RL78/G10

Serial Array Unit (CSI Slave Communication) CC-RL

Introduction

This application note describes how the serial array unit (SAU) performs communication tasks using the CSI slave communication feature. The SAU is selected by the CS signal sent from the master and performs single transmission/reception, continuous transmission, continuous reception, and continuous transmission/reception operations. To ensure reliable communication, it adopts a simple protocol and handles commands and processing in response to commands. The SAU also performs handshake processing using the BUSY signal to establish synchronization with the master.

Target Device

RL78/G10

When applying the sample program covered in this application note to another microcomputer, modify the program according to the specifications for the target microcomputer and conduct an extensive evaluation of the modified program.

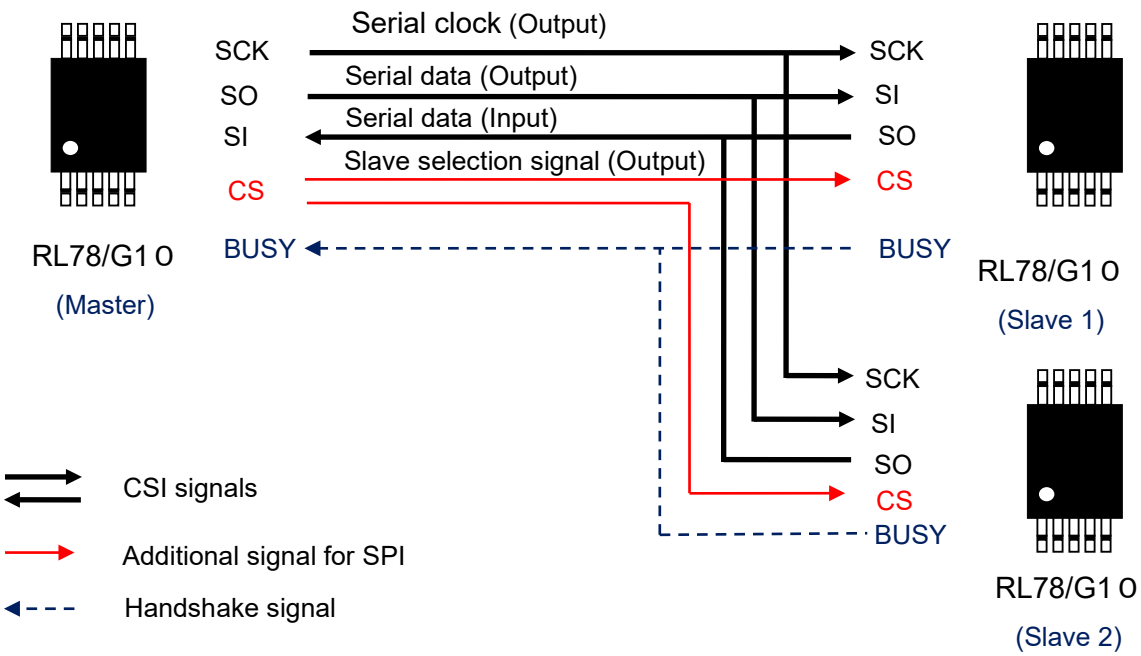
Contents

1. Specifications

The serial array unit (SAU) described in this application note performs CSI slave communication using the serial array unit (SAU). Selected by the CS signal from the master, the SAU performs single transmission/reception, continuous transmission, continuous reception, or continuous transmission/reception while performing handshaking using the BUSY signal. (Although CS is a negative logic signal, the bar that should normally appear over the signal name is omitted in this document.)

1.1 Outline of CSI Communication

CSI is a protocol for clock synchronous serial communication using three signal lines, namely, serial clock (SCK), serial input data (SI), and serial output data (SO). SPI (Serial Peripheral Interface) uses an additional signal, CS (Chip Select), which is used to select the slave device. The relationship among these signals is shown in figure 1.1.



SCK signal: Clock signal for indicating the communication timing, output by the master and input to the slaves.
SO signal: Serial output data signal. Connected to the SI signal pin of the target device.
SI signal: Serial input data signal. Connected to the SO signal pin of the target device.
CS signal: Used by the master to select the target slave device.
BUSY signal: Handshake signal that is introduced by this application note.

Figure 1.1 Outline of CSI Communication

The CSI communication master first selects the slave with which it wants to communicate with the CS signal (this is an SPI operation). The master outputs the SCK signals and place data on the SO signal line and inputs data from the SI signal line in synchronization with the SCK signals. In CSI communication, the slave needs to become ready for communication by the time the master starts communication (sending the SCK signals). In this application note, the BUSY signal is introduced as the signal for indicating the slave(RL78/G10 devices operate as the slaves) is ready for communication. Since the slave cannot catch up with the master in processing speed when it is selected by the CS signal or when the master starts communication, it controls the BUSY signal so that the master checks the signal whenever starting communication.

1.2 Outline of Communication

Communication is done in 1 ms slot units. In each slot, command transmission from the master and communication processing according to the command are processed. Figure 1.2 shows the outline of slot processing and table 1.1 lists the commands that are to be used.

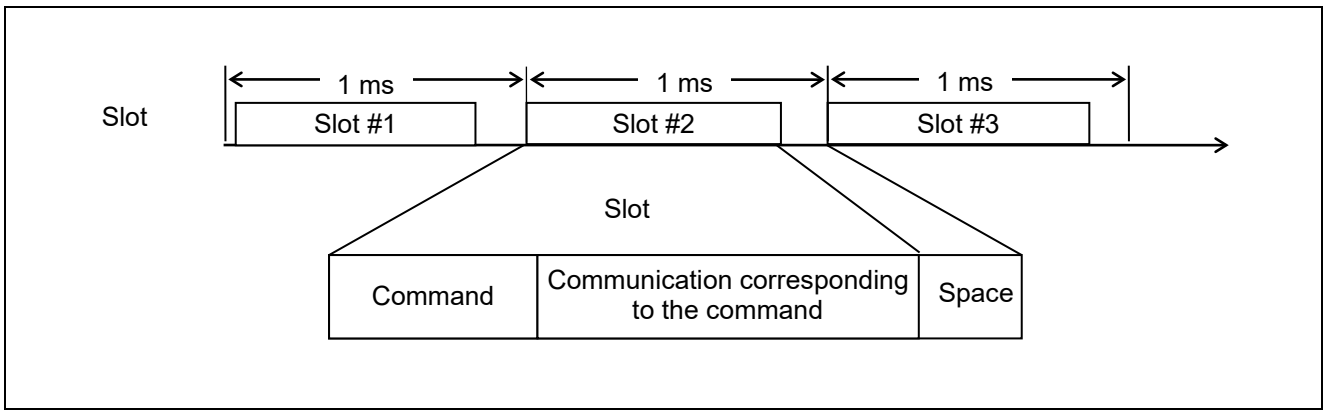


Figure 1.2 Outline of Slots

Table 1.1 Commands to be Used

Command	Operation Outline
Status check	Checks the number of data characters that the slave can transmit or receive.
Receive	Receives data from the slave in continuous mode.
Transmit	Transmits data to the slave in continuous mode.
Transmit/receive	Transmits and receives data to and from the slave in continuous mode.

The SAU as a slave returns the size of the transmit and receive buffers in response to the status check command. When the slave receives data, it takes the complement of the received data as the data to be sent next time.

The CSI channel to be used can be changed easily by editing a header file. (however, the CSI channel can only be changed in 16-pin products).

Table 1.2 lists the peripheral functions that are used and their uses. Figures 1.3 to 1.6 show the CSI communication operations. Unless specifically noted, CSIp is represented by CSI00.

Table 1.2 Peripheral Functions to Be Used and Their Uses

Peripheral Function	Use
Serial array unit m	Performs CSI master communication using the SCKp signal (clock output), SIp signal (receive data), and SOp signal (transmit data). p: 00/01 Note
External interrupt	INTP0: P137 (CS signal input)
Port	P04 (BUSY signal output)

Note: 16-pin products only

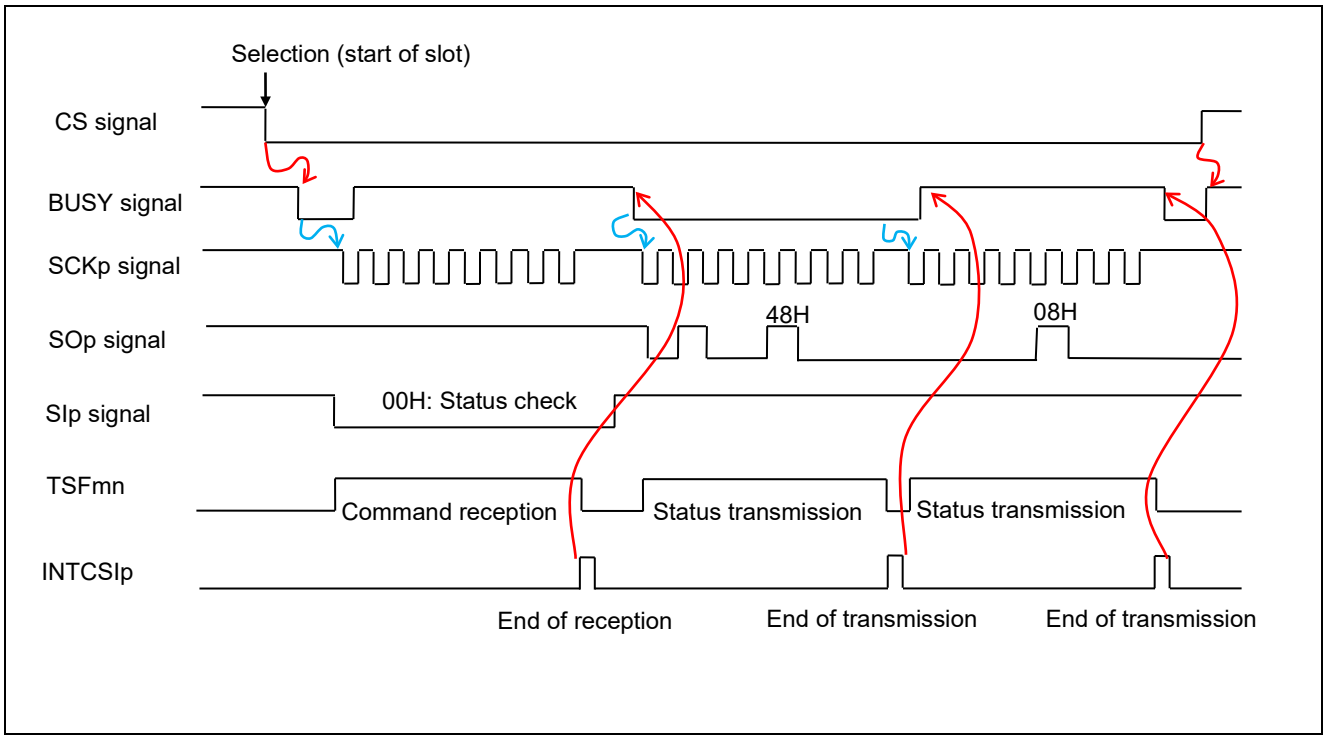


Figure 1.3 Timing Chart of Status Check Command

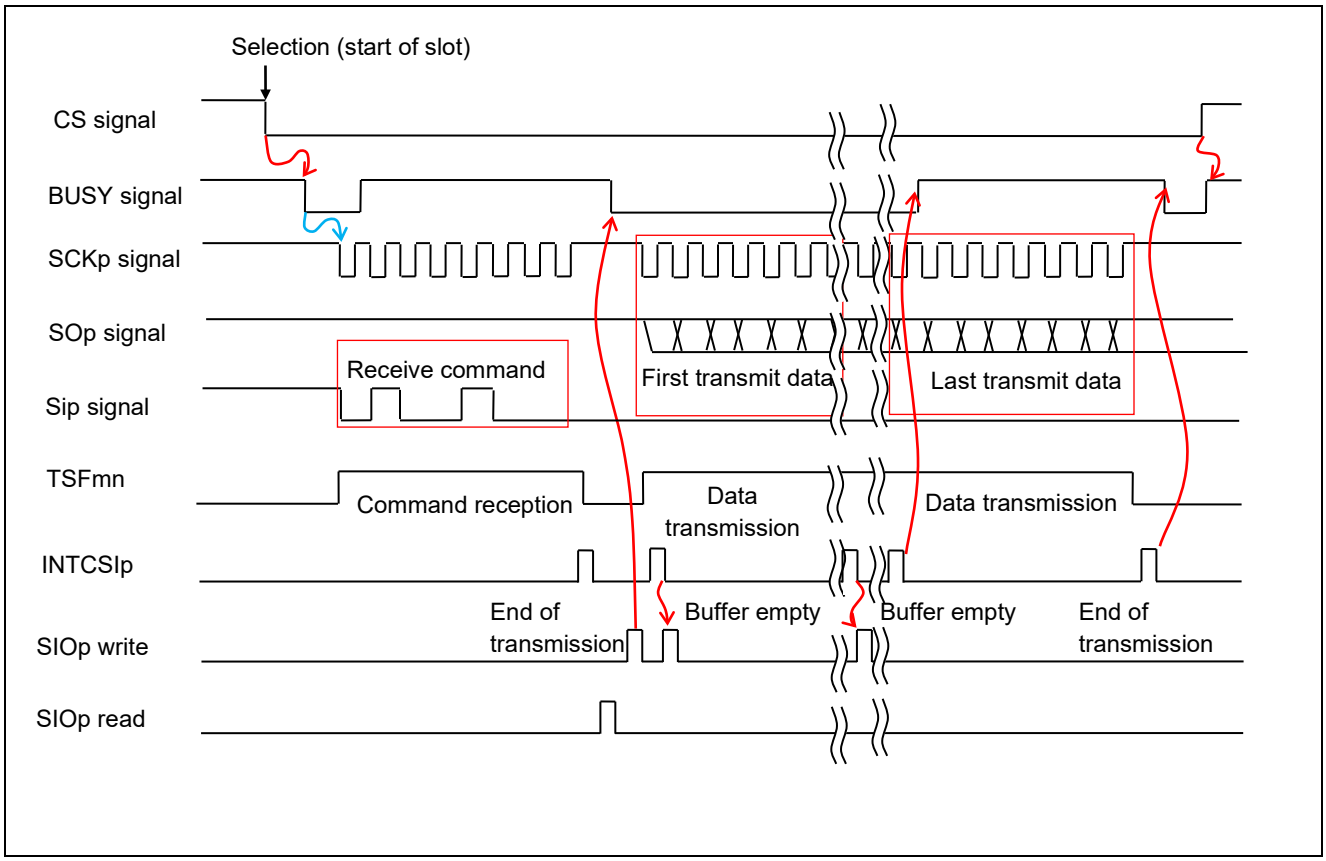


Figure 1.4 Timing Chart of Receive Command

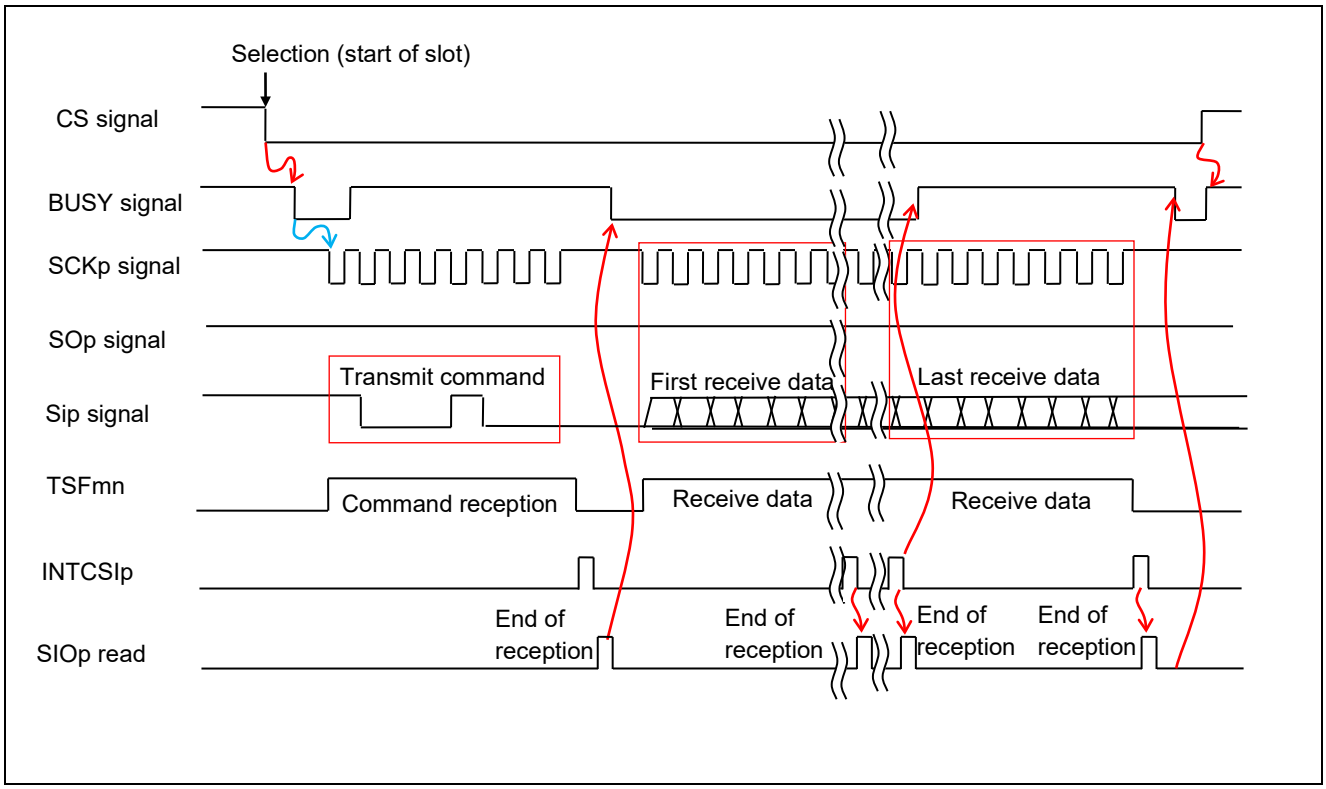


Figure 1.5 Timing Chart of Transmit Command

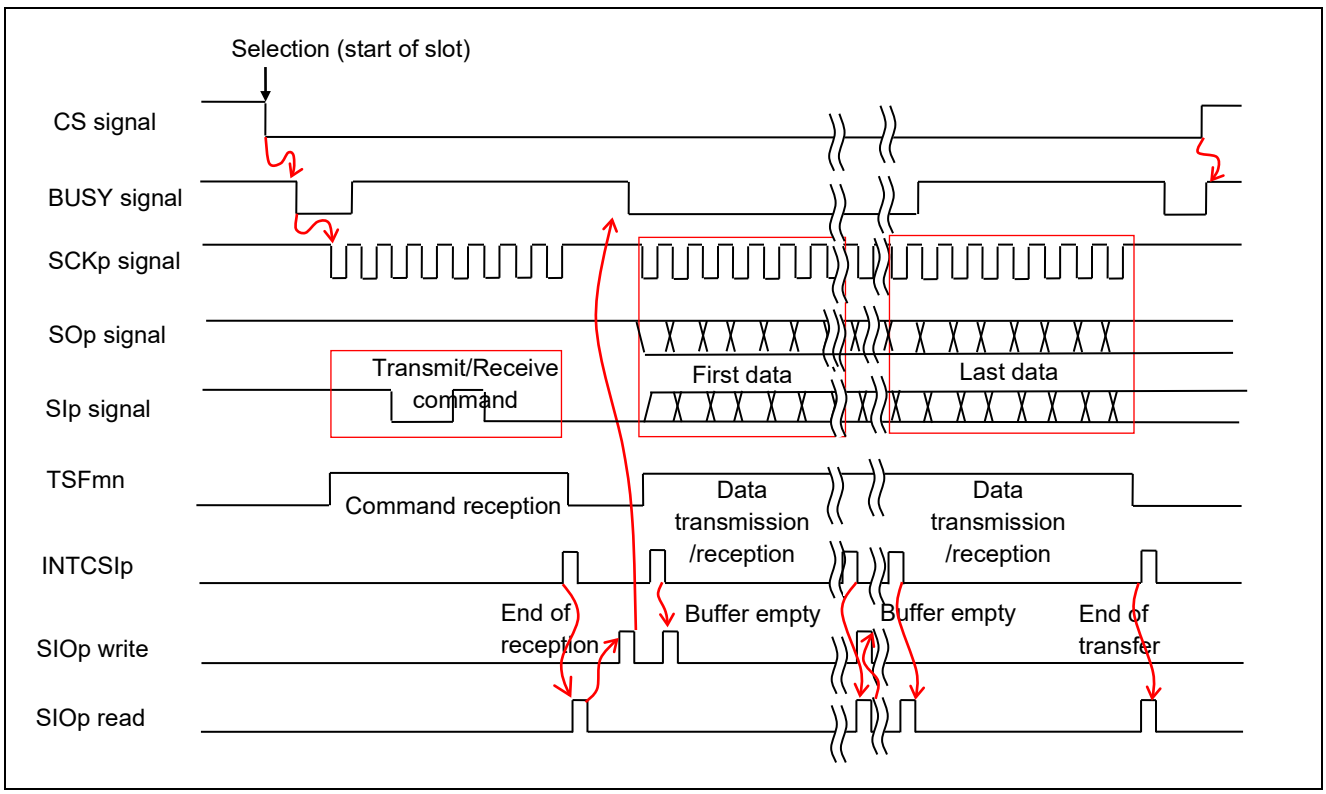


Figure 1.6 Timing Chart of Transmit/Receive Command

1.3 Communication Format

The characteristics of the CSI communication format that is used by the sample code are listed in table 1.3.

Table 1.3 Communication Format

Item	Specification	Remarks
Communication speed	1 Mbps	About 200 kbps at minimum
Data bit length	8 bits/character	
Transfer order	MSB first	
Communication type	Type1	
Communication mode	Single transfer/continuous transfer	Continuous mode is used for data transfer.
Communication direction	Receive/transmit/transmit and receive	
Maximum number of characters transferred	63 characters/slot	8 characters by default

1.4 Communication Protocol (Hardware Handshake)

(1) Necessity of handshaking in the RL78/G10

The BUSY signal is not available for dedicated SPI slave devices such as EEPROM, A/D, and D/A because of high-speed response being involved when they are selected by the master or communication is started. This is because these devices are made always ready for communication by hardware. When using a general-purpose device such as the RL78/G10 as a slave under program control, on the other hand, the processing time taken by the software is always involved.

In such a circumstance, handshaking using the BUSY signal is adopted to secure a preparatory time for readying the RL78/G10 for communication when it is selected by the master or at the beginning of communication processing. Since it is likely that handshaking using the BUSY signal alone locks the slave in the BUSY state, the master is provided with a timeout mechanism.

(2) Handshaking processing

Figure 1.7 shows an example of handshaking processing for the status check command. To select the slave, the master waits until the BUSY signal goes low while measuring the time so as to detect a timeout condition after the falling edge of the CS signal. When the BUSY signal goes low before a timeout, the master sends the command. Upon completion of the command transmission, the master waits until the BUSY signal goes low again to start status receive processing. In this way, the master performs handshaking to get synchronized with the slave by checking the BUSY signal before initiating a new communication operation.

In this case, the master can dispense with the BUSY signal if it is configured always to start processing after the lapse of a timeout time.

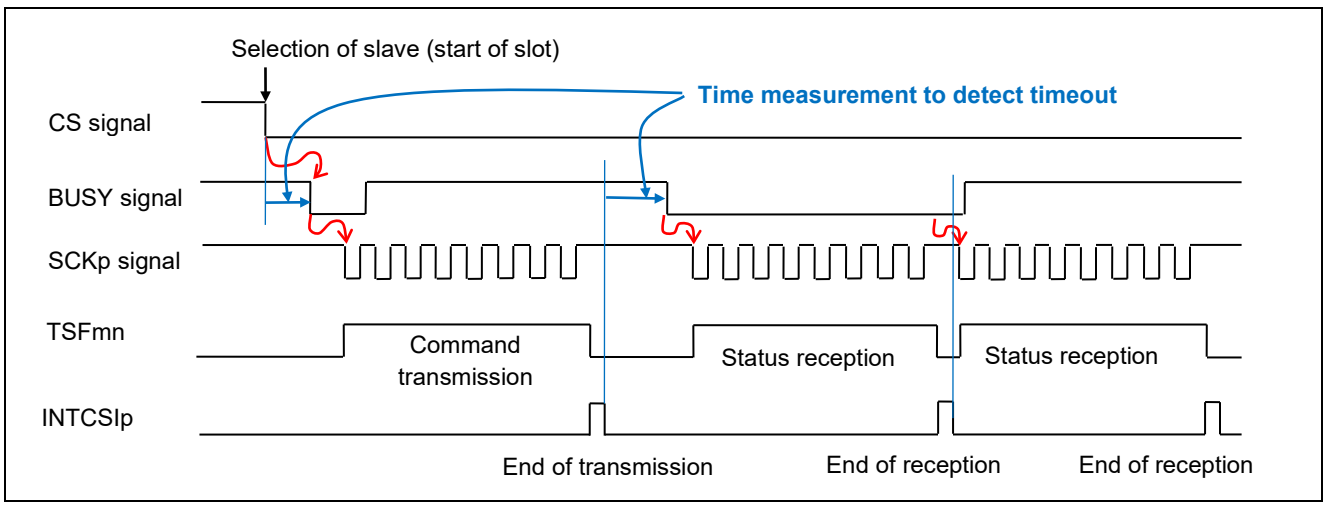


Figure 1.7 Handshaking Example (Processing at Master)

(3) Response time in the RL78/G10

The response time of the RL78/G10 is divided into the time up to the acceptance of an interrupt and the time required for the subsequent processing. The interrupt acceptance time is subject to whether interrupts are acceptable and to the instruction that is being executed when the interrupt request is generated. Whether interrupts are acceptable depends on the running program. In this example, it is assumed that interrupts are always acceptable. After a vector interrupt is received, 11 to 18 clock cycles are required for its acceptance, depending on the instruction that is currently being executed. In practice, when limited to the instruction used to wait for an interrupt, it is required to consider four clock cycles for a conditional branch instruction. So, at most 14 clock cycles are required for accepting an interrupt.

(a) When selecting a slave using the CS signal, for example, the processing program will look like as shown below and the BUSY signal goes low in 13 clocks. A total of 24 clocks is required, accounting for a response time of 1 μs.

```
IINTP0:
    BT P13.7,    $NOTSELECTED    ; branch if CS is not active
SELECTED:
    CLR1   PM_SOp           ; set SOp port to output
    MOV    SIOp,    #0xFF       ; set dummy data for start
    ONEB   CSISTS           ; set the number of received data(1)
    CLR1   BUSYSIG          ; fall down BUSY signal
    CLR1   BUSYMODE         ; output low BUSY signal
    CALL   !STARTCSIp       ; start CSIp
```

Entry to INTP0

Processing block to be executed when the device is selected by the CS signal. The device is immediately enabled for CSI operation and made ready for receiving commands from the master.
Subsequently, the BUSY signal is set low.

(b) The portion of the processing block that takes the longest processing time is the one that receives and analyzes a command and causes a branch to the necessary processing. An excerpt of the relevant processing block is shown below. Since different processing routines process different interrupts, the address of the actual processing routine is stored in the variable RCSISUBADDR in advance, so that a branch is made to that processing routine immediately when an INTCSIp interrupt is accepted. In the case that is shown below, 57 clocks are required in the worst case to cause a branch to the command processing block, entailing a total of 69 clocks. Subsequently, the individual command processing can start. Since the communication mode is reconfigured at the beginning of each command processing, communication can be started in approximately 35 clocks.

Summing up these, it is possible that the device becomes ready for communication in approximately 4 μs after the reception of the command is completed.

```
IINTCSIp:
        MOVW  AX,     RCSISUBADDR  ; get actual routine address
        BR    AX                   ; branch to actual routine
CSITXEND:
        MOV   A,      SIOp         ; get receved data
        MOV   RRXDATA, A           ; save received data
        CALL  !SSETEMPTYINT
        MOVW  RCSISUBADDR, #LOWW CSIBFEMP ; set transfer start address
        CLRB  CSISTS               ; receive end
        RETI

MAIN_LOOP2:
        CALL  !SCHKRXEND           ; CSIp command receive check
        BT    BUSYMODE, $MAIN_LOOP      ; branchi if CS is high
        BNZ   $MAIN_LOOP2          ; wait for command
        MOV   RCOMBUF, A           ; save command in buffer
        AND   RCOMBUF, #0B00111111 ; get data number
        AND   A,      #0B11000000  ; get command bit
        SHRW  AX,     13           ; A7 -> X2,A6 -> X1
        ADDW  AX,     #LOWW CCMDLIST      ; get table address
        MOVW  HL,     AX
        MOVW  AX,     ES:[HL]      ; get subroutine address
        CALL  AX                   ; go to actual routine
```

Branch processing to actual processing block (branch in 6 clocks)

Actual processing block for command reception
(exit in 10 clocks)

Main command wait block

Wait for completion of command reception in this loop.
(loop of 19 clocks)

Command analysis block.
References the table and causes a branch to pertinent command processing block.
(12 clocks are required.)

2. Operation Check Conditions

The sample code contained in this application note has been checked under the conditions listed in the table below.

Table 2.1 Operation Check Conditions

Item	Description
Microcontroller used	RL78/G10 (R5F10Y16ASP)
Operating frequency	• High-speed on-chip oscillator (HOCO) clock: 20 MHz • CPU/peripheral hardware clock: 20 MHz
Operating voltage	5.0 V (Operation is possible over a voltage range of 2.9 V to 5.5 V.) SPOR detection: Falling edge: VDD<2.84V Rising edge: VDD>=2.90V
Integrated development environment (CS+)	CS+ for CC V3.01.00 from Renesas Electronics Corp.
Assembler (CS+)	CC-RL V1.01.00 from Renesas Electronics Corp.
Integrated development environment (e^2 studio)	e^2 studio V4.0.2.008 from Renesas Electronics Corp.
Assembler (e^2 studio)	CC-RL V1.01.00 from Renesas Electronics Corp.
Integrated development environment (IAR)	IAR Embedded Workbench for Renesas RL78 V4.21.3 from IAR Systems.
Assembler (IAR)	IAR Assembler for Renesas RL78 V4.21.2.2420 from IAR Systems.
Board to be used	RL78/G10 target board (QB-R5F10Y16-TB)

3. Related Application Notes

The application notes that are related to this application note are listed below for reference.

RL78/G10 Initialization CC-RL (R01AN2668E) Application Note
RL78/G10 Serial Array Unit (CSI Master Communication) CC-RL (R01AN3076E) Application Note

4. Description of the Hardware

4.1 Hardware Configuration Example

Figure 4.1 shows an example of hardware configuration that is used for this application note.

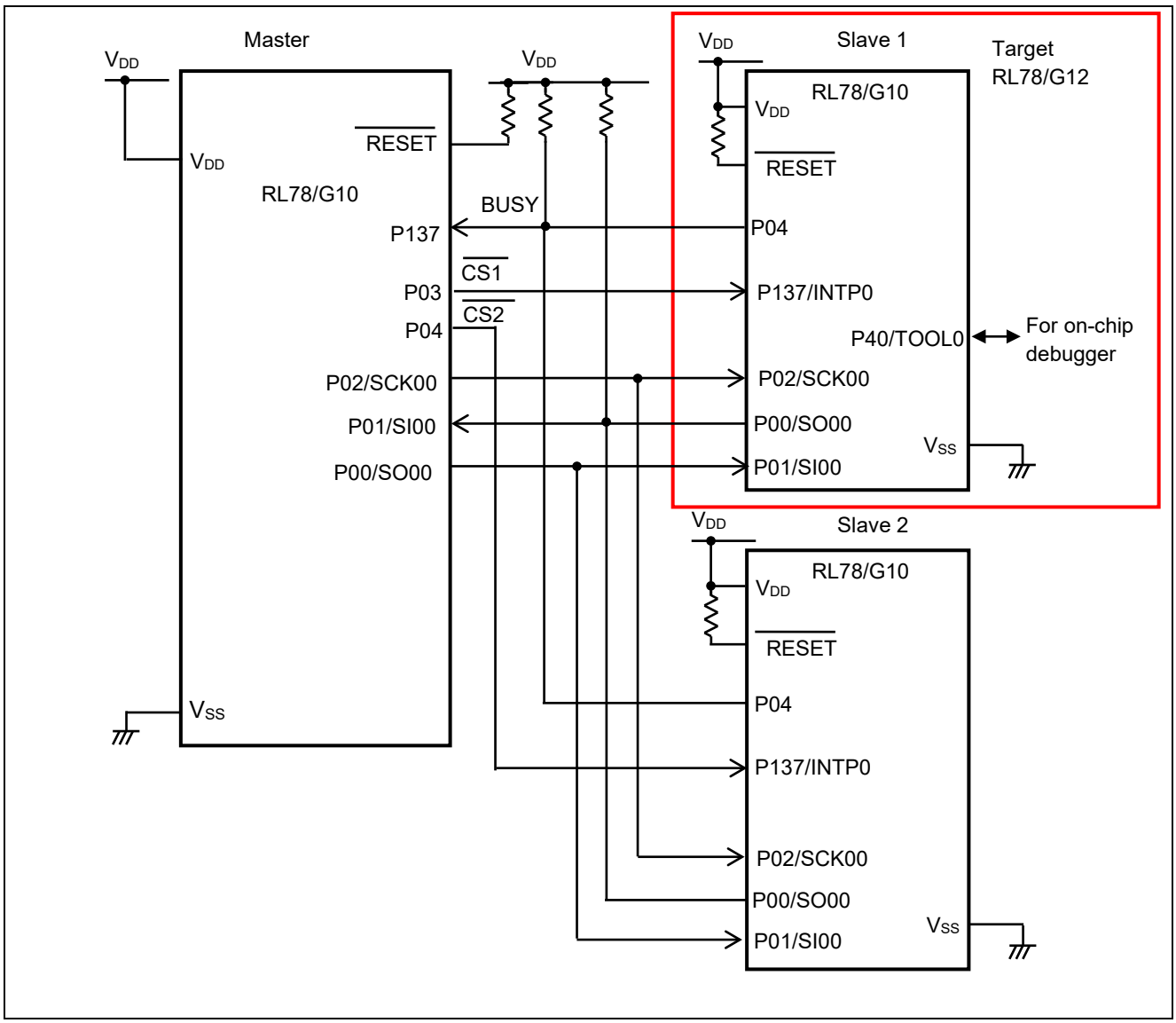


Figure 4.1 Hardware Configuration

Cautions: 1. The purpose of this circuit is only to provide the connection outline and the circuit is simplified accordingly. When designing and implementing an actual circuit, provide proper pin treatment and make sure that the hardware's electrical specifications are met (connect the input-only ports separately to V_{DD} or V_{SS} via a resistor).

2. V_{DD} must be held at not lower than the reset release voltage (V_{SPOR}) that is specified as SPOR.

3. In the example of wiring in the hardware configuration for the sample code from Renesas (R01AN3077), RL78/G10 devices are mounted as slaves 1 and 2. The sample code operates so that signal output is only by the slave selected by the CS signal (the non-selected pin is placed in the high impedance state). Accordingly, the P04 pins of slaves 1 and 2 are directly connectable (this also applies to the P00/SO00 pins on the slave side).

4.2 List of Pins to be Used

Table 4.1 lists the pins to be used and their function.

Table 4.1 Pins to be Used and their Functions

Pin Name	I/O	Description
P02/ ANI1/SCK00/SCL00/PCLBUZ0/KR3	Output	Serial clock output pin
P01/ANI0/SI00/RXD0/SDA00/KR2	Input	Data receive pin
P00/SO00/TXD0/INTP1	Output	Data transmit pin
P137/INTP0（CS）	Input	Select signal
P04/ANI3/TI01/TO01/KR5（BUSY）	Output	BUSY response signal

Note: The channel to be used must be specified in an include file (DEV&CSI_CH.inc). The default value is CSI00. The pins and interrupt to be used are automatically set according to the channel to be used.

5. Description of the Software

5.1 Operation Outline

This sample code, after completion of initialization, initializes the memory and waits for selection by the master. When selected as a slave, it waits for a command from the master and takes necessary actions according to the command that is received.

(1) Initialize the CSI.
<Conditions for setting the CSI>

- Use SAU0 channel 0 as CSI00 Note.
- Use SCKp input as the transfer clock.
- Assign the clock input to the P10/SCK00 pin Note, the data input to the P11/SI00 pin Note, and the data output to the P12/SO00 pin Note. In this stage, however, the P12/SO00 pin Note is not set up for output.
- Set the data length to 8 bits.
- Set the phase between the data and clock to type 1.
- Set the order of data transfer mode to MSB first.
- Turn on the transmission/reception mode and set the interrupt type (INTCSI00) Note to the buffer empty interrupt.
- Set the priority of the interrupt (INTCSI00) Note to the lowest (level 3 (default)).

Note: Two channels are available for use in 16-pin products.
The channel to be used must be specified in an include file (DEV&CSI_CH.inc). The default value is CSI00. The pins and interrupt to be used are automatically set according to the channel to be used.

(2) When initialization is completed, the sample code initializes the memory and waits for selection by the master with the CS signal.
Transition occurs in combination with the interrupt-triggered processing.

(3) When the sample code takes the following actions when the falling edge of the CS signal is detected by the INTP0 edge detection mode interrupt function:
1) Enables the CSIp for operation, enables the SOp for output, sets the BUSY signal low, and waits for the start of command reception processing.
2) When the reception of the command from the master starts and a buffer empty interrupt occurs, the sample code outputs a BUSY signal (sets it high) and wait for the end of reception processing.
3) Returns to step (2) when the CS signal goes high.

(4) Commands
Each communication operation begins with the reception of a 1-byte command. Table 5.1 lists the command formats.

Table 5.1 Command Formats

Command Code		Command Outline
Status check	00000000B	Checks the number of data characters that the slave can transmit or receive. The following responses can be made by the slave: 01xxxxxxB: The number of characters that the slave can transmit is xxxxxxB 00xxxxxxB: The number of characters that the slave can receive is xxxxxxB
Receive	01xxxxxxB	The master receives xxxxxxB bytes of data.
Transmit	10xxxxxxB	The master transmits xxxxxxB bytes of data.
Transmit/Receive	11xxxxxxB	Transmits and receives xxxxxxB bytes of data.

1) When command reception is completed and the INTCSIp interrupt is accepted, the interrupt processing block stores the received data in the buffer and clears the flag (CSISTS).
2) The main processing block waits for CSISTS to be cleared, reads the command from the buffer, stores the lower 6 bits of the received command in the data count buffer (RCOMBUF), reads the processing address associated with the received command with a table reference, and causes a

branch to that address (CALL AX). (For details, refer to paragraph (3), RL78/G10 response time, in section 1.4, Communication Protocol (Hardware Handshake).)

(5) Preparation for receiving the next command
1) Activates the CSIp for reception processing upon receipt of a next command and sets the BUSY signal low.
2) Return to 2) in (3).

(6) Status check command processing (SREADSTS)
1) Loads the HL register with the pointer to the transmit data (status).
2) Loads the A register with the transmit data count (2).
3) Calls the continuous transmission subroutine (SSEQTXSUB).
4) Waits for end of transmission.
5) Terminates processing when the transmission processing is completed.

(7) Master receive command processing (SMSTRRX)
1) Loads the HL register with the pointer to the transmit data.
2) Loads the A register with the transmit data count (RCOMBUF).
3) Calls the continuous transmission subroutine (SSEQTXSUB).
4) Waits for end of transmission.
5) Terminates processing when the transmission processing is completed.

(8) Master transmit command processing (SMSTRTX)
1) Loads the HL register with the pointer to the receive data.
2) Loads the A register with the transmit data count (RCOMBUF).
3) Calls the continuous transmission subroutine (SSEQTXSUB).
4) Waits for end of transmission.
5) Stores the complement of the received data in the transmit buffer and terminates processing.

(9) Transmit/receive command processing
1) Loads the HL register with the pointer to the transmit data.
2) Loads the DE register with the pointer to the receive data.
3) Loads the A register with the transmit data count (RCOMBUF).
4) Calls the continuous transmission subroutine (SSEQTXSUB).
5) Waits for end of reception.
6) Stores the complement of the received data in the transmit buffer and terminates processing.

5.2 List of Option Byte Settings

Table 5.2 summarizes the settings of the option bytes.

Address	Value	Description
000C0H	01101110B	Disables the watchdog timer. (Stops counting after the release from the reset state.)
000C1H	01111111B	SPOR detection voltage: Falling edge: VDD<2.84V Rising edge: VDD>=2.90V
000C2H	11100000B	HOCO: 20 MHz
000C3H	10000101B	Enables the on-chip debugger.

Table 5.2 Option Byte Settings

5.3 List of Constants

Tables 5.3 and 5.4 list the constants that are used in this sample program.

Table 5.3 Constants for the Sample Program (1/2)

Constant	Defined in	Setting	Description
CLKFREQ	DEV&CSI_CH .inc	20000	RL78/G10 operating clock frequency in kHz (24 MHz)
BAUDRATE	↑	1000	Communication speed in kbps (1 Mbps)
DIVIDE	↑	CLKFREQ/BAUDRATE	Frequency division ratio necessary for attain the specified communication speed
SDRDATA	↑	(DIVIDE - 1) * 2	Value to be set in SDR to specify the communication speed
INTERVAL	↑	1	Slot interval in ms units (1 ms) (not used)
TDRDATA	↑	(CLKFREQ/128)*INTERVAL-1	Value to be set in TDR03H (not used)
SMR0nH	↑	SMR00H Note	Channel mode setting register
SMR0nL		SMR00L Note	Channel mode setting register
SCR0nH	↑	SCR00H Note	Channel communication operation setting register
SCR0nL		SCR00L Note	Channel communication operation setting register
SDR0nH	↑	SDR00H Note	Channel serial data register
SIOp	↑	SIO00 Note	Lower 8 bits of channel serial data
SSR0n	↑	SSR00 Note	Channel status register
SIR0n	↑	SIR00 Note	Channel flag clear trigger register
TRGONn	↑	00000001B Note	Value for SSmL and STmL
SOEON	↑	TRGONn	For setting in channel output enable register (enable)
SOEOFF	↑	11111110B Note	For setting in channel output enable register (disable)
SOHIGH	↑	TRGONn	Used to set value into channel output register.
PM_SCKp	↑	PM0.2 Note	SCK signal port mode register
PM_SIp	↑	PM0.1 Note	SI signal port mode register
PM_SOp	↑	PM0.0 Note	SO signal port mode register

Table 5.4 Constants for the Sample Program (2/2)

Constant	Defined In	Setting	Description
P_SCKp	↑	P0.2[Note]	SCK signal port
P_SIp	↑	P0.1[Note]	SI signal port
P_SOp	↑	P0.0[Note]	SO signal port
CSIIFp	↑	CSIIF00[Note]	Channel interrupt request flag
CSIMKp	↑	CSIMK00[Note]	Channel interrupt master register
CRXMODE	↑	0100000000000111B	Value to be loaded in SCR register in receive mode
CTXMODE	↑	1000000000000111B	Value to be loaded in SCR register in transmit mode
CTRXMODE	↑	1100000000000111B	Value to be loaded in SCR register in transmit/ receive mode
CSMRDATA	↑	0000000000100000B	Initial value for SMR register
BUSYSIG	r_main.asm	P0.4	Port for outputting the BUSY signal
BUSYMODE	↑	PM0.4	Port mode register for outputting the BUSY signal
CRXDTNO	↑	8	Size of receive data buffer (in bytes)
CTXDTNO	↑	8	Size of transmit data buffer (in bytes)
CCMDLIST	↑	SREADSTS	Address of status check command processing block
		SMSTRRX	Address of master reception processing block
		SMSTRTX	Address of master transmission processing block
		SMSTRTXRX	Address of transmission/reception processing block

Note: Values vary when CSI01 is used.

5.4 List of Variables

Table 5.5 lists the global variables that are used in this sample program.

Table 5.5 Global Variables for the Sample Program

Type	Variable Name	Contents	Function Used
16 bits	RCSISUBADDR	Address of INTCSIp processing block	IINTCSIp, SETCSIMODE, CSIBFEMP, CSITXEND, STXNEXT, STRXNEXT, STX1DTST, SSEQRXSUB, SSEQTXSUB, SSEQTRXSUB
8-bit array	RSNDBUF	Transmit data buffer	main, SMSTXEND, STXNEXT, STRXNEXT, SSEQTXSUB, SSEQTRXSUB,
8-bit array	RRCVBUF	Receive data buffer	main, SRXNEXT, SRXEND, STRXNEXT2
8 bits	RCOMBUF	Command buffer from master	main, SMSTRRX, SMSTRTX, SMSTXEND, SMSTRTXRX
8 bits	RTXDTNO	Counter for number of characters transmitted	main, SREADSTS
8 bits	RRXDTNO	Counter for number of characters received	main, SREADSTS
8 bits	RRXDATA	Receive buffer	main, CSITXEND, SWAITTXEND, SCHKRXEND,
8 bits	CSISTS	Number of remaining characters to be transferred	IINTP0, SETCSIMODE, CSITXEND, SRXNEXT, STXNEXT, STXEND, STRXNEXT2, STRXNEXT, STX1DTST, SWAITTXEND, SCHKRXEND, SSEQRXSUB, SWAITSTREND, SSEQTXSUB, SSEQTRXSUB

5.5 List of Functions (Subroutines)

Table 5.6 summarizes the functions (subroutines) that are used in this sample program.

Table 5.6 List of Functions (Subroutines)

Function Name	Outline
RESET_START	Makes initial settings of hardware and calls the main function.
SINIPORT	Sets up the I/O port.
SINICLK	Sets up the clock generation circuit.
SINISAU	Initialize CSIp.
SINIINTP0	Initialize INTP0 for CS signal detection.
SREADSTS	Process status check command.
SMSTRRX	Process master receive command.
SMSTRTX	Process master transmit command.
SMSTXEND	End reception. Perform receive data complement processing.
SMSTRTXRX	Process transmit/receive command.
SETCSIMODE	Set up CSIp select release state (halt in transmission/reception mode).
IINTP0	Process CS signal edge detection interrupts.
IINTCSIp	Process INTCSIp interrupt entry.
CSIBFEMP	Process 1-character transfer start interrupts. Start the BUSY signal.
CSITXEND	Process 1-character transmit end interrupts (store receive data in RRXDATA).
SRXNEXT	Process data receive end interrupts in continuous reception mode.
STXNEXT	Process buffer empty interrupts in continuous transmission mode.
SRXEND	Process receive end interrupts for last data in continuous transmission/reception mode.
STXEND	Process transmit end interrupts for last data in continuous transmission mode.
STRXNEXT	Process transmit start interrupts in continuous transmission/reception mode.
STRXNEXT2	Process buffer empty interrupts in continuous transmission/reception mode.
STX1DTST	Turn on transmission/reception mode.
SRX1DTST	Turn on transmission/reception mode and start 1-character reception processing.
SWAITTXEND	Wait for end of 1-character reception (load receive data in A register).
SCHKRXEND	Check 1-character transfer state. Set Z flag to 1 if end of transfer.
SSEQRXSUB	Start continuous reception processing.
SWAITSTREND	Wait for end of continuous transfer.
SSEQTXSUB	Start continuous transmission processing.
SSEQTRXSUB	Wait for end of continuous transmission/reception.
SSEQRXSUB2	Waits for start of last data transfer.
SSETENDINT	Set up transfer end interrupts.
SSETEMPTYINT	Set up buffer empty interrupts.
SCHNG2TRX	Stop operation temporarily and enable transmission/reception mode (buffer empty interrupts).
SCHNG2TXS	Common processing for mode setup
SCHNG2TX	Stop operation temporarily and enable transmission mode (buffer empty interrupts).
SCHNG2RX	Stop operation temporarily and enable reception mode (transfer end interrupt).
STARTCSIp	Start CSI.
STOPCSIp	Stop CSI.

5.6 Function (Subroutine) Specifications

This section describes the specifications for the functions that are used in the sample program.

[Function Name] RESET_START

Synopsis	Makes initial settings of hardware and calls the main function.
Explanation	This function calls the main routine after setting the stack pointer and making initial settings for the hardware.
Arguments	None
Return value	None
Remarks	None

[Function Name] SINIPORT

Synopsis	Sets up the I/O port.
Explanation	This function sets the bits of port registers other than those for CSI-related pins and the CS signal to 0. This function sets unused pins as outputs where possible.
Arguments	None
Return value	None
Remarks	None

[Function Name] SINICLK

Synopsis	Sets up the clock generation circuit.
Explanation	This function makes initial settings of the registers related to the clock generation circuit.
Arguments	None
Return value	None
Remarks	None

[Function Name] SINISAU

Synopsis	Initialize CSIp.
Explanation	This function sets up the CSIp for type 1, 8 bits, MSB first, and transmission/reception on transfer end interrupts. The SOp output pin is not set up for output.
Arguments	None
Return value	None
Remarks	None

[Function Name] SINIINTP0

Synopsis	Initialize INTP0 interrupt.
Explanation	This function sets INTP0 to both edge detection mode.
Arguments	None
Return value	None
Remarks	None

[Function Name] SETCSIMODE

Synopsis	Process CSIp select release state setup.
Explanation	This function sets the BUSY signal high (subsequently disables output), configures the SOp output pin for input, stops the CSIp, and sets the interrupt timing type to buffer empty interrupt in transmission/reception mode. The function initializes the INTCSIp processing address, clears CSISTS, and prepares for next selection.
Arguments	None
Return value	None
Remarks	None

[Function Name] SREADSTS

Synopsis	Process status check command.
Explanation	This function transmits the status with the continuous transmission subroutine.
Arguments	None
Return value	None
Remarks	None

Function Name] SMSTRRX

Synopsis	Process master receive command.
Explanation	This function sets up pointer and number of communication characters and transmits data with the continuous transmission subroutine.
Arguments	None
Return value	None
Remarks	None

[Function Name] SMSTRTX

Synopsis	Process master transmit command.
Explanation	This function sets up pointer and number of communication characters and transmits data with the continuous reception subroutine.
Arguments	None
Return value	None
Remarks	None

[Function Name] SMSTRTXRX

Synopsis	Process transmit/receive command.
Explanation	This function sets up pointer and number of communication characters and transmits data with the continuous transmission/reception subroutine.
Arguments	None
Return value	None
Remarks	None

[Function Name] SMSTXEND

Synopsis	Received Data Complement Processing Function
Explanation	This function converts the received data into its complement.
Arguments	None
Return value	None
Remarks	None

[Function Name] IINTP0

Synopsis	Process INTP0 edge detection interrupt.
Explanation	If P13.7 is low, this function recognizes the device is selected, starting the CSIp to start command reception processing. If P13.7 is high, the function recognizes the device is not selected, stopping the CSIp.
Arguments	None
Return value	None
Remarks	None

The functions (subroutines) given below are the functions that actually process INTCSIp interrupts.

[Function Name] IINTCSIp

Synopsis	Process INTCSIp interrupt.
Explanation	This function performs the entry processing for an INTCSIp interrupt and causes a branch to the address that is stored in the variable RCSISUBADDR.
Arguments	None (The variable RCSISUBADDR contains the address of the actual interrupt processing.)
Return value	None
Remarks	None

[Function Name] CSIBFEMP

Synopsis	Process 1-character transfer start interrupt.
Explanation	This function is invoked when a 1-character transfer begins and a buffer empty interrupt occurs. The function sets the BUSY signal high, changes the INTCSIp timing to end of transfer, and sets the address of the next INTCSIp interrupt processing (RCSISUBADDR) to that of transfer end interrupt processing (CSITXEND).
Arguments	None
Return value	None
Remarks	None

[Function Name] CSITXEND

Synopsis	Process 1-character transmission end interrupt.
Explanation	This function is invoked by a 1-character of transfer end interrupt. The function stores the receive data in the buffer variable (RRXDATA), changes the INTCSIp timing to buffer empty interrupt, and sets the address of the next INTCSIp interrupt processing (RCSISUBADDR) to that of buffer empty interrupt processing (CSIBFEMP).
Arguments	None
Return value	None
Remarks	None

[Function Name] SRXNEXT

Synopsis	Process data receive end interrupt in continuous reception mode.
Explanation	This function is invoked by a 1-character of receive end interrupt in continuous reception mode and stores the receive data in the required buffer. If the data count (CSISTS) is set to 1, the function sets the address of the next INTCSIp interrupt processing (RCSISUBADDR) to that of receive end interrupt processing for last data (SRXEND) and sets the BUSY signal high when the reception of the last data starts.
Arguments	None
Return value	None
Remarks	None

[Function Name] STXNEXT

Synopsis	Process continuous transmission mode buffer empty interrupt.
Explanation	This function is invoked by a buffer empty interrupt in continuous transmission mode. If the data count (CSISTS) is not set to 1, the function decrements the value of CSISTS by 1 and write the data into the SIOp. If the number of communication characters is set to 1, the function sets the BUSY signal high, changes the interrupt timing to end of transfer, and sets the address of the next INTCSIp interrupt processing (RCSISUBADDR) to transmit end interrupt processing for last data (STXEND).
Arguments	None
Return value	None
Remarks	None

[Function Name] SRXEND

Synopsis	Process continuous transmission/reception mode last data reception completion interrupt.
Explanation	This function is invoked by a last data reception completion interrupt. The function stores the receive data in the required buffer and sets the data count (CSISTS) to 0.
Arguments	None
Return value	None
Remarks	None

[Function Name] STXEND

Synopsis	Process continuous transmission mode last data transmission completion interrupt.
Explanation	This function is invoked by transmit end interrupt for last data. The function sets the number of communication characters (CSISTS) to 0.
Arguments	None
Return value	None
Remarks	None

[Function Name] STRXNEXT

Synopsis	Process continuous transmission/reception mode transfer start interrupt.
Explanation	This function is invoked by the first buffer empty interrupt occurring in transmission/reception mode. The function writes the next data into the SIOp if the data count (CSISTS) is not set to 1. If the number of communication characters is set to 1, the function sets the BUSY signal high, changes the INTCSIp timing to transfer end interrupt, and sets the address of the next INTCSIp interrupt processing (RCSISUBADDR) to that of receive end interrupt processing for last data (SRXEND).
Arguments	None
Return value	None
Remarks	None

[Function Name] STRXNEXT2

Synopsis	Process continuous transmission mode buffer empty interrupt.
Explanation	This function is invoked by the second and subsequent buffer empty interrupts in continuous transmission/reception mode. The function stores the receive data in the receive data buffer, decrements the number of communication characters by 1, and performs the above STRXNEXT processing.
Arguments	None
Return value	None
Remarks	None

The functions (subroutines) given below are available as general-purpose functions.

[Function Name] SCHKRXEND

Synopsis	Check 1-character transfer state.	
Explanation	This function checks CSISTS to examine the transmission or reception state. The function returns with the Z flag set to 0 if the communication is not yet completed and with the Z flag set to 1 by loading receive data into the A register if the communication is completed.	
Arguments	None	
Return value	Z flag	: [1: Communication complete, 0: Communication in progress]
	A register	: Receive data (contents of RRCVBUF) if communication is completed
Remarks	None	

[Function Name] STX1DTST

Synopsis	Turn on transmission mode and perform 1-character transmission start processing.	
Explanation	This function places the CSIp in transmission/reception mode. This function writes data from the A register into the SIOp and starts communication processing. The function loads transfer start processing (CSIBFEMP) into next INTCSIp interrupt processing address (RCSISUBADDR) and sets the number of work-in-progress characters to 1 before returning.	
Arguments	A register	Transmit data
Return value	None (However, CSISTS is set to 1)	
Remarks	None	

[Function Name] SRX1DTST

Synopsis	Start 1-character reception processing.
Explanation	This function places the CSIp in transmission/reception mode. The function writes dummy data (FFH) into the SIOp and starts receive processing. It loads transfer start processing (CSIBFEMP) into next INTCSIp interrupt processing address (RCSISUBADDR) and sets the number of work-in-progress characters to 1 before returning.
Arguments	None
Return value	None
Remarks	None

[Function Name] SWAITTXEND

Synopsis	Perform 1-character transmission(/reception) completion wait processing.	
Explanation	This function waits for the end (CSISTS = 0) of the transmission processing that is started by the STXDATAST function. When the transmission is completed, the function reads the value of the receive data buffer (RRXDATA) into the A register.	
Arguments	None	
Return value	A register	Receive data
Remarks	The transmit end interrupt is processed by CSITXEND (CSISTS is set to 0).	

[Function Name] SSEQTXSUB

Synopsis	Start continuous transmission processing.	
Explanation	This function places the CSIp in transmission mode and starts transmission processing to send the number of characters specified in the A register from the buffer designated by the HL register. The function sets the address of the next INTCSIp interrupt processing (RCSISUBADDR) to that of the buffer empty interrupt processing (STXNEXT). It then sets the number of work-in-progress characters (CSISTS) to the value of the A register and loads the HL register on the register bank 1 with the buffer pointer before returning. The function returns with a Z flag value of 1 if the receive data count in the A register is 0.	
Arguments	HL register	: Address of area for storing the transmit data
	A register	: Number of characters to be transmitted
Return value	Z flag	: [0: Normal startup, 1: Number of characters is 0.] (CSISTS is set to the number of characters at normal startup time.)
Remarks	None	

[Function Name] SSEQRXSUB

Synopsis	Start continuous reception processing.	
Explanation	This function places the CSIp in the reception mode and activates the function that receives the number of characters designated in the A register from the buffer designated by the HL register. It also sets the address of the next INTCSIp interrupt processing (RCSISUBADDR) to that of buffer empty interrupt processing (STRXNEXT). The function places the value of the A register in the number of work-in-progress characters (CSISTS) and loads the DE register with the pointer to the buffer. The function returns with a Z flag value of 1 if the value of the receive data count in the A register is 0.	
Arguments	HL register	: Address of area for storing receive data
	A register	: No of receive characters
Return value	Z flag	: [0: Normal startup, 1: Number of characters is 0.] (CSISTS is set to the number of characters at normal startup time.)
Remarks	None	

[Function Name] SSEQTRXSUB

Synopsis	Start continuous transmission/reception processing.	
Explanation	This function places the CSIp in the transmission/reception mode and activates the function that transmits the number of characters designated in the A register from the buffer designated by the HL register and stores the received data in the buffer designated by the DE register. It also sets the address of the next INTCSIp interrupt processing (RCSISUBADDR) to that of buffer empty interrupt processing (STRXNEXT). The function places the value of the A register in the number of work-in-progress characters (CSISTS), loads the HL register on the register bank 1 with the pointer to the transmit data buffer, and loads the DE register with the pointer to the area for storing the received data. The function returns with a Z flag value of 1 if the value of the receive data count in the A register is 0.	
Arguments	HL register	: Address for storing transmit data
	DE register	: Address for storing receive data
	A register	: Number of transfer characters
Return value	Z flag	: [0: Normal startup, 1: Number of characters is 0.] (CSISTS is set to the number of characters at normal startup time.)
Remarks	None	

[Function Name] SSEQRXSUB2

Synopsis	Wait for start of last data transfer
Explanation	The function sets the address of the next INTCSIp interrupt processing (RCSISUBADDR) to the address of SRXNEXT. Read the CSIp state. If communication from the master has been started, set the BUSY signal to the high level.
Arguments	なし
Return value	なし
Remarks	なし

[Function Name] SWAITSTREND

Synopsis	Wait for end of continuous transfer.
Explanation	This function waits until the number of characters (CSISTS) reaches 0 during end of wait processing that is common to continuous reception, transmission, and transmission/reception processing.
Arguments	None
Return value	None
Remarks	None

[Function Name] SSETENDINT

Synopsis	Set up transfer end interrupts.
Explanation	This function sets the CSIp interrupt timing to end of transfer.
Arguments	None
Return value	None
Remarks	None

[Function Name] SSETEMPTYINT

Synopsis	Set up buffer empty interrupts.
Explanation	This function sets the CSIp interrupt timing to buffer empty interrupts.
Arguments	None
Return value	None
Remarks	None

[Function Name] SCHNG2TX

Synopsis	Set CSIp in transmission mode.
Explanation	This function stops the CSI temporarily and enables the transmission mode. The interrupt timing is set to buffer empty interrupt.
Arguments	None
Return value	None
Remarks	None

[Function Name] SCHNG2RX

Synopsis	Set CSIp in reception mode.
Explanation	This function stops the CSI temporarily and enables the reception mode. The interrupt timing is set to end of transfer.
Arguments	None
Return value	None
Remarks	None

[Function Name] SCHNG2TRX

Synopsis	Set CSIp in transmission/reception mode
Explanation	This function stops the CSI temporarily and enables the transmission/reception mode. The interrupt timing is set to buffer empty interrupt.
Arguments	None
Return value	None
Remarks	None

[Function Name] SCHNG2TXS

Synopsis	Mode Setup Common Processing
Explanation	Common processing for mode setup. This function stops the CSIp temporarily and changes the setting for the mode to that set in the AX register so that operations can proceed. The interrupt timing is set for the end of transfer.
Arguments	なし
Return value	なし
Remarks	なし

[Function Name] STARTCSIp

Synopsis	Enable CSIp.
Explanation	This function enables the CSIp (SE0n = 1), clears the interrupt request, and unmasks interrupts to enable interrupts.
Arguments	None
Return value	None
Remarks	None

[Function Name] STOPCSIp

Synopsis	Disable CSIp.
Explanation	This function disables (masks) interrupts and disables the CSIp.
Arguments	None
Return value	None
Remarks	None

5.7 Flowcharts

Figure 5.1 shows the overall flow of the sample program described in this application note.

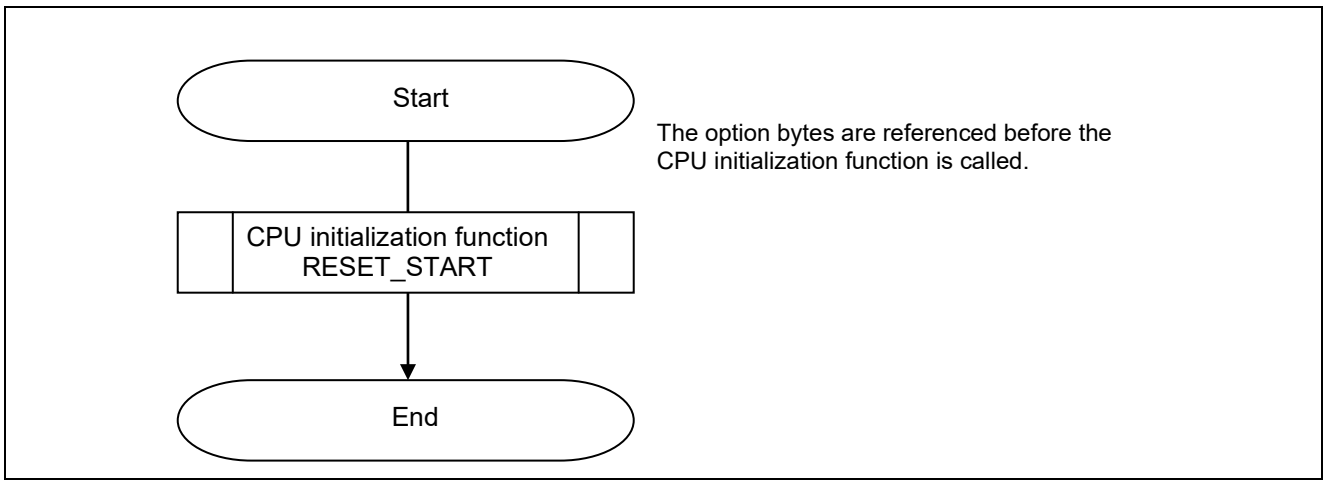


Figure 5.1 Overall Flow

5.7.1 CPU Initialization Function

Figure 5.2 shows the flowchart for the CPU initialization function.

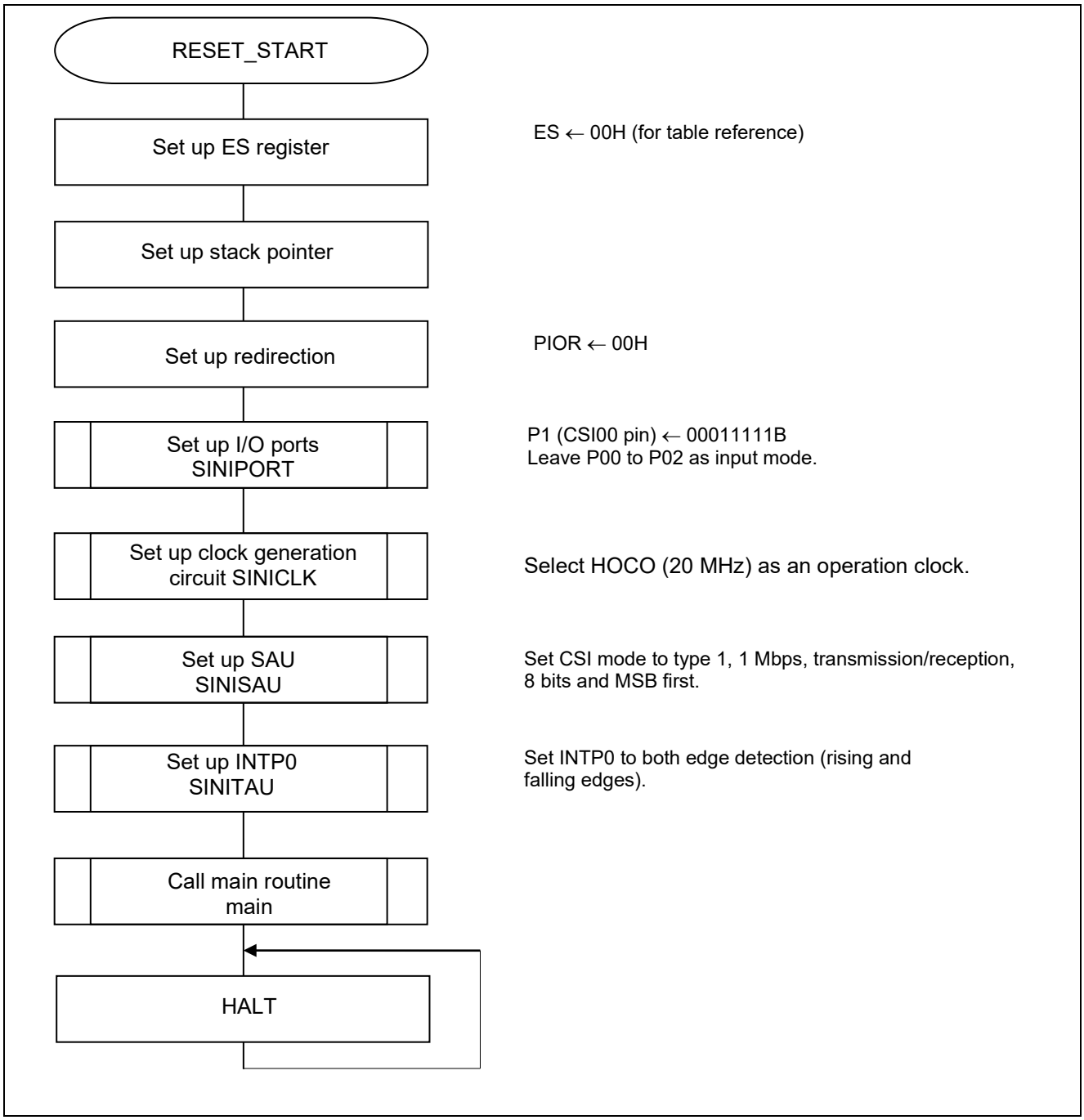


Figure 5.2 CPU Initialization Function

5.7.2 I/O Port Setup

Figure 5.3 shows the flowchart for I/O port setup.

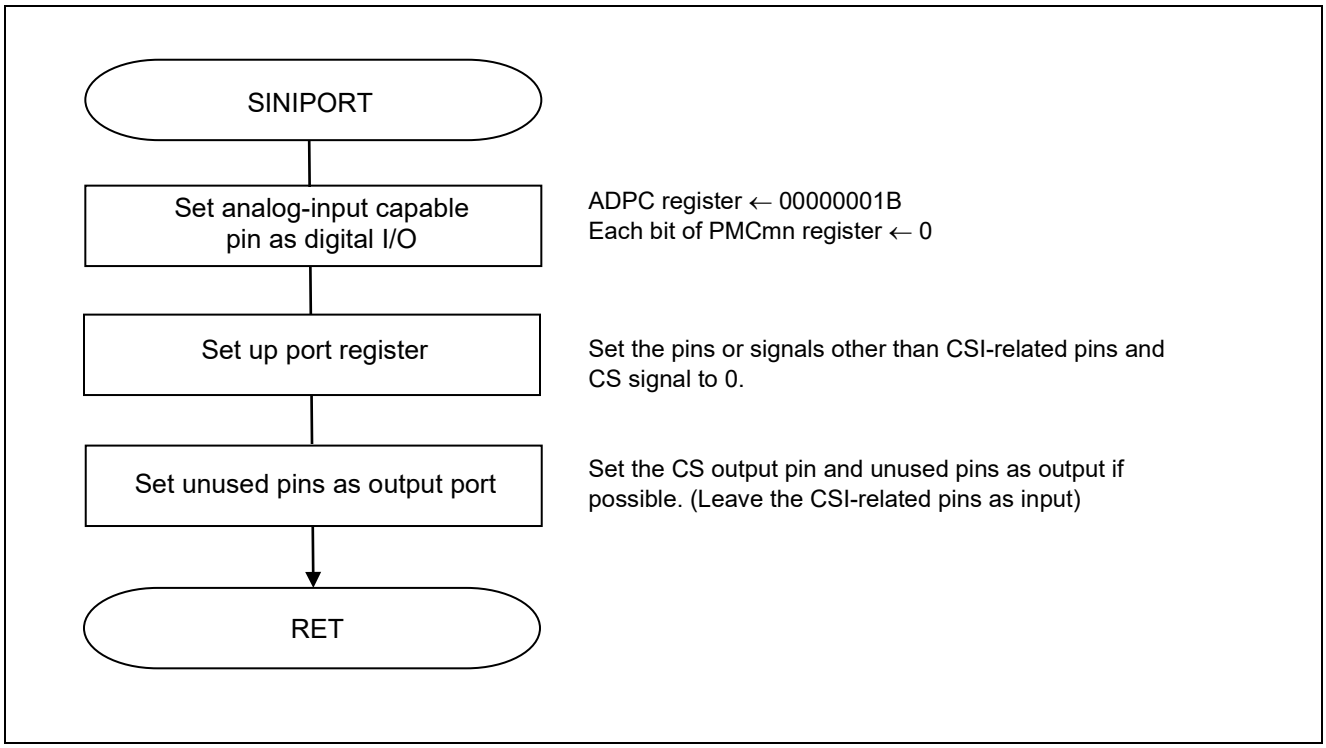


Figure 5.3 I/O Port Setup

Note: Refer to the section entitled "Flowcharts" in RL78/G10 Initialization Application Note (R01AN2668E) for the configuration of the unused ports.

Caution: Provide proper treatment for unused pins so that their electrical specifications are observed. Connect each of any unused input-only ports to V_{DD} or V_{SS} via separate resistors.

5.7.3 Setting up the Clock Generator Circuit

Figure 5.4 shows the flowchart for clock generation circuit setup.

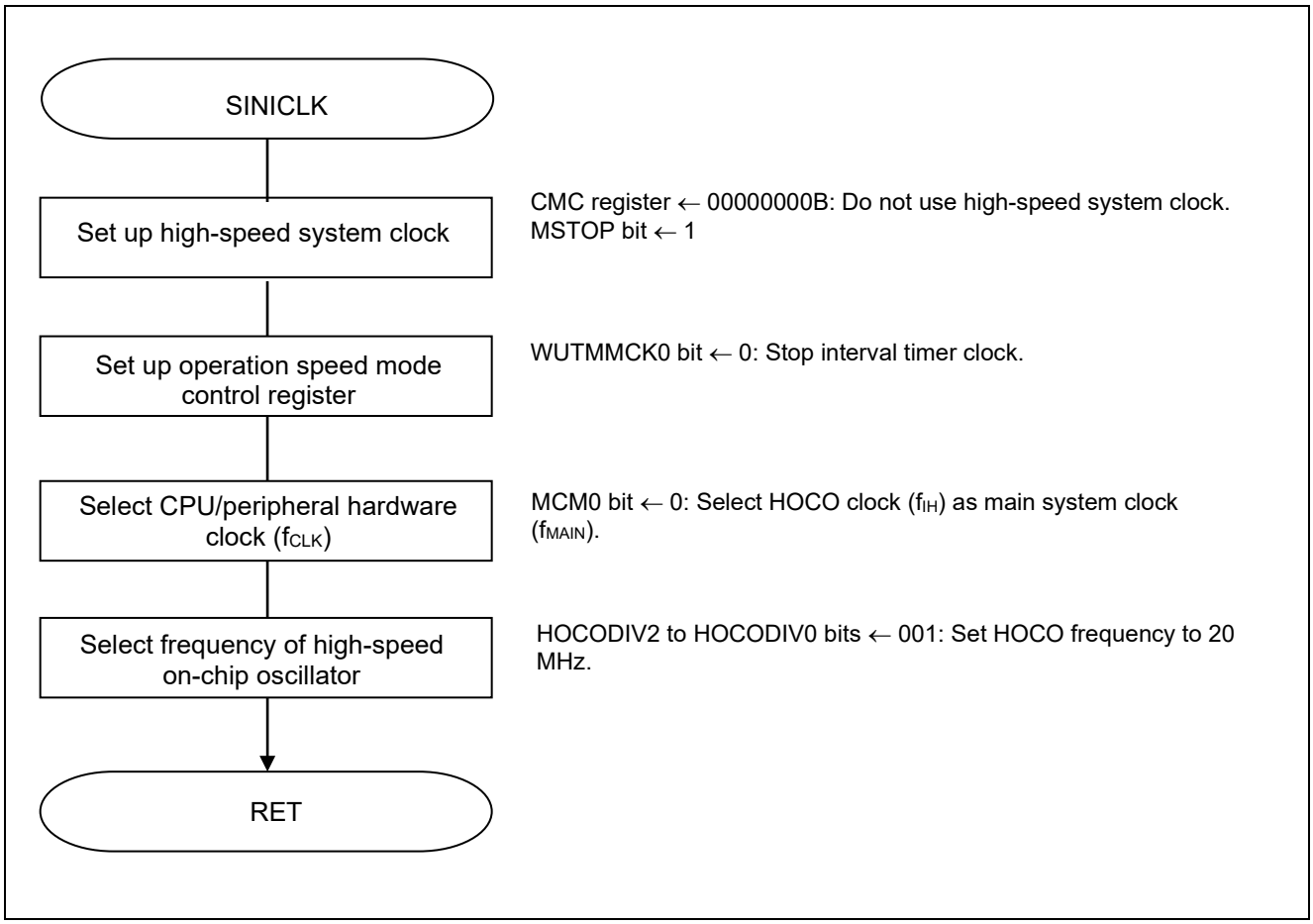


Figure 5.4 Clock Generator Circuit Setup

Caution: For details on the procedure for setting up the clock generation circuit (SINICLK), refer to the section entitled "Flowcharts" in RL78/G10 Initialization CC-RL Application Note (R01AN2668E).

5.7.4 SAU Setup

Figure 5.5 shows the flowchart for SAU setup.

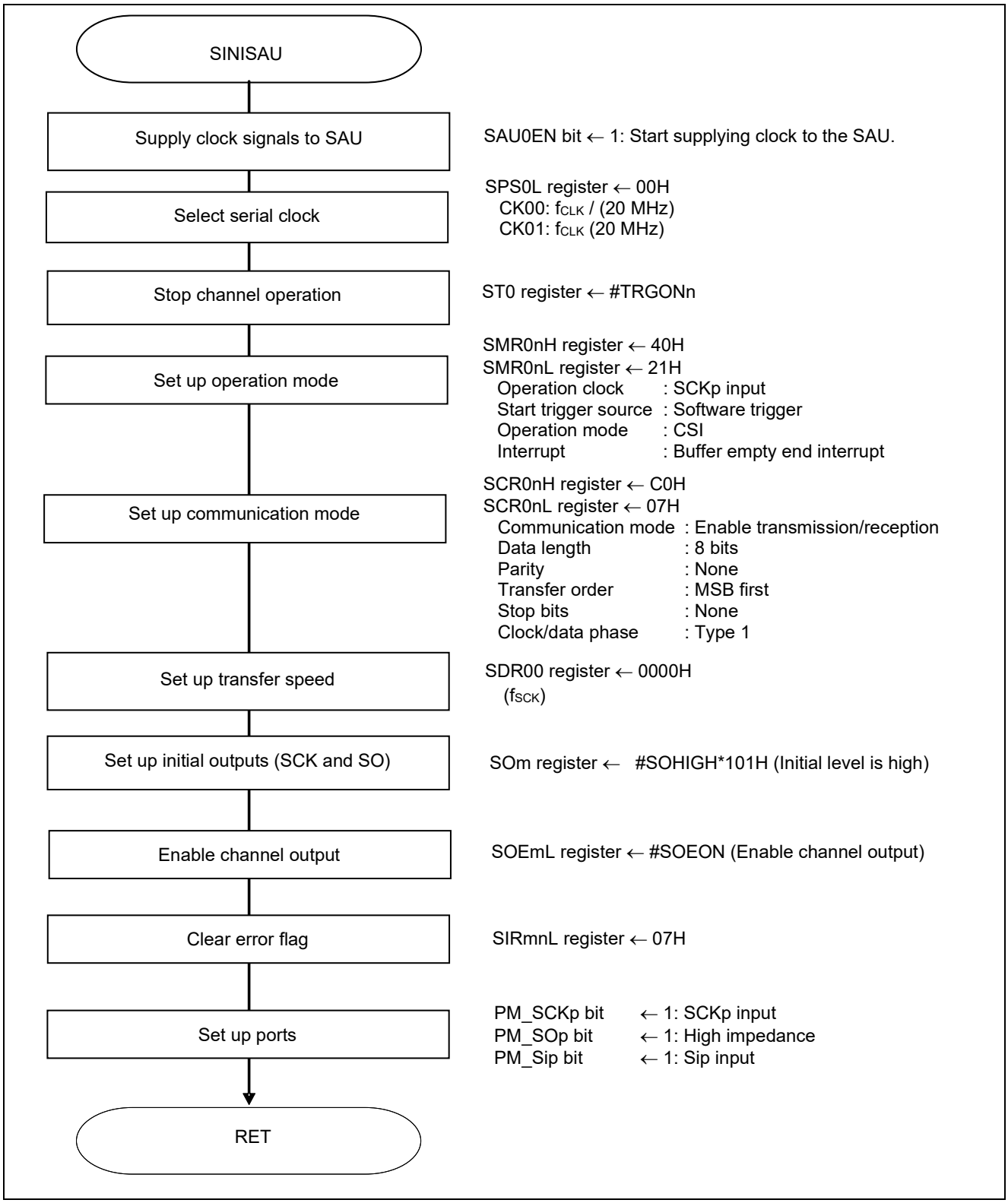


Figure 5.5 SAU Setup

Start supplying clock to the SAU

- Peripheral enable register 0 (PER0)
  Start supplying clock signals.

Symbol: PER0

7	6	5	4	3	2	1	0
TMKAEN[Note]	CMPEN[Note]	ADCEN	IICA0EN[note]	0	SAU0EN	0	TAU0EN
x	x	x	x	0	**1**	0	x

Bits 2

SAUmEN	Control of serial array unit n input clock supply
0	**Stops supply of input clock.**
1	**Enables supply of input clock.**

Selecting a serial clock

- Serial clock selection register m (SPS0)
  Select an operation clock for SAU.

Symbol: SPS0

7	6	5	4	3	2	1	0
PRS013	PRS012	PRS011	PRS010	PRS003	PRS002	PRS001	PRS000
0	**0**	**0**	**0**	**0**	**0**	**0**	**0**

Bits 7 to 0

PRS 0n3	PRS 0n2	PRS 0n1	PRS 0n0	Selection of operation clock (CK0n) (n = 0 to 1)					
					f_{CLK}= 1.25MHz	f_{CLK}= 2.5MHz	f_{CLK}= 5MHz	f_{CLK}= 10MHz	**f_{CLK}= 20MHz**
0	**0**	**0**	**0**	**f_{CLK}**	1.25 MHz	2.5 MHz	5 MHz	10 MHz	**20 MHz**
0	0	0	1	$f_{CLK}/2$	625 kHz	1.25 MHz	2.5 MHz	5 MHz	10 MHz
0	0	1	0	$f_{CLK}/2^2$	313 kHz	625 kHz	1.25 MHz	2.5 MHz	5 MHz
0	0	1	1	$f_{CLK}/2^3$	156 kHz	313 kHz	625 kHz	1.25 MHz	2.5 MHz
0	1	0	0	$f_{CLK}/2^4$	78 kHz	156 kHz	313 kHz	625 kHz	1.25 MHz
0	1	0	1	$f_{CLK}/2^5$	39 kHz	78 kHz	156 kHz	313 kHz	625 kHz
0	1	1	0	$f_{CLK}/2^6$	19.5 kHz	39 kHz	78 kHz	156 kHz	313 kHz
0	1	1	1	$f_{CLK}/2^7$	9.8 kHz	19.5 kHz	39 kHz	78 kHz	156 kHz
1	0	0	0	$f_{CLK}/2^8$	4.9 kHz	9.8 kHz	19.5 kHz	39 kHz	78 kHz
1	0	0	1	$f_{CLK}/2^9$	2.5 kHz	4.9 kHz	9.8 kHz	19.5 kHz	39 kHz
1	0	1	0	$f_{CLK}/2^{10}$	1.22 kHz	2.5 kHz	4.9 kHz	9.8 kHz	19.5 kHz
1	0	1	1	$f_{CLK}/2^{11}$	625 Hz	1.22 kHz	2.5 kHz	4.9 kHz	9.8 kHz
1	1	0	0	$f_{CLK}/2^{12}$	313 Hz	625 Hz	1.22 kHz	2.5 kHz	4.9 kHz
1	1	0	1	$f_{CLK}/2^{13}$	152 Hz	313 Hz	625 Hz	1.22 kHz	2.5 kHz
1	1	1	0	$f_{CLK}/2^{14}$	78 Hz	152 Hz	313 Hz	625 Hz	1.22 kHz
1	1	1	1	$f_{CLK}/2^{15}$	39 Hz	78 Hz	152 Hz	313 Hz	625 Hz

Note: 16-pin products only

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

Transiting to communication stopped state

- Serial channel stop register 0 (ST0)
  Stop communication

Symbol: ST0

7	6	5	4	3	2	1	0
0	0	0	0	0	0	ST01	ST00
0	0	0	0	0	0	**1**	**1**

Bits 1 and 0

ST0n	Operation stop trigger of channel n
0	No trigger operation
1	**Sets the SE0n bit to 0 and stops communication**

Setting up channel operation mode

- Serial mode register mn (SMR0nH, SMR0nL)
  Interrupt source
  Operation mode
  Select transfer clock.
  Select f_{MCK}.

Symbol: SMR0nH

7	6	5	4	3	2	1	0
CKS 0n	CCS 0n	0	0	0	0	0	STS 0n[Note]
0	**1**	0	0	0	0	0	**0**

Symbol: SMR0nL

7	6	5	4	3	2	1	0
0	SIS 0n0[Note]	1	0	0	MD 0n2	MD 0n1	MD 0n0
0	0	1	0	0	**0**	**0**	**0**

Bit 7(SMR0nH)

CKS0n	Selection of operation clock (f_{MCK}) of channel n
0	**Prescaler output clock CK00 set by the SPSm register**
1	Prescaler output clock CK01 set by the SPSm register

Bit 6(SMR0nH)

CCS0n	Selection of transfer clock (TCLK) of channel n
0	Divided operation clock f_{MCK} set by the CKSmn bit
1	**Clock input from the SCK pin.**

Bit 0(SMR0nH)

STS0n[Note]	Selection of start trigger source
0	**Only software trigger is valid**
1	Valid edge of the RxD pin (selected for UART reception)

Remarks: n: channel number (n=0, 1)

Note: SMR01H, SMR01L registers only

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

Symbol: SMR0nH

7	6	5	4	3	2	1	0
CKS 0n	CCS 0n	0	0	0	0	0	STS 0n[Note]
0	**1**	0	0	0	0	0	**0**

Symbol: SMR0nL

7	6	5	4	3	2	1	0
0	SIS 0n0[Note]	1	0	0	MD 0n2	MD 0n1	MD 0n0
0	0	1	0	0	**0**	**0**	**0**

Bit 6(SMR0nL)

SIS0n0[Note]	Selection of start trigger source
0	**Only software trigger is valid**
1	Valid edge of the RxD pin (selected for UART reception)

Bits 2 and 1(SMR0nL)

MD0n2	MD0n1	Setting of operation mode of channel n
0	**0**	**CSI mode**
0	1	UART mode
1	0	Simplified I^2C mode
1	1	Setting prohibited

Bit 0(SMR0nL)

MD0n0	Selection of interrupt source of channel n
0	Transfer end interrupt
1	**Buffer empty interrupt**

Remarks: n: channel number (n=0, 1)

Note: SMR01H, SMR01L registers only

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

Setting up channel communication mode

- Serial communication operation register mn (SCR0n)
  Setup data length, data transfer order, and operation mode.

Symbol: SCR0nH

7	6	5	4	3	2	1	0
TXE 0n	RXE 0n	DAP 0n	CKP 0n	0	EOC 0n	PTC 0n1	PTC 0n0
1	**1**	**0**	**0**	0	**0**	**0**	**0**

Symbol: SCR0nL

7	6	5	4	3	2	1	0
DIR 0n	0	SLC 0n1 Note	SLC 0n0	0	1	1	DLS 0n0
0	0	**0**	**0**	0	1	1	**1**

Bits 7 and 6 (SCR0nH)

TXE0n	RXE0n	Setting of operation mode of channel n
0	0	Disable communication.
0	1	Reception only
1	0	Transmission only
1	**1**	**Transmission/reception**

Bit 2 (SCR0nH)

EOC0n	Selection of masking of error interrupt signal (INTSREn)
0	**Masks error interrupt INTSRE0.**
1	Enables generation of error interrupt INTSREx.

Bits 1 and 0 (SCR0nH)

PTCmn 1	PTCmn 0	Setting of parity bit in UART mode	
		Transmission	Reception
0	**0**	**Does not output the parity bit.**	**Receives without parity.**
0	1	Outputs 0 parity.	No parity judgment
1	0	Outputs even parity.	Judged as even parity
1	1	Outputs odd parity.	Judged as odd parity

Bit 7 (SCR0nL)

DIRmn	Selection of data transfer sequence in CSI and UART modes
0	**Inputs/outputs data with MSB first.**
1	Inputs/outputs data with LSB first.

Bits 5 and 4 (SCR0nL)

SLC0n1	SLC0n0	Setting of stop bit in UART mode
0	**0**	**No stop bit**
0	1	Stop bit length = 1 bit
1	0	Stop bit length = 2 bits
1	1	Setting prohibited

Remarks: n: channel number (n=0, 1)

Note: SCR00L registers only

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

Symbol: SCR0nH

7	6	5	4	3	2	1	0
TXE 0n	RXE 0n	DAP 0n	CKP 0n	0	EOC 0n	PTC 0n1	PTC 0n0
1	**1**	**0**	**0**	0	**0**	**0**	**0**

Symbol: SCR0nL

7	6	5	4	3	2	1	0
DIR 0n	0	SLC 0n1 Note	SLC 0n0	0	1	1	DLS 0n0
0	0	**0**	**0**	0	1	1	**1**

Bits 1 and 0 (SCR0nL)

DLS0n0	Setting of data length in CSI mode
0	7-bit data length
1	**8-bit data length**

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

Setting up channel transfer clock

- Serial data register mn (SDR0nH, SDR0nL)
  Transfer clock frequency: f_{MCK}/20 (= 1 MHz)

Symbol: SDR0nH Symbol: SDR0nL

7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
0	**0**	**0**	**0**	**0**	**0**	**0**	0	x	x	x	x	x	x	x	x

Bit 7 to 1 (SDR0nH)

SDRmnH[7:1]							Transfer clock setting by dividing operation clock (f_{MCK})
0	**0**	**0**	**0**	**0**	**0**	**0**	f_{MCK} /2, **f_{SCK}**
0	0	0	0	0	0	1	f_{MCK} /4
0	0	0	0	0	1	0	f_{MCK} /6
0	0	0	0	0	1	1	f_{MCK} /8
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
1	1	1	1	1	1	0	f_{MCK} /254
1	1	1	1	1	1	1	f_{MCK} /256

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

Setting initial output level

- Serial output register 0 (SO0)
  Initial output: 1
- Serial clock output register 0 (CKO0)
  Initial output: 1

Symbol: SO0

7	6	5	4	3	2	1	0
0	0	0	0	0	0	SO01	SO00
0	0	0	0	0	0	**0/1**	**0/1**

Bit n

SOmn	Serial clock output of channel n
0	Serial data output value is “0”.
1	**Serial data output value is “1”.**

Symbol: CKO0

7	6	5	4	3	2	1	0
0	0	0	0	0	0	CKO01	CKO00
0	0	0	0	0	0	**0/1**	**0/1**

Bit n

CKO0n	チャネル n のシリアル・データ出力
0	シリアル・クロック出力値が“0”
1	**シリアル・クロック出力値が“1”**

Enabling target channel data output

- Serial output enable register m (SOEm/SOEmL)
  Enable output

Symbol: SOE0

7	6	5	4	3	2	1	0
0	0	0	0	0	0	SOE01	SOE00
0	0	0	0	0	0	**0/1**	**0/1**

Bit n

SOEmn	Serial output enable/disable of channel n
0	Disables output by serial communication operation.
1	**Enables output by serial communication operation.**

Remark: n: channel number (n=0, 1)

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

Clearing the error flags

- Serial flag clear trigger register 0n (SIR0n)
  Clear the error flags

Symbol: SIR0n

7	6	5	4	3	2	1	0
0	0	0	0	0	FECT0n[Note]	PECT0n	OVCT0n
0	0	0	0	0	**1**	**1**	**1**

Bit 2

FECT0n	Clear trigger of framing error of channel n
0	Not cleared
1	**Clears the FEF0n bit of the SSR0n registers to 0.**

Bit 1

PECT0n	Clear trigger of parity error of channel n
0	Not cleared
1	**Clears the PEF0n bit of the SSR0n registers to 0.**

Bit 0

OVCT0n	Clear trigger of overrun error of channel n
0	Not cleared
1	**Clears the OVF0n bit of the SSR0n registers to 0.**

Configuring the interrupt mask

- Interrupt mask flag register 0L (MK0L)
  Disable interrupt processing.

Symbol: MK0L (10-pin products)

7	6	5	4	3	2	1	0
TMMK00	TMMK01H	SREMK0	SRMK0	SRMK0 CSIMK00 IICMK00	PMK1	PMK0	WDTIMK
x	x	x	x	**1**	x	x	x

CSIMK00	Interrupt processing control
0	**Enables interrupt processing.**
1	**Disables interrupt processing.**

Note: SIR01 register only

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

Port setting (For CSI00)

- Port register 0 (P0)
- Port mode register 0 (PM0)
  Port setting for each of serial clock, transmit data and receive data.

Symbol: P0

7	6	5	4	3	2	1	0
P07 Note	P06 Note	P05 Note	P04	P03	P02	P01	P00
x	x	x	x	x	**1**	**1**	**1**

Bit 2-0

P0n	Output data control (in output mode)
0	0 is output
1	**1 is output**

Symbol: PM0

7	6	5	4	3	2	1	0
PM07Note	PM06Note	PM05Note	PM04	PM03	PM02	PM01	PM00
x	x	x	x	x	**1**	**1**	**1**

Bit 2

PM02	P02 pin I/O mode selection
0	Output mode (output buffer on)
1	**Input mode (output buffer off)**

Bit 1

PM01	P01 pin I/O mode selection
0	Output mode (output buffer on)
1	**Input mode (output buffer off)**

Bit 0

PM00	P00 pin I/O mode selection
0	Output mode (output buffer on)
1	**Input mode (output buffer off)**

Note: 16-pin products only.
Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

5.7.5 INTP0 Setup

Figure 5.6 shows the flowchart for INTP0 setup.

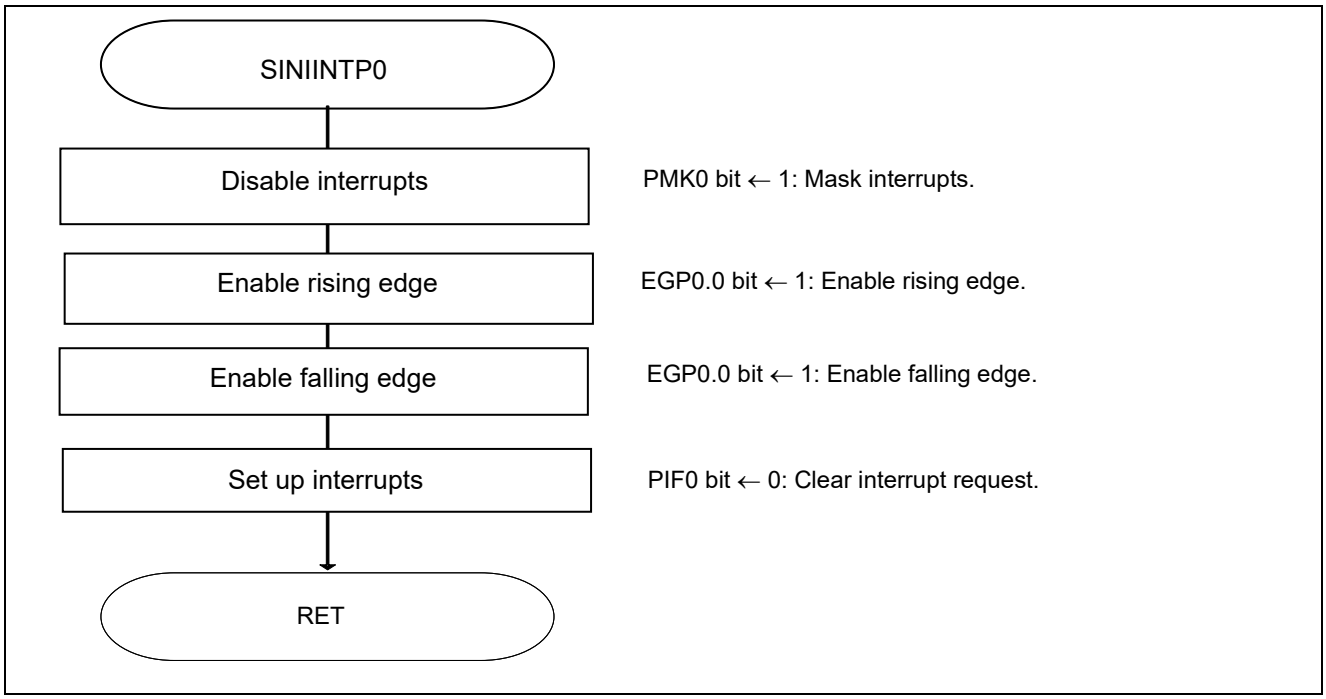


Figure 5.6 INTP0 Setup

5.7.6 Main Processing

Figures 5.7 and 5.8 show the flowcharts for the main processing.

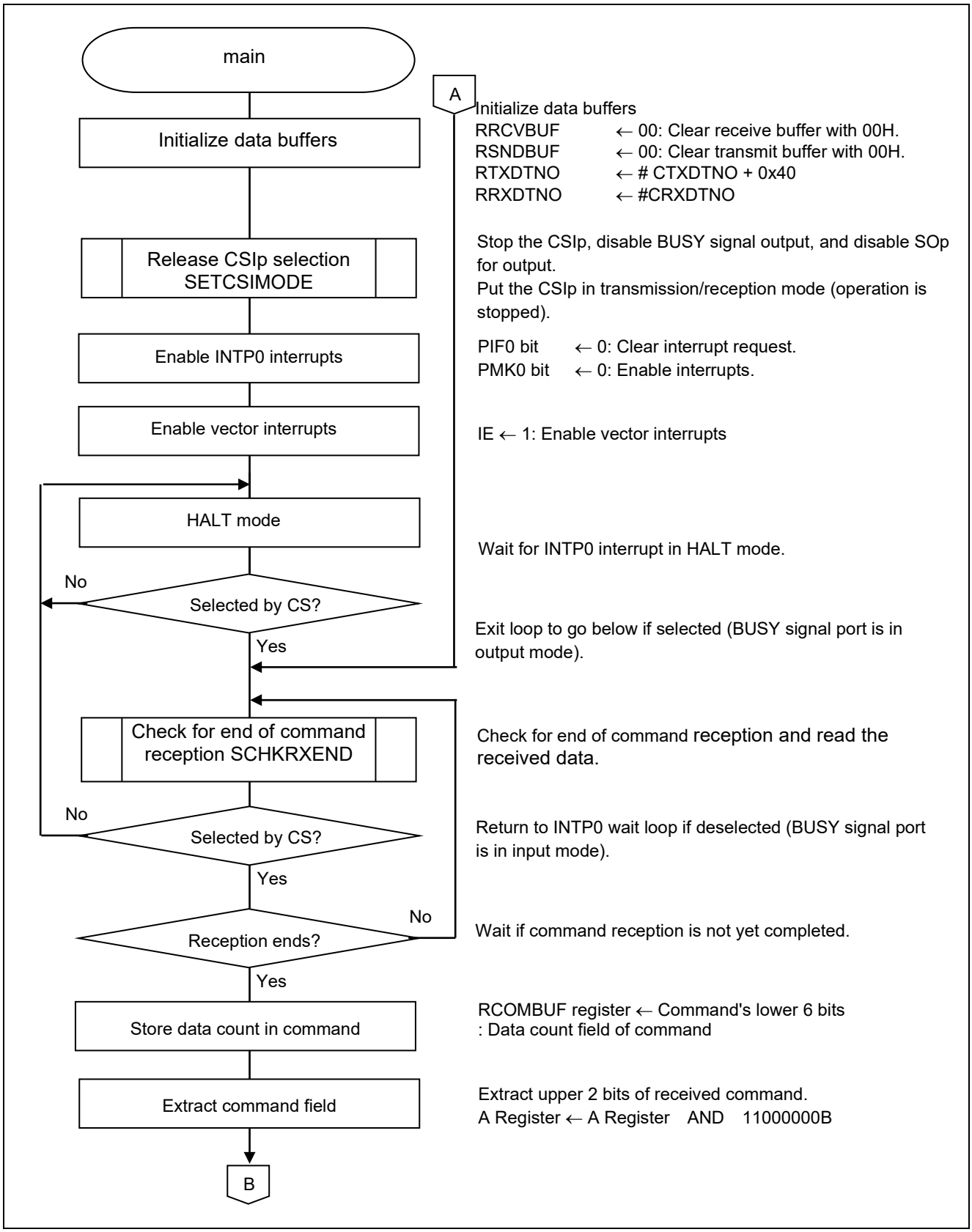


Figure 5.7 Main Processing (1/2)

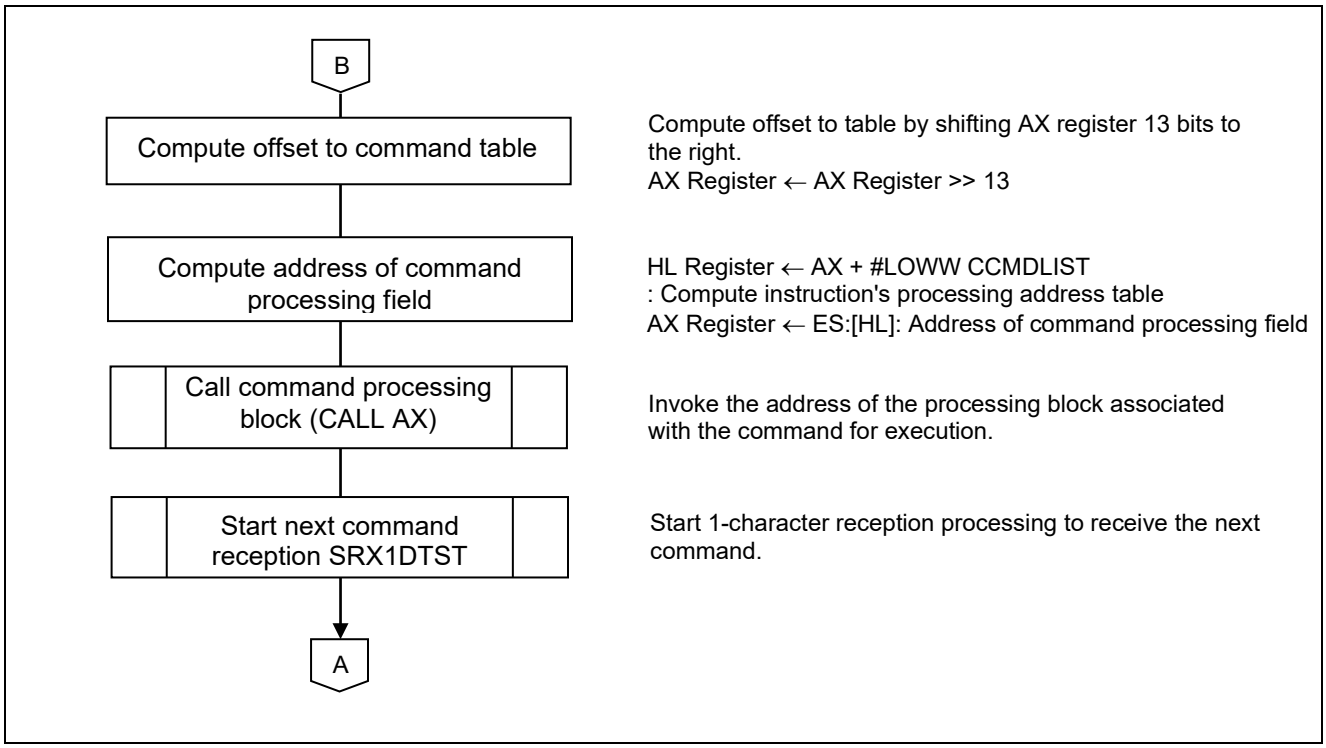


Figure 5.8 Main Processing (2/2)

5.7.7 CSIp Select Release Processing

Figure 5.9 shows the flowchart for the CSIp select release processing function.

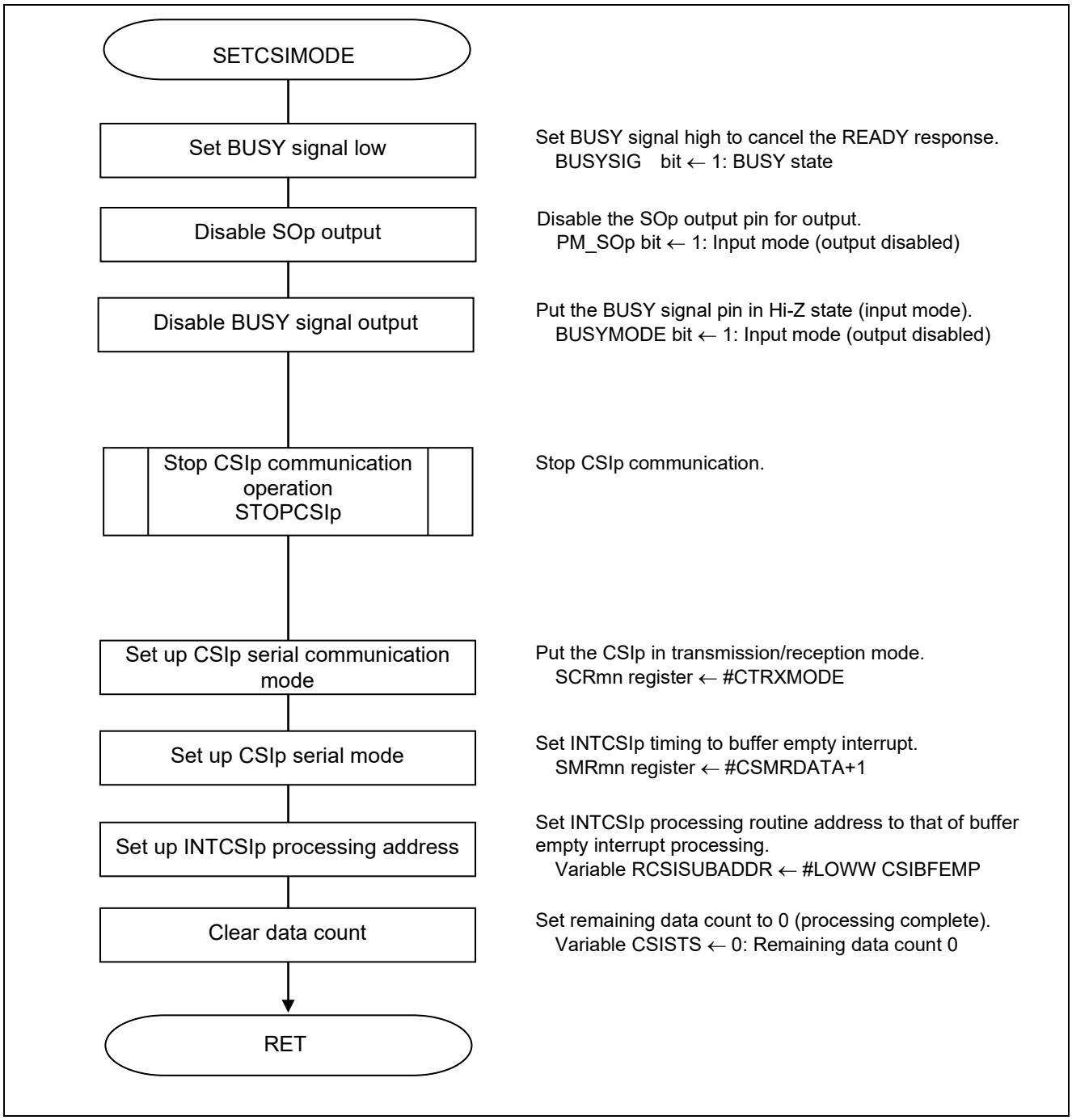


Figure 5.9 CSIp Select Release Processing Function

Port setting (For CSI00)

- Port mode register 0 (PM0)
  Disable the transmit data port for output.

Symbol: PM0

7	6	5	4	3	2	1	0
PM07[Note]	PM06[Note]	PM05[Note]	PM04	PM03	PM02	PM01	PM00
x	x	x	x	x	x	x	**1**

Bit 0

PM00	P00 I/O mode selection
0	Output mode (output buffer on)
1	**Input mode (output buffer off)**

Port setting (P04 for handshaking)

- Port register 0 (P0)
- Port mode register 0 (PM0)
  Set the BUSY signal port to high level, then disable its output.

Symbol: P0

7	6	5	4	3	2	1	0
P07[Note]	P06 [Note]	P05 [Note]	P04	P03	P02	P01	P00
x	x	x	**1**	x	x	x	x

Bit 4

P04	Output data control (in output mode)
0	Output 0 (READY)
1	**Output 1 (BUSY)**

Symbol: PM0

7	6	5	4	3	2	1	0
PM07[Note]	PM06[Note]	PM05[Note]	PM04	PM03	PM02	PM01	PM00
x	x	x	x	**1**	x	x	x

Bit 4

PM04	P04 (BUSY signal output) I/O mode selection
0	Output mode (output buffer on)
1	**Input mode (output buffer off)**

Note: 16-pin products only.

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

Transiting to channel stop state (for CSI00)

- Serial channel stop register 0 (ST0)
  Stop communication.

Symbol: ST0

7	6	5	4	3	2	1	0
0	0	0	0	0	0	ST01	ST00
0	0	0	0	0	0	**1**	**1**

Bits 1-0

ST0n	Operation stop trigger of channel 00
0	No trigger operation
1	**Clears the SE00 bit to 0 and stops the communication operation**

Setting up the channel to stop communication processing

- Serial communication operation register mn (SCR0nH, SCR0nL)
  Setup data length, data transfer order, and operation mode.

Symbol: SCR0nH

7	6	5	4	3	2	1	0
TXE 0n	RXE 0n	DAP 0n	CKP 0n	0	EOC 0n	PTC 0n1	PTC 0n0
1	**1**	**0**	**0**	0	**0**	**0**	**0**

Symbol: SCR0L

7	6	5	4	3	2	1	0
DIR 0n	0	SLC 0n1 Note	SLC 0n0	0	1	1	DLS 0n0
0	0	**0**	**0**	0	1	1	**1**

Bits 7 and 6 (SCR0nH)

TXE0n	RXE0n	Setting of operation mode of channel n
0	0	Disable communication
0	1	Reception only
1	0	Transmission only
1	**1**	**Transmission/reception**

Bit 2 (SCR0nH)

EOC0n	Selection of masking of error interrupt signal (INTSREn)
0	**Masks error interrupt INTSRE0.**
1	Enables generation of error interrupt INTSREx.

Bits 1 and 0 (SCR0nH)

PTCmn 1	PTCmn 0	Setting of parity bit in UART mode	
		Transmission	Reception
0	**0**	**Does not output the parity bit.**	**Receives without parity.**
0	1	Outputs 0 parity.	No parity judgment
1	0	Outputs even parity.	Judged as even parity
1	1	Outputs odd parity.	Judged as odd parity

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

Symbol: SCR0nH

7	6	5	4	3	2	1	0
TXE 0n	RXE 0n	DAP 0n	CKP 0n	0	EOC 0n	PTC 0n1	PTC 0n0
1	**1**	**0**	**0**	0	**0**	**0**	**0**

Symbol: SCR0nL

7	6	5	4	3	2	1	0
DIR 0n	0	SLC 0n1 Note	SLC 0n0	0	1	DLS 0n1	DLS 0n0
0	0	**0**	**0**	0	1	**1**	**1**

Bit 7 (SCR0nL)

DIR0n	Selection of data transfer sequence in CSI and UART modes
0	**Inputs/outputs data with MSB first.**
1	Inputs/outputs data with LSB first.

Bits 5 and 4 (SCR0nL)

SLC0n1	SLC0n0	Setting of stop bit in UART mode
0	**0**	**No stop bit**
0	1	Stop bit length = 1 bit
1	0	Stop bit length = 2 bits
1	1	Setting prohibited

Bits 0 (SCR0nL)

DLS0n0	Setting of data length in CSI mode
0	7-bit data length
1	**8-bit data length**

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

Setting up channel operation mode

- Serial mode register 0n (SMR0n)
  Interrupt source, Operation mode, Select transfer clock, Select f_{MCK}.

Symbol: SMR0nH

7	6	5	4	3	2	1	0
CKS 0n	CCS 0n	0	0	0	0	0	STS 0n Note
0	**1**	0	0	0	0	0	**0**

Symbol: SMR0nL

7	6	5	4	3	2	1	0
0	SIS 0n0 Note	1	0	0	MD 0n2	MD 0n1	MD 0n0
0	**0**	1	0	0	**0**	**0**	**1**

Bit 7 (SMR0nH)

CKS0n	Selection of operation clock (f_{MCK}) of channel n
0	**Prescaler output clock CK00 set by the SPSm register**
1	Prescaler output clock CK01 set by the SPSm register

Bit 6 (SMR0nH)

CCS0n	Selection of transfer clock (TCLK) of channel n
0	Divided operation clock f_{MCK} set by the CKSmn bit
1	**Clock input from the SCK pin.**

Bit 0 (SMR0nH)

STS0n Note	Selection of start trigger source
0	**Only software trigger is valid**
1	Valid edge of the RxD pin (selected for UART reception)

Bit 6 (SMR0nL)

SIS 0n0	Controls inversion of level of receive data of channel n in UART mode
0	**Falling edge is detected as the start bit.**
1	Rising edge is detected as the start bit.

Bits 2 and 1 (SMR0nL)

MDmn2	MDmn1	Setting of operation mode of channel n
0	**0**	**CSI mode**
0	1	UART mode
1	0	Simplified I^2C mode
1	1	Setting prohibited

Bit 0 (SMR0nL)

MDmn0	Selection of interrupt source of channel n
0	Transfer end interrupt
1	**Buffer empty interrupt**

Remark: n: channel number (n=0, 1)
Note: SMR01H, SMR01L registers only.
Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

Interrupt setting (for CSI00)

- Interrupt request flag register (IF0L)
  Clear the interrupt request flag
- Interrupt mask flag register (MK0L)
  Set up interrupt mask

Symbol: IF0L (10-pin products)

7	6	5	4	3	2	1	0
TMIF00	TMIF01H	SREIF0	SRIF0	STIF0 CSIIF00 IICIF00	PIF1	PIF0	WDTIIF
x	x	x	x	**0**	x	x	x

Bit 3

CSIIF00	Interrupt request flag
0	**No interrupt request signal is generated**
1	Interrupt request is generated, interrupt request status

Symbol: MK0L (10-pin products)

7	6	5	4	3	2	1	0
TMMK00	TMMK01H	SREMK0	SRMK0	SRMK0 CSIMK00 IICMK00	PMK1	PMK0	WDTIMK
x	x	x	x	**1**	x	x	x

Bit 3

CSIMK00	Interrupt processing control
0	Enables interrupt processing.
1	**Disables interrupt processing.**

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

5.7.8 Status Check Command Processing

Figure 5.10 shows the flowchart for the status check command processing function.

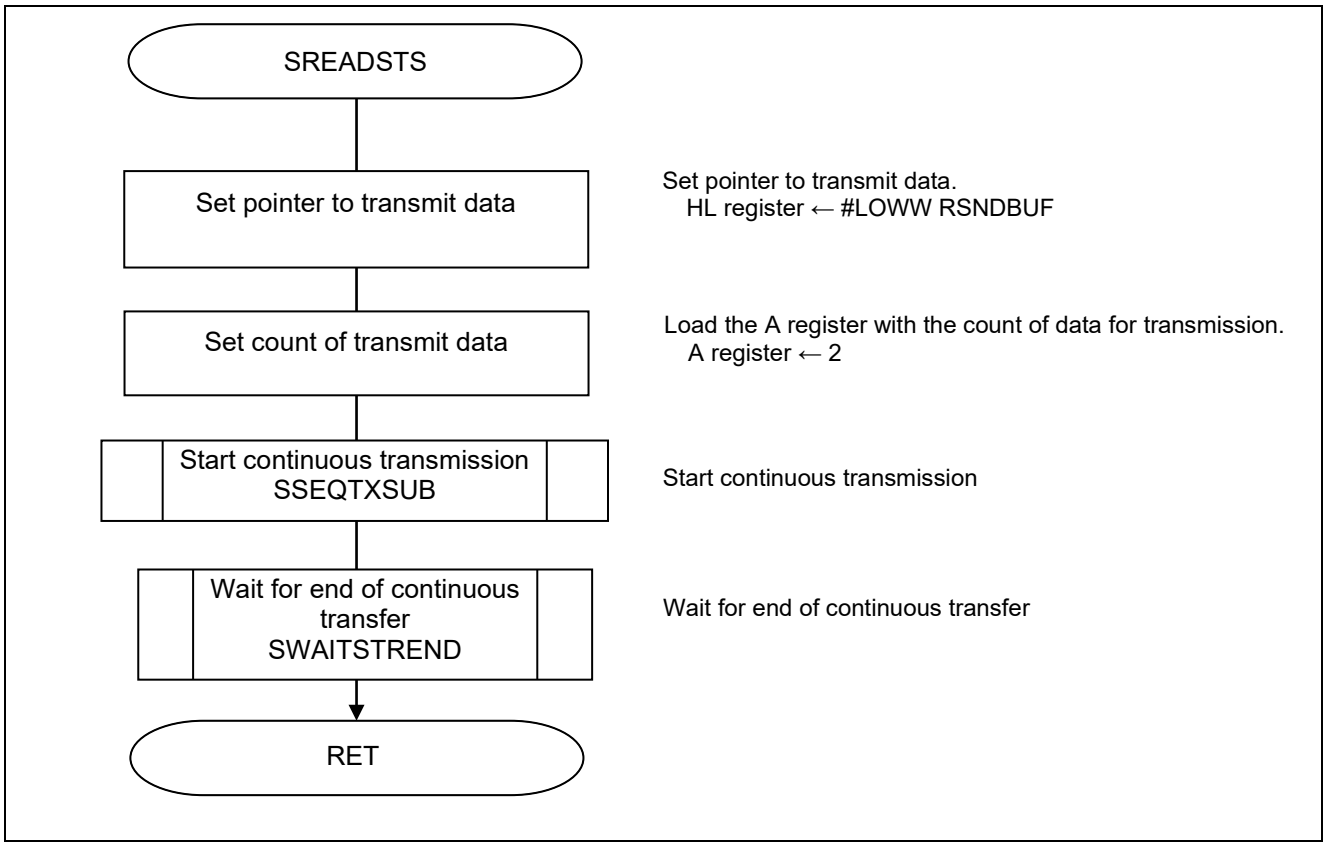


Figure 5.10 Status Check Command Processing Function

5.7.9 Master Receive Command Processing

Figure 5.11 shows the flowchart for the master receive command processing function.

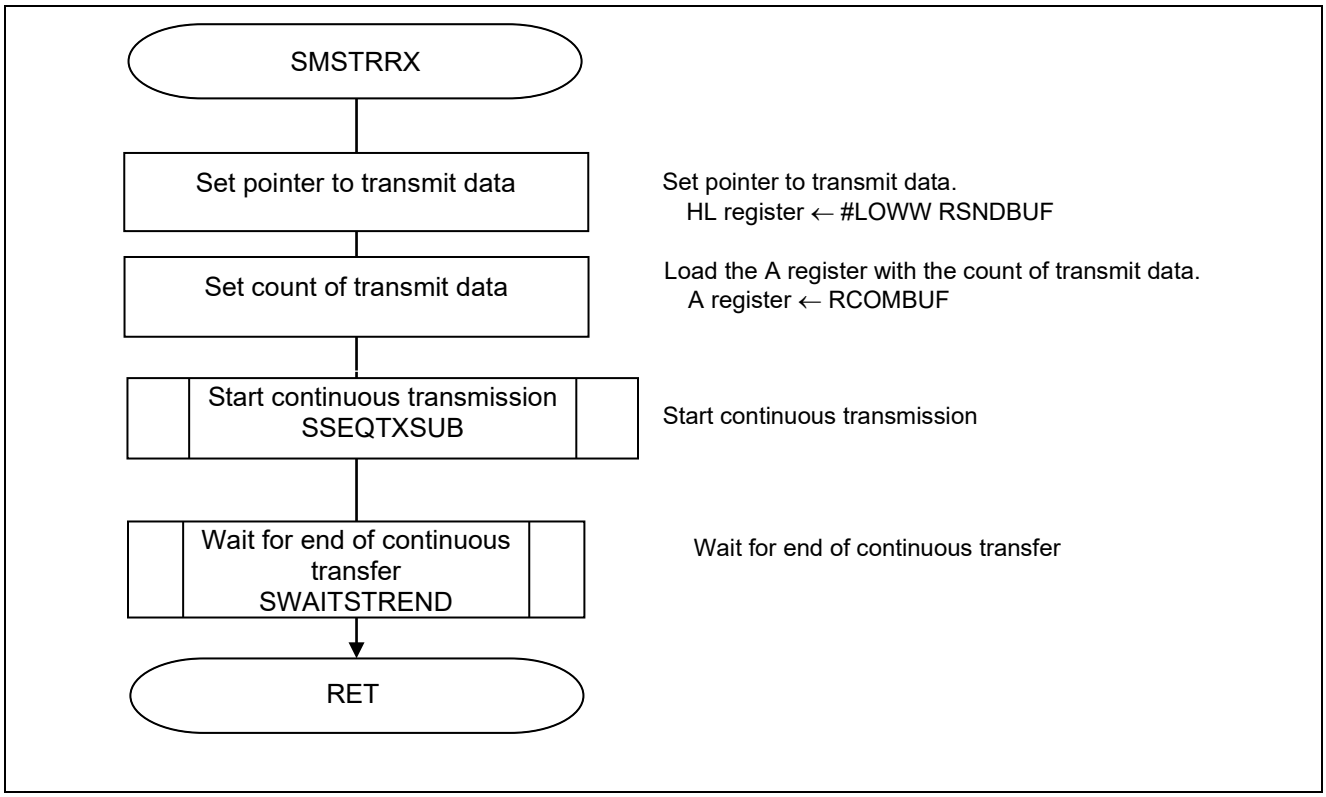


Figure 5.11 Master Receive Command Processing Function

5.7.10 Master Transmit Command Processing Function

Figure 5.12 shows the flowchart for the master transmit command processing function

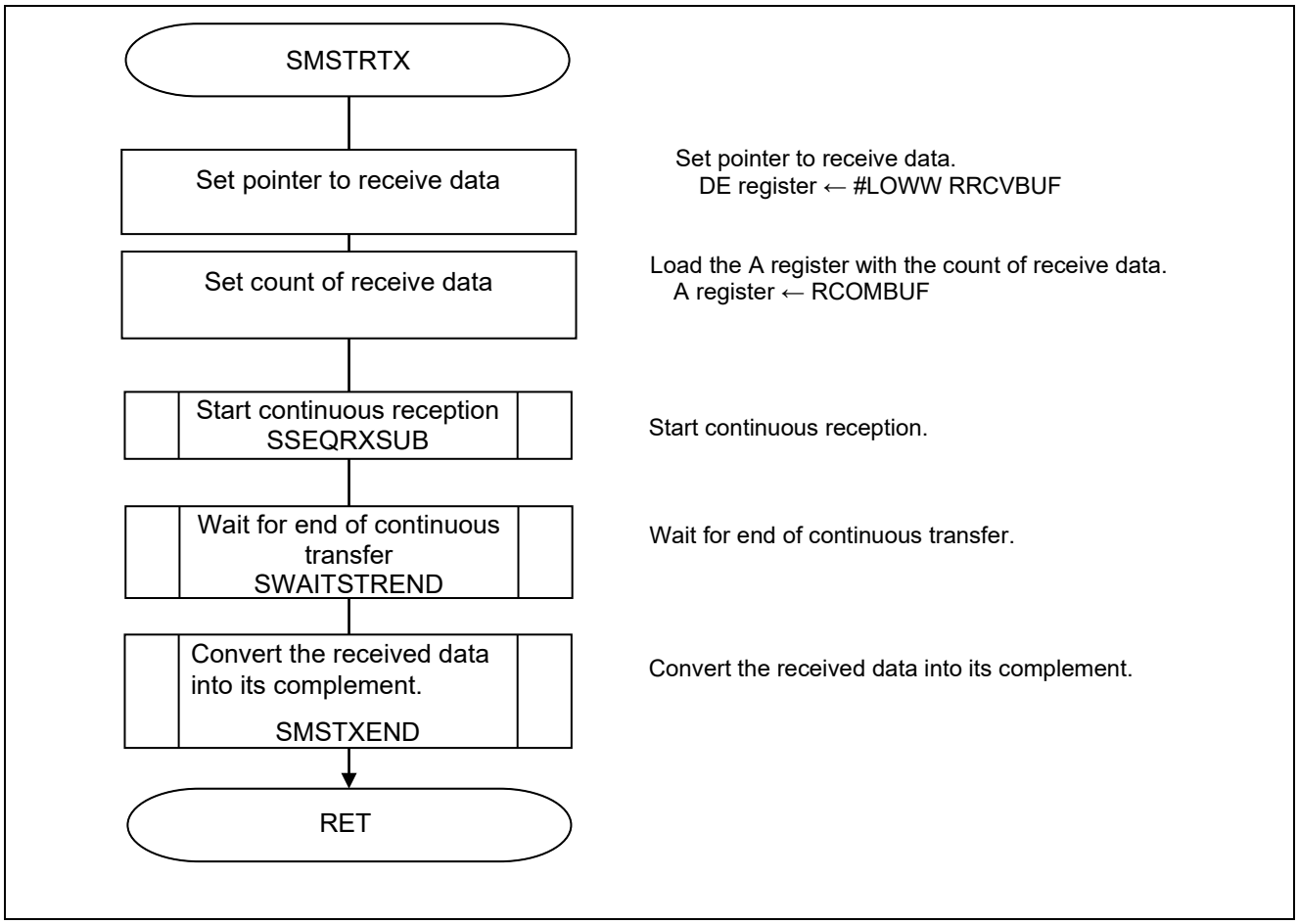


Figure 5.12 Master Tramsmit Command Processing Function

5.7.11 Transmit/Receive Command Processing

Figure 5.13 shows the flowchart for the transmit/receive command processing function.

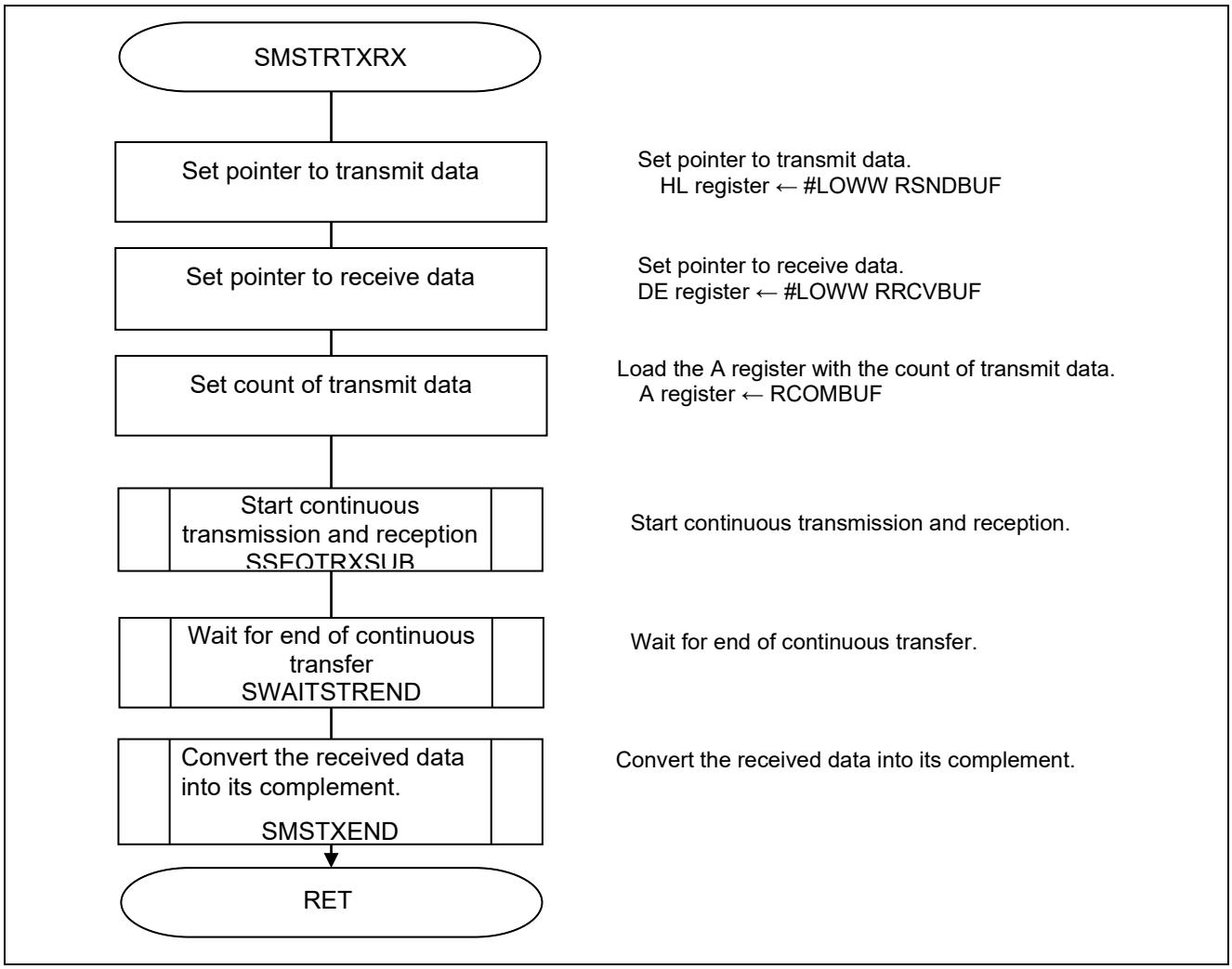


図 5.13 送受信コマンド処理関数

5.7.12 Received Data Complement Processing

Figure 5.14 shows the flowchart for the received data complement processing function

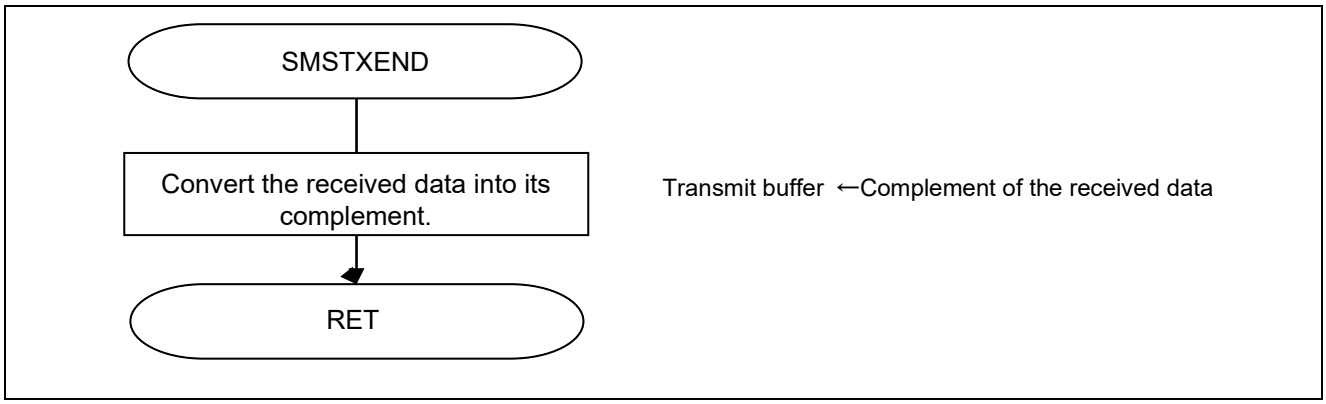


Figure 5.14 Received Data Complement Processing Function

5.7.13 INTP0 Interrupt Processing Function

Figure 5.10 shows the flowchart for the INTP0 interrupt processing function.

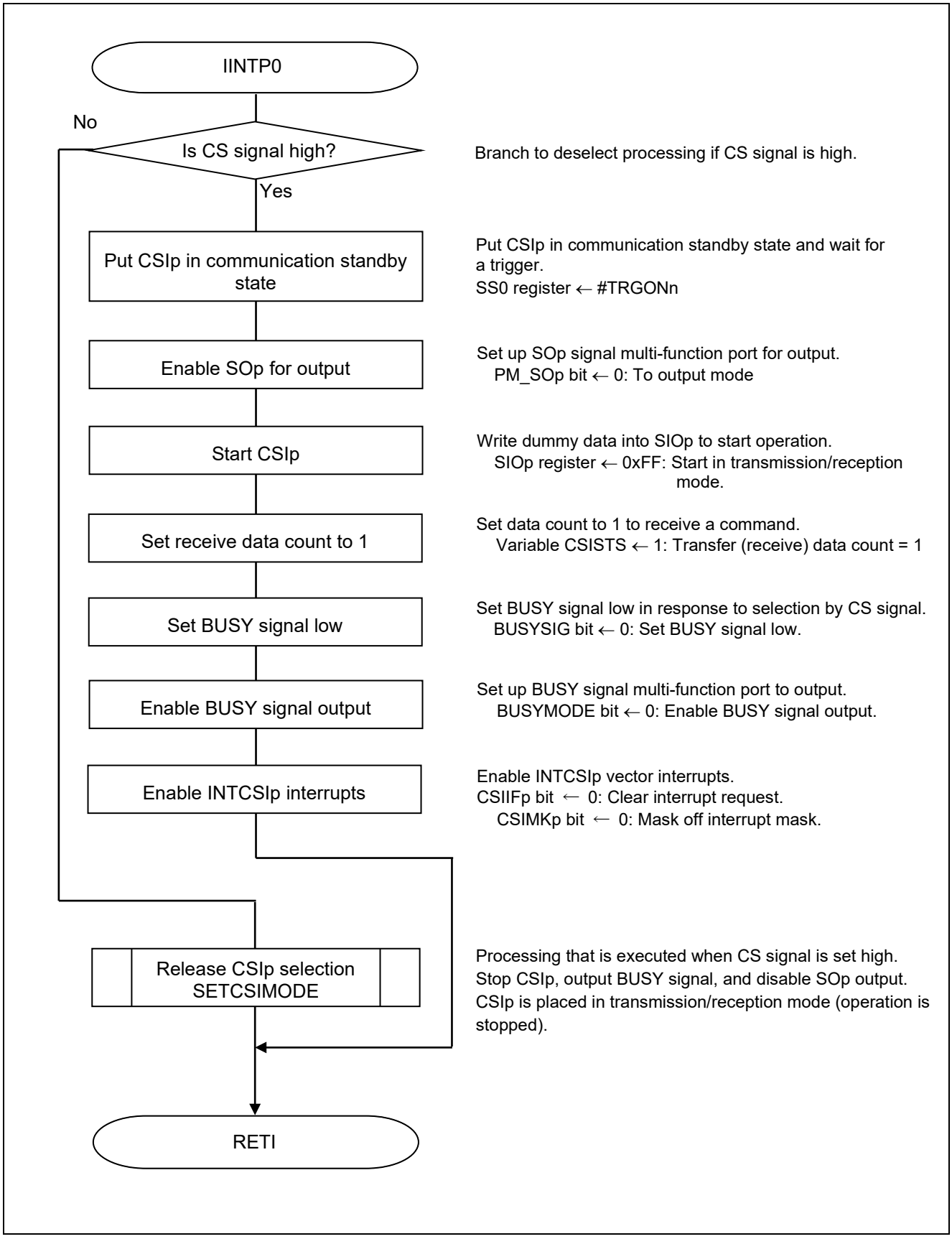


Figure 5.10 INTP0 Interrupt Processing Function

Putting channel into communication standby state (For CSI00)

- Serial channel start register 0 (SS0)
  Put channel in communication standby state.

Symbol: SS0

7	6	5	4	3	2	1	0
0	0	0	0	0	0	SS01	SS00
0	0	0	0	0	0	**1**	**1**

Bit 1-0

SS0n	Operation start trigger of channel 00
0	No trigger operation
1	**Sets the SE0n bit to 1 and enters the communication wait status.**

Port setting (For CSI00)

- Port mode register 1 (PM0)
  Set the transmit data port for output.

Symbol: PM0

7	6	5	4	3	2	1	0
PM07Note	PM06Note	PM05Note	PM04	PM03	PM02	PM01	PM00
x	x	x	x	x	x	x	**0**

Bit 0

PM00	P00 pin I/O mode selection
0	**Output mode (output buffer on)**
1	Input mode (output buffer off)

Note: 16-pin product only

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

Port setting (P04 for handshaking)

- Port register 0 (P0)
- Port mode register 0 (PM0)
  Set the BUSY signal port low, then enable it for output.

Symbol: P0

7	6	5	4	3	2	1	0
P07 Note	P06 Note	P05 Note	P04	P03	P02	P01	P00
x	x	x	**0**	x	x	x	x

Bit 3

P04	Output data control (in output mode)
0	**Output 0 (READY)**
1	Output 1 (BUSY)

Symbol: PM0

7	6	5	4	3	2	1	0
PM07Note	PM06Note	PM05Note	PM04	PM03	PM02	PM01	PM00
x	x	x	**0**	x	x	x	x

Bit 4

PM04	P04 (BUSY signal output) I/O mode selection
0	**Output mode (output buffer on)**
1	Input mode (output buffer off)

Note: 16-pin products only

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

Interrupt setting (for CSI00)

- Interrupt request flag register (IF0L)
  Clear the interrupt request flag
- Interrupt mask flag register (MK0L)
  Cancel interrupt mask

Symbol: IF0L (10-pin products)

7	6	5	4	3	2	1	0
TMIF00	TMIF01H	SREIF0	SRIF0	STIF0 CSIIF00 IICIF00	PIF1	PIF0	WDTIIF
x	**0**	x	x	x	x	x	x

Bit 6

CSIIF00	Interrupt request flag
0	**No interrupt request signal is generated**
1	Interrupt request is generated, interrupt request status

Symbol: MK0L (10-pin products)

7	6	5	4	3	2	1	0
TMMK00	TMMK01H	SREMK0	SRMK0	STMK0 CSIMK00 IICMK00	PMK1	PMK0	WDTIMK
x	**0**	x	x	x	x	x	x

Bit 6

CSIMK00	Interrupt processing control
0	**Enables interrupt processing.**
1	Disables interrupt processing.

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

5.7.14 INTCSIp Interrupt Entry Processing

Figure 5.11 shows the flowchart for INTCSIp interrupt entry processing.

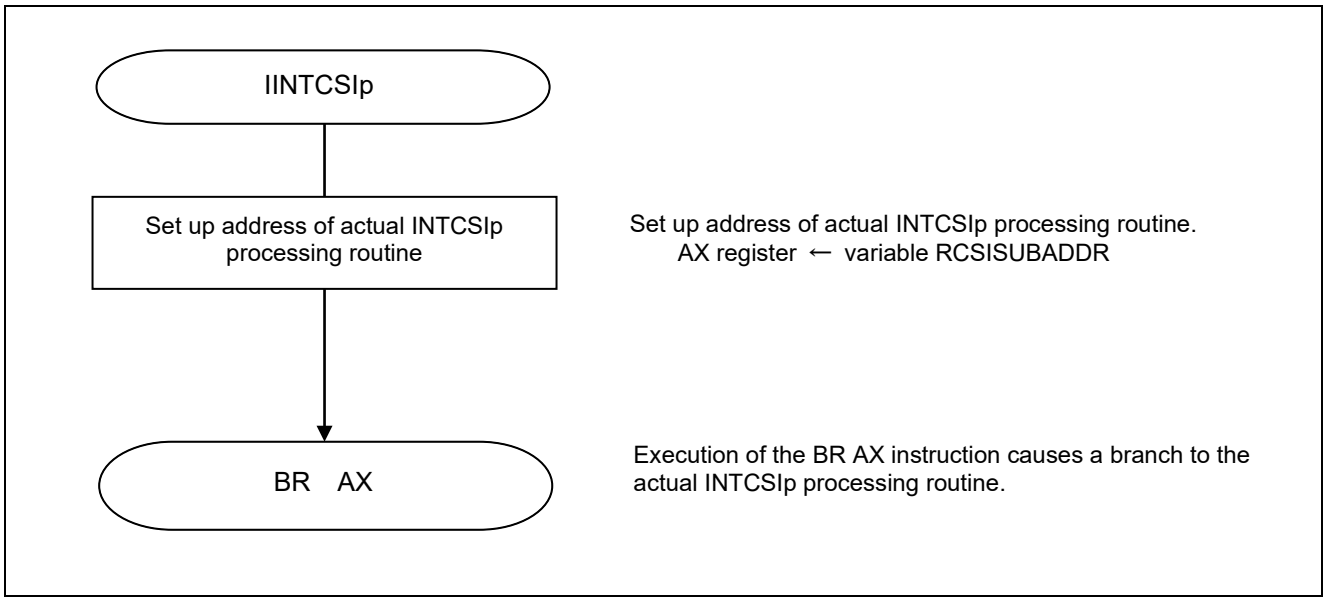


Figure 5.11 INTCSIp Interrupt Entry Processing

5.7.15 1-character Transfer Start Interrupt Processing

Figure 5.12 shows the flowchart for 1-character transfer start interrupt processing.

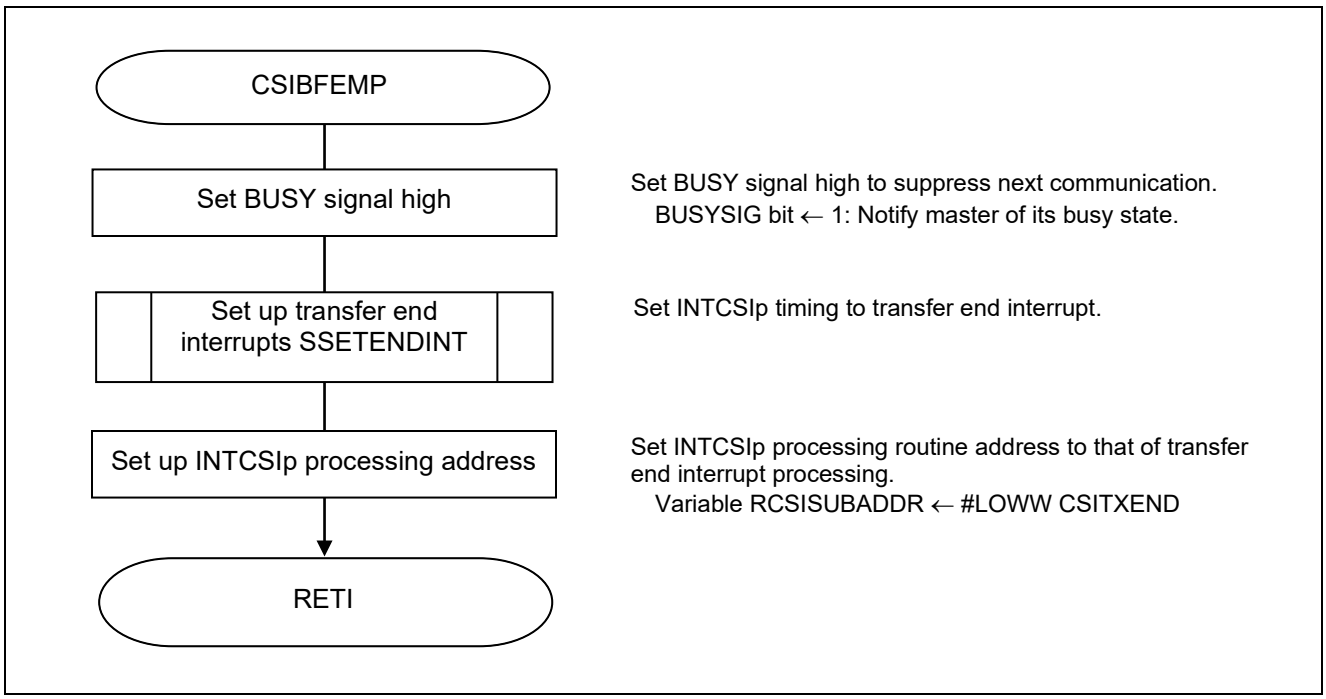


Figure 5.12 1-character Transfer Start Interrupt Processing

5.7.16 1-character Transmit End Interrupt Processing

Figure 5.13 shows the flowchart for 1-character transmit end interrupt processing.

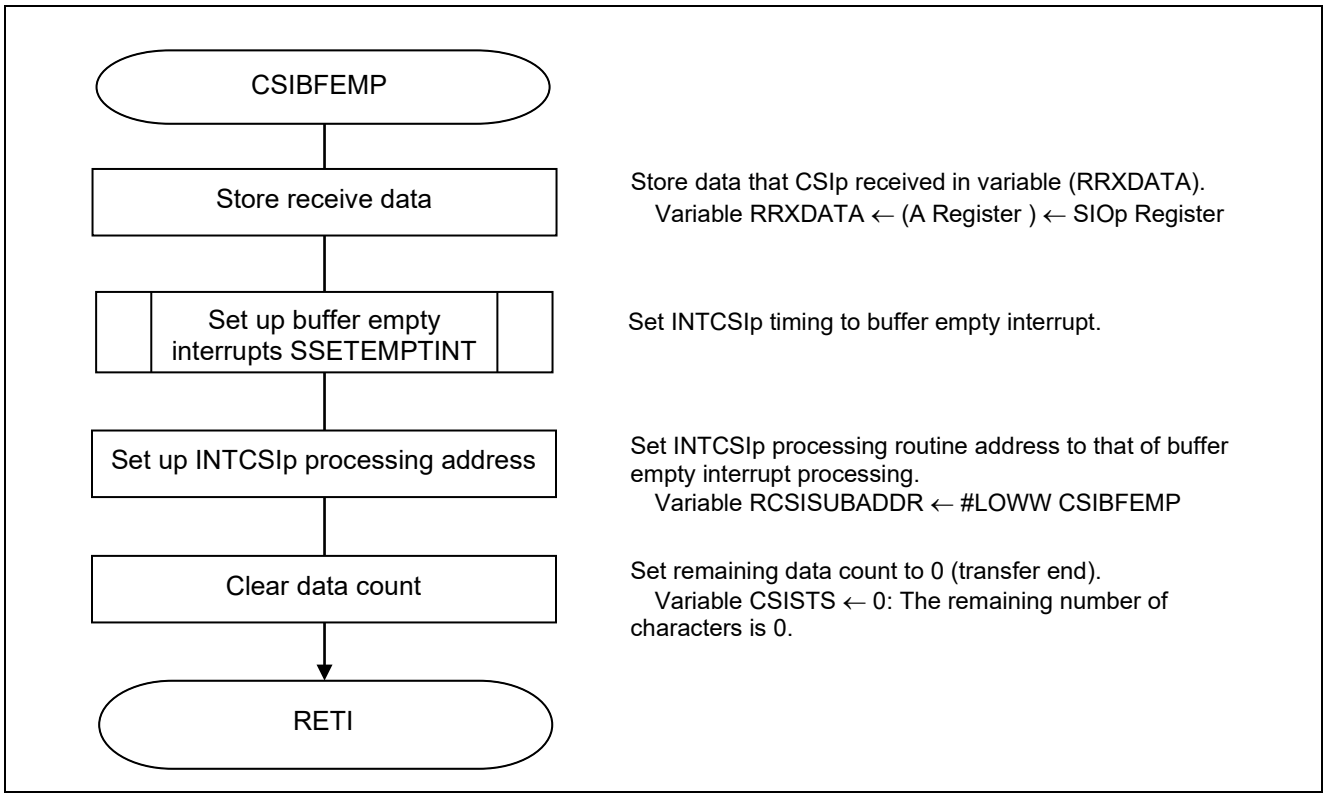


Figure 5.13 1-character Transmit End Interrupt Processing

5.7.17 Data Receive End Interrupt Processing in Continuous Reception Mode

Figure 5.14 shows the flowchart for data receive end interrupt processing in continuous reception mode.

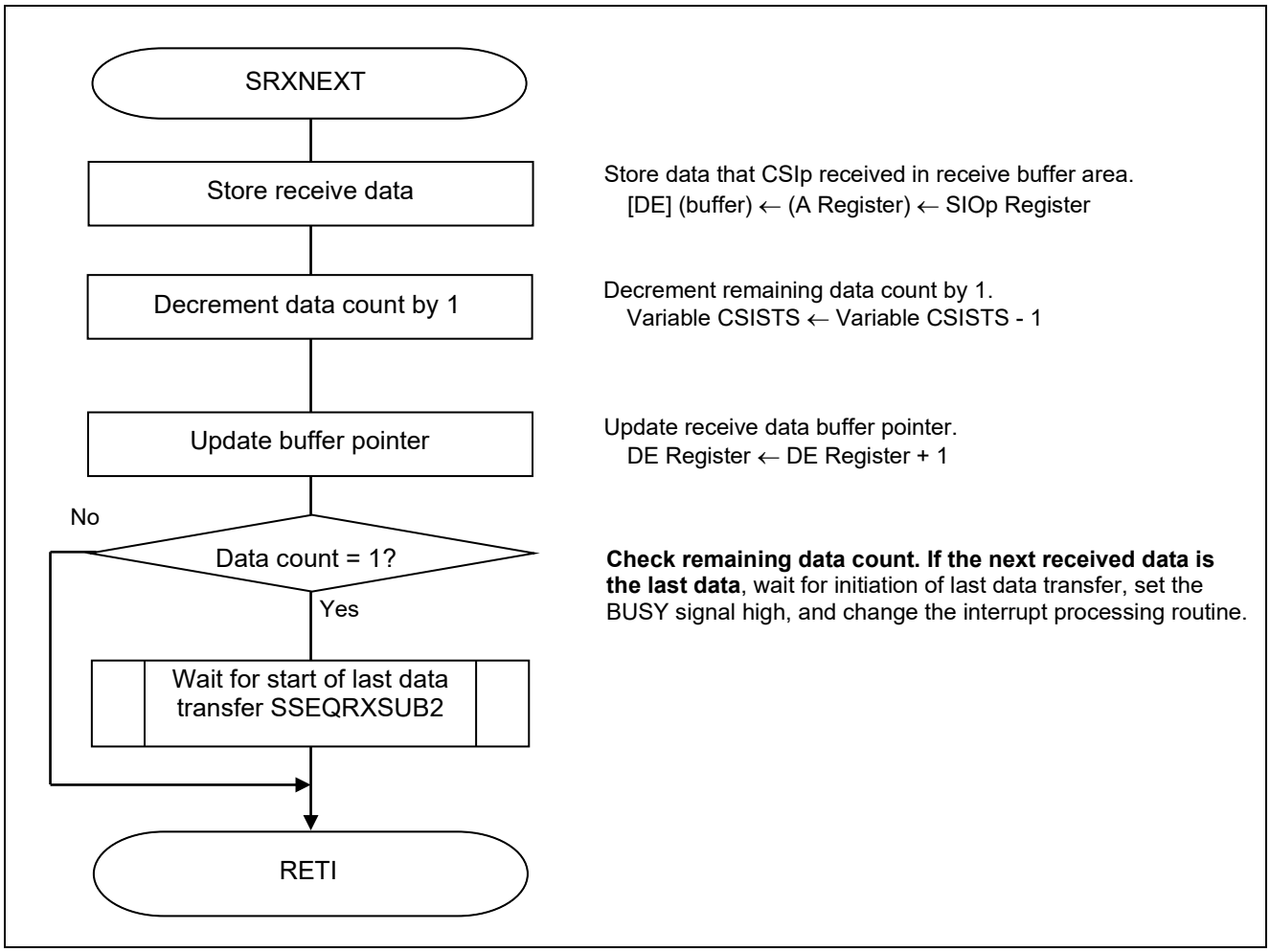


Figure 5.14 Data Receive End Interrupt Processing in Continuous Reception Mode

5.7.18 Data Transmit End Interrupt Processing in Continuous Transmission Mode

Figure 5.15 shows the flowchart for data transmit end interrupt processing in continuous transmission mode.

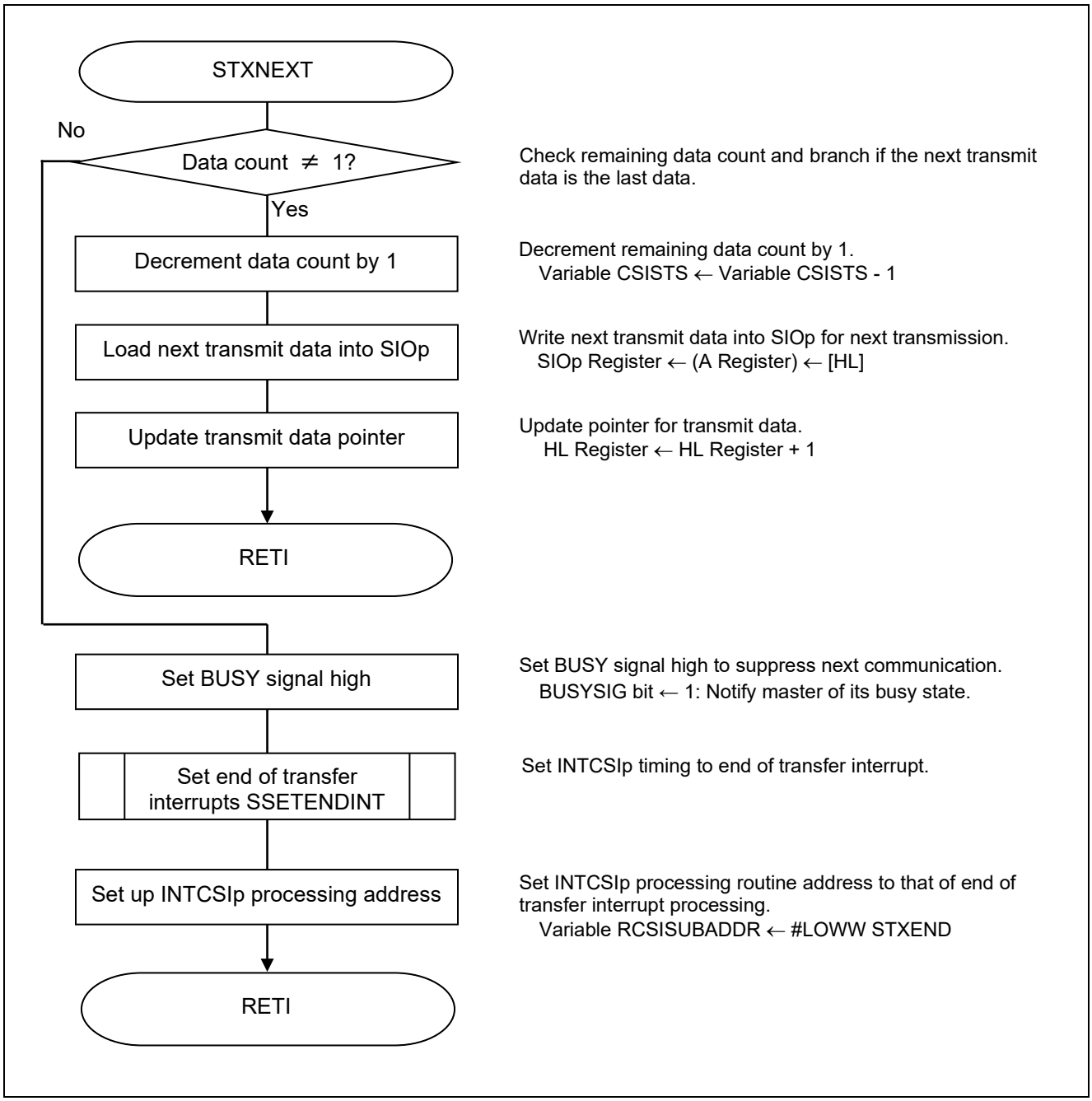


Figure 5.15 Data Transmit End Interrupt Processing in Continuous Transmission Mode

5.7.19 Receive End Interrupt Processing for Last Data in Continuous Transmission/Reception Mode

Figure 5.16 shows the flowchart for receive end interrupt processing for last data in continuous transmission/reception mode.

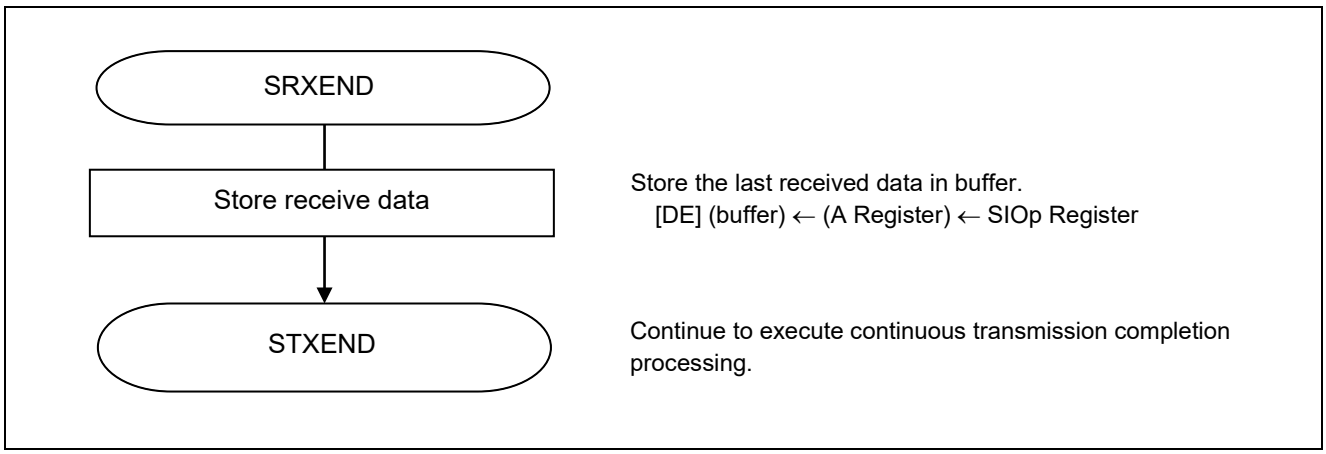


Figure 5.16 Receive End Interrupt Processing for Last Data in Continuous Transmission/Reception Mode

5.7.20 Transmit End Interrupt Processing for Last Data in Continuous Transmission Mode

Figure 5.17 shows the flowchart for transmit end interrupt processing for last data in continuous transmission mode.

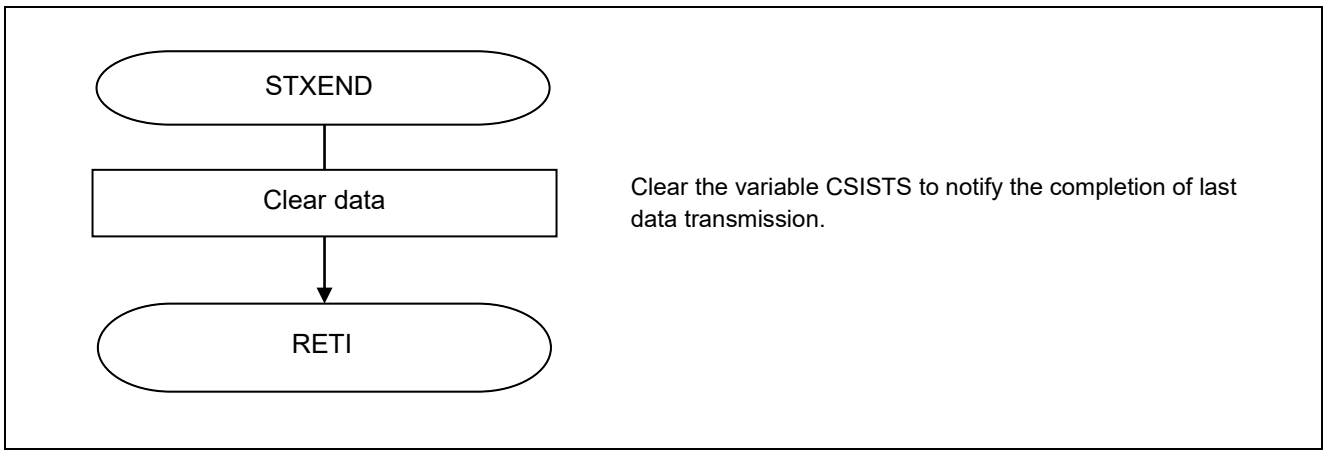


Figure 5.17 Transmit End Interrupt Processing for Last Data in Continuous Transmission Mode

5.7.21 Communication Start Interrupt Processing in Continuous Transmission/Reception Mode

Figure 5.19 shows the flowchart for communication start interrupt processing in continuous transmission/reception mode.

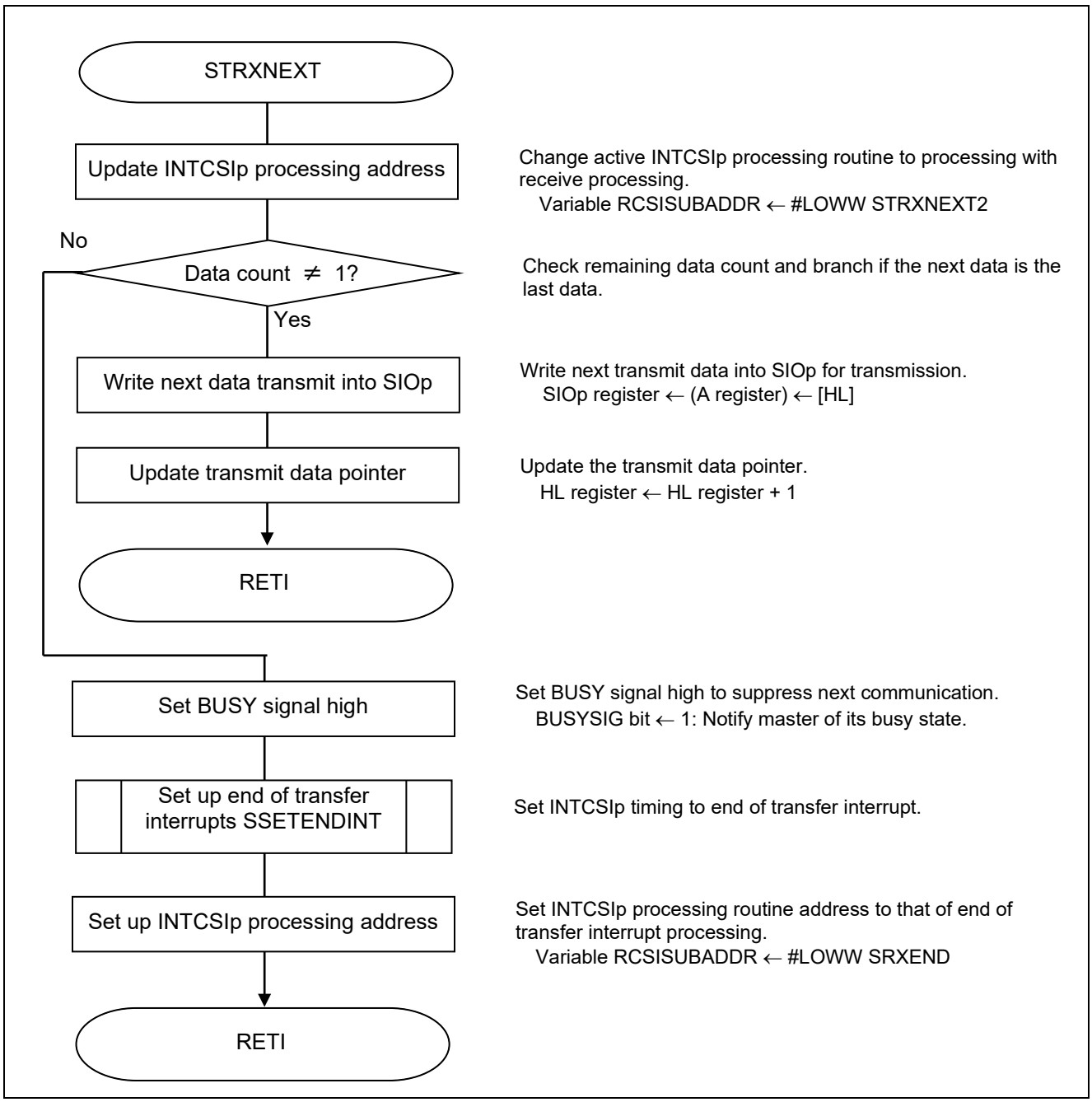


Figure 5.19 Communication Start Interrupt Processing in Continuous Transmission/Reception Mode

5.7.22 Buffer Empty Interrupt Processing in Continuous Transmission/Reception Mode

Figure 5.18 shows the flowchart for buffer empty interrupt processing in continuous transmission/reception mode.

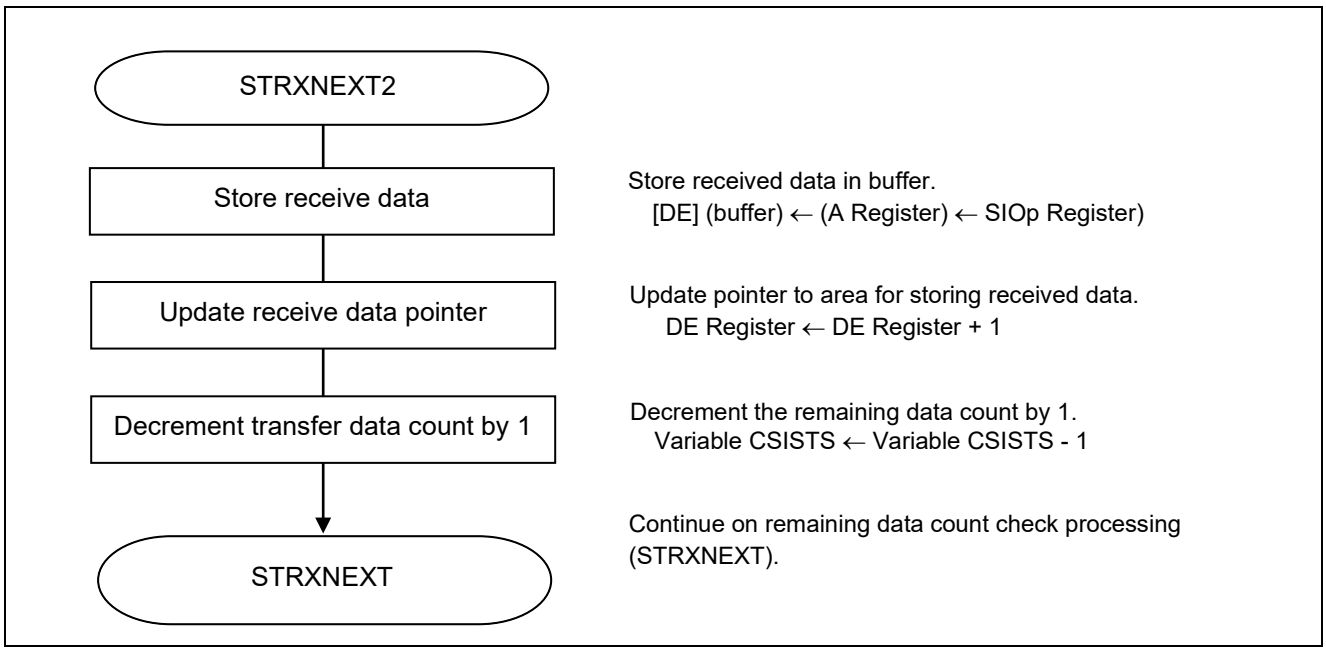


Figure 5.18 Buffer Empty Interrupt Processing in Continuous Transmission/Reception Mode

5.7.23 1-Character Transfer State Check Function

Figure 5.24 shows the flowchart for the 1-character transfer state check Function.

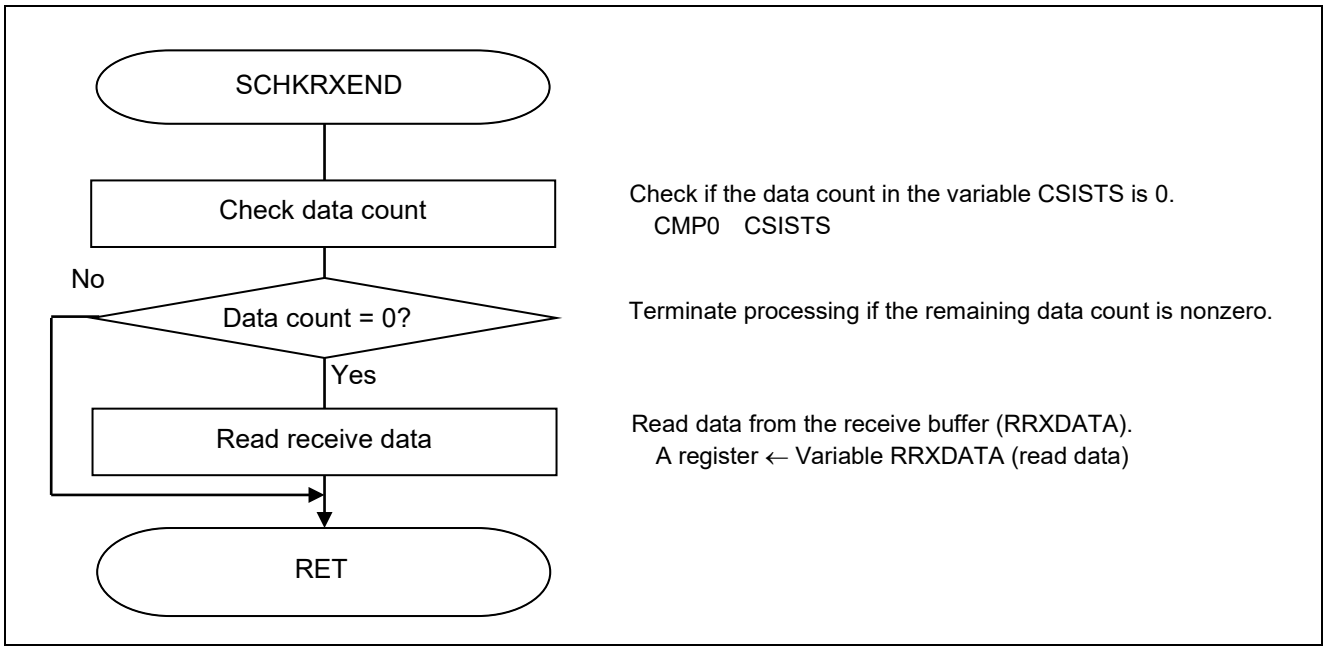


Figure 5.24 1-Character Transfer State Check Function

Given below is a collection of subroutines that are used for basic continuous data communication processing. Two functions, for start and wait processing, are used in pair. Set up the parameters given below when invoking startup processing. The CSIp communication mode is automatically set up.

Continuous transmission processing

HL register = Address of the transmit buffer

A register = Transmit data count (1 to 255)

Continuous reception processing

HL register = Address of the buffer for storing the received data

A register = Receive data count (1 to 255)

Continuous transmission/reception processing

HL register = Address of the transmit buffer

DE register = Address of the buffer for storing the received data

A register = Transmit data count (1 to 255)

5.7.24 1-character Transmission Start Processing Function 1

Figure 5.20 shows the flowchart for 1-character transmission start processing function 1.

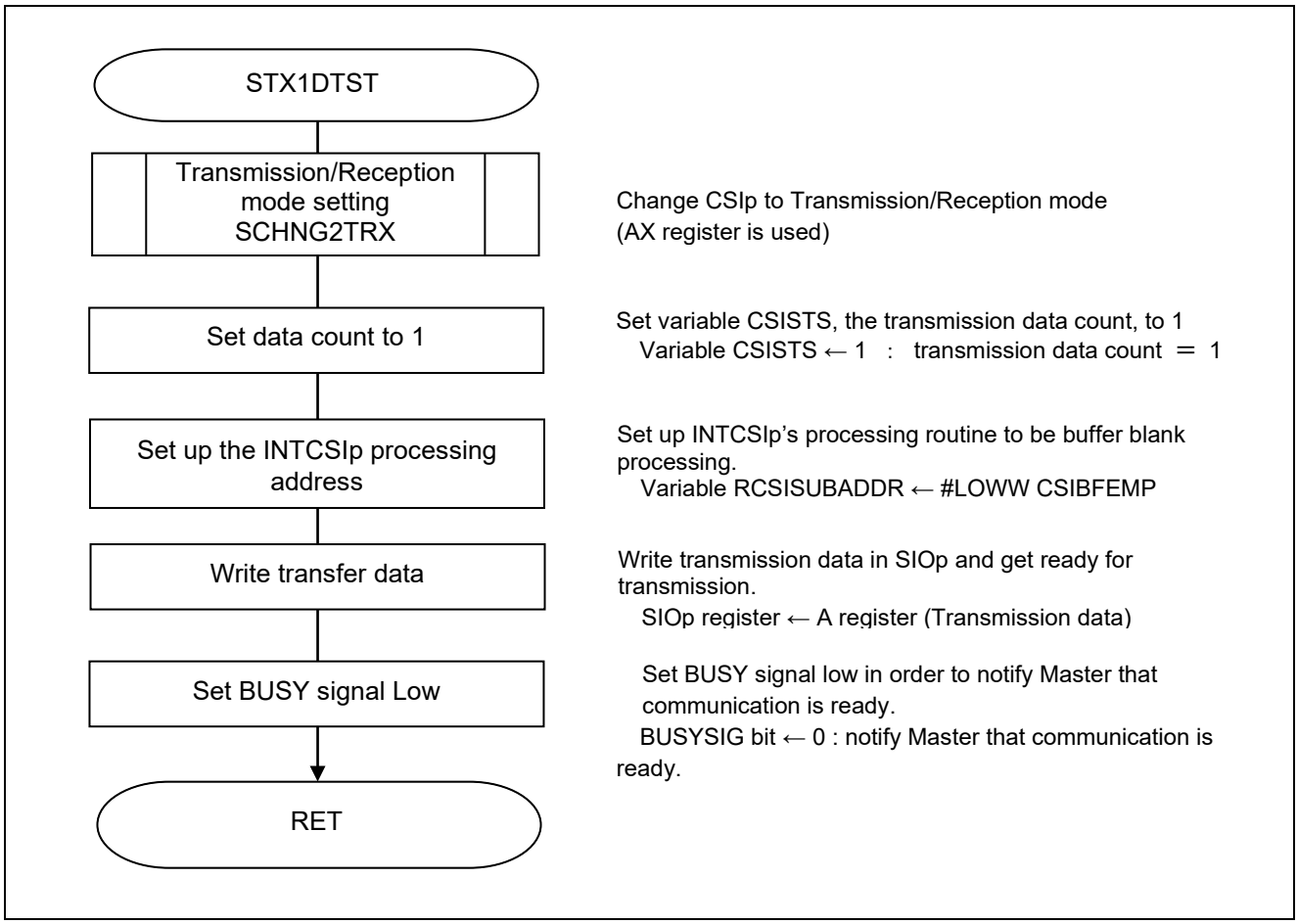


Figure 5.20 1-character Transmission Start Processing Function 1

1-character Reception Start Processing Function

Figure 5.22 shows the flowchart for the 1-character reception start processing function.

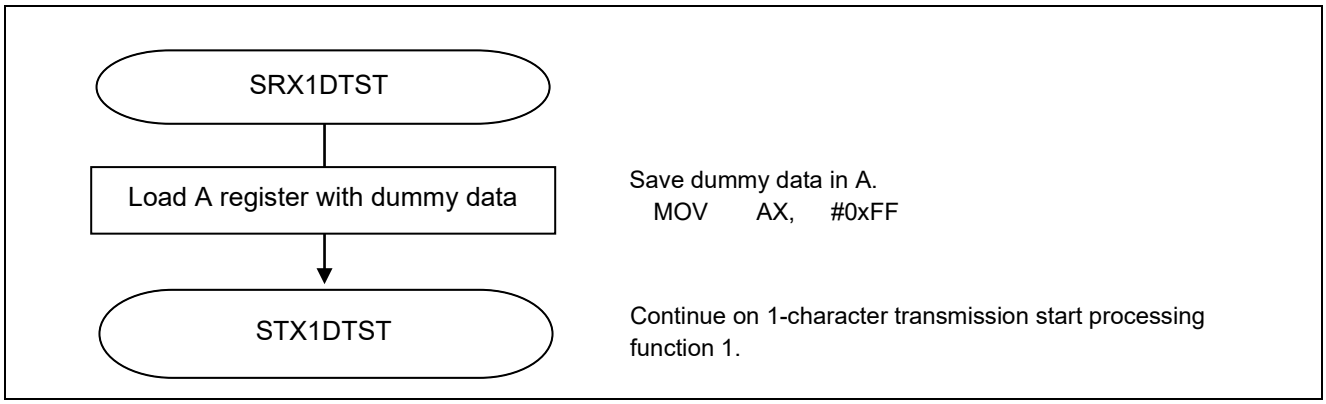


Figure 5.22 1-character Reception Start Processing Function

5.7.25 1-character Transmission/Reception End Wait Processing Function

Figure 5.23 shows the flowchart for the 1-character transmission/reception end wait processing function.

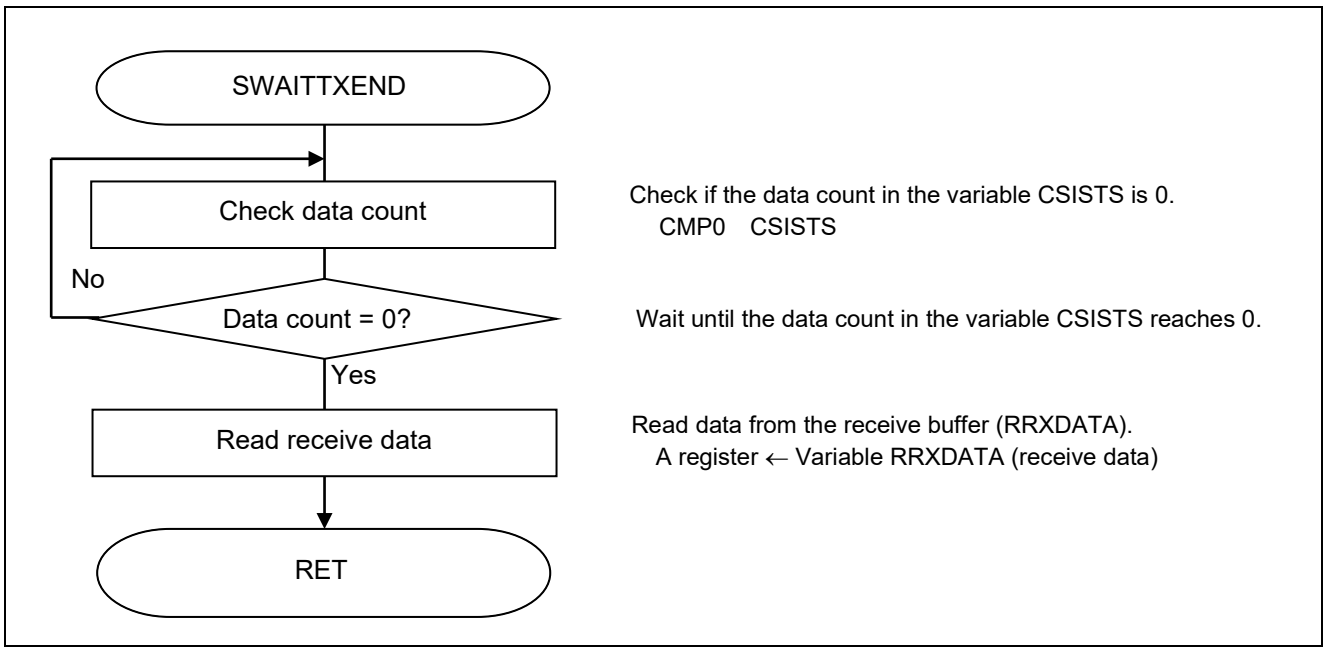


Figure 5.23 1-character Transmission/Reception End Wait Processing Function

5.7.26 Continuous Transmission Start Processing

Figure 5.27 shows the flowcharts for continuous transmission start processing

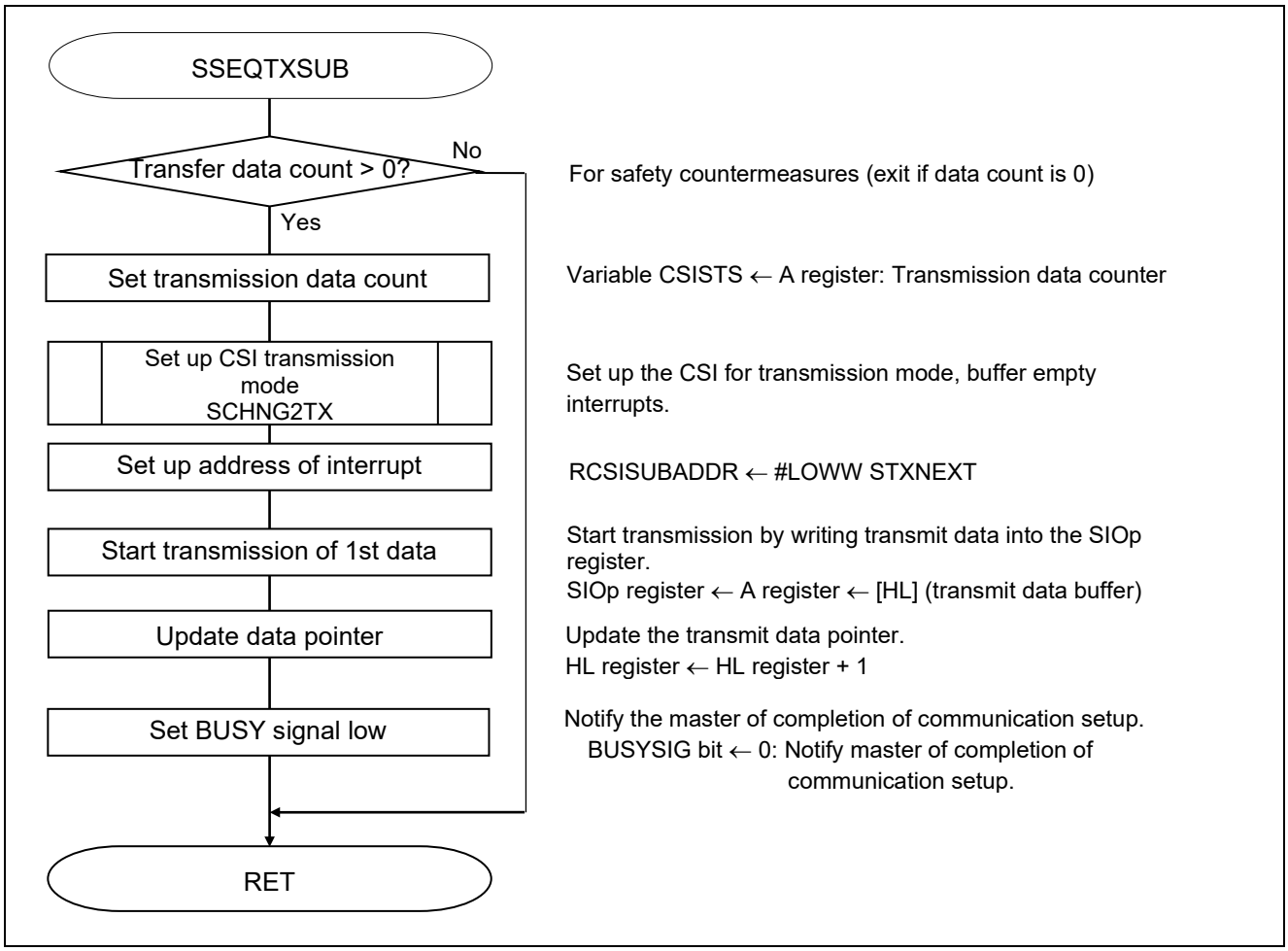


Figure 5.27 Continuous Transmission Start Processing

5.7.27 Continuous Reception Start Processing

Figure 5.25 shows the flowchart for continuous reception start processing.

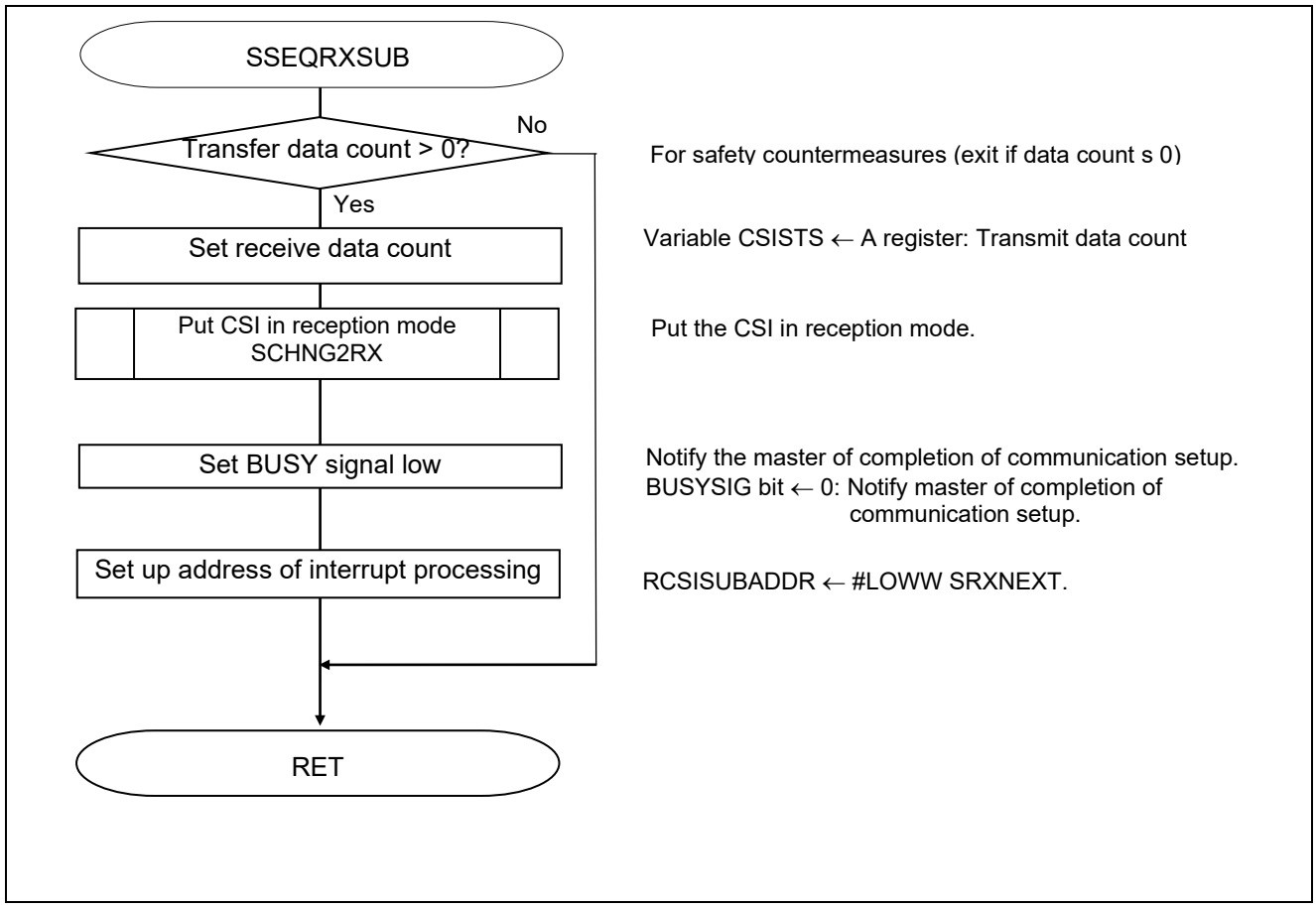


Figure 5.25 Continuous Reception Start Processing

5.7.28 Continuous Transmission/Reception Start Processing

Figure 5.28 shows the flowchart for continuous transmission/reception start processing.

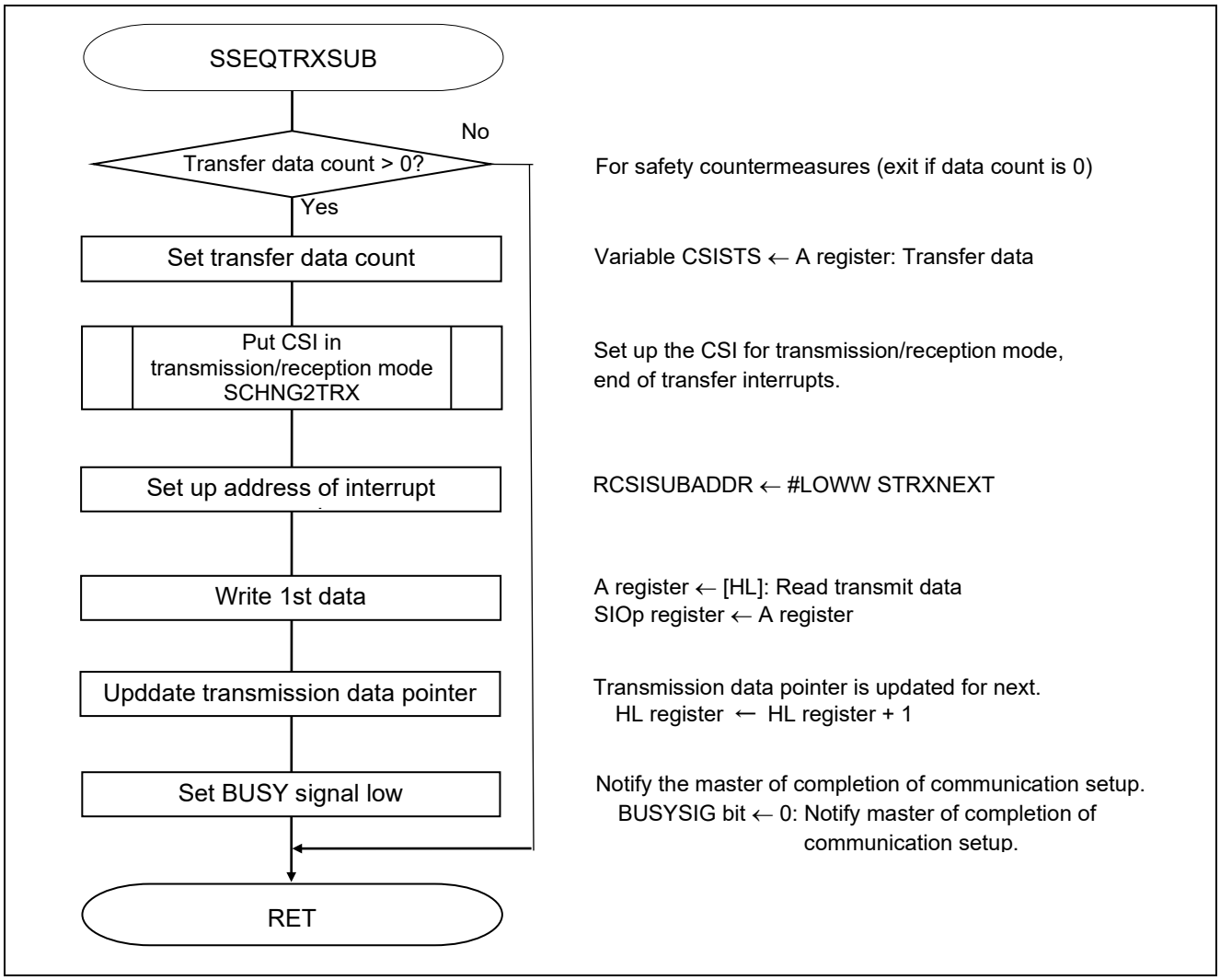


Figure 5.28 Continuous Transmission/Reception Start Processing

5.7.29 Last Data Transfer Start Wait Processing Function

Figure 5.32 shows the flowchart for last data transfer start wait processing.

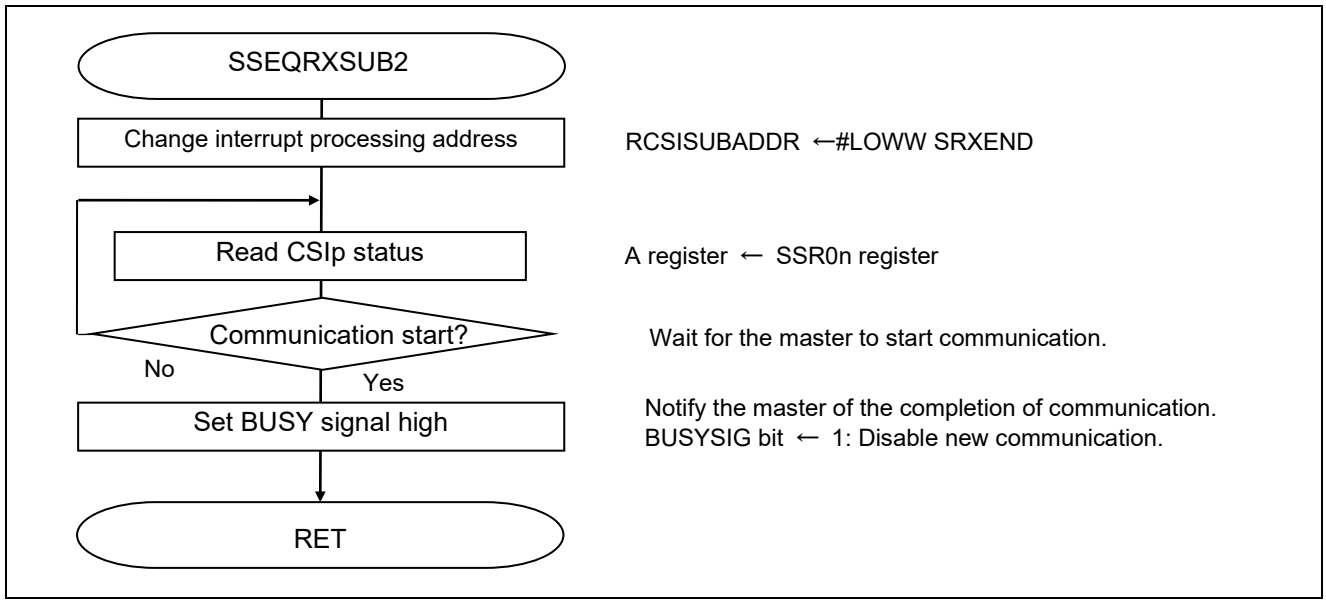


Figure 5.32 Last Data Transfer Start Wait Processing

Checking communication status

- Serial status register 0n (SSR0n)
  Reads CSIp communication status.

Symbol：SSR0n

7	6	5	4	3	2	1	0
0	TSF0n	BFF0n	0	0	FEF0n[Note]	PEF0n	OVF0n
0	**0/1**	x	0	0	x	x	x

Bit 6

TSF0n	Communication status indication flag of channel n
0	**Communication is stopped or suspended.**
1	**Communication is in progress.**

Remark n: Channel number (n = 0, 1)

Note Provided in SSR01 register only.
Caution For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

5.7.30 Continuous Transfer End Wait Processing

Figure 5.26 shows the flowchart for continuous transfer end wait processing.

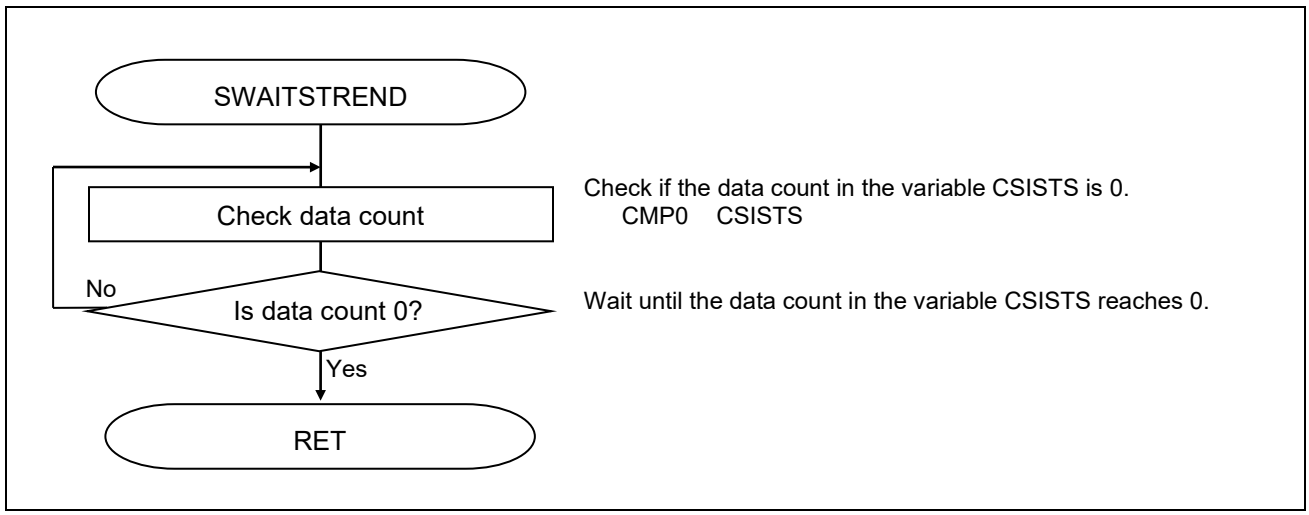


Figure 5.26 Continuous Transfer End Wait Processing

5.7.31 Transfer End Interrupt Setup Processing

Figure 5.29 shows the flowchart for transfer end interrupt setup processing.

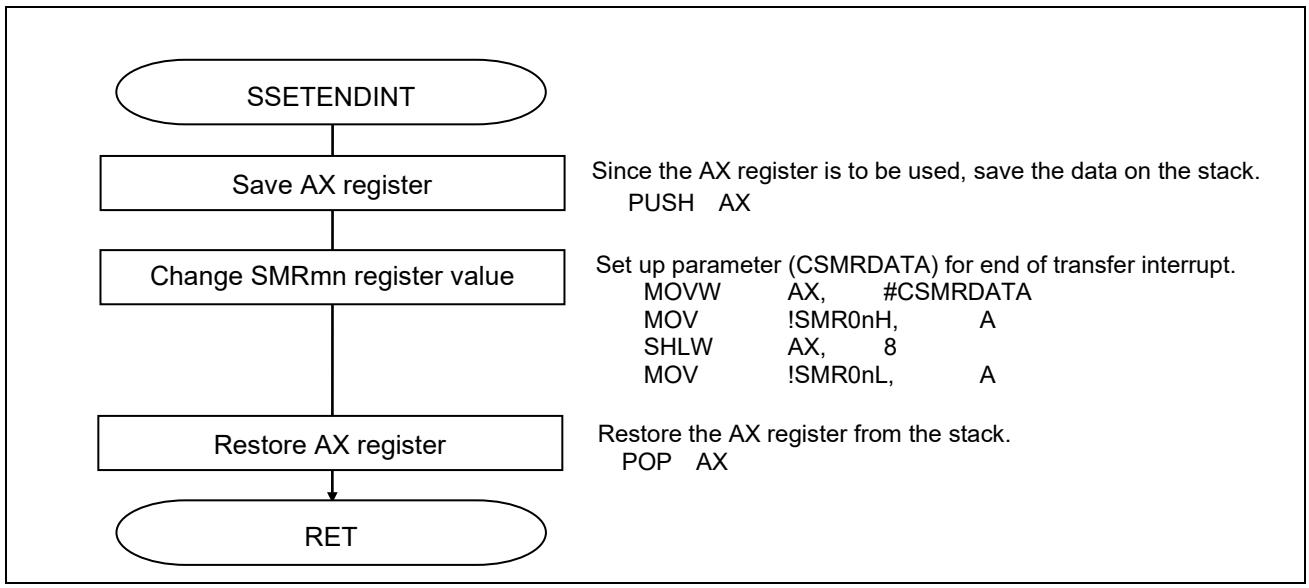


Figure 5.29 Transfer End Interrupt Setup Processing

Setting up the channel operating mode

- Serial mode register mn (SMR0nH, SMR0nL)
  Interrupt source and end of transfer interrupt

Symbol: SMR0nH

7	6	5	4	3	2	1	0
CKS 0n	CCS 0n	0	0	0	0	0	STS 0n[Note]
0	0	0	0	0	0	0	0

Symbol: SMR0nL

7	6	5	4	3	2	1	0
0	SIS 0n0[Note]	1	0	0	MD 0n2	MD 0n1	MD 0n0
0	0	1	0	0	0	0	**0**

Bit 0 (SMR0nL)

MDmn0	Selection of interrupt source of channel n
0	**Transfer end interrupt**
1	**Buffer empty interrupt**

Note: 16-pin products only.

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

5.7.32 Buffer Empty Interrupt Setup Processing

Figure 5.30 shows the flowchart for buffer empty interrupt setup processing.

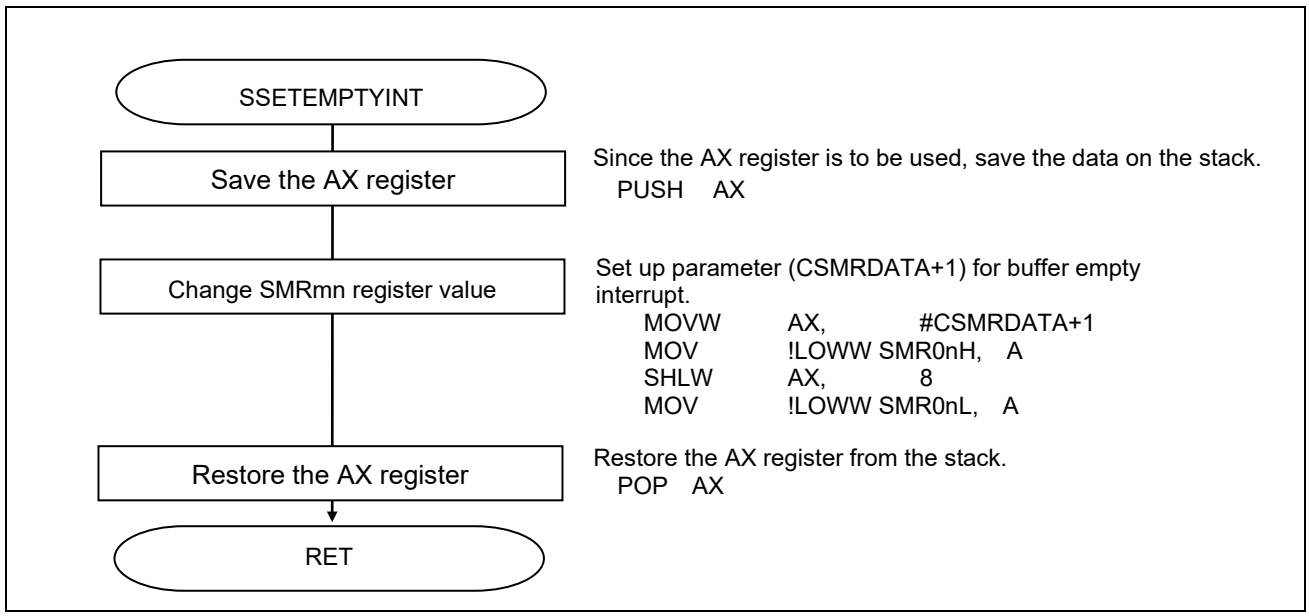


Figure 5.30 Buffer Empty Interrupt Setup Processing

Setting up the channel operating mode

- Serial mode register mn (SMR0nH, SMR0nL)
  Interrupt source and buffer empty interrupt

Symbol: SMR0nH

7	6	5	4	3	2	1	0
CKS 0n	CCS 0n	0	0	0	0	0	STS 0n[Note]
0	0	0	0	0	0	0	0

Symbol: SMR0nL

7	6	5	4	3	2	1	0
0	SIS 0n0[Note]	1	0	0	MD 0n2	MD 0n1	MD 0n0
0	0	1	0	0	0	0	**1**

Bit 0 (SMR0nL)

MDmn0	Selection of interrupt source of channel n
0	**Transfer end interrupt**
1	**Buffer empty interrupt**

Remark: n: channel number (n=0, 1)
Note: SMR01H, SMR01L registers only.
Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

5.7.33 Transmission Mode Setup Processing

Figure 5.36 shows the flowchart for transmission mode setup processing.

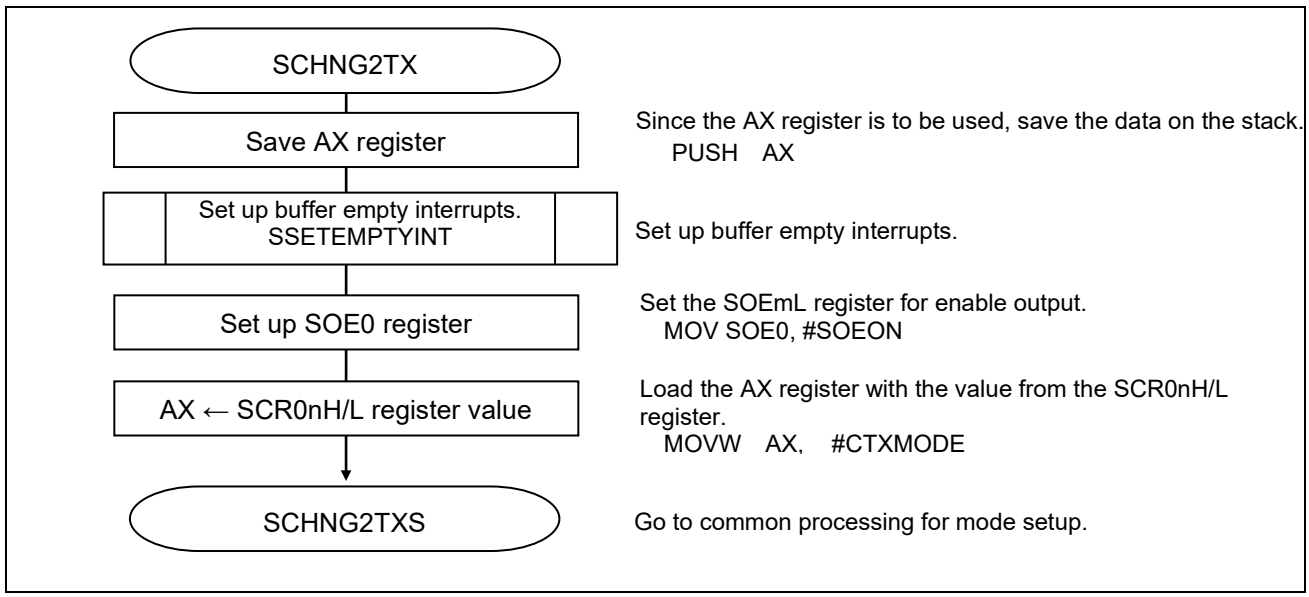


Figure 5.36 Transmission Mode Setup Processing

5.7.34 Reception Mode Setup Processing

Figure 5.37 shows the flowchart for reception mode setup processing.

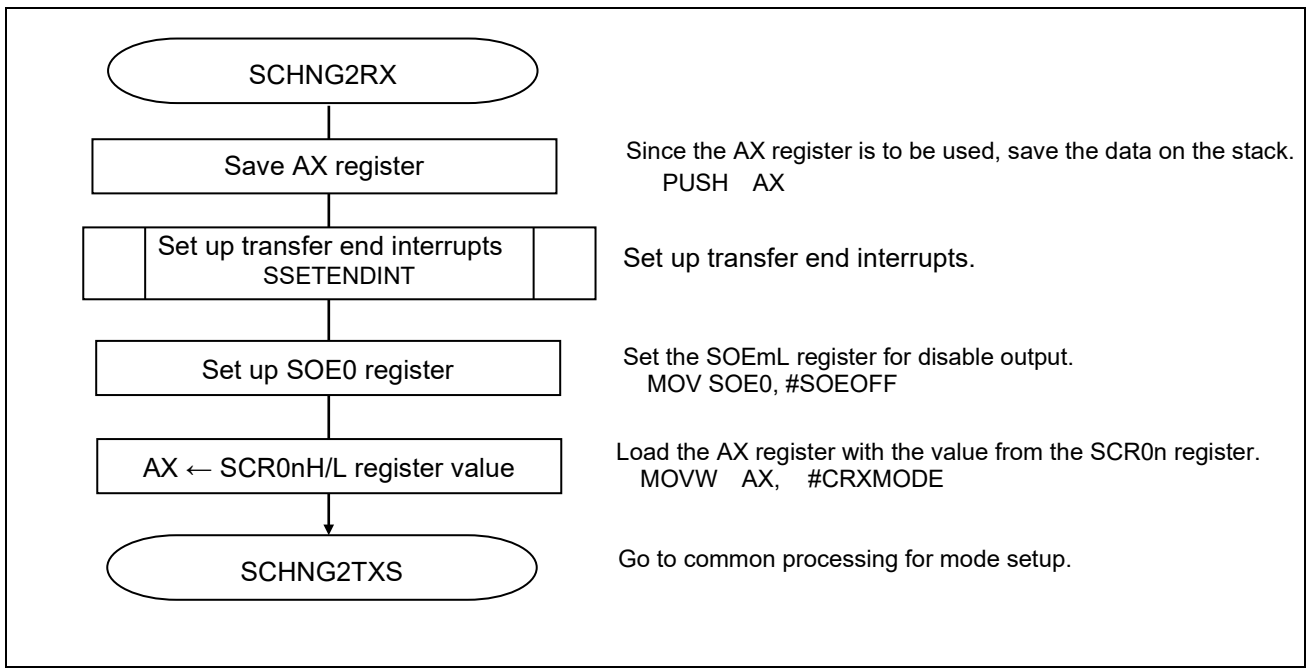


Figure 5.37 Reception Mode Setup Processing

5.7.35 Transmission/Reception Mode Setup Processing

Figure 5.38 shows the flowchart for transmission/reception mode setup processing.

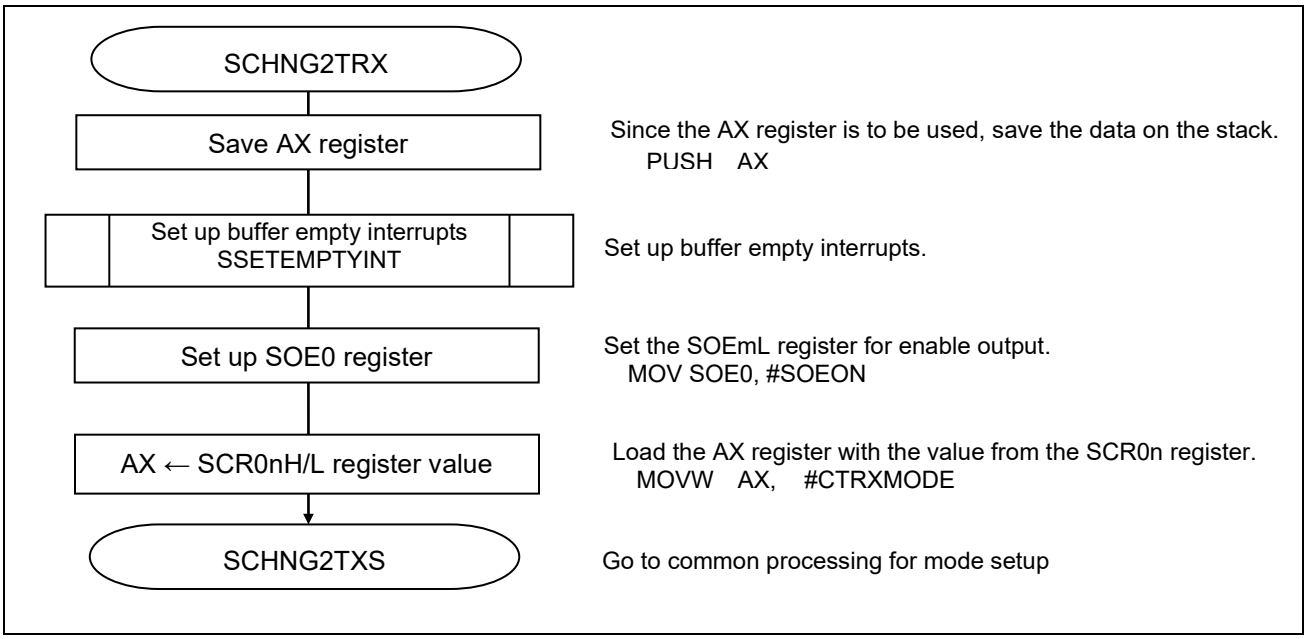


Figure 5.38 Transmission/Reception Mode Setup Processing

5.7.36 Mode Setup Common Processing

Figure 5.39 shows the flowchart for mode setup common processing.

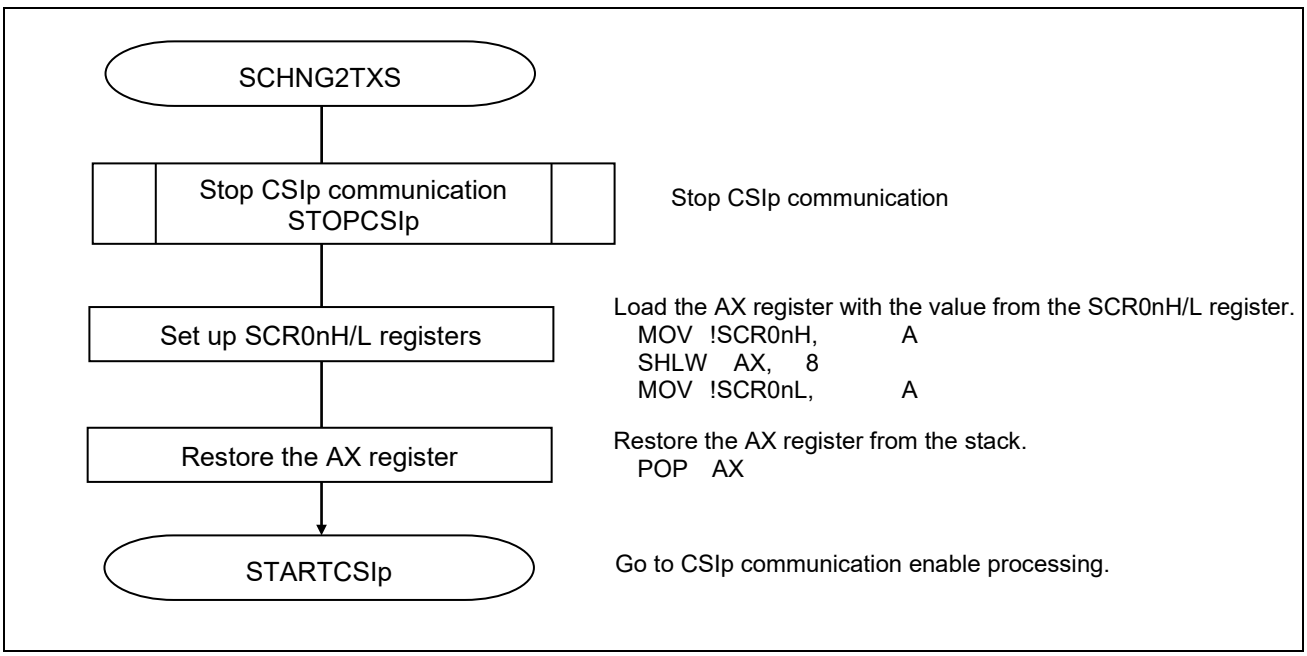


Figure 5.39 Mode Setup Common Processing

Interrupt setting (CSI00)

- Interrupt mask flag register (MK0L)

Set the interrupt mask

Symbol: MK0L (10-pin products)

7	6	5	4	3	2	1	0
TMMK00	TMMK01H	SREMK0	SRMK0	SRMK0 CSIMK00 IICMK00	PMK1	PMK0	WDTIMK
x	x	x	x	**1**	x	x	x

Bit 3

CSIMK00	Interrupt processing control
0	Enables interrupt processing.
1	**Disables interrupt processing.**

Transiting to communication stopped

- Serial channel stop register 0 (ST0)

Stop communication

Symbol: ST0

7	6	5	4	3	2	1	0
0	0	0	0	0	0	ST01	ST00
0	0	0	0	0	0	**1**	**1**

Bits 1 and 0

ST0n	Operation stop trigger of channel n
0	No trigger operation
1	**Sets the SE0n bit to 0 and stops communication**

Setting up channel communication operation

- Serial communication operation setting register 0n (SCR0nH, SCR0nL)

Operation mode

Symbol: SCR0nH

7	6	5	4	3	2	1	0
TXE 0n	RXE 0n	DAP 0n	CKP 0n	0	EOC 0n	PTC 0n1	PTC 0n0
0/1	**0/1**	**0**	**0**	0	**0**	**0**	**0**

Symbol: SCR0nL

7	6	5	4	3	2	1	0
DIR 0n	0	SLC 0n1 Note	SLC 0n0	0	1	1	DLS 0n0
0	0	**0**	**0**	0	1	1	**1**

Bits 7 and 6 (SCR0nH)

TXE0n	RXE0n	Setting of operation mode of channel n
0	0	Reception only
0	**1**	**Transmission only**
1	**0**	**Transmission/reception**
1	**1**	**Reception only**

Remark n: Channel number (n = 0, 1)

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

5.7.37 CSIp Communication Enable Processing

Figure 5.34 shows the flowchart for CSIp communication enable processing.

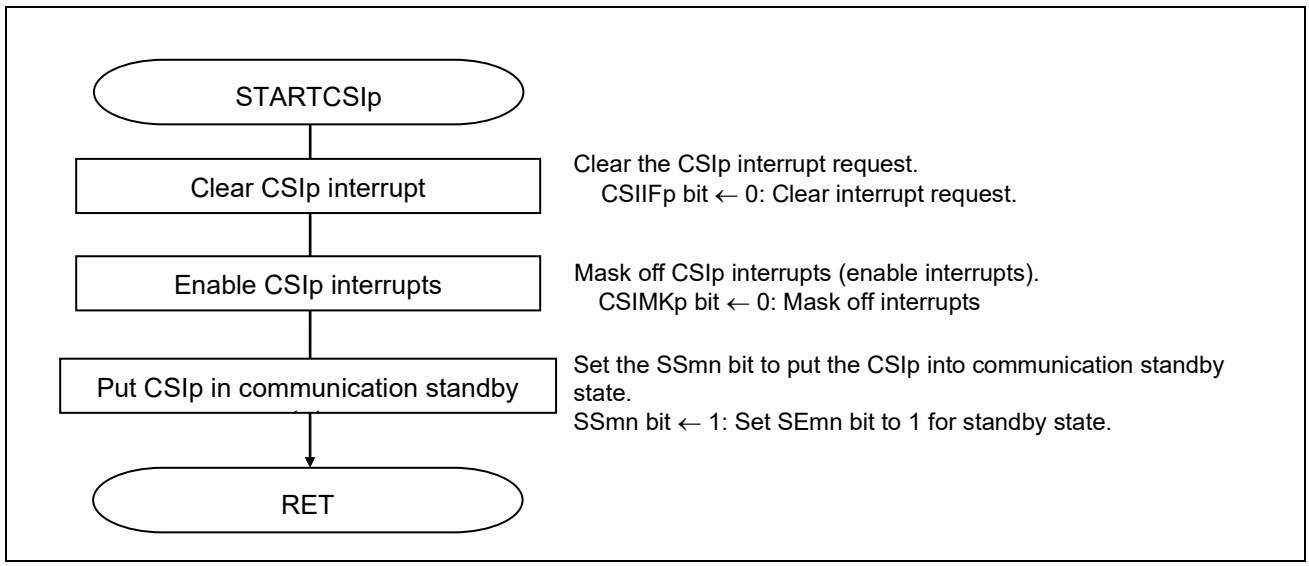


Figure 5.34 CSIp Communication Enable Processing

Transiting to communication standby state

- Serial channel startup register m (SS0)
  Start operation.

Symbol: SS0

7	6	5	4	3	2	1	0
0	0	0	0	0	0	SS01	SS00
0	0	0	0	0	0	**1**	**1**

Bits 1 to 0

SS0	Operation start trigger of channel n
0	No trigger operation
1	**Sets the SEmn bit to 0 and enters the communication wait status.**

Remark: n:channel number (n=0, 1)

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

Interrupt setting

- Interrupt request flag register (IF0L)
  Clear the interrupt request flag
- Interrupt mask flag register (MK0L)
  Clear the interrupt mask

Symbol: IF0L (10-pin products)

7	6	5	4	3	2	1	0
TMIF00	TMIF01H	SREIF0	SRIF0	STIF0 CSIIF00 IICIF00	PIF1	PIF0	WDTIIF
x	**0**	x	x	x	x	x	x

Bit 6

TMIF01H	Interrupt request flag
0	**No interrupt request signal is generated**
1	Interrupt request is generated, interrupt request status

Symbol: MK0L (10-pin products)

7	6	5	4	3	2	1	0
TMMK00	TMMK01H	SREMK0	SRMK0	STMK0 CSIMK00 IICMK00	PMK1	PMK0	WDTIMK
x	**0**	x	x	x	x	x	x

Bit 6

TMMK01H	**Interrupt processing control**
0	**Enables interrupt processing.**
1	Disables interrupt processing.

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

5.7.38 CSIp Communication Termination Processing

Figure 5.35 shows the flowchart for CSIp communication termination processing.

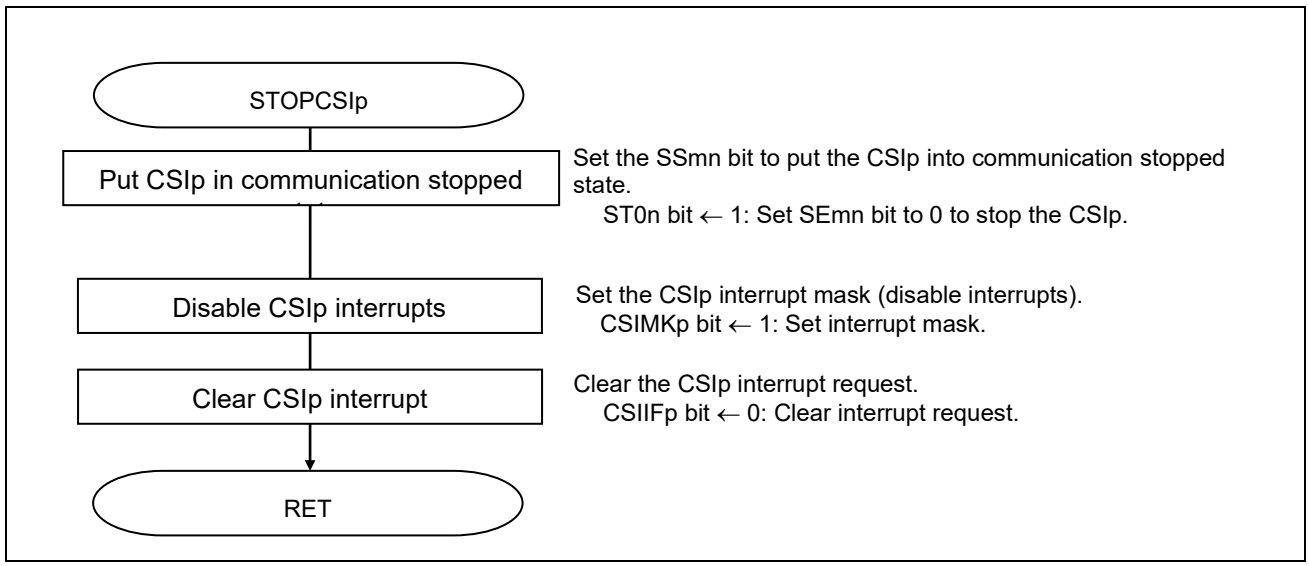


Figure 5.35 CSIp Communication Termination Processing

Transiting to communication stopped

- Serial channel startup register m (ST0)
  Stop operation.

Symbol: ST0

7	6	5	4	3	2	1	0
0	0	0	0	0	0	ST01	ST00
0	0	0	0	0	0	**1**	**1**

Bits 1-0

SS0n	Operation start trigger of channel n
0	No trigger operation
1	**Sets the SE0n bit to 0 and enters the communication wait status.**

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

Interrupt setting (10-pin products)

- Interrupt request flag register (IF0L)
  Clear the interrupt request flag
- Interrupt mask flag register (MK0L)
  Set the interrupt mask

Symbol: IF0L (10-pin products)

7	6	5	4	3	2	1	0
TMIF00	TMIF01H	SREIF0	SRIF0	STIF0 CSIIF00 IICIF00	PIF1	PIF0	WDTIIF
x	x	x	x	**0**	x	x	x

Bit 3

CSIIF00	Interrupt request flag
0	**No interrupt request signal is generated**
1	Interrupt request is generated, interrupt request status

Symbol: MK0L (10-pin products)

7	6	5	4	3	2	1	0
TMMK00	TMMK01H	SREMK0	SRMK0	SRMK0 CSIMK00 IICMK00	PMK1	PMK0	WDTIMK
x	x	x	x	**1**	x	x	x

Bit 3

CSIMK00	Interrupt processing control
0	Enables interrupt processing.
1	**Disables interrupt processing.**

Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

6. Changing the Channel to be Used

6.1 Definition File

The channel to be used for CSI master communication is defined in an include file (DEV&CSI_CH.inc). Note that the available channels vary with the device.

6.2 Major Items of the Definition File

The include file defines the following constants that the user can modify. Never modify the value of the other constants. The CPU clock frequency is defined to refer to the clock frequency of the CPU that is actually used in the user system. The user cannot use this definition to change the clock frequency of the CPU.

- CPU clock frequency (CLKFREQ) in kHz : The initial value is 20000 (20MHz).
- CSI communication speed (BAUDRATE) in kbps : The initial value is1000 (1 M bps).
- Microcontroller to be used : The initial value is R5F10Y16ASP.
- CSI channel to be used : The initial value is CSI00.

6.3 Changing the Transfer Rate

The transfer rate is defined as shown below. For a CPU clock frequency of 24 MHz, the user can change the transfer rate between 200 kbps and 2000 kbps by changing "1000" to a desired value between 200 and 2000. It is necessary to modify the program to use a transfer rate outside this value range.

```
;*****************************************************************
;
;     Communication definitions
;
;*****************************************************************

CLKFREQ      .EQU    20000              ; kHz
BAUDRATE     .EQU    1000               ; kpbs
```

6.4 Changing the Microcontroller to be Used

When changing the microcontroller to be used, create a new project with CS+ and specify the desired device in the project.
The microcontroller to be used is defined as shown below. Line that has become a ".SET 1" is enabled. Set the ".SET 0" to the other device.

```
;*****************************************************************
;
;          device select
;
;*****************************************************************
R5F10Y16     .SET    1     ; 10 pins
R5F10Y14     .SET    0     ; 10 pins
R5F10Y44     .SET    0     ; 16 pins
R5F10Y46     .SET    0     ; 16 pins
R5F10Y47     .SET    0     ; 16 pins
```

Defines the microcontroller that is in use.

6.5 Changing the Channel to be Used

The channel to be used is defined as shown below. Select the desired channel from the channels that are available for the microcontroller to be used and delete the leading semicolon (';') from the line defining the desired channel. At the same time, append a semicolon at the beginning of the line for the channel that had been selected until now. **The program will not run normally if two or more channels area selected.**

```
;*****************************************************************************
;
;           Communication channel select
;
;*****************************************************************************
$IF( R5F10Y14+R5F10Y16==1 )
;=============================================================  Definition for 10-pin products .
;           for R5F10Y16 and R5F10Y14
;           CSI00 only
;=============================================================
CSI00           .SET      1                   ; CSI00 is selected
CSI01           .SET      0                   ; CSI01 is not selected now

$ELSE                                                          Definition for 16-pin products .
;=============================================================
;           for R5F10Y44 , R5F10Y46 and R5F10Y47
;           select CSI00 or CSI01
;=============================================================
CSI00           .SET      1                   ; CSI00 is selected
CSI01           .SET      0                   ; CSI01 is not selected now
;CSI01          .SET      1                  ; CSI01 is selected
$ENDIF
```

6.6 Reference

Once the channel to be used is defined, the constants to be used by the program are set to the values that conform to the newly defined channel by the definitions given below, so that the user need not be aware of the channel he or she is to use.

Port initialization is accomplished by directly referencing the microcontroller and channel definitions that are provided separately from these definitions.

```
$IF( CSI00 )
SMR0nH   EQU     SMR00H          ; Serial mode register(High)
SMR0nL   EQU     SMR00L      ; Serial mode register(Low)
SCR0nH   EQU     SCR00H      ; Serial communication operation setting register(High)
SCR0nL   EQU     SCR00L      ; Serial communication operation setting register(Low)
SDR0nH   EQU     SDR00H      ; Serial data register(High)
SIOp  EQU     SIO00           ; Serial data register(Low)
SSR0n     EQU     SSR00           ; Serial status register
SIR0n EQU     SIR00           ; Serial flag clear trigger register
TRGONn   EQU     0B00000001 ; for trigger SS00/ST00
SOEON    EQU     TRGONn              ; for turn on SOE00
SOEOFF   EQU     0B11111110 ; for turn off SOE00
SOHIGH   EQU     TRGONn          ; for set SO bit
PM_SCKp EQU     PM0.2       ; port mode register bit for SCK
PM_SIp   EQU     PM0.1       ; port mode register bit for SI
PM_SOp   EQU     PM0.0       ; port mode register bit for SO
P_SCKp   EQU     P0.2        ; port register for SCK
P_SIp EQU     P0.1            ; port register for SI
P_SOp    EQU     P0.0            ; port register for SO
CSIIFp   EQU     CSIIF00     ; interrupt request flag
CSIMKp   EQU     CSIMK00         ; interrupt mask register
$ENDIF
```

7. Sample Code

The sample code is available on the Renesas Electronics Website.

8. Documents for Reference

RL78/G10 User's Manual: Hardware (R01UH0384E)

RL78 Family User's Manual: Software (R01US0015E)

RL78 Family CubeSuite+ Startup Guide (R01AN1232E)

RL78/G10 Serial Array Unit (CSI Master Communication) (R01AN1460E)

(The latest versions of the documents are available on the Renesas Electronics Website.)

Technical Updates/Technical Brochures

(The latest versions of the documents are available on the Renesas Electronics Website.)

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Nov. 27, 2015	—	First edition issued
1.10	June. 24, 2022	—	Operation check condition is updated.

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.