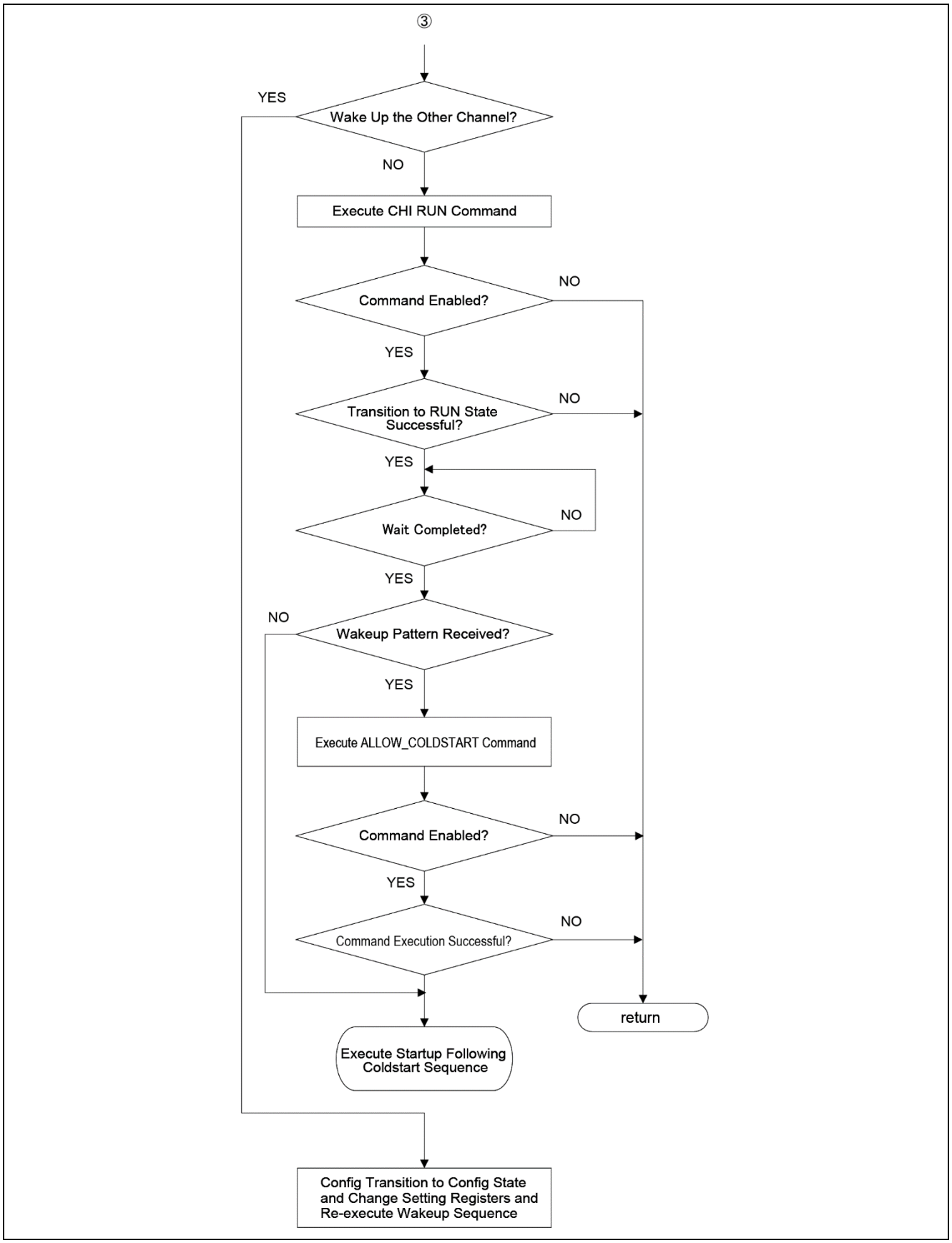


Figure 3-6 Example Wakeup Sequence（4）

3.3.1 Example of Processing in RECEIVED_HEADER or COLLISION_HEADER State

When in the RECEIVED_HEADER (WSV bits of the FLXAnFRCCSV register = “001”) or COLLISION_HEADER (WSV bits = “011”) state, the Wakeup process is aborted, assuming that the cluster in which the local node attempts to execute Wakeup is already in the communication state.

An operational overview example is shown in Figure 3-8.

In the operational overview example, the local node transitions to the Startup state as the leading Cold_start node.

For the control procedure example, refer to Figure 3-4. Since both channels are active, control step ② is executed.

。

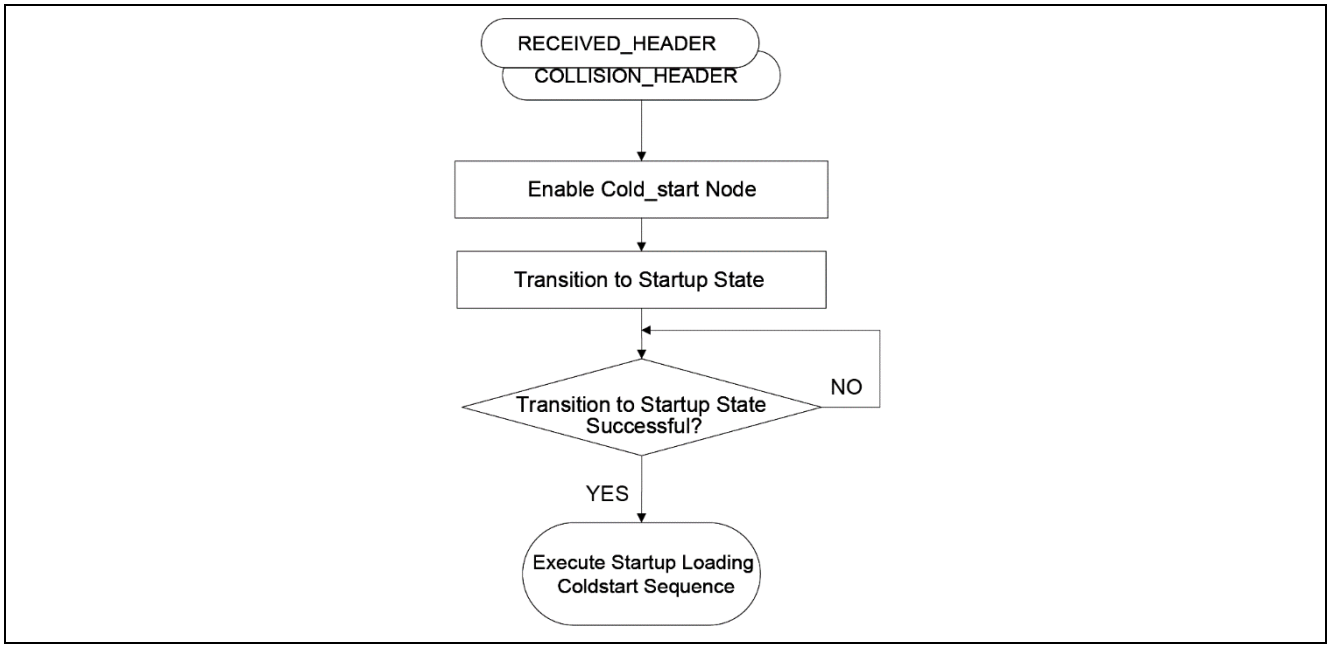


Figure 3-7 Example Setting Procedure

3.3.2 Example of Processing Procedure for RECEIVED_WUP, COLLISION_WUP, or TRANSMITTED States

When in the RECEIVED_WUP state (WSV bits of the FLXAnFRCCSV register = “010”) or COLLISION_WUP state (WSV bits = “100”), the Wakeup process is aborted because the channel that the node is attempting to wake up is already undergoing Wakeup by another node.。

In the TRANSMITTED state (WSV bits = “110”), the Wakeup pattern has been successfully transmitted, and the node transitions to the Startup state. In this case, the node waits for the duration required for other nodes to perform the Wakeup process.

If only one channel is active and the node does not execute Wakeup on the other channel, it transitions to the Startup state by the CHI RUN command (CMD bit of the FLXAnFRSUCC1 register). In this case, the node waits for other nodes to perform Wakeup on the remaining channel. The node then acts as a Following Cold_start Node, waiting for the Startup of other Cold_start nodes and integrating accordingly.

Figure 3-9 shows an example of the setting procedure.

Figure 3-8 Example Setting Procedure

3.3.3 Example of Processing in COLLISION_UNKNOWN State

When the state is COLLISION_UNKNOWN (WSV bits of the FLXAnFRCCSV register = “101”), if no valid wakeup pattern or valid frame header is received and the internal wakeup timer reaches the specified value, the wakeup process stops by exiting the WAKEUP_DETECT state.

When the communication controller is in this state, it can be assumed that there is significant noise interference or an abnormality in communication.

When executing the wakeup process again, wait for a certain period to allow other nodes to complete their internal processing.

【Note】 The Listen Timeout Noise timer value can be set using LTN0 to LTN3 in the FLXAnFRSUCC2 register.

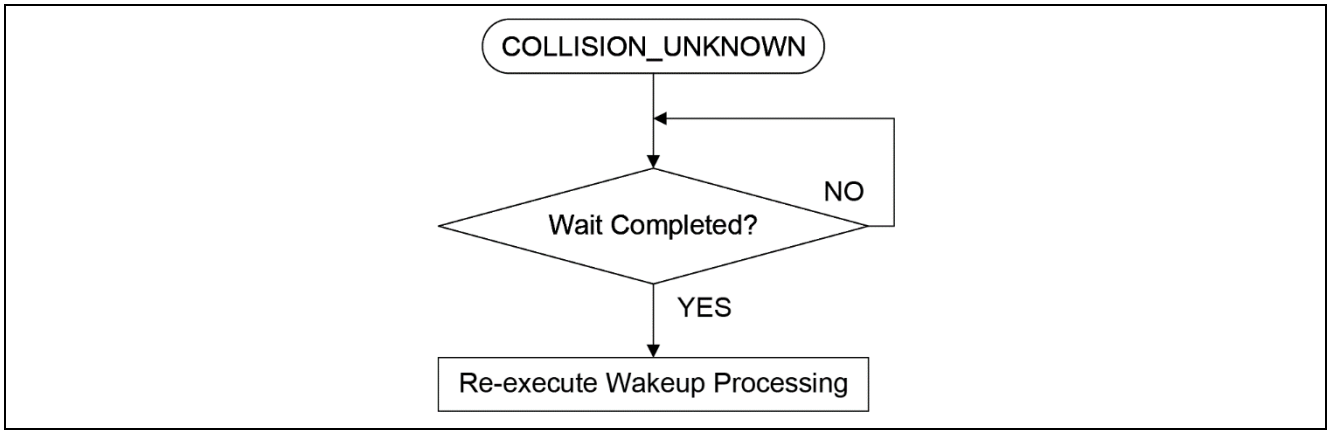


Figure 3-9 Example Control Procedure

3.4 Startup State

In FlexRay, the clocks of all nodes in the cluster must be synchronized before communication starts. Each node performs the startup process to align its clock with that of the Cold-start node.

At least two Cold-start nodes are required to start up the cluster.

For the settings required to operate as a Startup node or Sync node, refer to section “1. Initial Settings.”

Figure 3-11 shows the state transition diagram during startup.

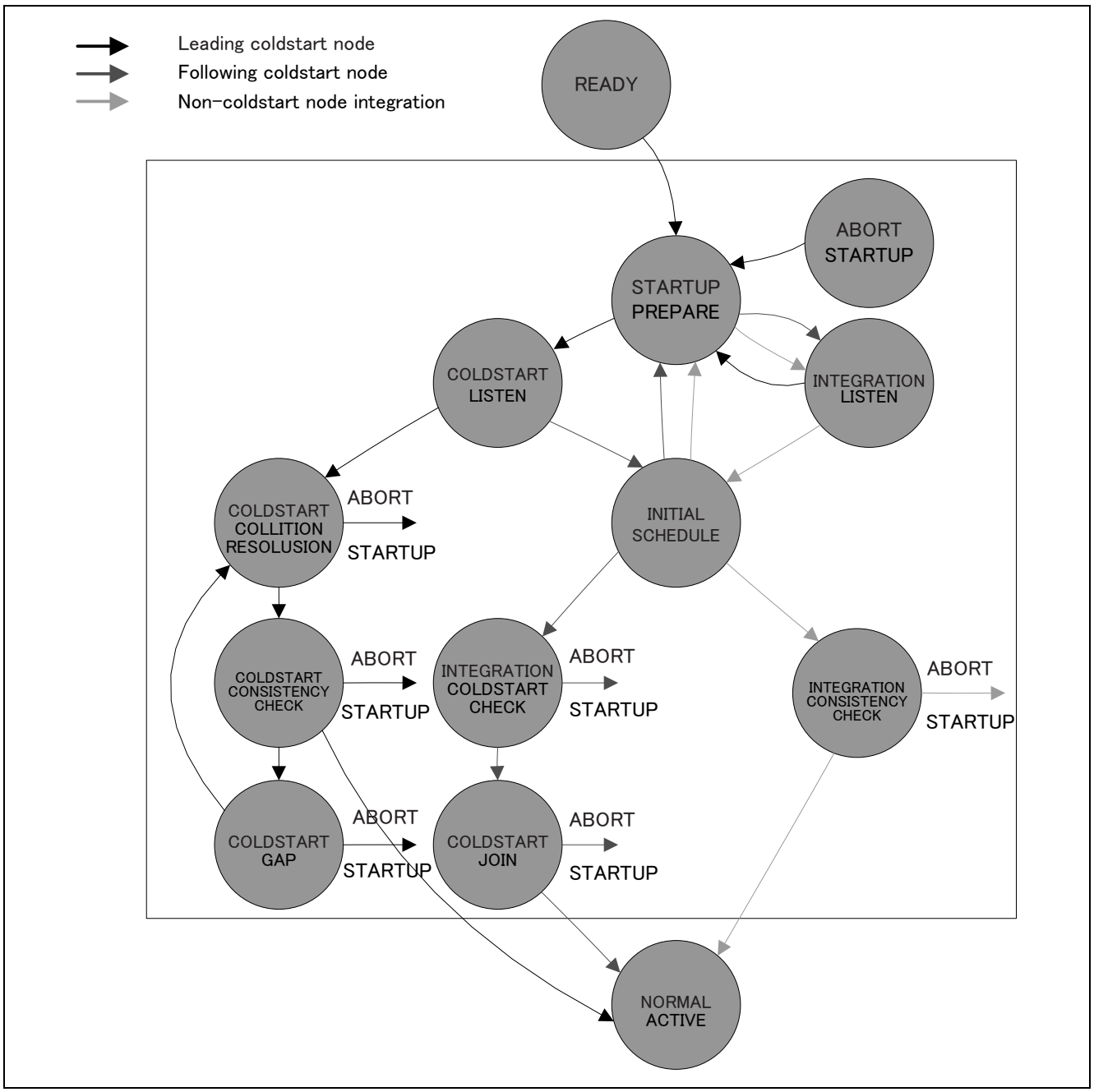


Figure 3-10 State Transitions during Startup

3.4.1 Operation of Cold Start Node

When the node is in the Ready state (POCS bits in the FLXAnFRCCSV register = “000001”), executing the CHI RUN command (CMD bits in the FLXAnFRSUCC1 register = “0100”) transitions the node to the Startup state.

The number of cold start attempts is set using the CSA0 to CSA4 bits in the FLXAnFRSUCC1 register.

Set the number of Cold_start attempts for Cold_start nodes to 2 or more, and ensure that the same value is configured for all Cold_start nodes in the cluster.

A Startup node can start the Startup process as a Cold_start node when the CSI bit in the FLXAnFRCCSV register is “0”.
This bit is cleared (“0”) by executing the CHI ALLOW_COLDSTART command (CMD bits in the FLXAnFRSUCC1 register = “1001”).

However, if communication is already in progress, starting the Startup process simultaneously when a Cold_start node transitions to the Startup state may interfere with the ongoing communication.
To avoid this, the Cold_start node must transition to the Startup state with the CSI bit set to “1” (Startup disabled).

In this case, after executing the CHI RUN command (transitioning to the Startup state), the Cold_start node enters the INTEGRATION_LISTEN state.

If no communication activity is detected, execute the CHI ALLOW_COLDSTART command (CMD bits in the FLXAnFRSUCC1 register = “1001”) to clear the CSI bit.
This transitions the node to the COLDSTART_LISTEN state.

【Note】 If the Cold_start fails, read the status and error registers.
When restarting the Startup process, ensure that the Cold_start is disabled using the CSI bit in the FLXAnFRCCSV register to prevent interference with ongoing communication.

The control procedure example for the Cold_start node is shown in Figure 3-12.

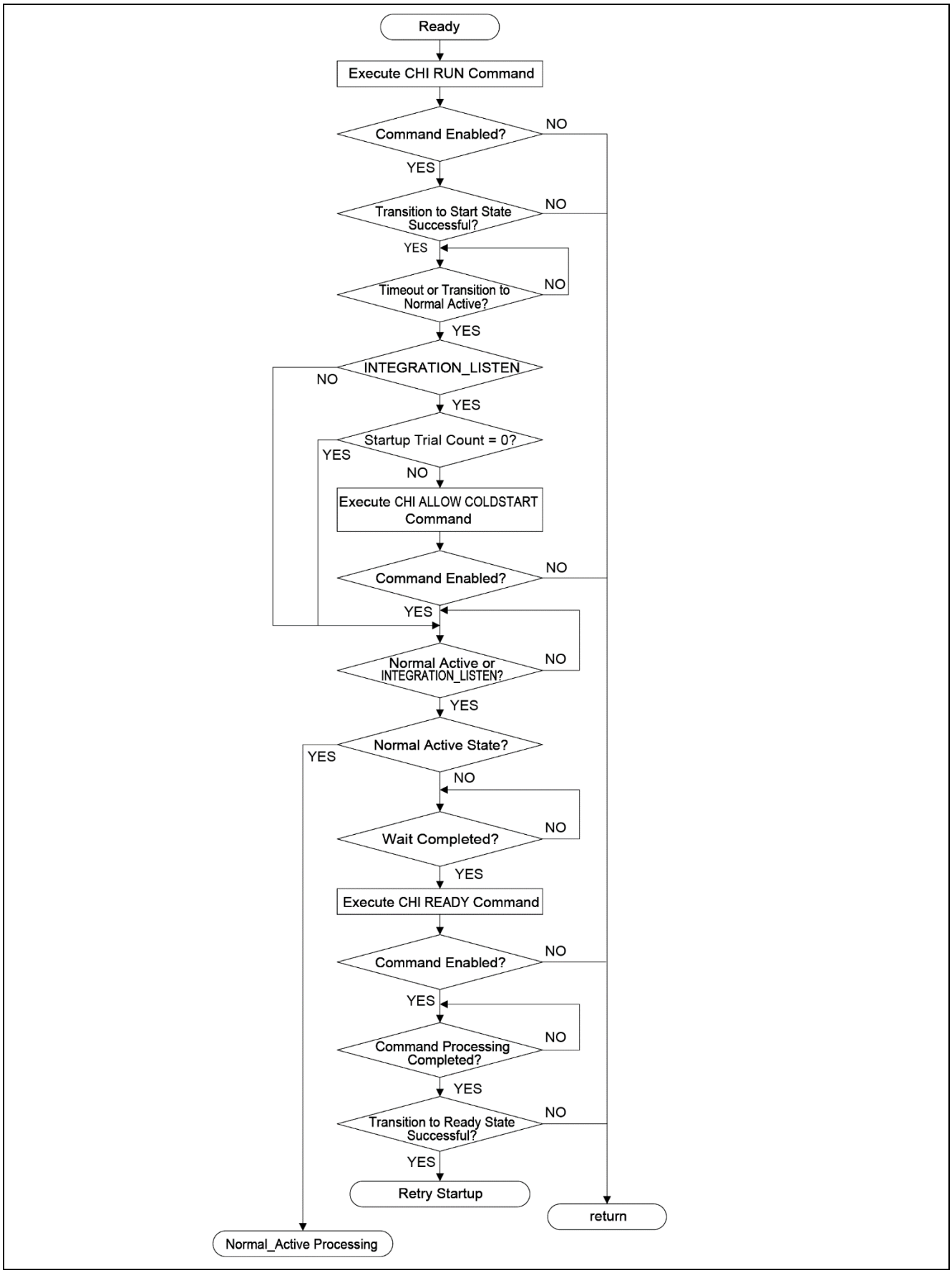


Figure 3-11 Example Cold-start Node Control Procedure

■Leading Cold_start Node Path

If no communication is detected, the node transitions to the COLDSTART_COLLISION_RESOLUTION state and transmits a CAS symbol. After the CAS symbol transmission, cycle 0 starts.

Although multiple Cold_start nodes may begin Startup simultaneously, the FlexRay protocol avoids this situation within the first four cycles.

If a Cold_start node receives a CAS symbol or a frame header during the first four cycles (cycles 0 to 3), it immediately returns to the COLDSTART_LISTEN state. Consequently, only one node remains in the Leading Cold_start node sequence.

At cycle 4, Cold_start nodes other than the Leading Cold_start node start transmitting frames.

The Leading Cold_start node transitions to the COLDSTART_CONSISTENCY_CHECK state after the COLDSTART_COLLISION_RESOLUTION state in four cycles. During cycles 4 and 5, it adjusts its clock by receiving Startup frames from other Cold_start nodes.

When the clock adjustment succeeds and the node successfully receives Startup frames from at least one other Cold_start node for two cycles, it transitions to the Normal_Active state.

The number of cold startup retries that a cold_start node can perform is set by CSA0 to CSA4 of the FLXAnFRSUCC1 register.
The remaining retry count can be checked using the RCA0 to RCA4 bits in the FLXAnFRCCSV register.
The values of RCA0 to RCA4 decrease by one for each retry.

A node can transition to the COLDSTART_COLLISION_RESOLUTION state only when this value is greater than “0”, and to the COLDSTART_LISTEN state only when it is greater than “1”.
When the retry count is set to “1”, the node is prohibited from initiating a Cold Start by itself, but it can still integrate with other nodes.

【Note】 Set the CSA0 to CSA4 bits to a value of “2” or more, and use the same value for all nodes in the cluster.

■Following Cold_start Node Path

The Following Cold_start node transitions to the COLDSTART_LISTEN state, receives valid Startup frames from the Leading Cold_start node for two cycles, and obtains schedule and clock correction information.

When it receives the first valid Startup frame, it immediately transitions to the INITIALIZE_SCHEDULE state.

After successfully receiving the second valid Startup frame and completing clock synchronization and schedule information acquisition, it transitions to the INTEGRATION_COLDSTART_CHECK state.

During the two cycles in the INTEGRATION_COLDSTART_CHECK state, the node receives all Sync frames from other nodes and performs clock correction.
After completing the clock correction without any errors and confirming that the Leading Cold_start node is active, it transitions to the COLDSTART_JOIN state.

In the COLDSTART_JOIN state, the node starts transmitting Startup frames.

In this state, the Leading Cold_start node and the Following Cold_start node mutually check their schedules. If a clock correction error is detected, the Startup process is aborted.

When at least one Startup frame is received in every even-numbered cycle, and at least one pair of Startup frames is received in every two consecutive cycles, the node transitions to the Normal_Active state.

As a result, the Following Cold_start node exits the Startup state at least one cycle later than the Leading Cold_start node.

3.4.2 Operation of Non Cold Start Node

Nodes other than Cold_start nodes integrate their clocks into those of the Cold_start nodes as Non-Cold_start nodes.

When the CHI RUN command (CMD bits in the FLXAnFRSUCC1 register = “0100”) is executed in the Ready state (POCS bits in the FLXAnFRCCSV register = “000001”), the node transitions to the Startup state.

Figure 3-13 shows an example Non-Cold_start node control procedure.。

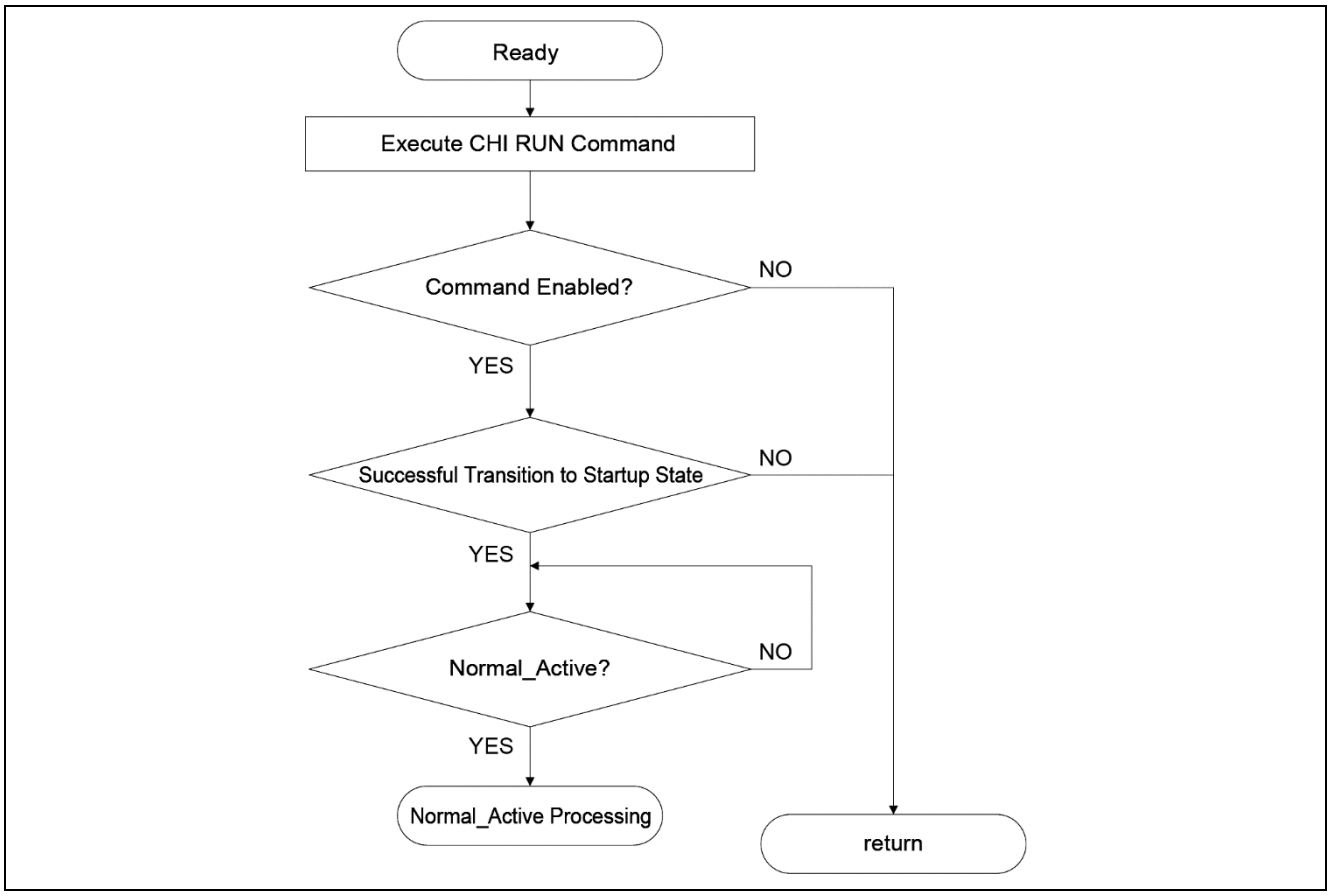


Figure 3-12 Example Non Cold-start Node Control Procedure

After executing the CHI RUN command (CMD bits in the FLXAnFRSUCC1 register = “0100”), the Non-Cold_start node transitions to the INTEGRATION_LISTEN state and monitors the status of the connected channel for a certain period.

When a valid Startup frame is received, the node transitions to the INITIALIZE_SCHEDULE state. After successfully synchronizing the clock with the second valid Startup frame and obtaining schedule information, it transitions to the INTEGRATION_CONSISTENCY_CHECK state.

In the INTEGRATION_CONSISTENCY_CHECK state, the node confirms that clock synchronization is correctly performed and that at least two Cold_start nodes are transmitting Startup frames containing the same schedule information as its own. If any error is detected, the process is aborted.

In this state, the Non-Cold_start node must receive two valid Startup frames, or Startup frames from the nodes it has integrated with, during the even-numbered cycles.
If reception fails, the node aborts the integration process.

In addition, during the first two cycles, the node must receive a pair of two valid Startup frames, or a pair of Startup frames from the nodes it has integrated with. If reception fails, the node aborts the integration process.

If two or more Startup frames cannot be received within the even-numbered cycles after the first two cycles, or if a pair of two valid Startup frames cannot be received within two cycles, the node aborts the integration process.

In this state, the node receives two valid Startup frames over two consecutive even-odd cycle pairs and then transitions to the Normal_active state.

As a result, the node exits the Startup state at the end of an odd-numbered cycle, which occurs at least two cycles after the Leading Cold_start node.

4. Data Setting to Message RAM and Data Reading from Message RAM

4.1 Structure of Transmit Frame Data

Set the message buffer header section (transmit/receive filtering conditions) using the FLXAnFRWRHS1–FLXAnFRWRHS3 registers.

Set the data section using the FLXAnFRWRDS1–FLXAnFRWRDS64 registers.

The data set in the FLXAnFRWRHS1–FLXAnFRWRHS3 and FLXAnFRWRDS1–FLXAnFRWRDS64 registers is transferred to the MessageRAM (message buffer) via the IBF (Input Buffer).

An example of the header section configuration is shown in Table 4-1, and an example of the control procedure is shown in Figure 4-1.

Table 4-1 Example Header Section Setting

Name	Setting Values	Description
Frame ID	1	
Cycle Filtering	Every communication cycle	
Transmit/Receive channel	Ch. A/B both	
Transmit/Receive configuration	Transmit	
Network management flag	Not used	
Transmit mode	Continuous transmit mode	
Transmit/Receive completion interrupt	Not used	
Header CRC	0xXXXX	
Data length	16 bytes	
Data pointer	Data section start address H'400	

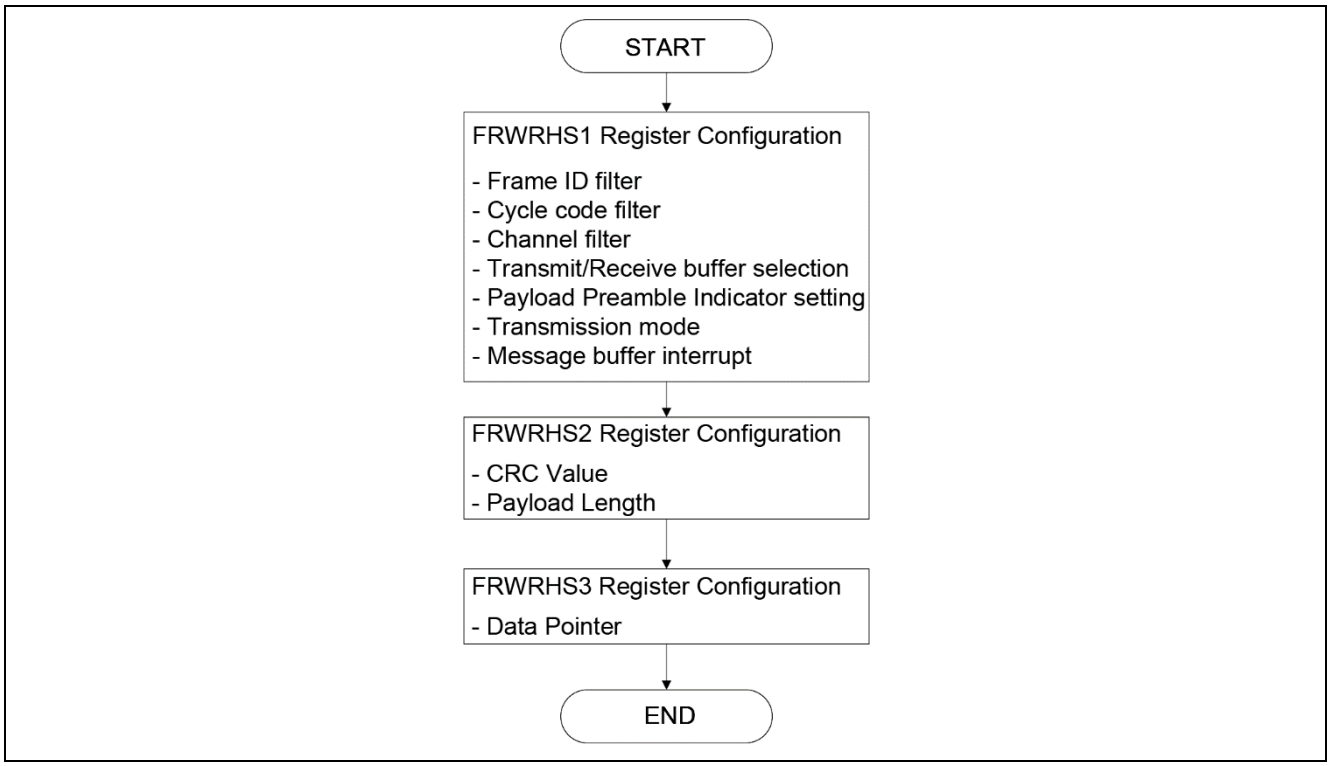


Figure 4-1 Example Control Procedure

■Header CRC

In FlexRay, a Header CRC is introduced to detect data corruption in the header section. The Header CRC value must be calculated from the values of the SyncFrameIndicator, StartupFrameIndicator, FrameID, and PayloadLength in the header section.

【Note】 According to the FlexRay specification, the communication controller does not calculate the Header CRC when transmitting a frame. The Header CRC must be pre-calculated by a configuration tool, host MCU, or similar, and set in the header section.

4.2 IBF (Input Buffer): Transfer to Message RAM

The FlexRay module writes data to the Message RAM via the IBF to ensure consistency of the transmit data. In addition, to improve data transfer performance, a dual-structure IBF consisting of a host side (CPU side) and a shadow side (Message RAM side) is implemented.

4.2.1 Transfer Method

(1) Set the data section and header section in the IBF (host side).

(FLXAnFRWRDS1 to FLXAnFRWRDS64 registers, FLXAnFRWRHS1 to FLXAnFRWRHS3 registers, and FLXAnFRIBCM register)

(2) Set the buffer number of the transfer destination in the IBRH bit of the FLXAnFRIBCR register.

(3) The write operation in (2) serves as a transfer request, causing the IBF host and IBF shadow to switch, and the transfer between the shadow side IBF and the Message RAM starts.

At this time, the IBSYS bit becomes “1,” indicating the busy state.

(4) When there is data to be transferred next, set the data section and header section again in the IBF (host side).

At this time, if the IBSYS bit is “1,” the IBSYH bit becomes “1.”

(5) When the transfer is completed, the IBF host and IBF shadow switch again, and the next pending transfer starts.

At this time, the IBSYS bit remains “1,” and the IBSYH bit becomes “0.”

【Note】 If the IBRH bit is set for the next transfer request while both the IBSYS bit and IBSYH bit are “1,” an error occurs and the IIBA bit in the FLXAnFREIR register becomes “1.”
In this case, the value in the IBF does not change.

The double-buffer structure of the input buffer is shown in Figure 4-2.

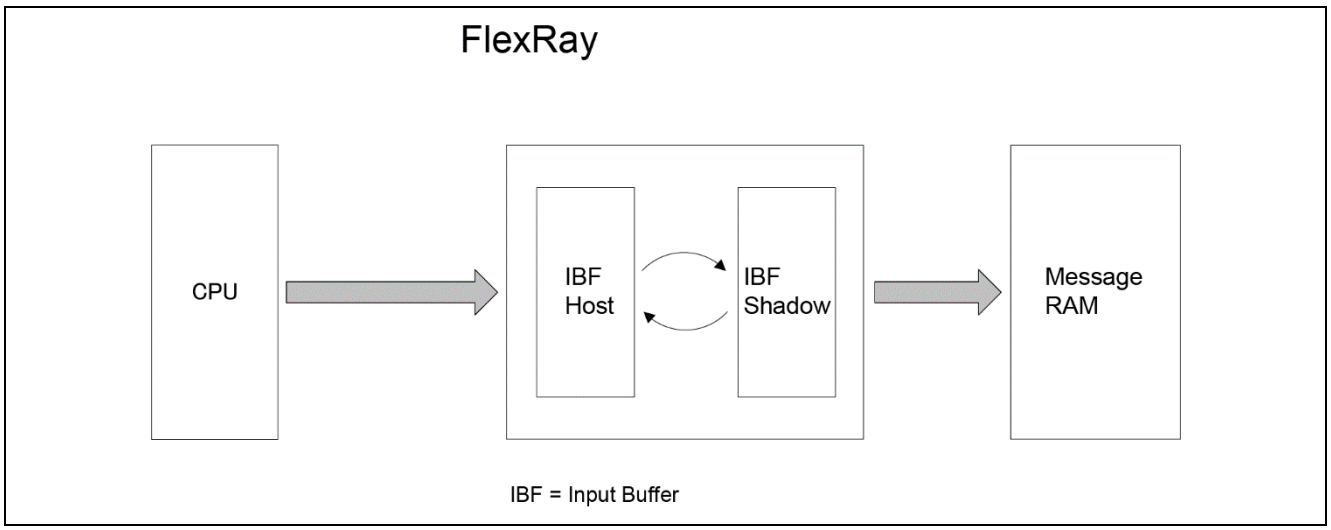


Figure 4-1 Double-Buffer Structure of the Input Buffer

The example of setting for transferring transmit frame data to Message Buffer 1 in the Message RAM is shown in Table 4-2, and the example control procedure is shown in Figure 4-3.

Table 4-1 Example Setting of Transmit Frame Data

Name	Setting Value	Description
Header Section	Transfer	
Data Section	Transfer	
Transmit Request	Transmit Request ON	
Destination Buffer Number	1	Since buffer numbers start from “0”, data is transferred to the second buffer.

【Note】 1. Do not transfer the header section to the buffers locked by the SEC0 to SEC1 bits in the FLXAnFRMRC register.
2. FIFO buffers are for receive-only. Do not transfer the data section to them.

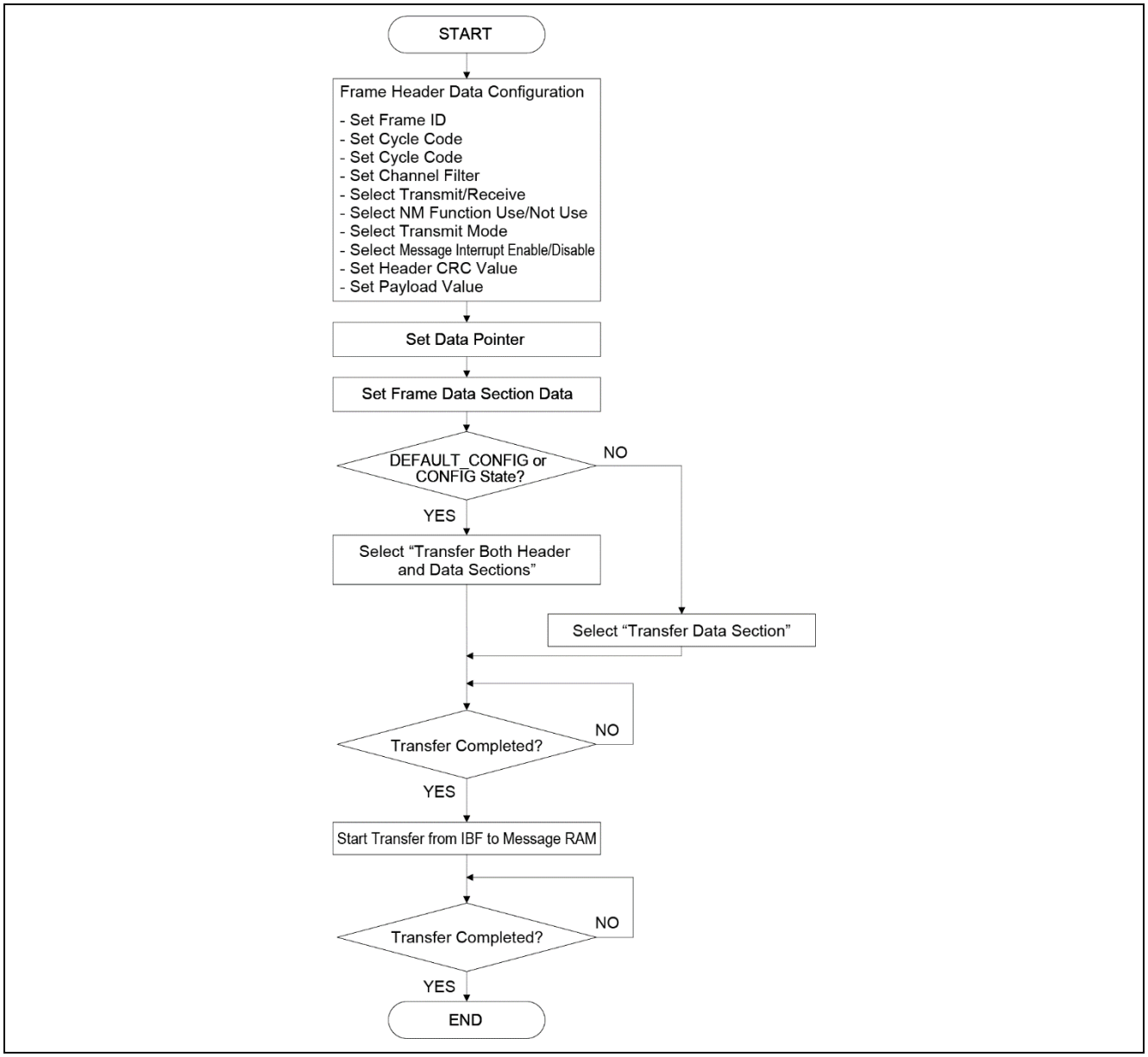


Figure 4-2 Example Control Procedure

4.3 OBF (Output Buffer): Read from Message RAM

The FlexRay module reads the received data from the Message RAM via the OBF to ensure data consistency.

A dual structure consisting of a host side (CPU side) and a shadow side (Message RAM side) is adopted.

The header section of the message buffer read from the Message RAM is stored in the FLXAnFRRDHS1 to FLXAnFRRDHS3 registers, and the data section is stored in the FLXAnFRRDDS1 to FLXAnFRRDDS64 registers.

4.3.1 Transfer Method

(1) Set the buffer number in the Message RAM to the OBRS bit in the FLXAnFROBCR register.

(2) After confirming that the OBSYS bit is “0”, set the REQ bit to “1”.

This operation starts the transfer from the message buffer to the OBF.

(3) When the transfer is completed, the OBSYS bit is automatically cleared to “0”.

(4) After confirming that the OBSYS bit is “0”, set the VIEW bit to “1”.

This operation switches the OBF host and OBF shadow of the OBF buffer.。

(5) Read the data from the OBF buffer.

【Note】 1. If the REQ bit is set to “1” while the OBSYS bit is “1”, an error occurs, and the IOBA bit in the FLXAnFREIR register is set to “1”.

2. When transferring data from a FIFO buffer, set the OBRS bit to the first FIFO buffer number.

The double-buffer structure of the output buffer is shown in Figure 4-4.

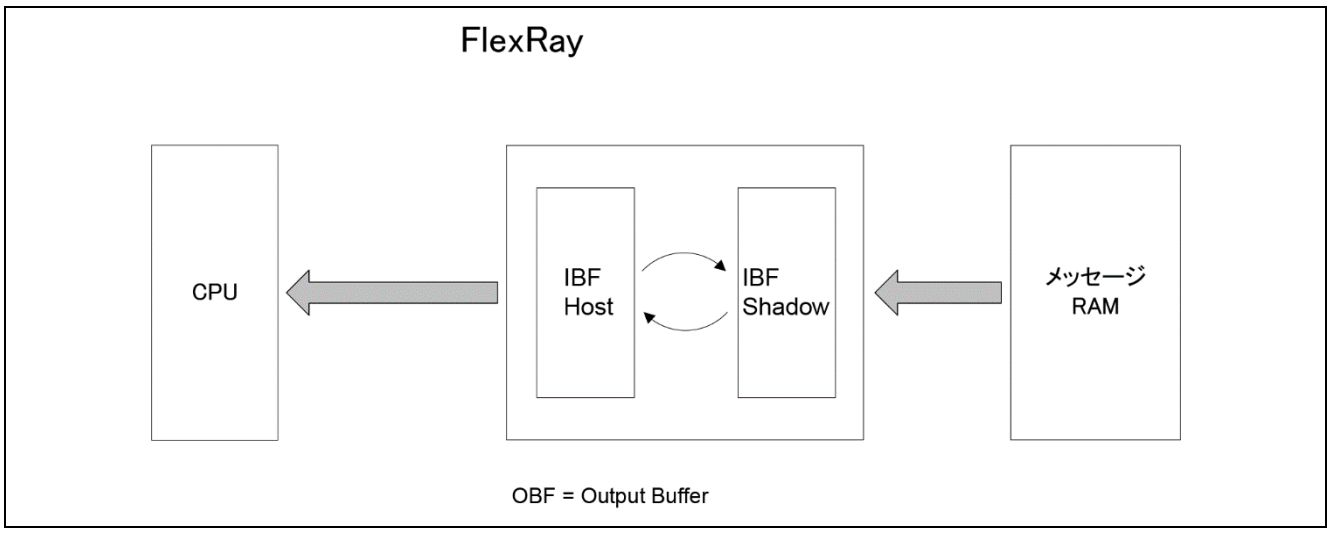


Figure 4-3 Double-Buffer Structure of Output Buffer

The example setting for reading a frame received in Message RAM Buffer 1 to the OBF is shown in Table 4-3, and the example control procedure is shown in Figure 4-5.

Table 4-2 Example Setting for Reading Received Frames

Name	Setting Value	Description
Header Section	Transfer	
Data Section	Transfer	
Destination Buffer Number	1	

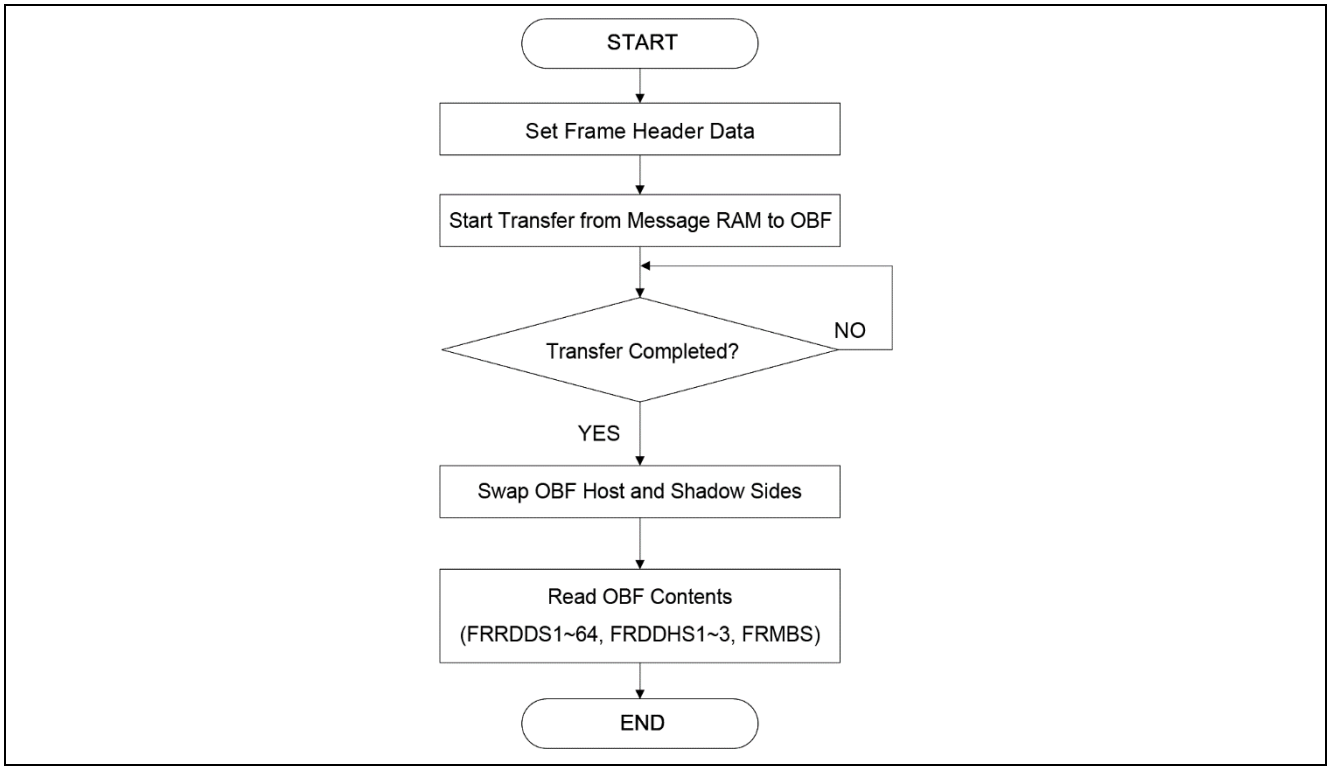


Figure 4-4 Example Control Procedure

5. Frame Transmission and Reception

5.1 Frame Transmission

FlexRay communication is based on the TDMA method. Therefore, users do not need to configure the detailed timing of frame transmission. The communication controller controls the frame transmission timing according to the pre-configured communication schedule.

The flow from transmit data preparation to frame transmission is shown below.

(1) Set the frame to be transmitted in the IBF. (At this time, set the CFG bit in the FLXAnFRWRHS1 register to “1”.)

(2) Transfer the transmit frame data from the IBF to the Message RAM.

If the STXRH bit in the header section is “1”, the transmit request flag in the corresponding FLXAnFRTXRQ1 to 4 register is set to “1”, and the buffer enters the transmit pending state.

(3) Frame transmission starts at the Action Point of the corresponding slot.

(4) If the transmit completion interrupt is enabled (the MBI bit in the FLXAnFRWRHS1 register is set to “1”), an interrupt request occurs (the TXI bit in the FLXAnFRSIR register is set to “1”) upon frame transmission completion.

At this time, the message buffer number that successfully transmitted the frame is stored in the MBT bit of the FLXAnFRMHDS register.

【Note】 The transfer of transmit frame data from the IBF buffer to the Message RAM must start before the beginning of the slot in which the frame is transmitted. (For example, to transmit a frame in slot “5”, the transfer from the IBF buffer to the Message RAM must start before the beginning of slot “5”.)

■ Continuous Transmission Mode and Single-Shot Mode

The transmit mode is selected using the TXM bit in the FLXAnFRWRHS1 register.

When the single-shot mode is selected, the TXR flag is cleared to “0” upon transmission completion.

When the continuous transmission mode is selected, the TXR flag remains “1” even after transmission completion.

■ Padding Function

In the static segment, when the payload length set in the message buffer header section is shorter than the value set in the SFDL bit of the FLXAnFRMHDC register, padding data is appended.
The padding value is “0”.

5.2 Frame Reception

In FlexRay communication, data transmission and reception are performed based on the TDMA method. Therefore, the communication controller not only checks whether the format of the received frame is valid, but also verifies whether the frame was received at the correct timing and whether various flags in the frame are valid.

The internal operation flow of the communication controller from data detection on the bus to storing the received frame is shown below.

(1) Check whether the frame format is valid.
If a CRC error or other format error is detected, it is regarded as a Syntax Error.

(2) Check whether the header contents are valid.
If an abnormal cycle counter value or other inconsistency is detected,
it is regarded as a Content Error.

(3) Check whether the conditions for storing the frame in the receive buffer are satisfied.

(4) Check whether the conditions for storing the frame in the receive FIFO are satisfied.

Among the frames that satisfy the reception conditions defined in the protocol, only those that meet the filtering conditions (Frame ID, Channel ID, and Cycle Counter) set for the receive buffer or receive FIFO are stored in the receive buffer.

When a frame is stored in the receive buffer, the corresponding flags in the FLXAnFRNDAT1–4 registers and the FLXAnFRMBSC1–4 registers are set to “1”.

At this time, the message buffer number in which the frame was successfully received is stored in the MBU bit of the FLXAnFRMHDS register.

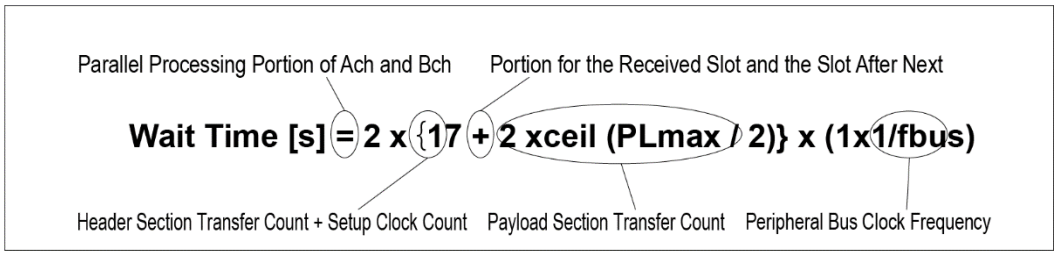
If the receive completion interrupt is enabled (the MBI bit in the FLXAnFRWRHS1 register is set to “1”) when configuring the header, an interrupt request (the RXI bit in the FLXAnFRSIR register is set to “1”) occurs upon frame reception completion.

【Note】 Each flag in the FLXAnFRNDAT1–FLXAnFRNDAT4 registers is automatically cleared when the data section is transferred from the Message RAM to the OBF. If only the header section of a frame is transferred to the OBF, the receive completion flags in the FLXAnFRNDAT1–FLXAnFRNDAT4 registers are not cleared.

■ Timing for Reading Received Messages

A received message can be reliably read by waiting for the worst-case time calculated by the following formula from the beginning of the slot immediately after the message is received.

Formula:



PLmax: Maximum Payload Length (word)

ceil(x): Minimum Integer Value Exceeding x

Examples of calculating the waiting time are shown below.

(1) When the payload size is 16 words (32 bytes) and the peripheral bus clock is 80 MHz:

Waiting time = 2 x {17 + 2 x ceil(16 / 2)} x {1 / (80 x 106)}
= 2 x (17 + 2 x 8) x {1 / (80 x 106)}
= 82.5 [nsec]

(2) When the payload size is 128 words (254 bytes) and the peripheral bus clock is 80 MHz:

Waiting time = 2 x {17 + 2 x ceil(128 / 2)} x {1 / (80 x 106)}
= 2 x (17 + 2 x 64) x {1 / (80 x 106)}
= 3.6 [usec]

6. Interrupts

6.1 Interrupt Control

FlexRay-related interrupts of RH850/P1x are shown in Table 6-1.
For details on interrupts, refer to RH850/P1x Hardware Manual, Chapter 6: Interrupts.

Table 6-1 FlexRay-Related Interrupts

Interrupt Sources	EIINT Interrupt Channel Number	Offset Address (Table Reference Method)
FlexRay 0 Interrupt	509	+3FAH
FlexRay 1 Interrupt	510	+3FCH
Timer 0 Interrupt	511	+3FEH
Timer 1 Interrupt	512	+400H
Timer 2 Interrupt	513	+402H
FIFO Transfer Interrupt	514	+404H
FIFO Transfer Warning Interrupt	515	+406H
Output Transfer Warning Interrupt	516	+408H
Output Transfer Complete Interrupt	517	+40AH
Input Queue Full Interrupt	518	+40CH
Input Queue Empty Interrupt	519	+40EH

6.2 FlexRay0 Interrupt and FlexRay1 Interrupt

FlexRay 0 and FlexRay 1 interrupts can be triggered under the following conditions:

- Frame transmit/receive completion
- Detection of communication errors
- Change of various status condition
- Timer counter reaches the specified value
- Transfer completion between IBF/OBF and Message RAM

Each interrupt can be enabled or disabled via the FLXAnFREIES (Set) / FLXAnFREIER (Reset) registers and the FLXAnFRSIES (Set) / FLXAnFRSIER (Reset) registers. Setting the corresponding bit of the Set register to “1” enables the interrupt. Setting the corresponding bit of the Reset register to “1” disables the interrupt (writing “0” has no effect on the register value).

The status of each interrupt is reflected in the FLXAnFREIR and FLXAnFRSIR registers regardless of whether the interrupt is enabled or disabled. To clear a status bit, write “1” to the corresponding bit (writing “0” has no effect on the register value).

Each interrupt can be assigned to either the FlexRay 0 interrupt or FlexRay 1 interrupt using the interrupt output selection registers (FLXAnFREILS / FLXAnFRSILS).

Interrupt setting examples are shown in Table 6-2, and the control procedure example when using the Channel A Error Detection Interrupt is shown in Figure 6-1.

Table 6-2 Example of Interrupts, Request Flags, and Interrupt Assignment

Interrupt Name	Request Flag	Interrupt Assignment
Receive Interrupt	FLXAnFRSIR_RXI	FlexRay 1 Interrupt
POC Error Mode Change	FLXAnFREIR_PEMC	FlexRay 0 Interrupt
Channel A Error Detection	FLXAnFREIR_EDA	FlexRay 0 Interrupt
Channel B Error Detection	FLXAnFREIR_EDB	FlexRay 0 Interrupt

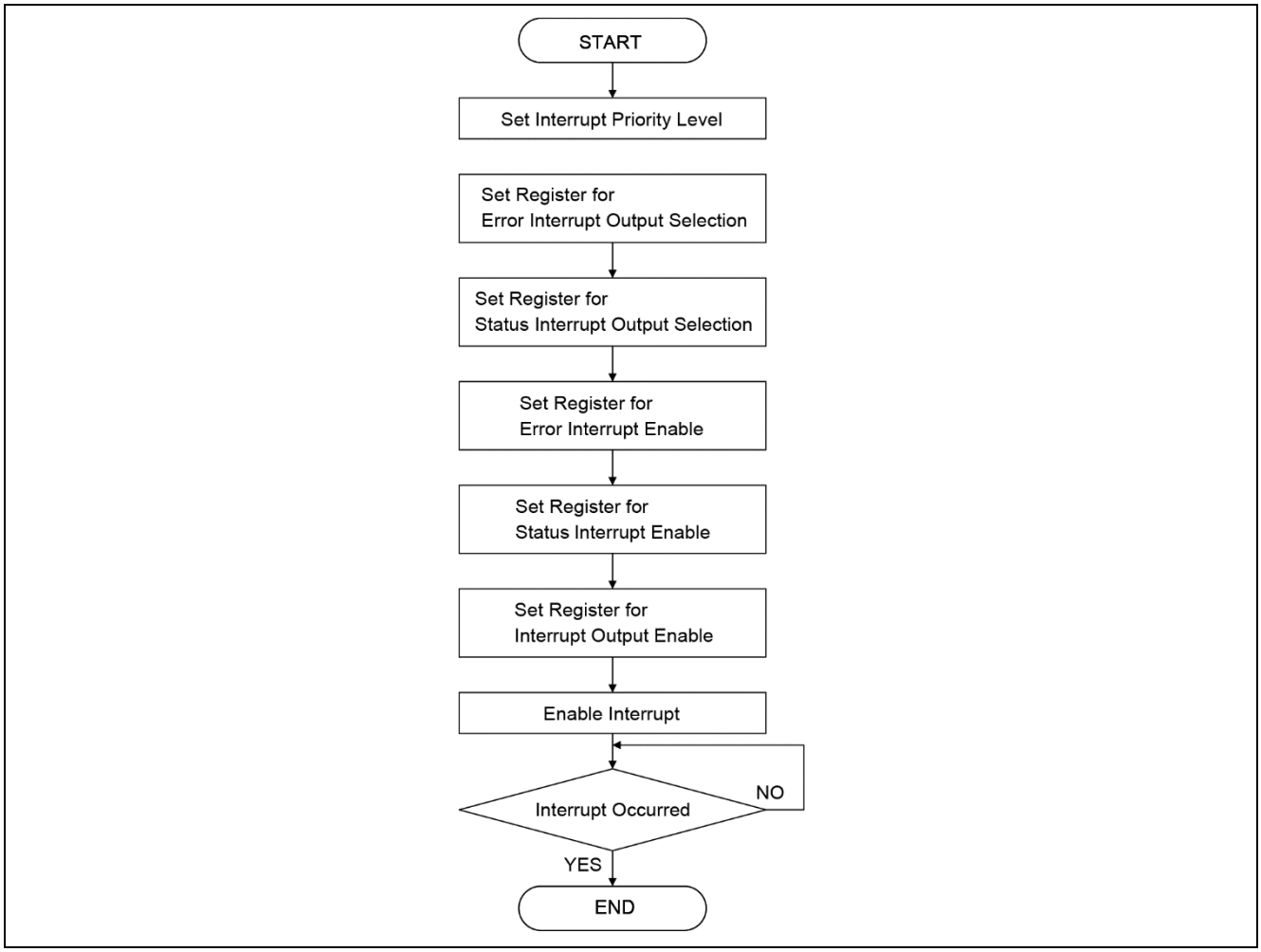


Figure 6-1 Example Control Procedure of Channel A Error Detection Interrupt

【Note】 When using the Receive Interrupt, set the MBI bit of the corresponding message buffer to “1”.

6.3 Clearing Interrupt Requests

In the FlexRay module, interrupt requests can be cleared by writing “1” to the corresponding bits of the interrupt status registers (FLXAnFREIR, FLXAnFRSIR).

When clearing interrupt requests, write a value with only the relevant bit set to “1” (immediate value) to the register. Using bitfield structures or logical operations to clear interrupt request bits may accidentally clear other interrupt requests.

7. Timer

The FlexRay module has the following two built-in timers.

- Timer 0
- Timer 1

7.1 Timer 0

It is an absolute timer. The interrupt generation timing is set using the cycle count value and the offset value (MT value) from the start of the cycle. Timer 0 starts when “1” is set to the T0RC bit in the FLXAnFRT0C register. The interrupt generation timing is set using the T0CC bit and the T0MO bit in the FLXAnFRT0C register.

【Note】 1. When setting timer values in the FLXAnFRT0C register, set the T0RC bit to “0” in advance to stop Timer 0.
2. Timer 0 operates only in the NORMAL_ACTIVE state. When the module transitions to any other state, Timer 0 stops.

The example settings for Timer 0 are shown in Table 7-1, and the example control procedure is shown in Figure 7-1.

Table 7-1 Example Settings of Timer 0

Name	Setting	Description
Cycle Counter Value	Every two Communication Cycles	Generates an interrupt in every even-numbered communication cycle
MT Offset Value	1,000 MT	Generates an interrupt 1,000 MT after the start of the communication cycle
Timer Operation Mode	Continuous Mode	Generates an interrupt request every time the specified condition is met

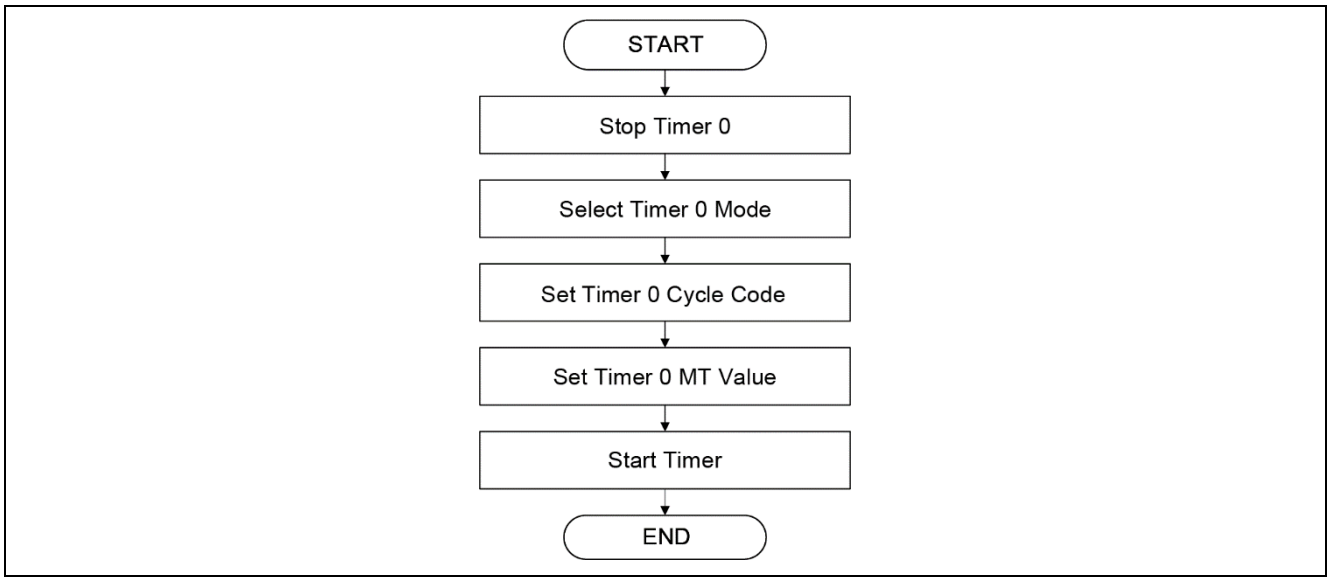


Figure 7-1 Example Control Procedure

7.2 Timer 1

It is a relative timer. The interrupt generation timing is set using the elapsed time (MT value) from the timer start. Timer 1 starts when “1” is set to the T1RC bit in the FLXAnFRT1C register. The interrupt generation timing is set using the T1MC bit in the FLXAnFRT1C register.

【Note】 1. When setting timer values in the FLXAnFRT1C register, set the T1RC bit to “0” in advance to stop Timer 1.
2. Timer 1 operates only in the NORMAL_ACTIVE state. When the module transitions to any other state, Timer 1 stops.

The example settings for Timer 1 are shown in Table 7-2, and the example control procedure is shown in Figure 7-2.

Table 7-2 Example Settings of Timer 1

Name	Setting	Description
MT Count	After 1000 MT	
Timer Operation Mode	Single-Shot Mode	Stops upon the generation of the Timer 1 interrupt

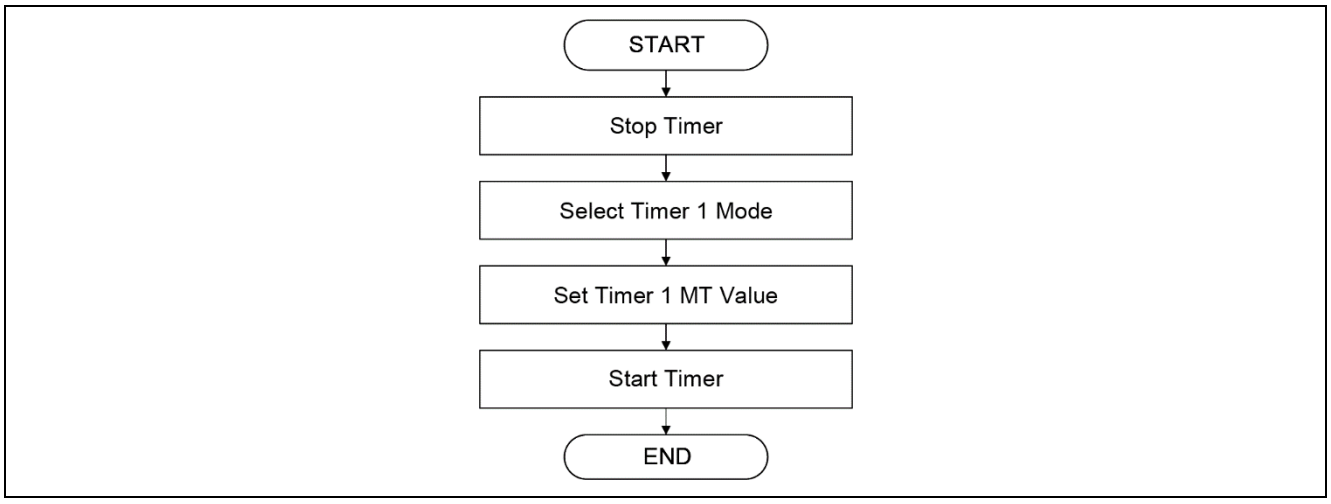


Figure 7-2 Example Control Procedure

8. ストップウォッチタイマ

The stopwatch timer captures the cycle counter value, slot counter value, and MT value when a trigger event occurs, and stores these values in the following registers:

- Cycle counter value : SCCV0–SCCV5 bits in the FLXAnFRSTPW1 register
- MT value: : SMTV0–SMTV13 bits in the FLXAnFRSTPW1 register
- Channel A slot counter value : SSCVA0–SSCVA10 bits in the FLXAnFRSTPW2 register
- Channel B slot counter value : SSCVB0–SSCVB10 bits in the FLXAnFRSTPW2 register

The trigger events of the stopwatch timer are as follows:

- Generation of a FlexRay 0 interrupt request
- Generation of a FlexRay 1 interrupt request
- Software trigger (SSWT bit = “1” in the FLXAnFRSTPW1 register)

【Note】 1. The FlexRay 0/1 interrupt trigger and software trigger cannot be enabled at the same time.
2. When the stopwatch trigger occurs, each value is captured at the time of the trigger event and at the start of the next MT following the trigger.

The example of the stopwatch trigger settings is shown in Table 8-1, and the example of the control procedure is shown in Figure 8-1.

Table 8-1 Example Settings of Stopwatch Trigger

Name	Setting	Description
FlexRay 0/1 Interrupt Trigger	Disabled	
Stopwatch Operation Mode	Single-Shot Mode	Stops upon the generation of a stopwatch interrupt
ソフトウェアトリガ	許可	

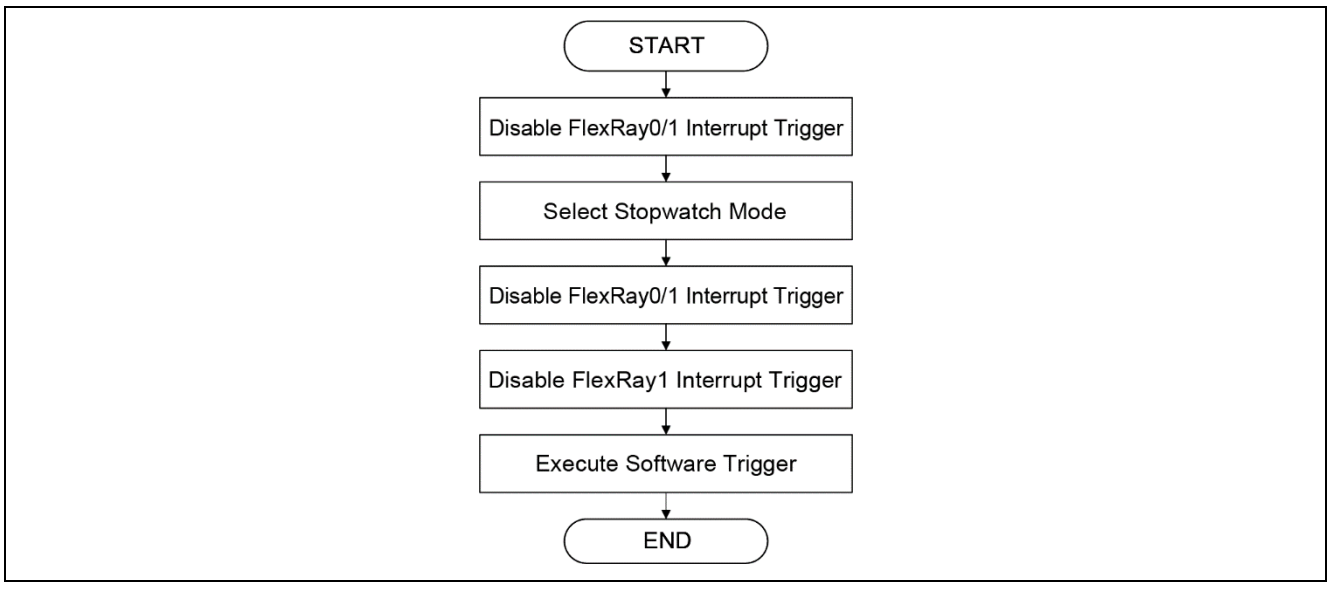


Figure 8-1 Example Control Procedure

9. Network Management Function

The FlexRay module incorporates a network management function that extracts the leading portion of frames received in the Static segment, for which the Payload Preamble Indicator (PPI) flag is set to “1,” as NM vectors.

The communication controller performs a bitwise OR of all NM vectors received within one communication cycle and stores the result as NM vector information. If the NM vector information of the current communication cycle differs from that of the previous cycle, an interrupt request is generated.

【Note】NM vector information (FLXAnFRNMV1–3 registers) is updated at the end of each communication cycle.

An example of network management function settings is shown in Table 9-1.

Table 9-1 Example Settings of Network Management Function

Name	Setting	Description
Network Management Function	Enabled	In the Sample Program, This Function Is Set as Unused
NM Vector Length	4 Bytes	Can be set from 0 to 12 bytes in 1-byte increments

Change the macro definition value NEMC_SET in the Node_1.h file as required.

10. Receive FIFO

The FlexRay module has a built-in receive FIFO function.

When configuring FIFO buffers in the Message RAM, the start buffer number of the FIFO buffer can be set using the FFB0–FFB7 bits of the FLXAnFRMRC register, and the end buffer number can be set using the LCB0–LCB7 bits of the FLXAnFRMRC register.

Up to 128 FIFO buffers can be configured. For details on Message RAM configuration, refer to 1. Initial Configuration.

If the received data does not meet the filtering conditions of the Static and Dynamic buffers but meets the filtering conditions of the FIFO buffer, it is stored in the FIFO buffer. In this case, the MBS bits of the corresponding message buffer are overwritten with the received frame's Frame ID, payload length, received cycle counter, and status.

10.1 FIFO Filtering

The filtering of FIFO buffers consists of channel filtering, Frame ID filtering, and cycle counter filtering.

10.1.1 Configuration of FIFO Rejection Filter

The reject conditions for storing received data in the FIFO buffer are set in the FLXAnFRFRF and FLXAnFRFRFM registers.

When the RSS bit of the FLXAnFRFRF register is set to "1" (default), all received messages in the Static segment are rejected by the FIFO.

When the RNF bit of the FLXAnFRFRF register is set to "1," received Null frames are not stored in the FIFO buffer.

If a Null frame is not rejected by the FIFO rejection filter, it is stored in the FIFO buffer like any other data. In this case, the corresponding bit in the NDAT register is set to "1."

The FID0–FID10 bits of the FLXAnFRFRF register are used to set the reject filtering conditions for Frame IDs. The MFID0–MFID10 bits of the FLXAnFRFRFM register determine whether the filtering conditions set in the FID0–FID10 bits of the FLXAnFRFRF register are applied or ignored, thereby specifying the actual reject Frame IDs. When an MFID0–MFID10 bit is set to "1," the corresponding FID0–FID10 bit setting is ignored.

Table 10-1 to Table 10-5 show examples of frame ID reject filter settings.

Table 10-1 Example Settings of Reject Filter 1

Bit Name	Setting Value
FLXAnFRFRF レジスタ　FID Bit	000 0000 0011b
FLXAnFRFRFM レジスタ　MFID Bit	000 0000 0000b
リジェクトフレーム ID	000 0000 0011b

“Explanation”
Frame ID = 3 is rejected.

Table 10-2 Example Settings of Reject Filter 2

Bit Name	Setting Value
FRF レジスタ　FID Bit	000 0000 0011b
FLXAnFRFRFM レジスタ　MFID Bit	000 0000 0001b
リジェクトフレーム ID	000 0000 001Xb（X は任意）

“Explanation”
Frame ID 2 and Frame ID 3 are rejected.

Table 10-3 Example Settings of Reject Filter 3

Bit Name	Setting Value
FLXAnFRFRF Register　FID Bit	000 0000 1000b
FLXAnFRFRFM Register　MFID Bit	000 0000 0111b
Reject Frame ID	000 0000 1XXXb（Don’t care ”X”）

“Explanation”
Frame ID 8 through Frame ID 15 are rejected.

Table 10-4 Example Settings of Reject Filter 4

Bit Name	Setting Value
FLXAnFRFRF Register　FID Bit	000 0000 1000b
FLXAnFRFRFM Register　MFID Bit	000 0000 0100b
Reject Frame ID	000 0000 1X00b（Don’t care ”X”）

“Explanation”
Frame ID 8 through Frame ID 12 are rejected.

Table 10-5 Example Settings of Reject Filter 5

Bit Name	Setting Value
FLXAnFRFRF Register　FID Bit	000 0000 1000b
FLXAnFRFRFM Register　MFID Bit	000 0001 1111b
Reject Frame ID	000 000X XXXXb（Don’t care ”X”）

“Explanation”
Frame ID 1 through Frame ID 31 are rejected.

10.2 Reading the FIFO Buffer

To read a frame stored in the receive FIFO, perform the “FIFO head buffer read process” for the OBF buffer.

To start the transfer trigger from the FIFO buffer to the OBF buffer in a state other than DEFAULT_CONFIG or CONFIG, set the first buffer number of the FIFO buffer to the OBRS0 to OBRS6 bits in the FLXAnFROBCR register. This write operation serves as a transfer request.

The first buffer number of the FIFO buffer can be referenced by the FFB0 to FFB7 bits in the FLXAnFRMRC register.

To read data from the FIFO buffer, perform the sequence shown below.
For details, refer to section 4.3 “OBF (Output Buffer): Reading from Message RAM.”

Table 10-6 shows an example of FIFO buffer readout settings, and an example of the control procedure is shown in Figure 10-1.

Table 10-6 Setting Example

Name	Setting	Description
Header Section Transfer	Enabled	
Data Section Transfer	Enabled	
First FIFO Buffer Number	8	

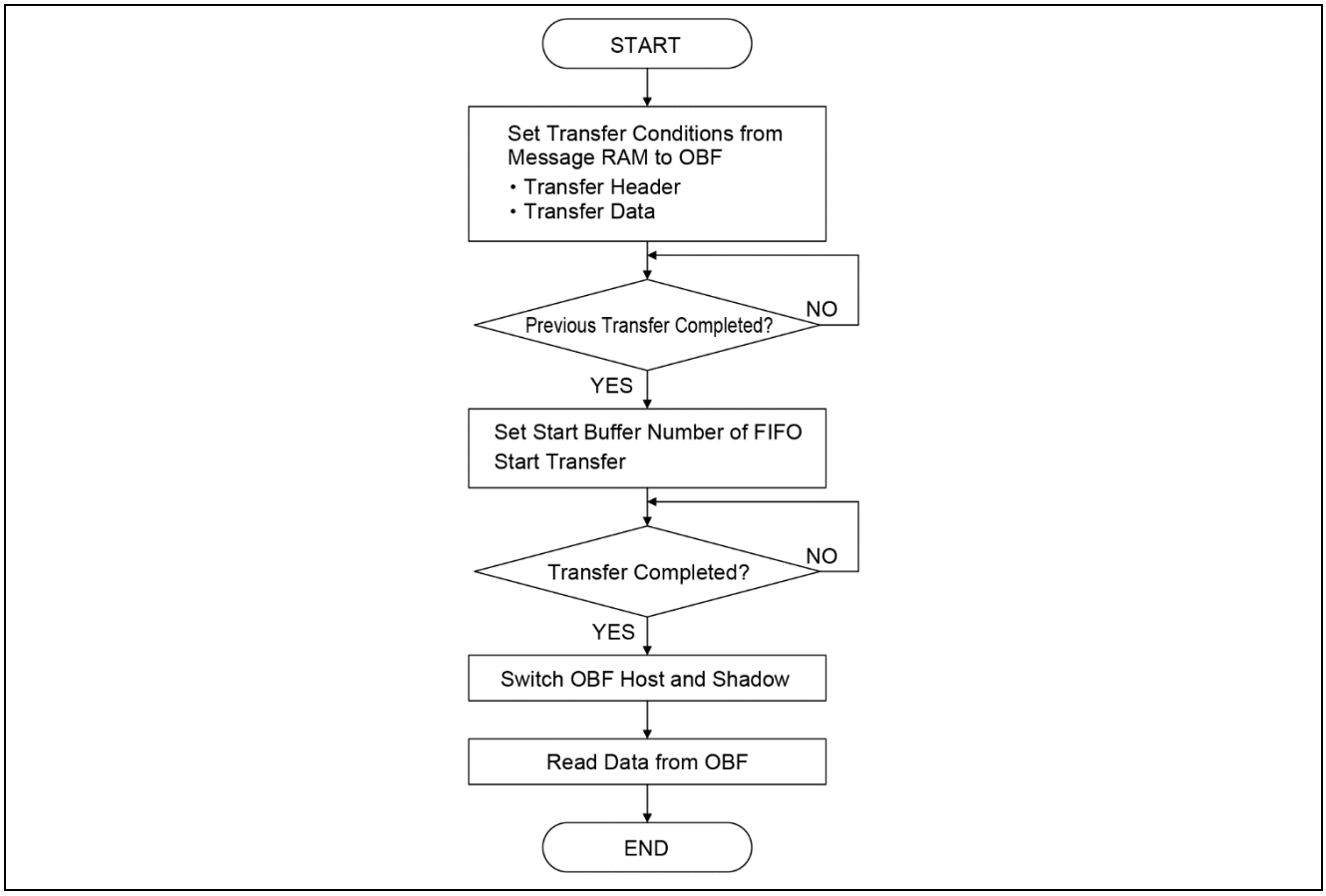


Figure 10-1 Example Control Procedure

【Note】 1. Do not perform the read process on any FIFO buffer other than the FIFO head buffer.
2. Do not perform a dummy read of the FIFO buffer.
If a dummy read is performed, the EFA (Empty FIFO Access) flag in the FLXAnFREIR register is set. Perform the read process only after confirming that data is stored in the FIFO by checking RFNE (Receive FIFO Not Empty) in the FLXAnFRSIR register or other indicators.

3. When data is read from the FIFO buffer, the FLXAnFRRDHS1 register retains only the frame ID (FID bits) of the received frame. All other bits are set to “0” at the time of reading.
4. When the data field is transferred from the FIFO buffer to the OBF buffer (RDSS bit in the FLXAnFROBCM register = “1”), the amount of data corresponding to the configured payload length (PLC0 to PLC6 bits in the FLXAnFRWRHS2 register and FLXAnFRRDHS2 register) is transferred.
   The payload length (PLR0 to PLR6 bits in the FLXAnFRRDHS2 register) of the received frame is not transferred.
   When a message is stored in the FIFO buffer, the received payload (PLR0 to PLR6 bits in the FLXAnFRRDHS2 register) and the configured payload length (PLC0 to PLC6 bits in the FLXAnFRWRHS2 register and FLXAnFRRDHS2 register) behave as follows:
   - PLR[6:0] > PLC[6:0] :
     The payload data stored in the message buffer is truncated to the configured payload length. The resulting length becomes PLC + 1.
   - PLR[6:0] ≤ PLC[6:0] :
     The received payload data is stored in the data field of the message buffer. The remaining bytes up to the number indicated by PLC are filled with indeterminate values.
   - PLR[6:0] = “0” :
     The data field of the message buffer is filled with indeterminate values.
   - PLC[6:0] = “0” :
     The message buffer does not contain a data field. No data is stored in the data field of the message buffer.

10.3 Writing to FIFO Buffer

Writing to the FLXAnFRWRHS1 to FLXAnFRWRHS3 registers for the FIFO buffer is controlled in the same manner as described in *Section 3, "Setting Data to Message RAM and Reading Data from Message RAM."*

When a message buffer is used as a FIFO buffer, the receive conditions set in the FLXAnFRWRHS1 register are ignored, and the receive conditions set in the FLXAnFRFRF and FLXAnFRFRFM registers are used instead.

An example of the write control procedure for the FIFO buffer located in message buffer 2 is shown in Figure 10-2.

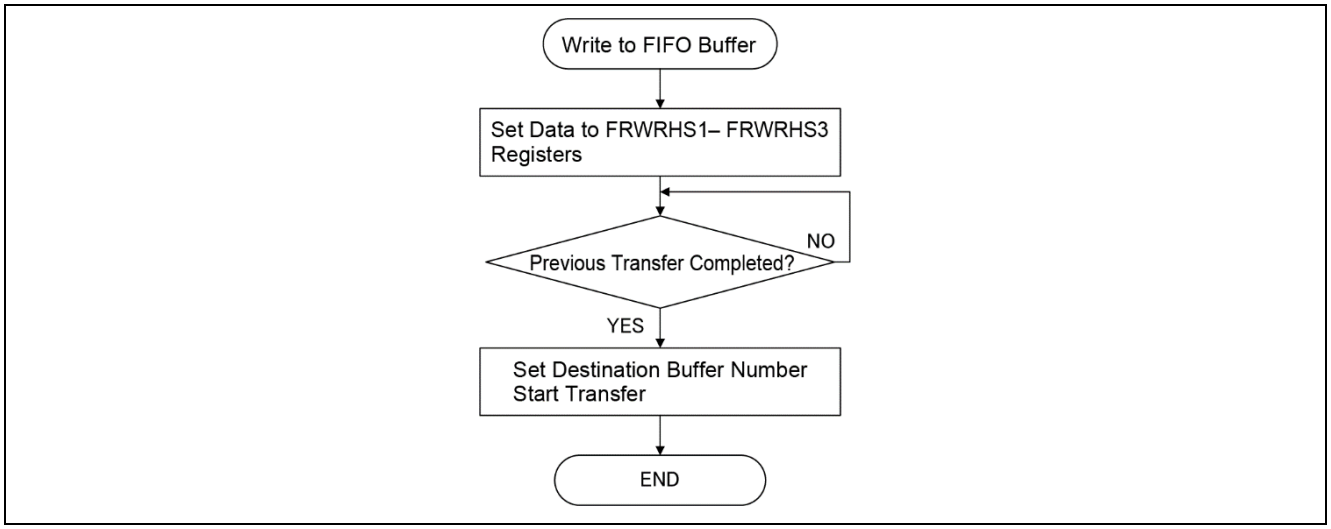


Figure 10-2 Example Control Procedure

【Note】The FIFO buffers must have the same data length (PLC0 to PLC6 bits in the FLXAnFRWRHS2 register).

12. Revision Record

Rev.	Issue date	Revised contents	
		Page	Points
1.00	Dec. 24, 2025	Pages	First edition

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

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

1. Precaution against Electrostatic Discharge (ESD)

   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on

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

   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins

   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.
5. Clock signals

   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.
6. Voltage application waveform at input pin

   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses

   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products

   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

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

Contact information

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

Trademarks

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