RL78/G10

Serial Array Unit (CSI Master Communication) CC-RL

Introduction

This application note describes how the serial array unit (SAU) performs communication tasks using the CSI master communication feature. As CSI applications, the SAU selects one of two slaves with the CS signal which is issued through a port and performs single transmission/reception, continuous transmission, continuous reception, and continuous transmission/reception operations. To ensure reliable communication, it adopts a simple protocol and a command set plus its compatible format. Since RL78/G10 units running in CSI slave mode are used as slave devices, it performs handshake processing using the BUSY signal.

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.

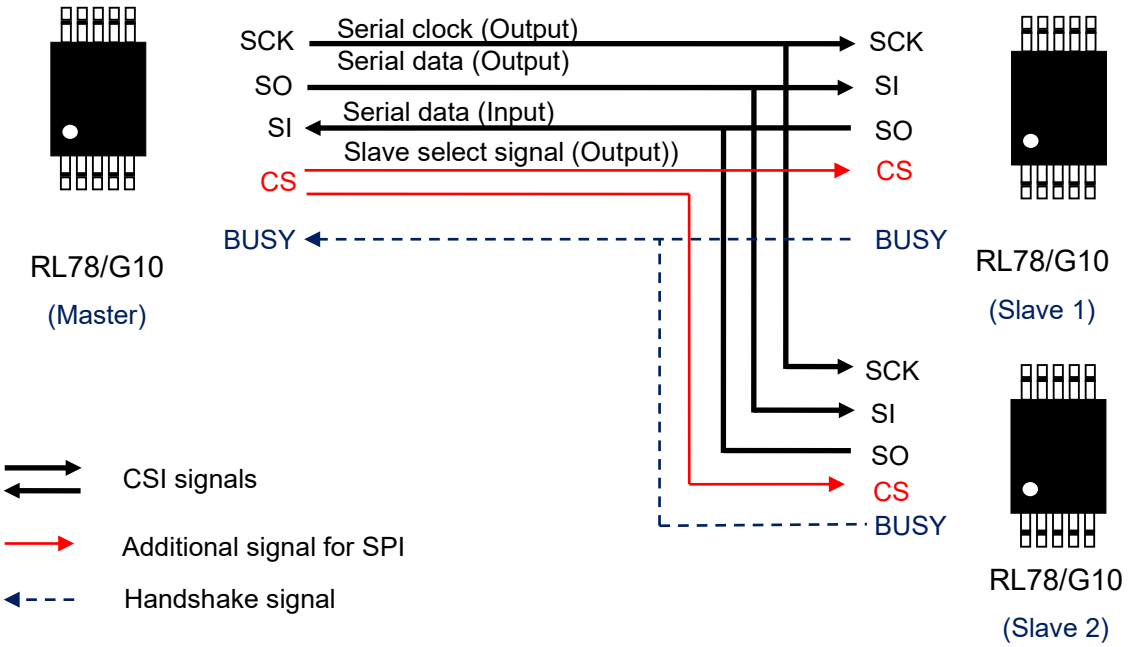
Contents

1. Specifications

The serial array unit (SAU) described in this application note performs CSI master communication using the serial array unit (SAU). The SAU outputs SPI CS signals through ports to a maximum of 2 slaves that are attached to select the target of communication and 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 in the example in this application note) is ready for communication. The master verifies this BUSY signal before initiating a communication session.

1.2 Outline of Communication

Communication is performed 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 to be used.

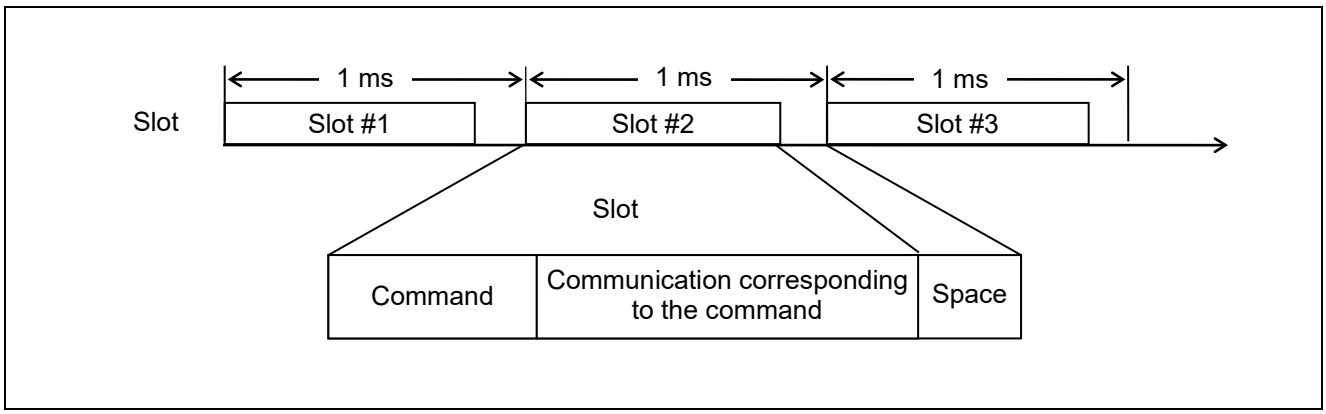


Figure 1.2 Outline of Slots

Table 1.1 Commands to be Used

Command	Outline of Operation
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 slave is designed to transmit the complement of the data it received in the next communication operation, so that the master can determine whether the data received by the slave is correct. The master prepares increment pattern data, e.g., 00, 01, 02, ..., as transmit data and updates the transmit data on each transmission operation.

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
Port	P03 (CS1 signal output), P04 (CS2 signal output), P137 (BUSY signal input)

Note: 16-pin products only

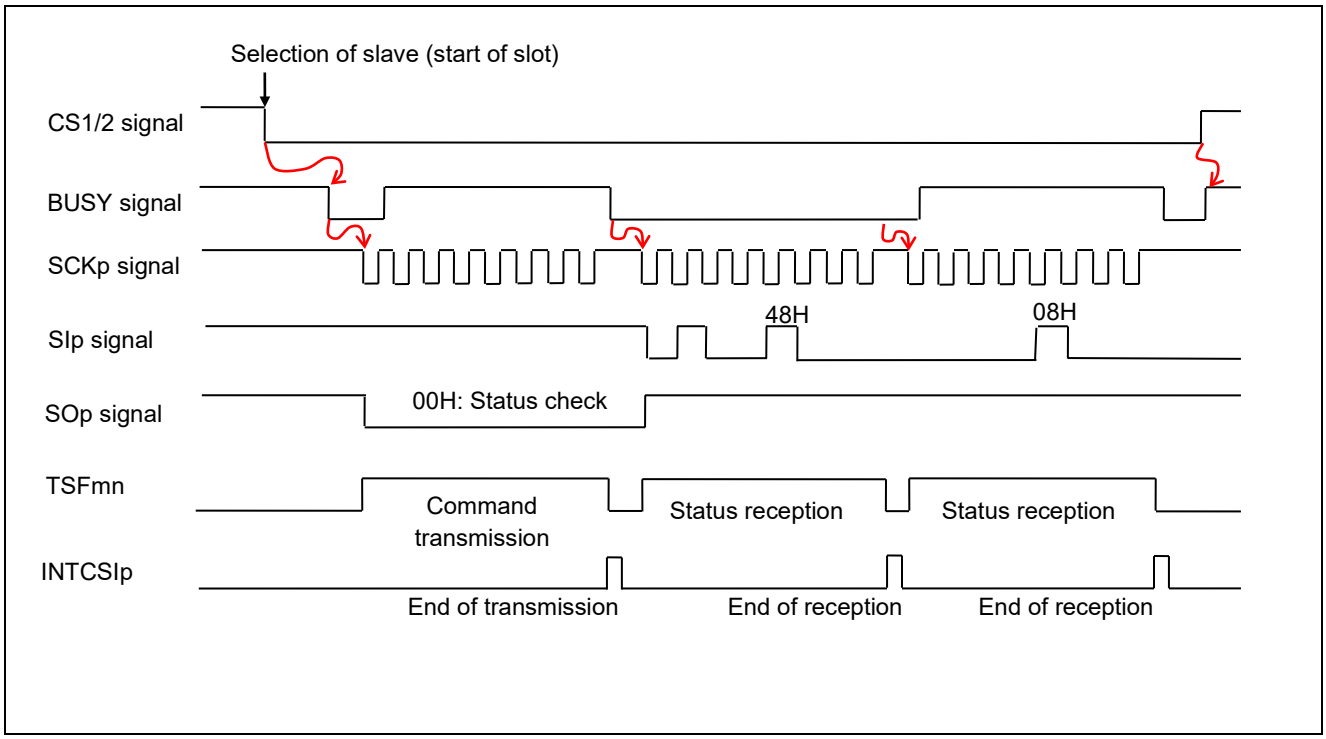


Figure 1.3 Timing Chart of Status Check Command

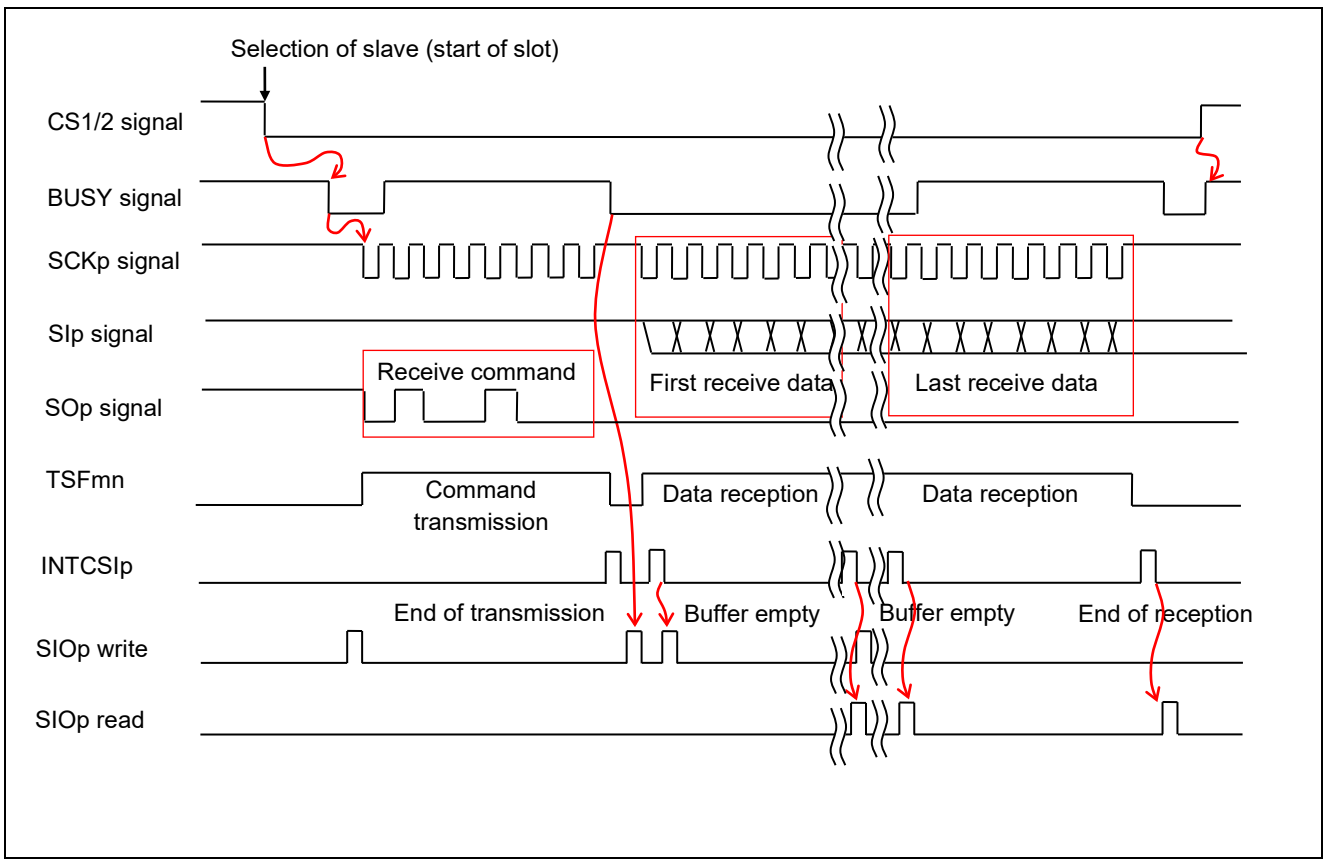


Figure 1.4 Timing Chart of Receive Command

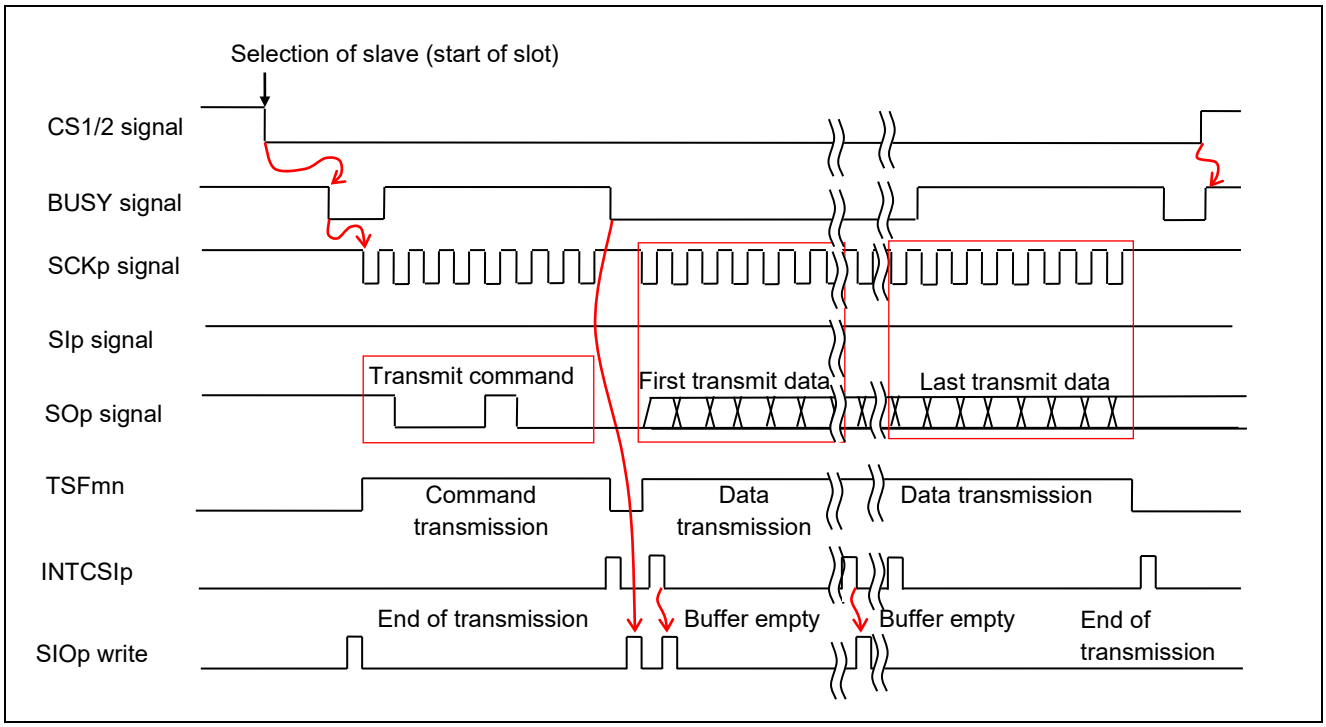


Figure 1.5 Timing Chart of Transmit Command

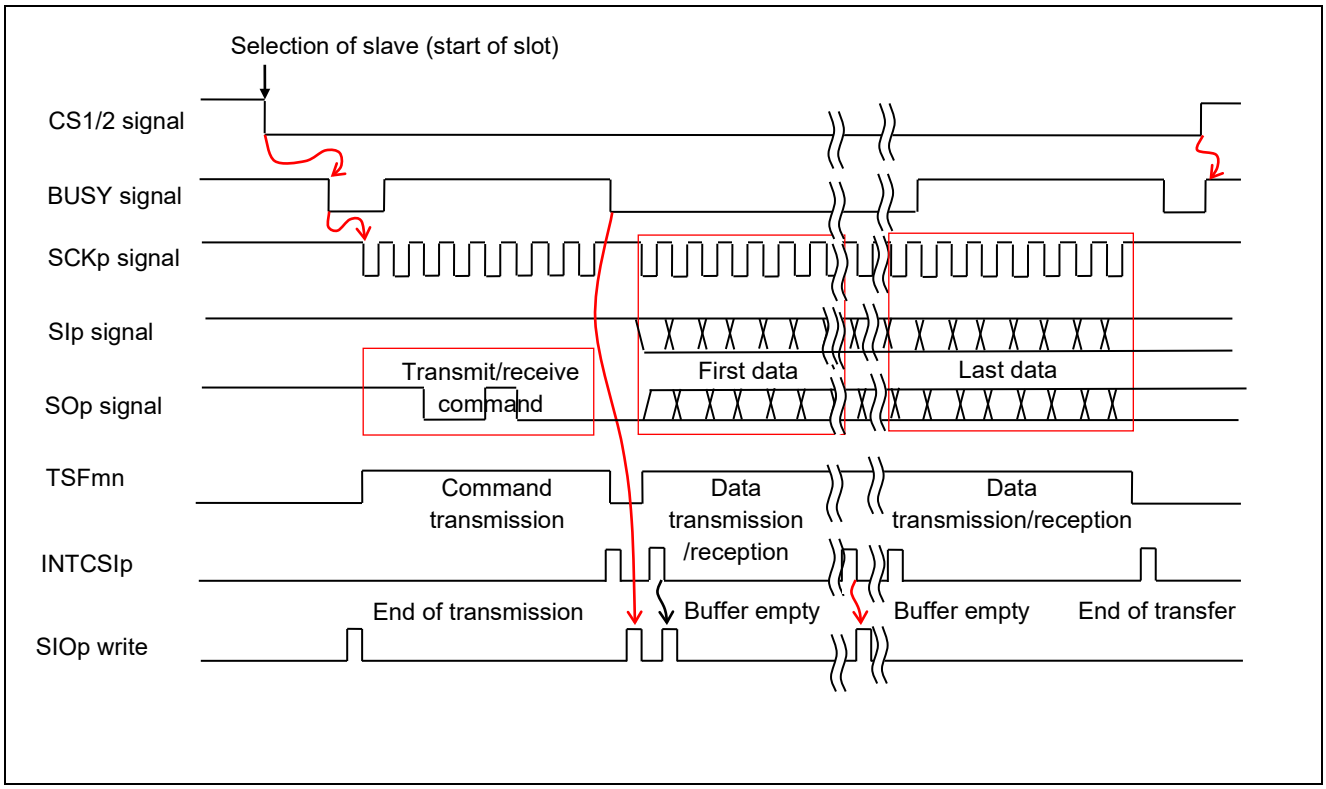


Figure 1.6 Timing Chart of Transmit/Receive Command

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	Type 1	
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)

The communication target is set to the RL78/G10 running in CSI slave mode and handshaking using the BUSY signal is adopted to have the setup time that is required for the communication operation on the slave side.

The BUSY signal is used to verify that the slave becomes ready for communication when selecting it with the CS signal or when sending a command. A timeout time of 10 μs is set up so that the master does not enter an unnecessary deadlock state when no slave is connected. If no response is returned from the slave within this period, the master terminates processing immediately, considering that the slave is in the busy state in which it is taking some action or that there is no slave available.

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.

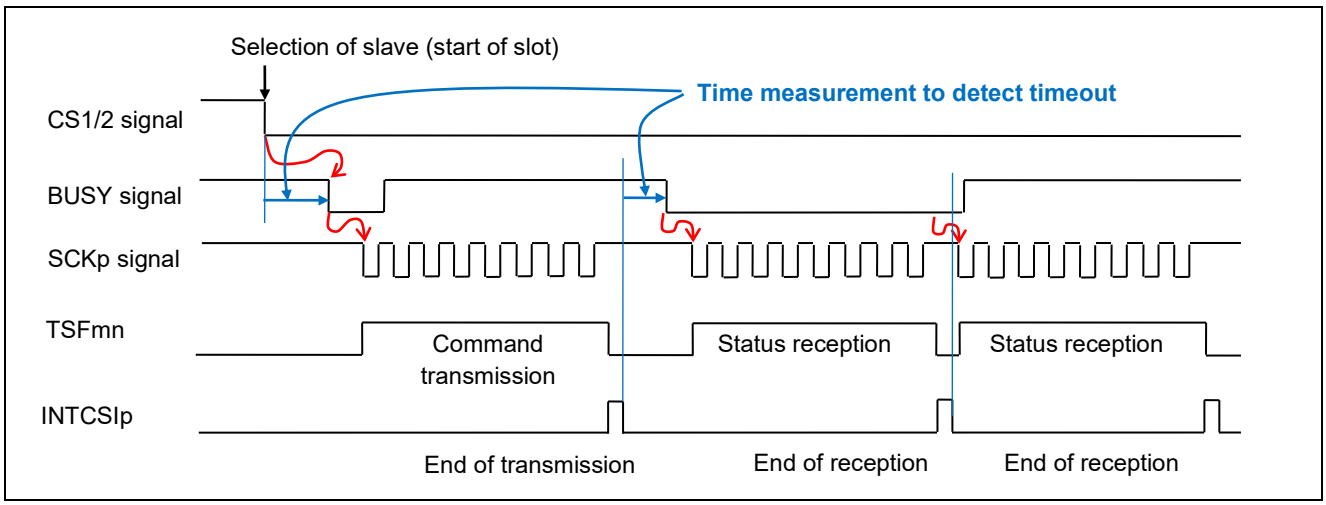


Figure 1.7 Handshaking Example

The BUSY signal is not available for dedicated SPI slave devices such as EEPROM, A/D, and D/A. This is because these devices are always ready for communication. A hardware measure to be taken when connecting these dedicated slave devices is to connect the BUSY signal input to V_{SS}. Timeout checking is accomplished by a dedicated subroutine (SWAITRDY). As software countermeasures, it is possible to dispense with the checking for the BUSY signal by modifying the subroutine so that it simply returns after

clearing the CY flag. Subsequently, perform communication according to the commands that are defined for the individual devices and their communication protocol.

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 voltage: 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 Slave Communication) CC-RL (R01AN3077E) 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.

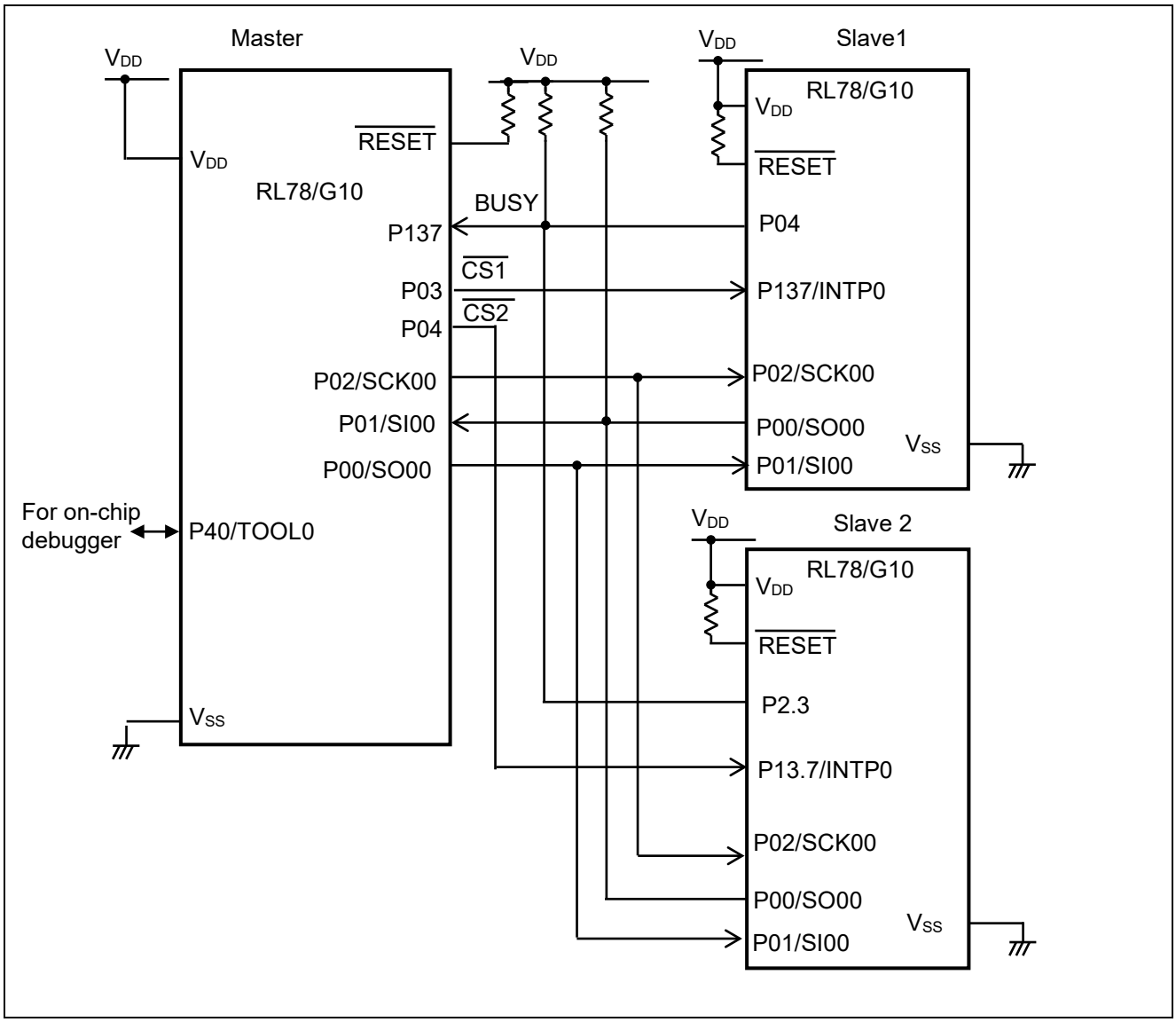


Figure 4.1 Hardware Configuration

Note: Only for 30-pin products.

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 (R01AN1459), 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/TI00/INTP0	Input	BUSY signal input from the slave
P03/ANI2/TO00/KR4/(INTP1)（CS1）	Output	Slave 1 select signal
P04/ANI3/TI01/TO01/KR5（CS2）	Output	Slave 2 select 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, selects a slave and performs communication operations such as status check, data transmission, data transmission/reception, and data reception in that order on the selected slave.

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

- Use SAU0 channel 0 as CSI00 Note.
- Use CK00 as the transfer clock.
- Assign the clock output to the P10/SCK00 pin Note, the data input to the P11/SI00 pin Note, and the data output to the P12/TxD0 pin Note.
- 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.
- Set the transfer rate to 1M bps.
- Use transmission end interrupt (INTCSI00) Note .
- 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) Initialize the timer.
<Conditions for setting the timer>

- Run channels 3 and 2 as 8-bit interval timers.
- Set the operating clock frequency to 187.5 kHz which is derived by dividing f_{CLK} by 128.
- Use the upper TM03H as a 1-ms interval timer.
- Use the lower TM03 as a 10-μs interval timer.

(3) After initialization is completed, the master initializes memory and performs communication with the slave according to the steps given below.
   1) Waits in the HALT state for 1ms interval interrupts (INTTM03H).
   2) When the master is released from the HALT state on an INTTM03H interrupt, it selects (by issuing the CS signal) the slave that is selected in the flag (RCSFLAG) and waits for a response from the slave.
   3) Proceeds to step 4) when the BUSY signal from the slave goes low. When a timeout is detected, the master deselects the slave and proceed with step 10).
   4) Transmits a status check command and receives the status from the slave. When a timeout is detected, the master deselects the slave and proceed with step 10).
   5) When an INTTM03H occurs, the master transmits the number of data characters specified in step 4) and generates the next transmit data and the expected value of the receive data. If the value of receiveddata does not match the expected value, the master enters the HALT mode and subsequent processing does not proceed. When a timeout is detected, the master deselects the slave and proceed with step 10).
   6) When an INTTM03H occurs, the master transmits and receives the number of characters specified in step 4). It also checks whether the received data matches the complement of the data that is transmitted in step 5). When a timeout is detected, the master deselects the slave and proceed with step 10).
   7) The master generates the next transmit data and the expected value of the receive data, then waits for an INTTM03H.
   8) When an INTTM03H occurs, the master receives the number of data characters specified in step 4). When a timeout is detected, the master deselects the slave and proceed with step 10).
   9) The master checks whether the received data matches the expected value.
   10) The master changes the state of the flag (RCSFLAG) to switch the target slave. Subsequently, the master repeats the steps starting at 1).

(4) Commands

Each communication operation begins with the transmission of a 1-byte command. The command formats are listed in table 5.1. The master transmits a status check command and receives the response from the slave in the first slot of a communication sequence. The number of received data characters or the size of buffer, whichever is smaller, is used as the data count for the next communication processing. The master then performs the next communication using this data count.

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
Reception	01xxxxxxB	The master receives xxxxxxB bytes of data.
Transmission	10xxxxxxB	The master transmits xxxxxxB bytes of data.
Transmission/reception	11xxxxxxB	Transmits and receives xxxxxxB bytes of data.

5.2 List of Option Byte Settings

Table 5.2 summarizes the settings of the option bytes.

Table 5.2 Option Byte Settings

Address	Value	Description
000C0H	11101110B	Disables the watchdog timer. (Stops counting after the release from the reset state.)
000C1H	11110111B	SPOR detection voltage: Falling edge: VDD < 2.84V Risnig edge: VDD >= 2.90V
000C2H	11111001B	HS mode, HOCO: 20 MHz
000C3H	10000101B	Enables the on-chip debugger.

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 (20 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) * 2H	Value to be set in SDR to specify the communication speed
INTERVAL	↑	1	Slot interval in ms units (1 ms)
TDRDATA	↑	(CLKFREQ/128) * INTERVAL - 1	Value to be set in TDR03H
SMR0nH	↑	SMR00H Note	Channel mode setting register (Higher bits)
SMR0nL		SMR00L Note	Channel mode setting register (Lower bits)
SCR0nH	↑	SCR00H Note	Channel communication operation setting register (Higher bits)
SCR0nL		SCR00L Note	Channel communication operation setting register (Lower bits)
SDR0nH	↑	SDR00H Note	Higher 8 bits of channel serial data
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 SS0 and ST0
SOEmL	↑	SOE0L Note	Channel output enable register
SOEON	↑	TRGONn	For setting in channel output enable register (enable)
SOEOFF	↑	11111110B Note	For setting in channel output enable register (disable)
SOHIGH	↑	TRGONn	For setting value in 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	↑	PM0.2 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	SCK signal port
P_SIp	↑	P0.1	SI signal port
P_SOp	↑	P0.0	SO signal port
CSIIFp	↑	CSIIF00	Channel interrupt request flag
CSIMKp	↑	CSIMK00	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	P13.7	Port for checking the BUSY signal
CS1SIG	↑	P0.3	CS1 output port
CS2SIG	↑	P0.4	CS2 output port
CRXDTNO	↑	8	Size of receive data buffer (in bytes)
CTXDTNO	↑	8	Size of transmit data buffer (in bytes)
STSCHKCMD	↑	00000000B	Status check command
MSTRDCMD	↑	01000000B	Master receive command
MSTWTCMD	↑	10000000B	Master transmit command
MSTRWCMD	↑	11000000B	Transmit/receive command
SELOFFSIG	↑	00001100B	Data for terminating slave selection

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 the program that performs actual processing when an INTCSIp interrupt occurs.	main, STXDATAST, SRXDATAST, SSEQRXSUB, SSEQTXSUB, SSEQTRXSUB, IINTCSIp, STXNEXT, STRXLAST
8-bit array	RSNDBUF1	Buffer for transmit data to slave 1	main, (SETTRXPNTR), SCHANGEDATA, SSEQTXSUB, SSEQTRXSUB, STXNEXT, STRXNEXT
8-bit array	RRCVBUF1	Buffer for receive data from slave 1	main, SRXNEXT, STRXNEXT, STRXEND
16 bits	RSTTS1	Number of characters that can be sent to or received from slave 1	main, SSTSCHK, STXCMD, SRXCMD, STRXCMD
8-bit array	RSNDBUF2	Buffer for transmit data to slave 2	main, (SETTRXPNTR), SCHANGEDATA, SSEQTXSUB, SSEQTRXSUB, STXNEXT, STRXNEXT
8-bit array	RRCVBUF 2	Buffer for receive data from slave 2	main, SRXNEXT, STRXNEXT, STRXEND
16 bits	RSTTS2	Number of characters that can be sent to or received from slave 2	main, SSTSCHK, STXCMD, SRXCMD, STRXCMD
8 bits	RCSFLAG	Slave to which LSB is to access	main, SETTRXPNTR, SSLAVSEL, SSTSCHK
8 bits	RRCVBUF	Used to store receive data transmitted in single transfer mode.	SWAITRXEND, CSITXEND
8 bits	CSISTS	Number of remaining characters to be transferred	STXDATAST, SWAITTXEND, SRXDATAST, SWAITRXEND, SSEQRXSUB, SWAITSTREND, SSEQTXSUB, SSEQTRXSUB, CSITXEND, SRXNEXT, STXNEXT, STXEND, STRXNEXT, STRXEND
8-bit array	RCMPDATA	Area for storing the expected value of the receive data	SCHANGEDATA, SCHKDTSUB

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 ports.
SINICLK	Sets up the clock generation circuit.
SINISAU	Initialize CSIp.
SINITAU	Initialize TM03 and TM03H.
SSTARTINTV	Start 1-ms interval timer.
SSLAVSEL	Output CS signal to slave identified by slave number.
SWAIT1MS	Wait for 1ms interval timing in HALT state.
SWAITRDY	Wait until BUSY signal goes low, until timeout.
SSTSCHK	Check slave status.
STXCMD	Transmit data to slave in continuous transmission mode.
SRXCMD	Receive data from slave in continuous reception mode.
STRXCMD	Transmit/receive data to and from slave in continuous transmission/reception mode.
SCHANGEDATA	Generate expected value from transmit data and next transmit data.
SETTRXPNTR	Set up pointer to data buffer that is determined by slave number.
SCHKDTSUB	Compare receive data with expected value.
STXDATAST	Start 1-character transmission processing (send data from A register).
SWAITTXEND	Wait for end of 1-character transmission.
SRXDATAST	Start 1-character reception processing.
SWAITRXEND	Wait for end of 1-character reception (load receive data in A register).
STRXREADY	Check 1-character transfer state, set Z flag to 1 if end of transfer.
SSEQTXSUB	Start continuous transmission processing (send data designated by HL number of times specified in A register.
SSEQRXSUB	Start continuous reception processing (receive number of characters specified in A register into buffer designated by HL).
SSEQTRXSUB	Start continuous transmission/reception processing (HL: transmit pointer, DE: receive pointer, A: number of characters).
SWAITSTREND	Wait for end of continuous transfer.
SSETENDINT	Set up transfer end interrupts.
SSETEMPTYINT	Set up buffer empty interrupts.
SCHNG2TX	Stop operation temporarily and enable transmission mode (transfer end interrupt).
SCHNG2RX	Stop operation temporarily and enable reception mode (transfer end interrupt).
SCHNG2TRX	Stop operation temporarily and enable transmission/reception mode (transfer end interrupt).
SCHNG2TXS	Common processing for mode setup
STARTCSIp	Enables CSI.
STOPCSIp	Disables CSI.
IINTCSIp	Start INTCSIp interrupt processing (branch to processing block).
CSITXEND	Process 1-character transfer end interrupts (load receive data into RRCVBUF).
SRXNEXT	Process 1-character transfer end interrupts in continuous reception mode.
STXNEXT	Process buffer empty interrupt in continuous transmission mode.
STXEND	Process transmit end interrupts in continuous transmission mode (set variable CSISTS to 0).
STRXNEXT	Process buffer empty interrupts in continuous transmission/reception mode.
STRXEND	Process transfer end interrupts in continuous transmission/reception mode.

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	Make initial settings of the CPU by starting up from a reset.
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	Set up the I/O ports
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	Set 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.
Arguments	None
Return value	None
Remarks	None

[Function Name] SINITAU

Synopsis	Initialize TM03.
Explanation	This functions sets up the TM03 as two 8-bit interval timers.
Arguments	None
Return value	None
Remarks	None

[Function Name] SSTARTINTV

Synopsis	Start TM03H.
Explanation	This functions starts the TM03H (1-ms interval timer).
Arguments	None
Return value	None
Remarks	None

[Function Name] SSLAVSEL

Synopsis	Perform slave selection processing.
Explanation	This functions generates the CS signal to the slave designated by RCSFLAG.0 and waits for a response from the slave. When a timeout is detected, the function turns off the CS signal.
Arguments	None
Return value	CY flag : [1: No slave response, 0: Slave response present]
Remarks	None

[Function Name] SWAIT1MS

Synopsis	Wait for 1-ms interval timing.
Explanation	This functions disables vector interrupts and waits for the occurrence of a TM03H interrupt in HALT mode.
Arguments	None
Return value	None
Remarks	None

[Function Name] SWAITRDY

Synopsis	Wait response from slave.
Explanation	This function starts the TM03 (for timeout measurement) and waits for a response from the slave (BUSY signal going low). When a timeout is detected before the timer startup, the function turns off the CS signal and terminates processing.
Arguments	None
Return value	CY flag : [1: No slave response, 0: Slave response present]
Remarks	None

[Function Name] SSTSCHK

Synopsis	Check slave status.
Explanation	This function sends the Check Status command to the selected slave and stores the number of transmittable or receivable characters, which is received from the selected slave, in a work area. The function signals an error if a timeout is detected or the status (data count) that is received proves invalid.
Arguments	None
Return value	CY flag : [1: Invalid slave response, 0: Normal slave response]
Remarks	On normal execution, the function loads RSTTS1 or RSTTS2 with the numbers of transmittable and receivable characters to and from the slave.

[Function Name] STXCMD

Synopsis	Transmit data to slave.	
Explanation	This function sends a master transmit command to the slave, places the CSIp in transmission mode, and transmits the number of characters that the slave can receive, which is stored in RSTTS1 or RSTTS2, from the transmit data buffer.	
Arguments	None	
Return value	CY flag	: [1: Invalid slave response, 0: Normal slave response]
Remarks	None	

[Function Name] SRXCMD

Synopsis	Receive data from slave.	
Explanation	This function sends a master receive command to the slave, places the CSIp in reception mode, and receives and stores the number of characters that the slave can transmit, which is stored in RSTTS1 or RSTTS2, into the receive data buffer.	
Arguments	None	
Return value	CY flag	: [1: Invalid slave response, 0: Normal slave response]
Remarks	None	

[Function Name] STRXCMD

Synopsis	Transmit/receive data to and from slave.	
Explanation	This function sends a master transmit/receive command to the slave, places the CSIp in transmission/reception mode, and transmits the number of characters that the slave can transmit and receive, which is stored in RSTTS1 or RSTTS2, from the transmit data buffer and receives and stores received data in the receive data buffer.	
Arguments	None	
Return value	CY flag	: [1: Invalid slave response, 0: Normal slave response]
Remarks	This function is called only when the numbers of the characters that the slave can transmit and receive are the same.	

[Function Name] SCHANGEDATA

Synopsis	Generate expected value from transmit data and next transmit data.
Explanation	This functions generates the expected value of the data to be received next from the slave from the data that has been transmitted and places the value in a variable area (RCMPDATA), then update the contents of the transmit buffer.
Arguments	None
Return value	None
Remarks	None

[Function Name] SETTRXPNTR

Synopsis	Set up buffer pointer associated with selected slave.	
Explanation	This functions loads the address of the area containing the transmit data associated with the slave designated by RCSFLAG.0 into the HL register and the address of the buffer for storing the receive data into the DE register.	
Arguments	None	
Return value	HL register	: Address for storing transmit data
	DE register	: Address for storing receive data
Remarks	None	

[Function Name] SCHKDTSUB

Synopsis	Compare receive data with expected value.	
Explanation	This functions compares the data that is received with the expected value. The result is returned with the CY flag.	
Arguments	None	
Return value	CY flag	: [1: Error detected in comparison result, 0: Comparison result is normal.]
Remarks	None	

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

[Function Name] STXDATAST

Synopsis	Start 1-character transmission processing.
Explanation	This function writes data from the A register into the SIOp and starts communication processing. The function loads the address of CSITXEND into RCSISUBADDR as an INTCSIp processing routine 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	The CSIp need be configured for transmission or transmission/reception.

[Function Name] SWAITTXEND

Synopsis	Wait for end of 1-character transmission.
Explanation	This function waits for the end of transmission processing (CSISTS = 0) that is started by the STXDATAST function.
Arguments	None
Return value	None
Remarks	The transmission end interrupts are processed by CSITXEND (CSISTS is set to 0).

[Function Name] SRXDATAST

Synopsis	Start 1-character reception processing.
Explanation	This function writes dummy data (0xFF) into the SIOp and starts receive processing. The function loads the address of CSITXEND into RCSISUBADDR as an INTCSIp processing routine and sets the number of work-in-progress characters to 1 before returning.
Arguments	None
Return value	None
Remarks	The CSIp needs to be configured for reception or transmission/reception.

[Function Name] SWAITRXEND

Synopsis	Wait for end of 1-character reception.
Explanation	This function waits for the end of reception processing (CSISTS = 0) that is started by the SRXDATAST function. Upon end of reception processing, the function reads the received characters from RRCVBUF.
Arguments	None
Return value	A register Receive data
Remarks	The receive data is stored in RRCVBUF by CSITXEND.

[Function Name] STRXREADY

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 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] 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 verifies the initiation of data transmission by testing the TSF bit. If the number of characters to be transmitted is 2 characters or more, the function changes the interrupt timing to that for buffer empty interrupts and loads RCSISUBADDR with the address of STXNEXT as an INTCSIp processing routine. If the number of characters to be transmitted is 1 character, the function loads RCSISUBADDR with the address of STXEND. The function returns after setting the value of the A register to the in-communication data count (CSISTS). It returns with a Z flag value of 1 if the number of receive characters 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: Data count 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 receive mode, loads the buffer designated by the HL register with the number of characters designated by the A register, and starts reception processing. The function loads the address of SRXNEXT into RCSISUBADDR as an INTCSIp processing routine and sets the A register to the number of work-in-progress characters (CSISTS) before returning. The function returns with the Z flag set to 1 if the number of received characters 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 transmission/reception mode and starts the function to transmit and receive the number of data designated by the A register from the buffer designated by the HL register. The function verifies the initiation of data transmission/reception processing by testing the TSF bit. If the number of characters to be transmitted is 2 characters or more, the function changes the interrupt timing to that for buffer empty interrupts and loads RCSISUBADDR with the address of STRXNEXT as an INTCSIp processing routine. If the number of characters to be transmitted is 1 character, the function loads RCSISUBADDR with the address of STRXEND. The function returns after setting the value of the A register to the in-communication data count (CSISTS). It returns with a Z flag value of 1 if the number of receive characters in the A register is 0.	
Arguments	HL register	: Address of area storing the transmit data
	DE register	: Address of area for storing receive data
	A register	: Number of transfer characters
Return value	Z flag	: [0: Normal startup, 1: Data count is 0.] (CSISTS is set to the number of characters at normal startup time.)
Remarks	None	

[Function Name] SWAITSTREND

Synopsis	Wait for end of continuous transfer.
Explanation	This function waits until the number of work-in-progress 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	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	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 end of transfer.
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 CSIp temporarily and enables the transmission/reception mode. The interrupt timing is set to end of transfer.
Arguments	None
Return value	None
Remarks	None

[Function Name] SCHNG2TXS

Synopsis	Common processing for mode setup
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	None
Return value	None
Remarks	None

[Function Name] STARTCSIp

Synopsis	Enable CSIp.
Explanation	This function enables the CSIp.
Arguments	None
Return value	None
Remarks	None

[Function Name] STOPCSIp

Synopsis	Disable CSIp.
Explanation	This function disables the CSIp.
Arguments	None
Return value	None
Remarks	None

[Function Name] IINTCSIp

Synopsis	Start INTCSIp interrupt processing.
Explanation	This function is activated on an INTCSIp and causes a branch to the address that is stored in the variable RCSISUBADDR.
Arguments	None
Return value	None
Remarks	None

[Function Name] CSITXEND

Synopsis	Perform 1-character transfer end interrupt processing.
Explanation	This function reads receive data from the CSIp into RRCVBUF and sets the number of work-in-progress characters (CSISTS) to 0.
Arguments	None
Return value	None
Remarks	None

[Function Name] SRXNEXT

Synopsis	Perform 1-character transfer end interrupt processing in continuous reception mode.
Explanation	This function reads receive data from the CSIp into the buffer area and decrements the number of characters (CSISTS) by 1. If the number of remaining characters is 2 or more, the function writes dummy data into the SIOp to start the receive function. If the number of remaining characters is 1, the function switches the interrupt timing to transfer end interrupt. The function terminates processing when the number of reaming characters is 0.
Arguments	None
Return value	None
Remarks	None

[Function Name] STXNEXT

Synopsis	Perform buffer empty interrupt processing in continuous transmission mode.
Explanation	If the number of remaining characters is 1, this function switches the interrupt timing to transfer end interrupt and changes the value of RCSISUBADDR to the address of STXEND as an INTCSIp processing routine. If the number of remaining characters is 2 or more, the function decrements the number of work-in-progress characters (CSISTS) by 1 and writes the data from the transmit data buffer into the SIOp.
Arguments	None
Return value	None
Remarks	None

[Function Name] STXEND

Synopsis	Perform transmit end interrupt processing in continuous transmission mode.
Explanation	This function performs transmit end interrupt processing in continuous transmission mode. The function sets the number of work-in-progress characters (CSISTS) to 0 to signals the end of communication.
Arguments	None
Return value	None
Remarks	None

[Function Name] STRXNEXT

Synopsis	Perform buffer empty interrupt processing in continuous transmission/reception mode.
Explanation	This function stores the receive data in the receive data buffer. If the number of remaining characters is 2 or more, this function decrements the number of work-in-progress characters (CSISTS) by 1 and writes the data from the transmit data buffer into the SIOp. If the number of remaining characters is 1, the function switches the interrupt timing to transfer end interrupt and changes the value of RCSISUBADDR to the address of STRXEND as an INTCSIp processing routine.
Arguments	None
Return value	None
Remarks	None

[Function Name] STRXEND

Synopsis	Perform transfer end interrupt processing in continuous transmission/reception mode.
Explanation	This function performs transfer end interrupt processing in continuous transmission/reception mode. The function stores the receive data in the receive data buffer and sets the number of work-in-progress characters (CSISTS) to 0 to signal the end of communication.
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.

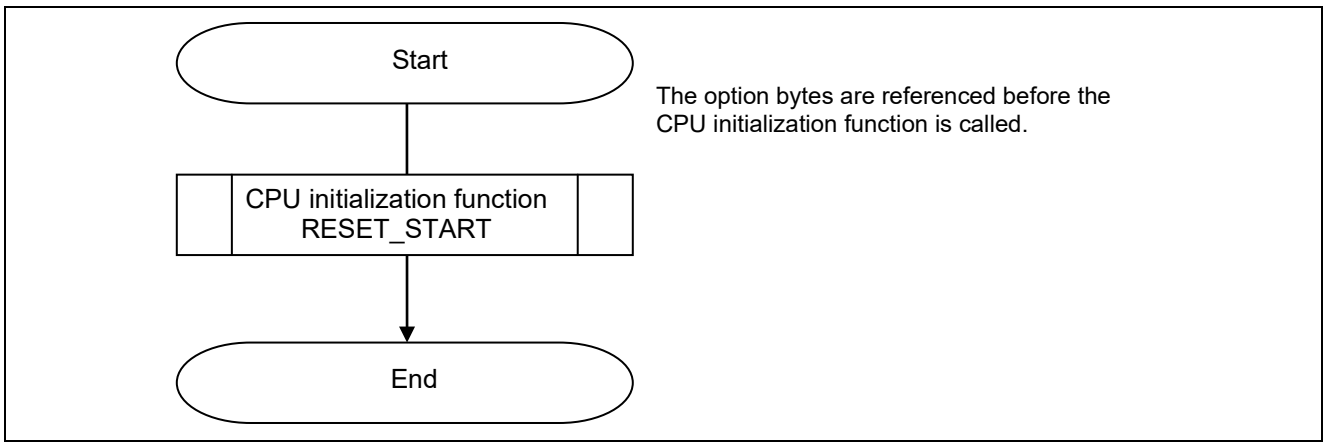


Figure 5.1 Overall Flow

5.7.1 CPU Initialization Function

Figure 5.2 shows the flowchart for the CPU initialization function.

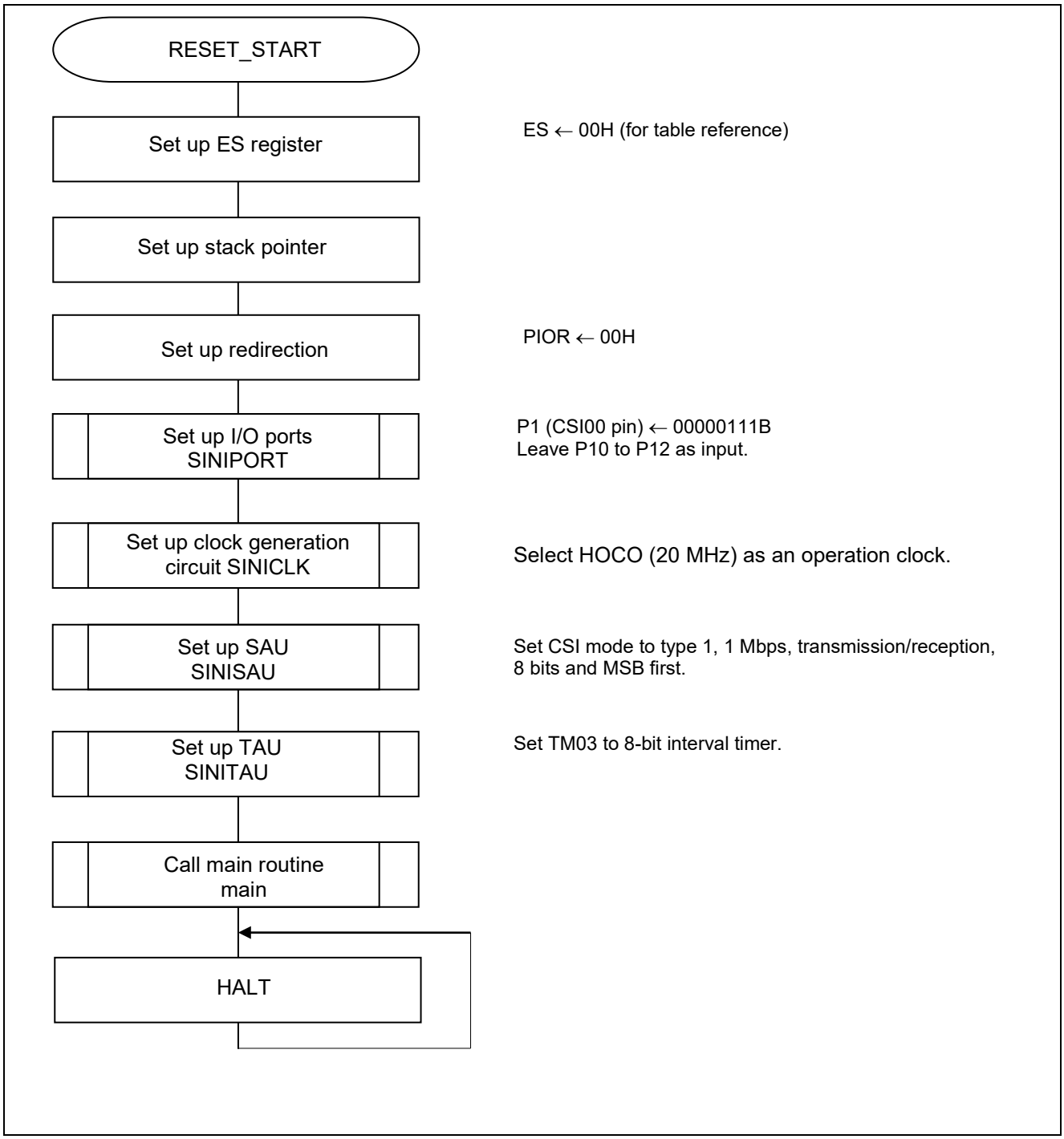


Figure 5.2 CPU Initialization Function

5.7.2 I/O Port Setup

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

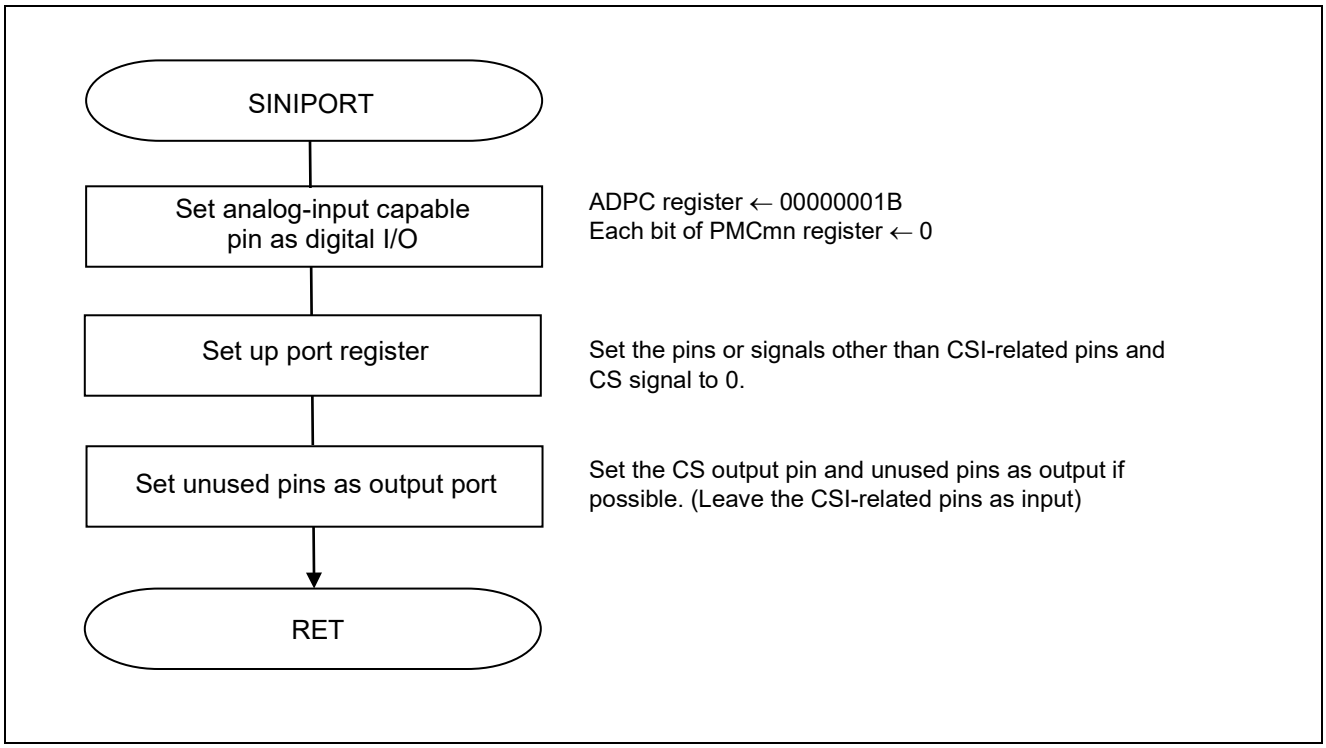


Figure 5.3 I/O Port Setup

Note: Refer to the section entitled "Flowcharts" in RL78/G10 Initialization CC-RL 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 Clock Generation Circuit Setup

Figure 5.4 shows the flowchart for clock generation circuit setup.

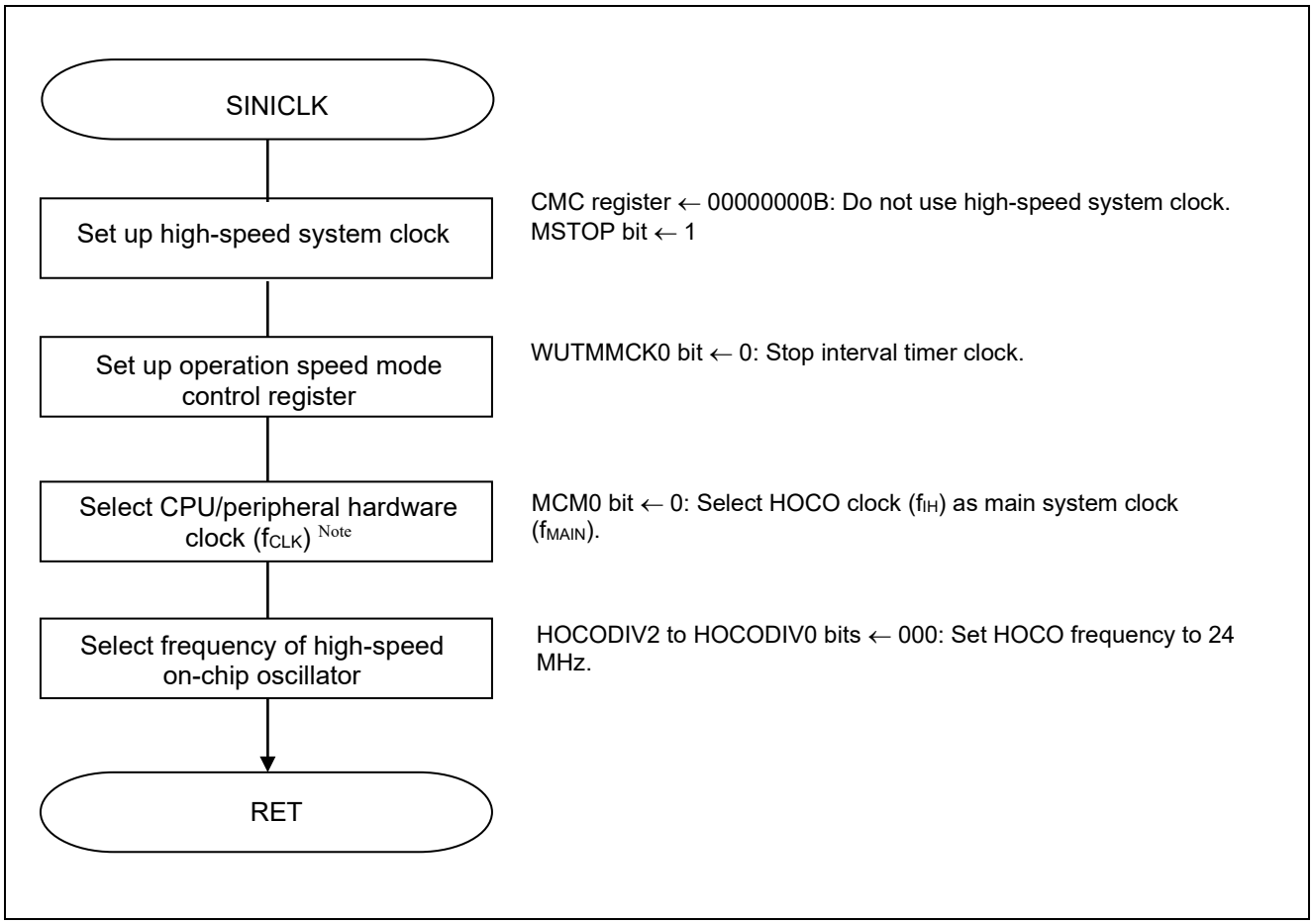


Figure 5.4 Clock Generator Circuit Setup

Note: 16-pin prodcuts only

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.

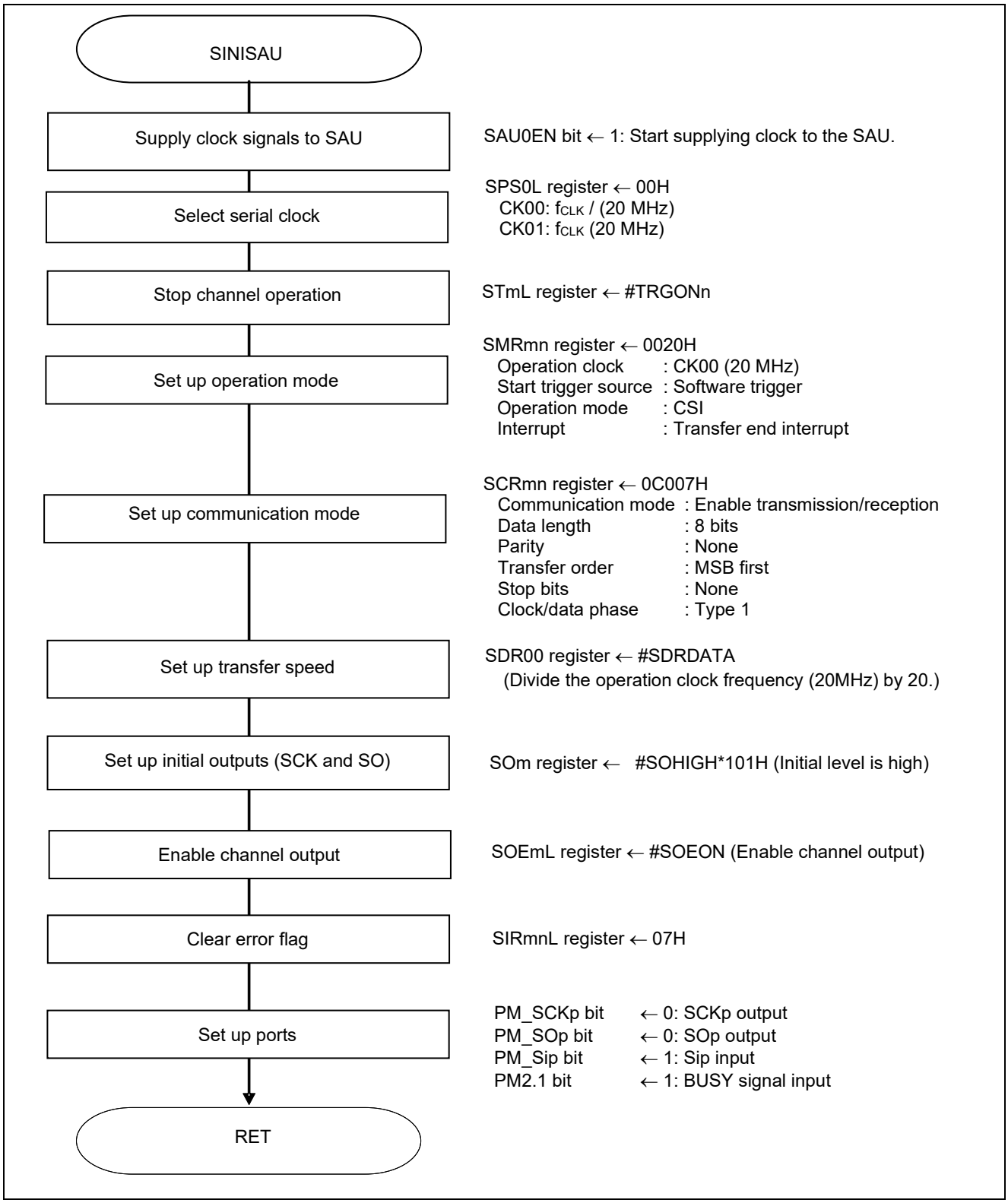


Figure 5.5 SAU Setup

Starting clock signal supply to 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 select 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	**0/1**	**0/1**

Bit n

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

Setting up channel operation mode

- Serial mode register mn (SMRnH, SMRnL)
  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	**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 7 (SMR0nH)

CKSmn	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)

CCSmn	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)

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

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

Note Provided in the SMR01H and 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	**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 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 Provided in the SMR01H and 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 (SCRnH, SCRnL)
  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)

SLCmn 1	SLCmn 0	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

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

Note: SCR00L register 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 0 (SCR0nL)

DLSmn0	Setting of data length in CSI mode
0	7-bit data length
1	**8-bit data length**

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

Note: SCR00L register only

Setting up channel transfer clock

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

Symbol: SDR0nH

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

Symbol: SDR0nL

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

Bits 7 to 1 (SDR0nH)

SDR0nH[7:1]							Transfer clock setting by dividing operation clock (f_{MCK})
0	0	0	0	0	0	0	f_{MCK} /2
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
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
0	**0**	**0**	**1**	**0**	**0**	**1**	**f_{MCK} /20**
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
1	1	1	1	1	1	0	f_{MCK} /254
1	1	1	1	1	1	1	f_{MCK} /256

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

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

Setting initial output level

- Serial output register m (SOm)
  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”.**

略号：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	Serial data output of channel n
0	Serial clock output value is "0".
1	**Serial clock output value is "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

SOE0n	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 mn (SIRmn)
  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 FEFmn bit of the SSRmn registers to 0.**

Bit 1

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

Bit 0

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

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

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) (n=0-2)
0	0 is output
1	**1 is 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	**0**	**1**	**0**

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 product only

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

5.7.5 Timer Array Unit Setup

Figure 5.6 shows the flowchart for setting up the timer array unit.

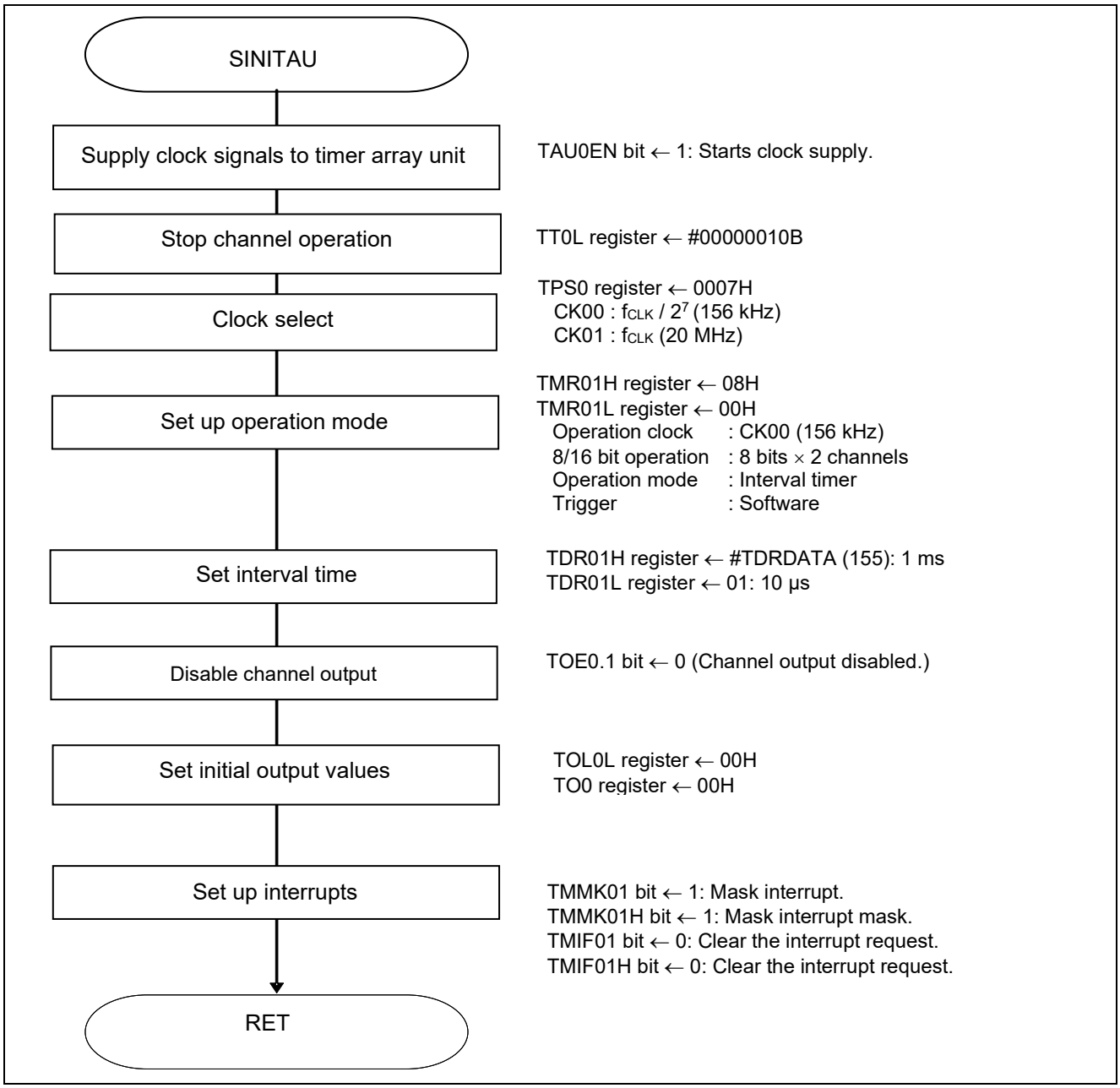


Figure 5.6 Timer Array Unit Setup

5.7.6 Main Processing

Figures 5.7 to 5.9 show the flowcharts for the main processing.

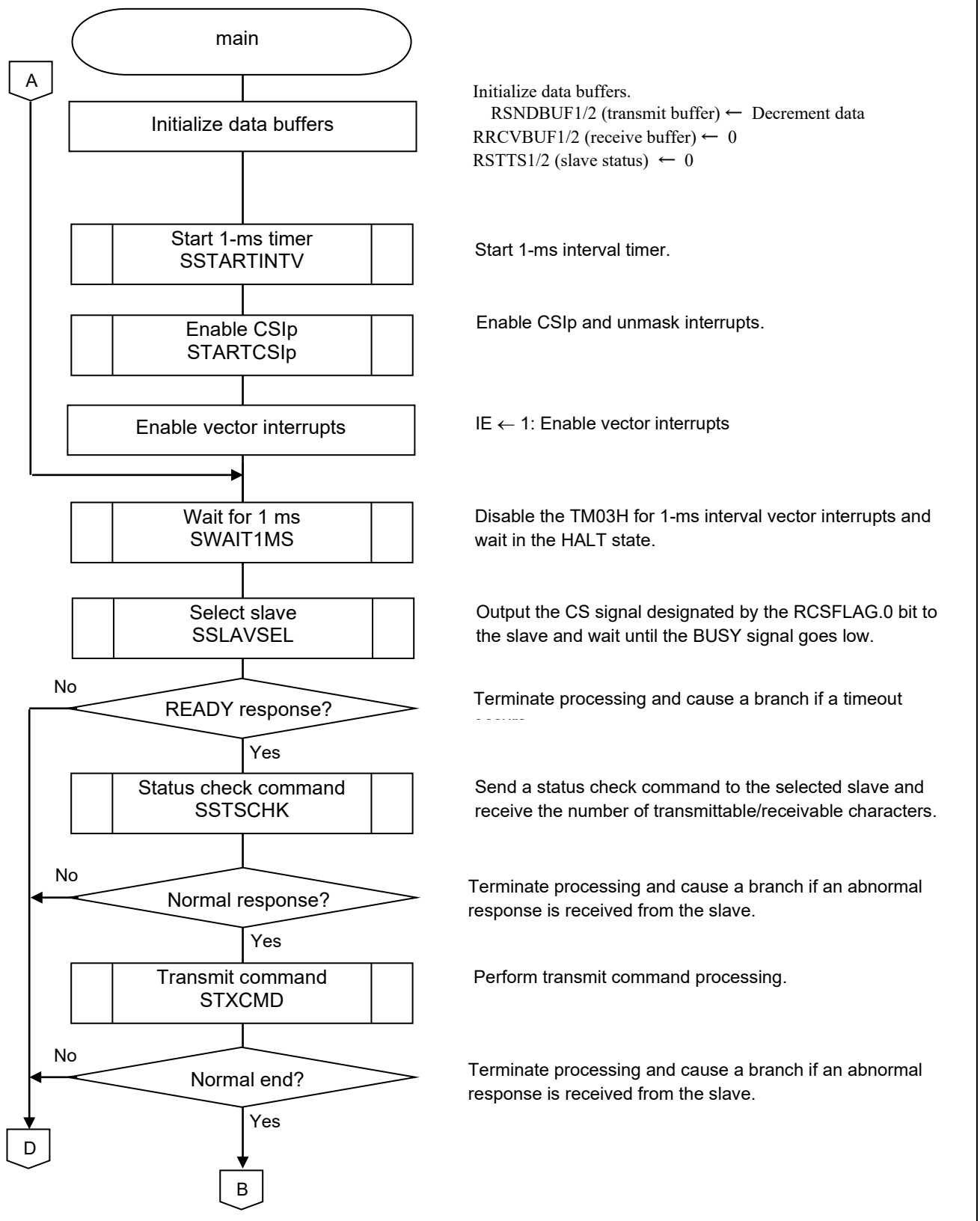


Figure 5.7 Main Processing (1/3)

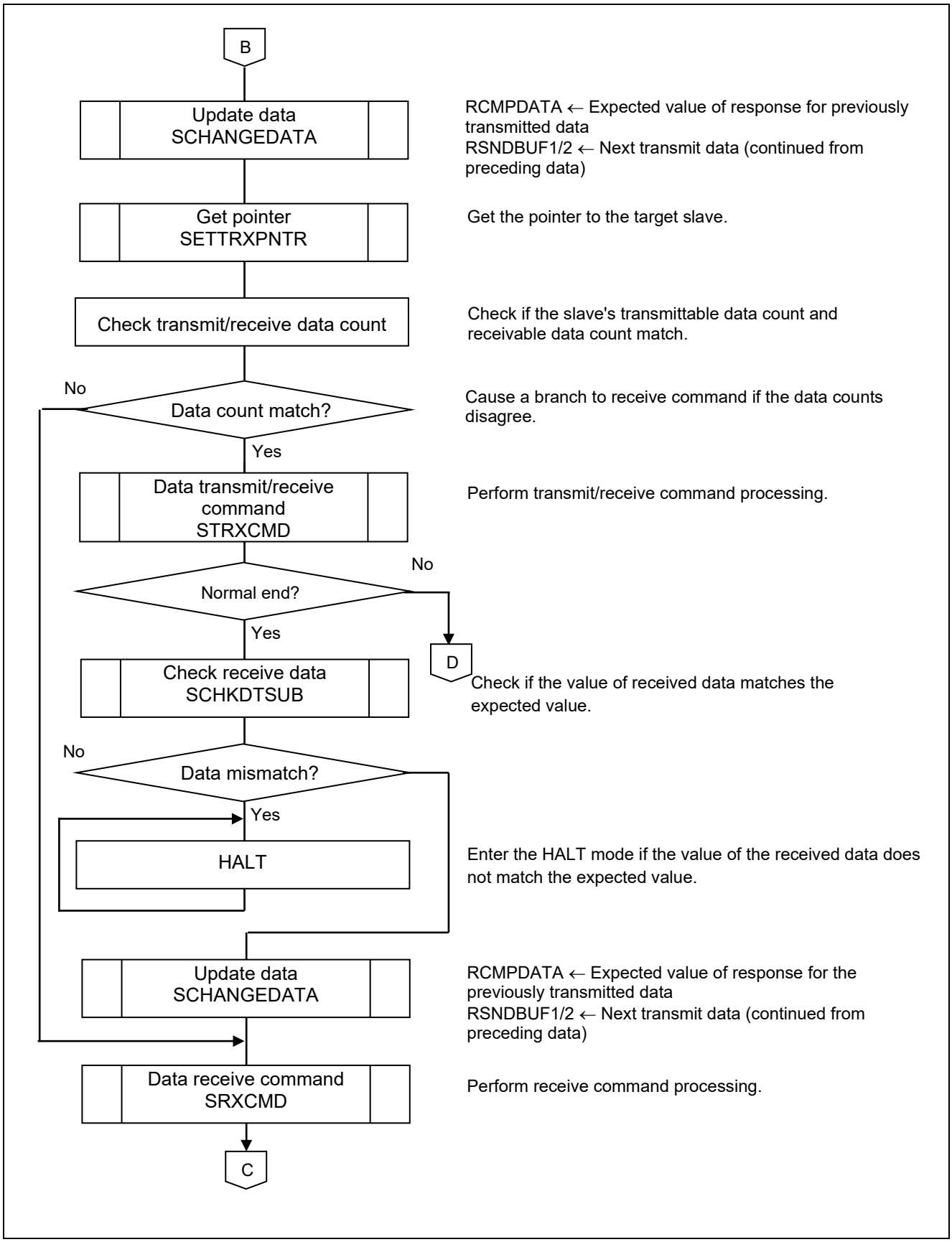


Figure 5.8 Main Processing (2/3)

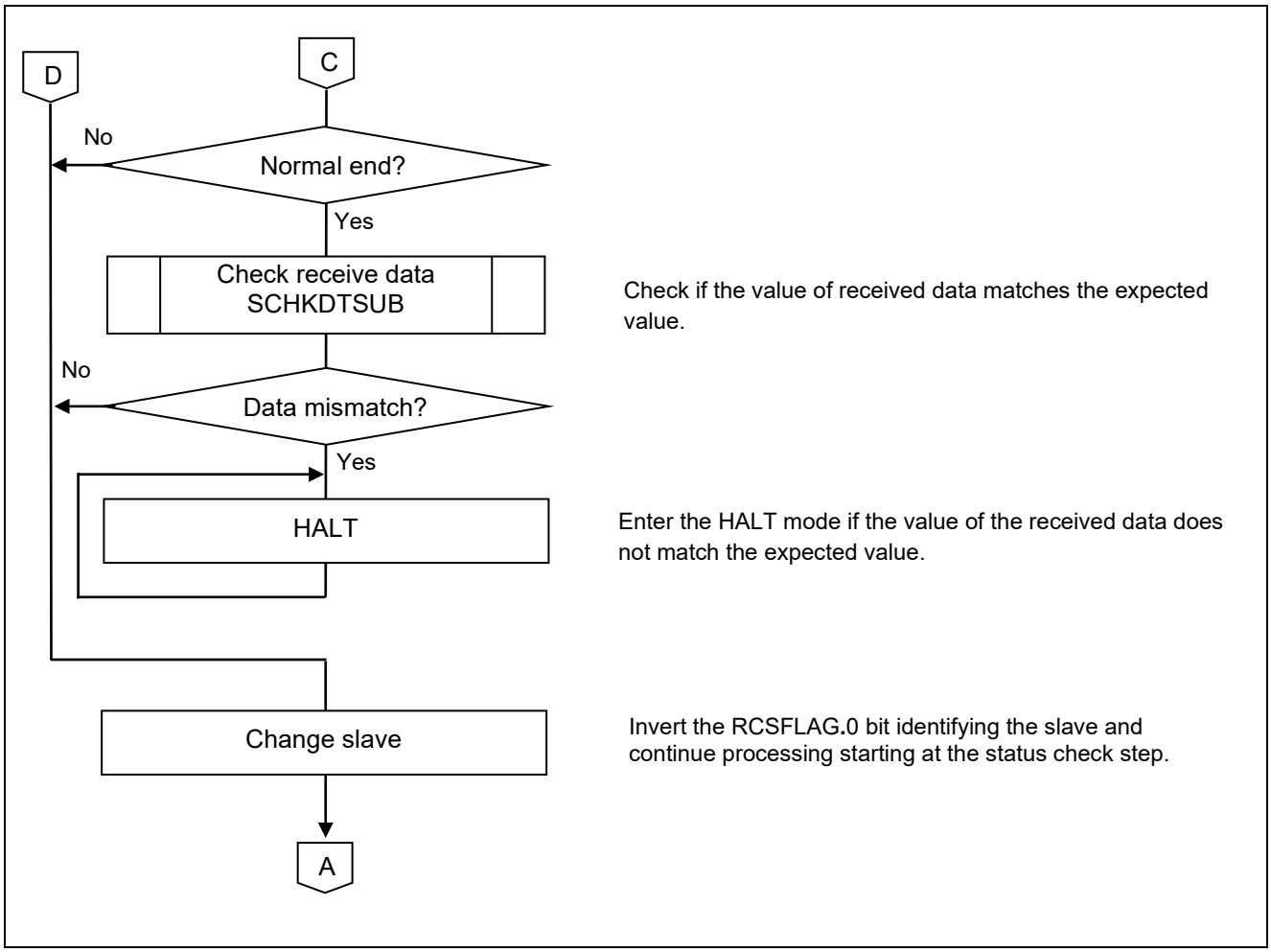


Figure 5.9 Main Processing (3/3)

5.7.7 1-ms Interval Timer Startup Processing

Figure 5.10 shows the flowchart for the 1-ms interval timer startup function.

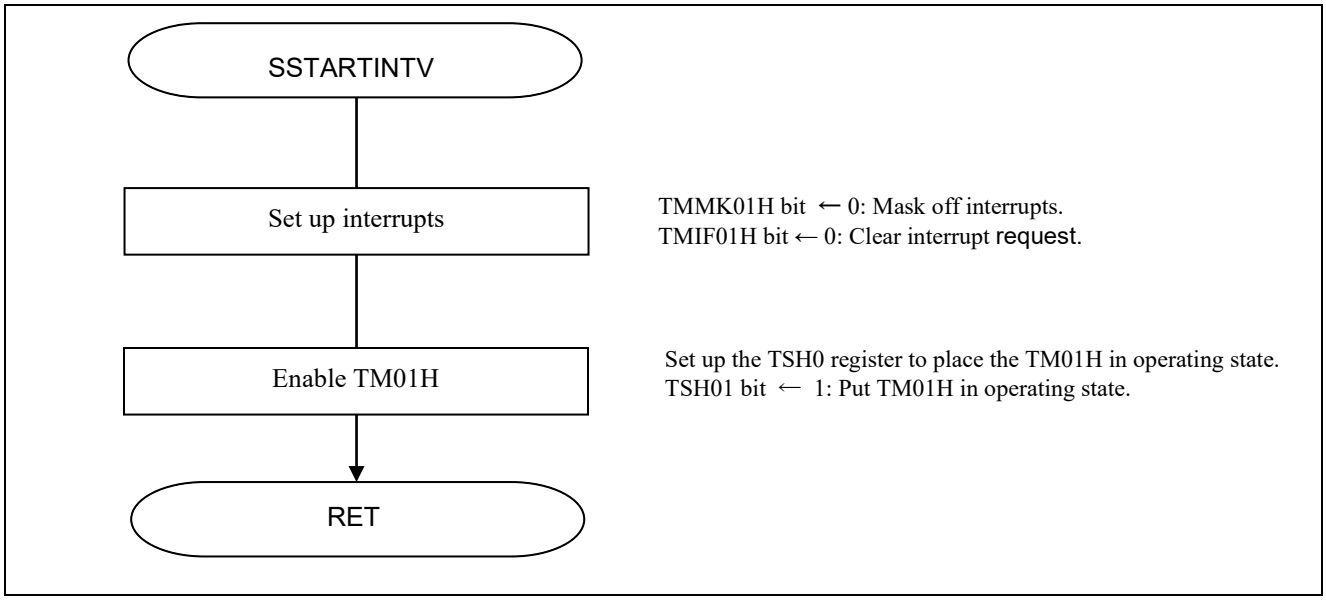


Figure 5.10 1-ms Interval Timer Startup Processing

Interrupt setting

- Interrupt request flag register (IF0L)
  Clear the interrupt request flag
- Interrupt mask flag register (MK0H)
  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

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

Symbol: MK0H (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

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.

Transiting to timer operating state

- Timer channel startup register 0 (TSH0)
  Start counting.

Symbol: TSH0

7	6	5	4	3	2	1	0
0	0	0	0	TSH03 Note	0	TSH01	0
0	0	0	0	x	0	**1**	0

Bit 1

TSH01	Operation start trigger of channel 01H
0	No trigger operation
1	**The TEH03 bit is set to 1 and the count operation becomes enabled.**

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

5.7.8 1-ms Interval Wait Function

Figure 5.11 shows the flowchart for the 1ms interval wait function.

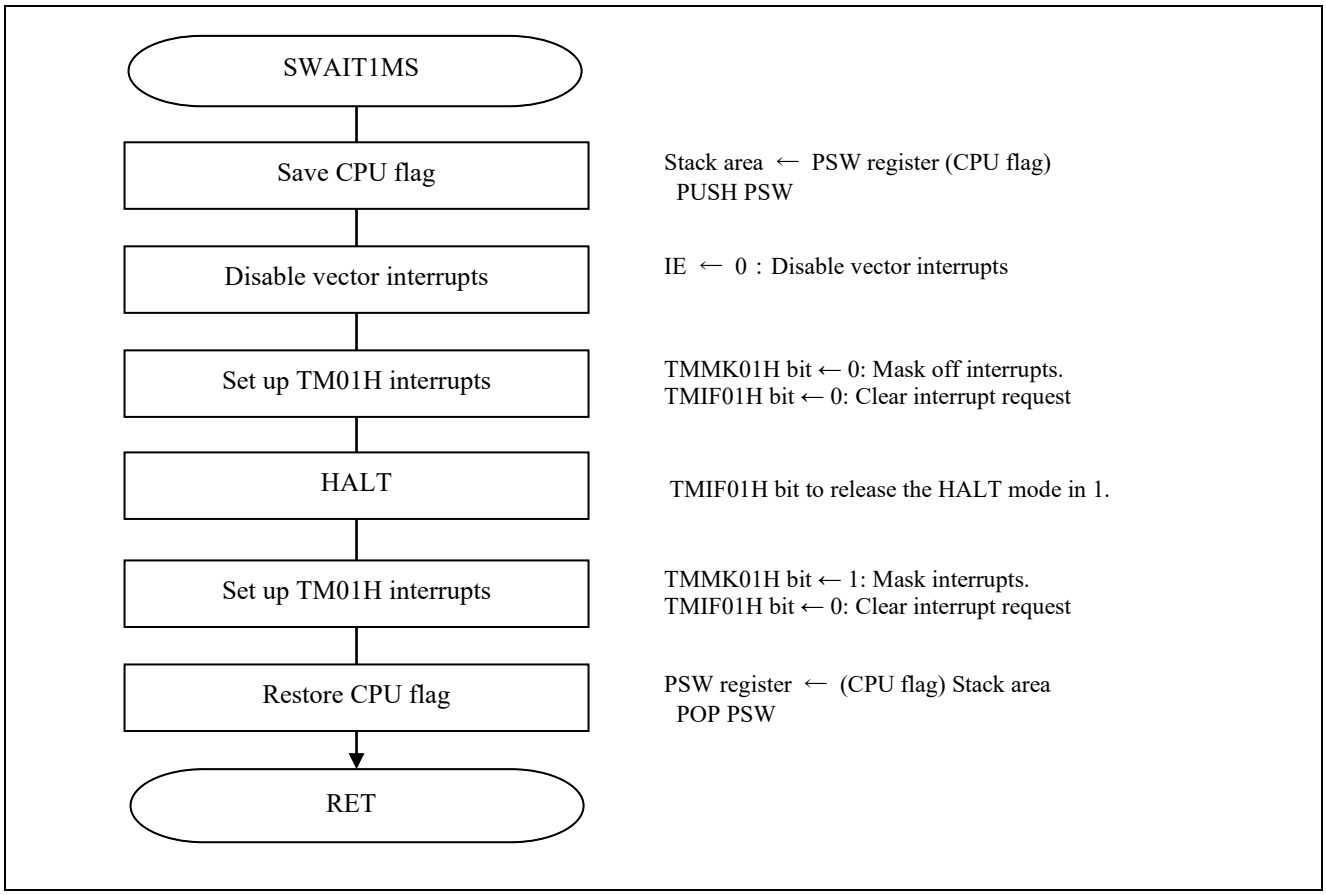


図 5.11 1-ms Interval Wait Function

Interrupt setting

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

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

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

5.7.9 Slave Selection Processing

Figure 5.12 shows the flowchart for the slave selection processing.

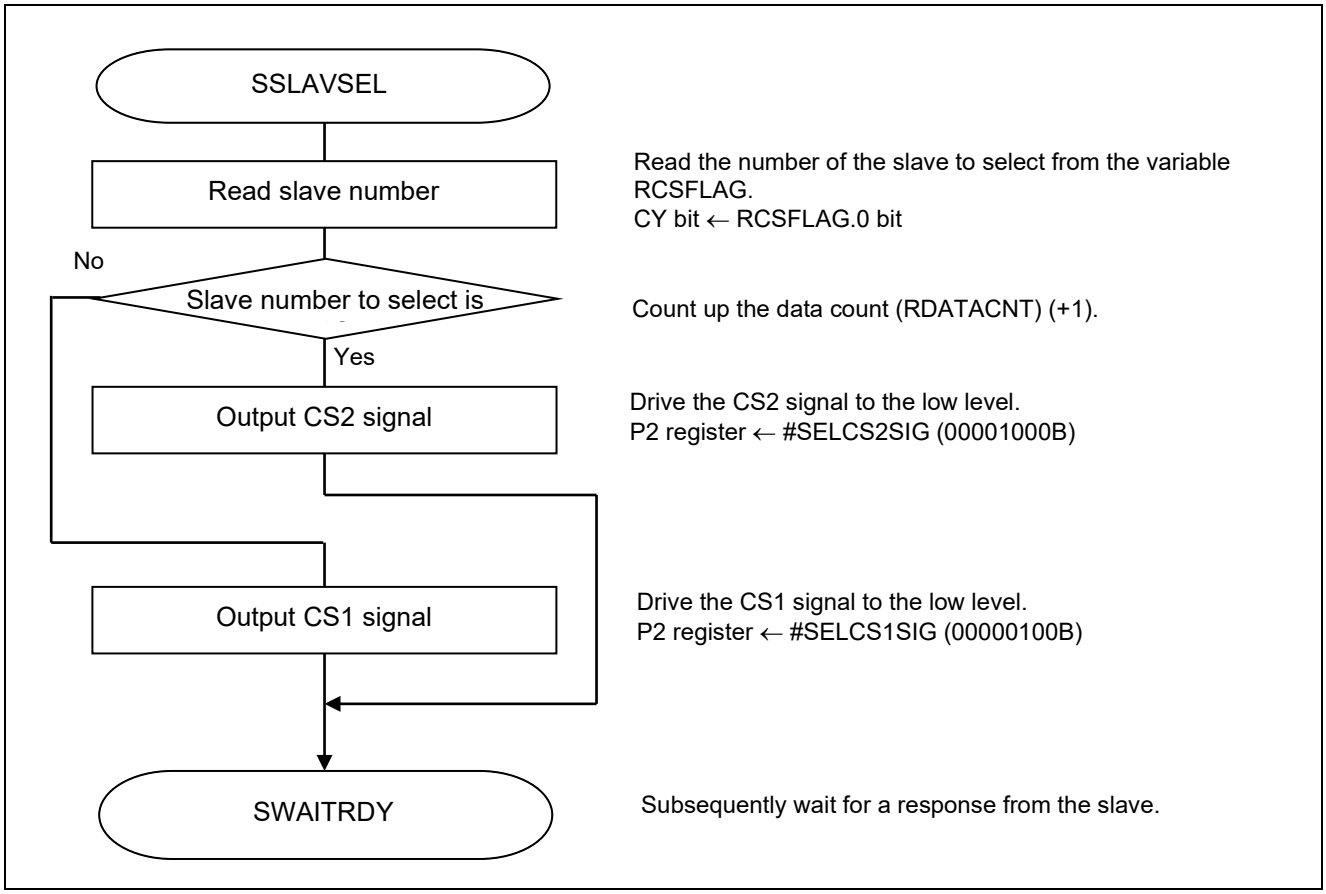


Figure 5.12 Slave Selection Processing

5.7.10 Slave Response Wait Processing

Figure 5.13 shows the flowchart for the slave response wait processing.

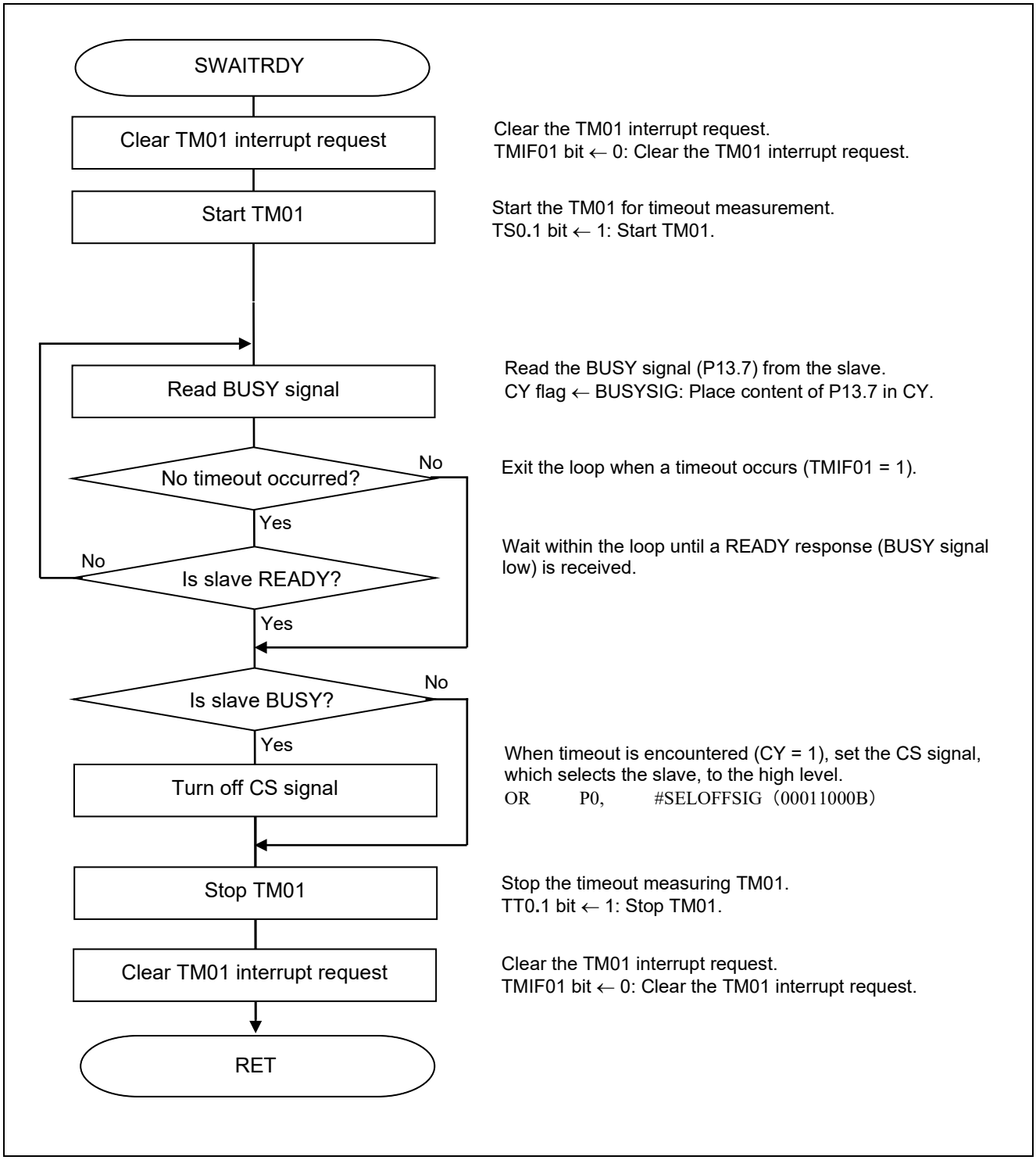


Figure 5.13 Slave Response Wait Processing

Transiting to timer operating state

- Timer channel startup register 0 (TS0)
  Start counting.

Symbol: TS0

7	6	5	4	3	2	1	0
0	0	0	0	TS03 Note	TS02 Note	TS01	TS00
0	0	0	0	x	x	**1**	x

Bit 3

TS01	Operation enable (start) trigger of channel 1
0	No trigger operation
1	**TE01 is set to 1 and the count operation becomes enabled.**

Transiting to timer stopped state

- Timer channel stop register 0 (TT0)
  Stop counting.

Symbol: TT0

7	6	5	4	3	2	1	0
0	0	0	0	TT03	TT02	TT01	TT00
0	0	0	0	x	x	**1**	x

Bit 3

TT01	Operation stop trigger of channel 1
0	No trigger operation
1	**TE01 is cleared to 0. Operation is stopped (stop trigger is generated.**

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

Interrupt setting

- Interrupt request flag register (IF0H)
  Clear the interrupt request flag
- Interrupt mask flag register (MK0H)
  Cancel interrupt mask

Symbol: IF0H (10-pin products)

7	6	5	4	3	2	1	0
0	0	0	0	0	KRIF	ADIF	TMIF01
0	0	0	0	0	x	x	**0**

Bit 0

TMIF01	Interrupt request flag
0	**No interrupt request signal is generated**
1	Interrupt request is generated, interrupt request status

Symbol: MK0H (10-pin products)

7	6	5	4	3	2	1	0
1	1	1	1	1	KRMK	ADMK	TMMK01
1	1	1	1	1	x	x	**1**

Bit 1

TMMK01	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.

5.7.11 Slave Status Check Processing

Figures 5.14 to 5.16 show the flowcharts for slave status check processing.

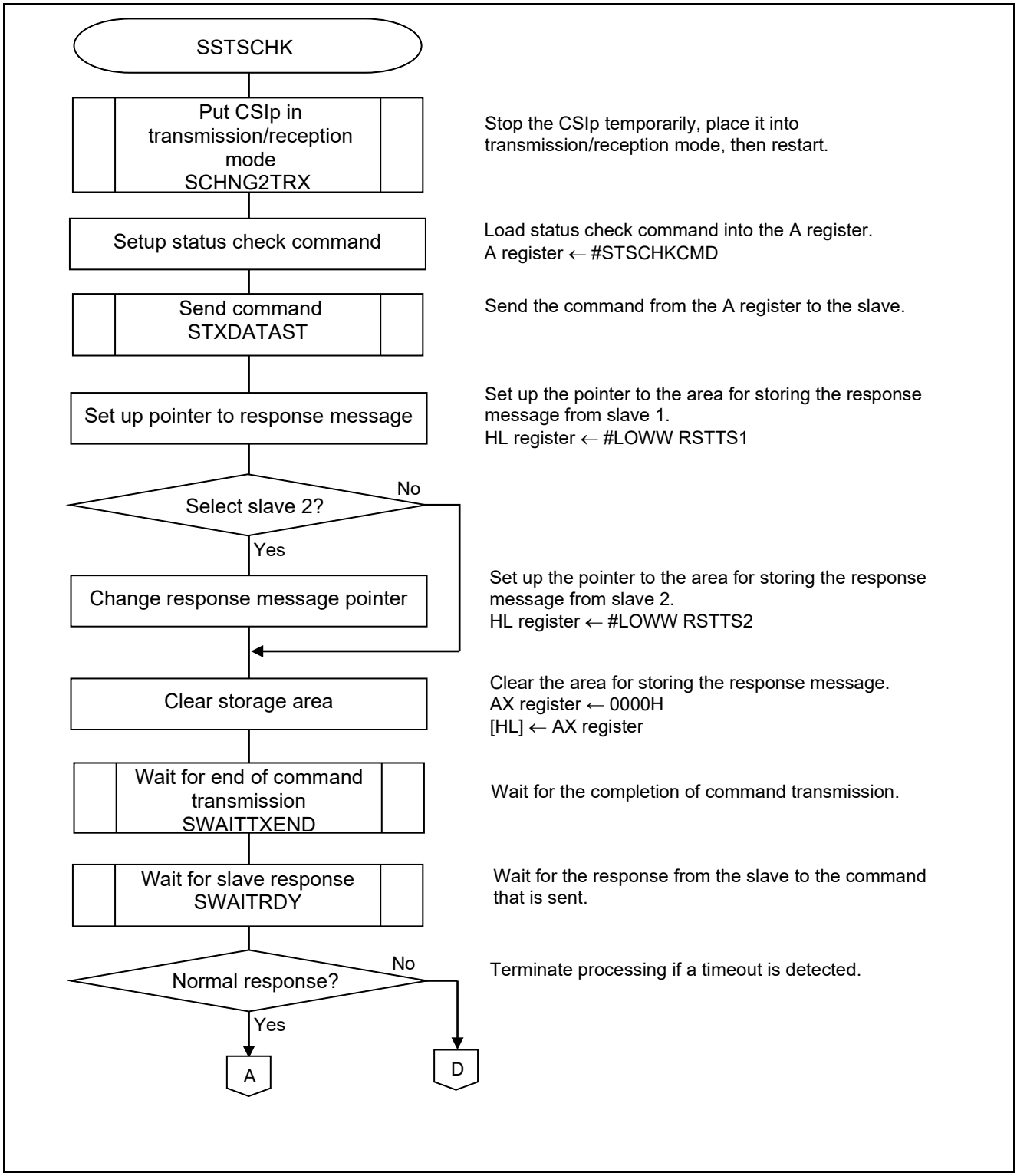


Figure 5.14 Status Check Processing (1/3)

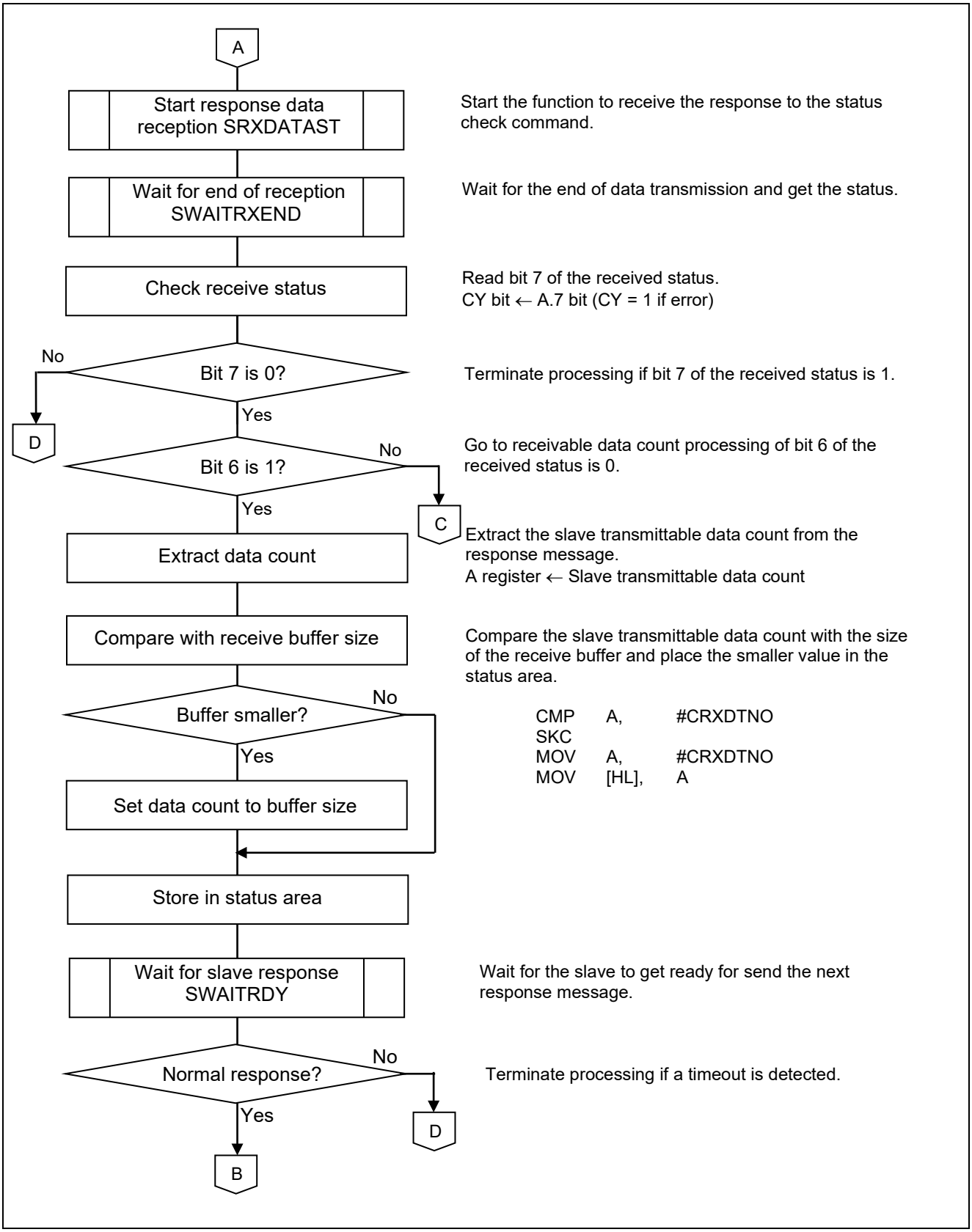


Figure 5.15 Status Check Processing (2/3)

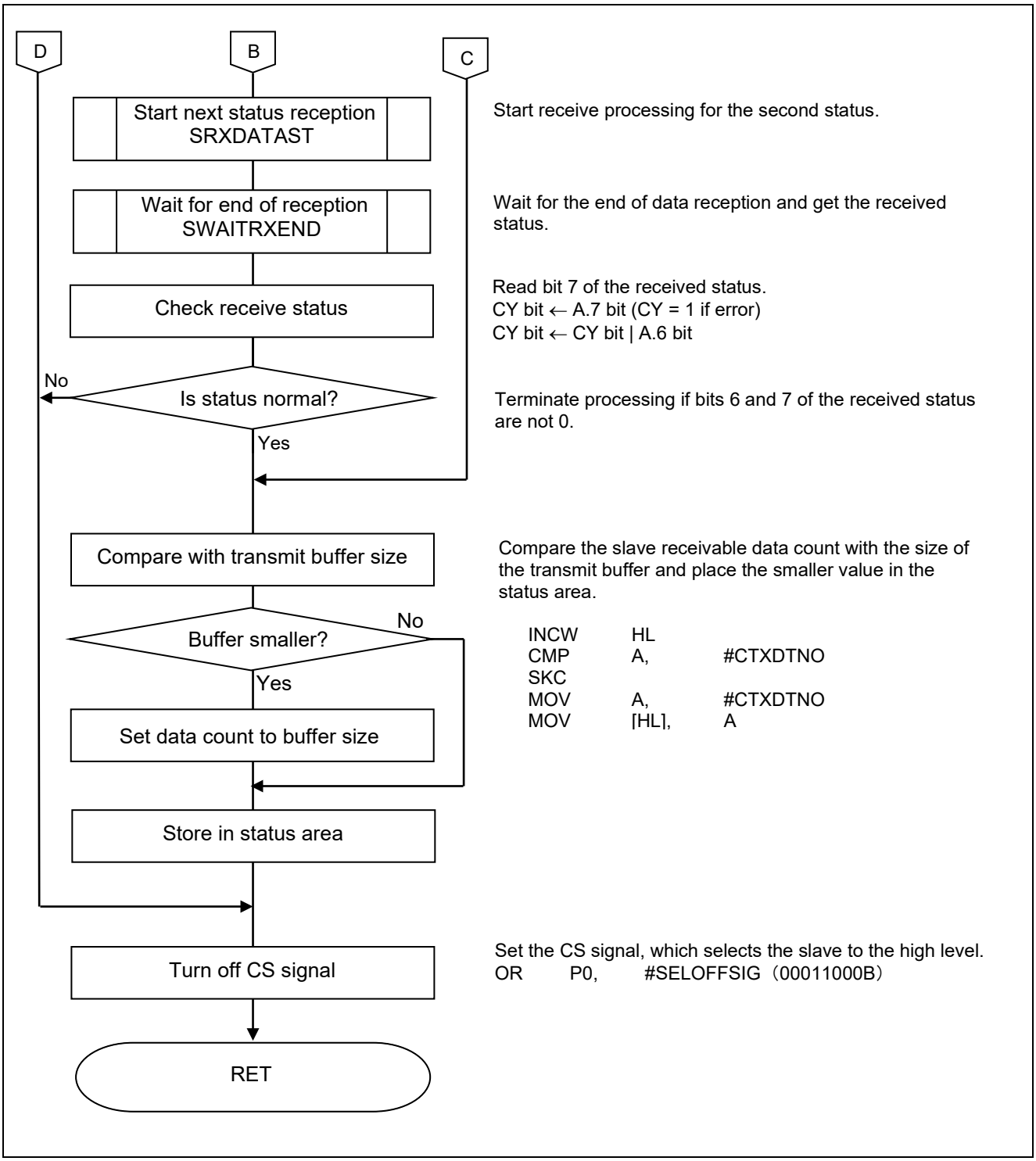


Figure 5.16 Status Check Processing (3/3)

5.7.12 Continuous Data Transmission Processing

Figures 5.17 and 5.18 show the flowcharts for the processing for transmitting data continuously to the slave.

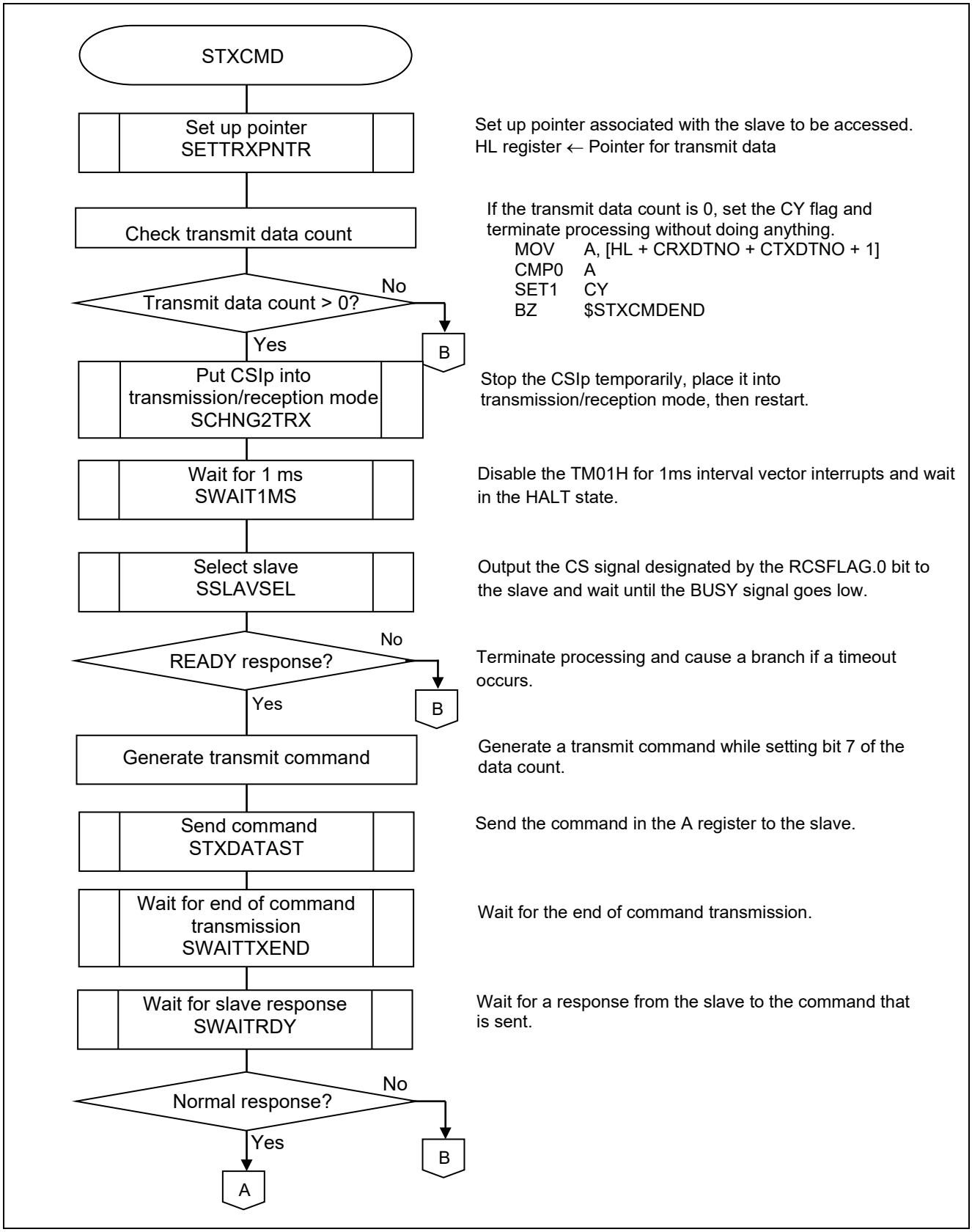


Figure 5.17 Continuous Data Transmission Processing (1/2)

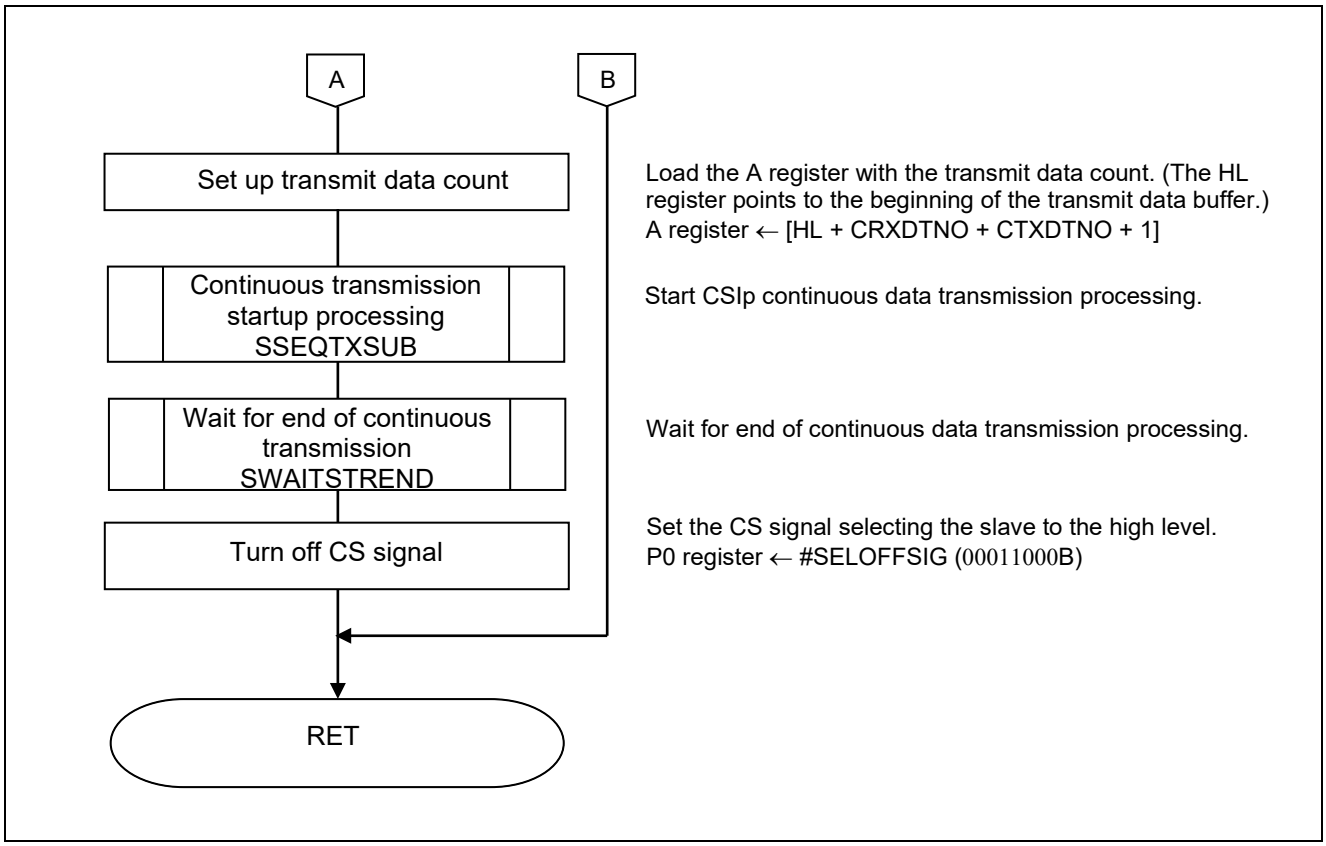


Figure 5.18 Continuous Data Transmission Processing (2/2)

5.7.13 Continuous Data Transmission/Reception Processing

Figures 5.19 and 5.20 show the flowcharts for the processing for transmitting and receiving data continuously to and from the slave.

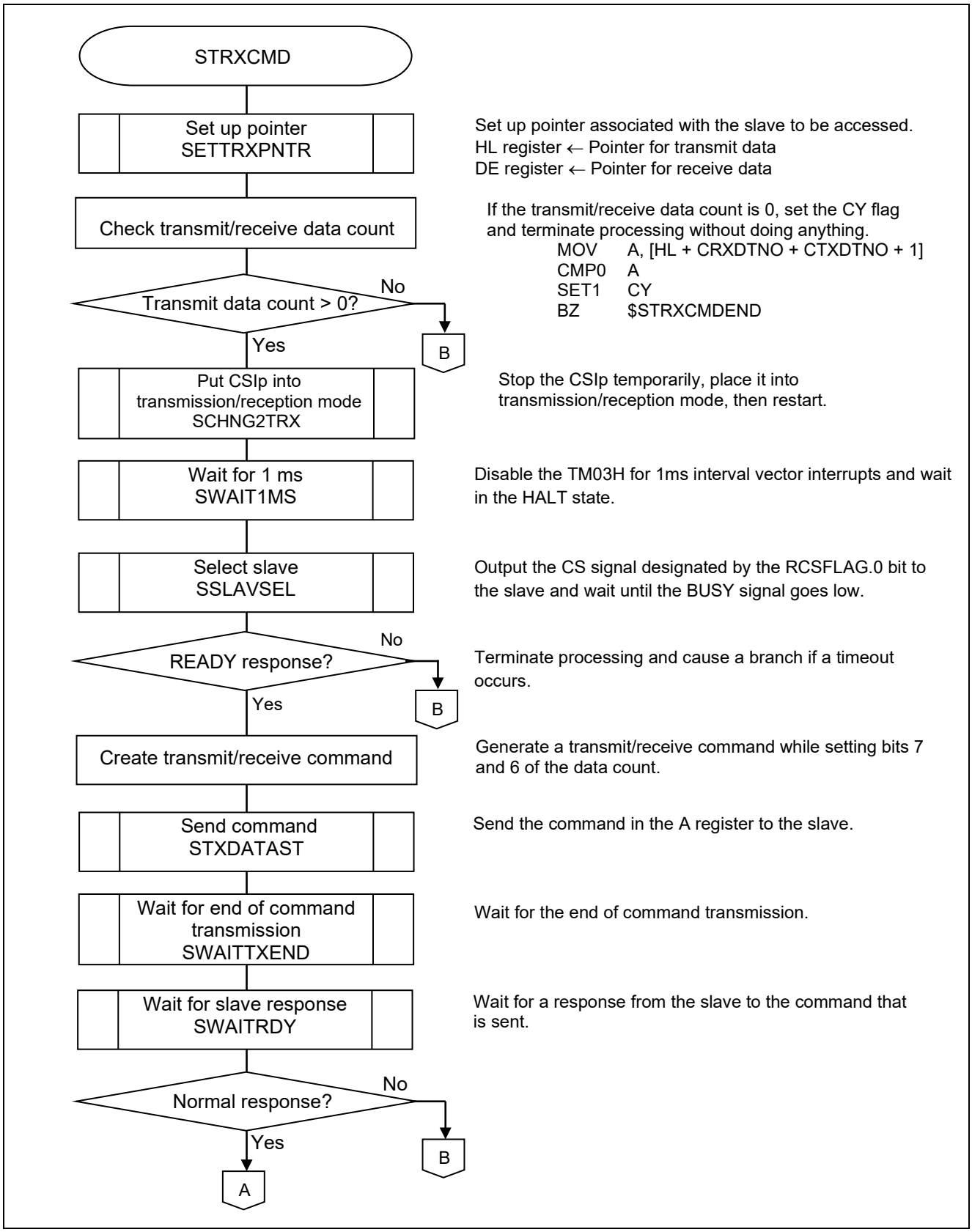


Figure 5.19 Continuous Data Transmission/Reception Processing (1/2)

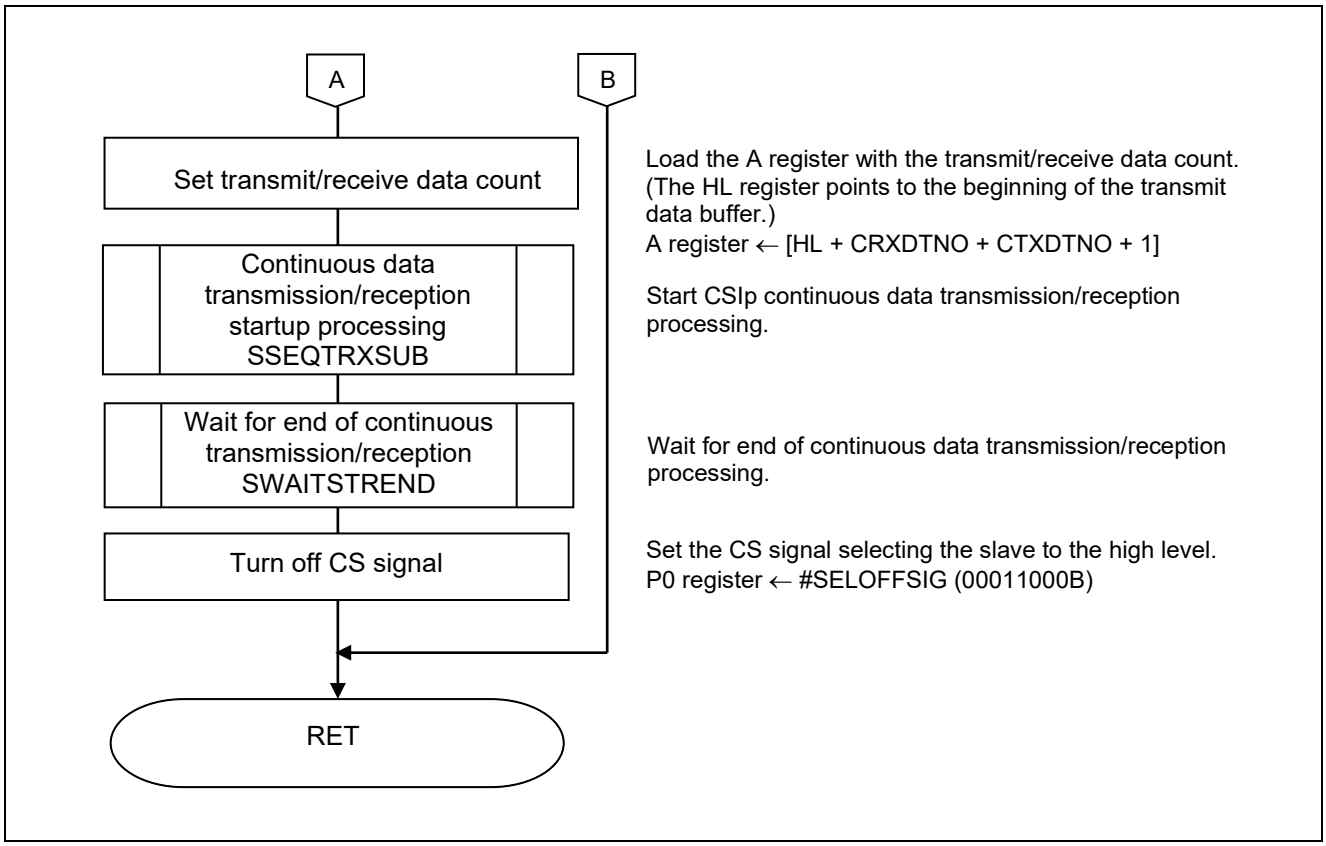


Figure 5.20 Continuous Data Transmission/Reception Processing (2/2)

5.7.14 Continuous Data Reception Processing

Figures 5.21 and 5.22 show the flowcharts for the processing for receiving data continuously from the slave.

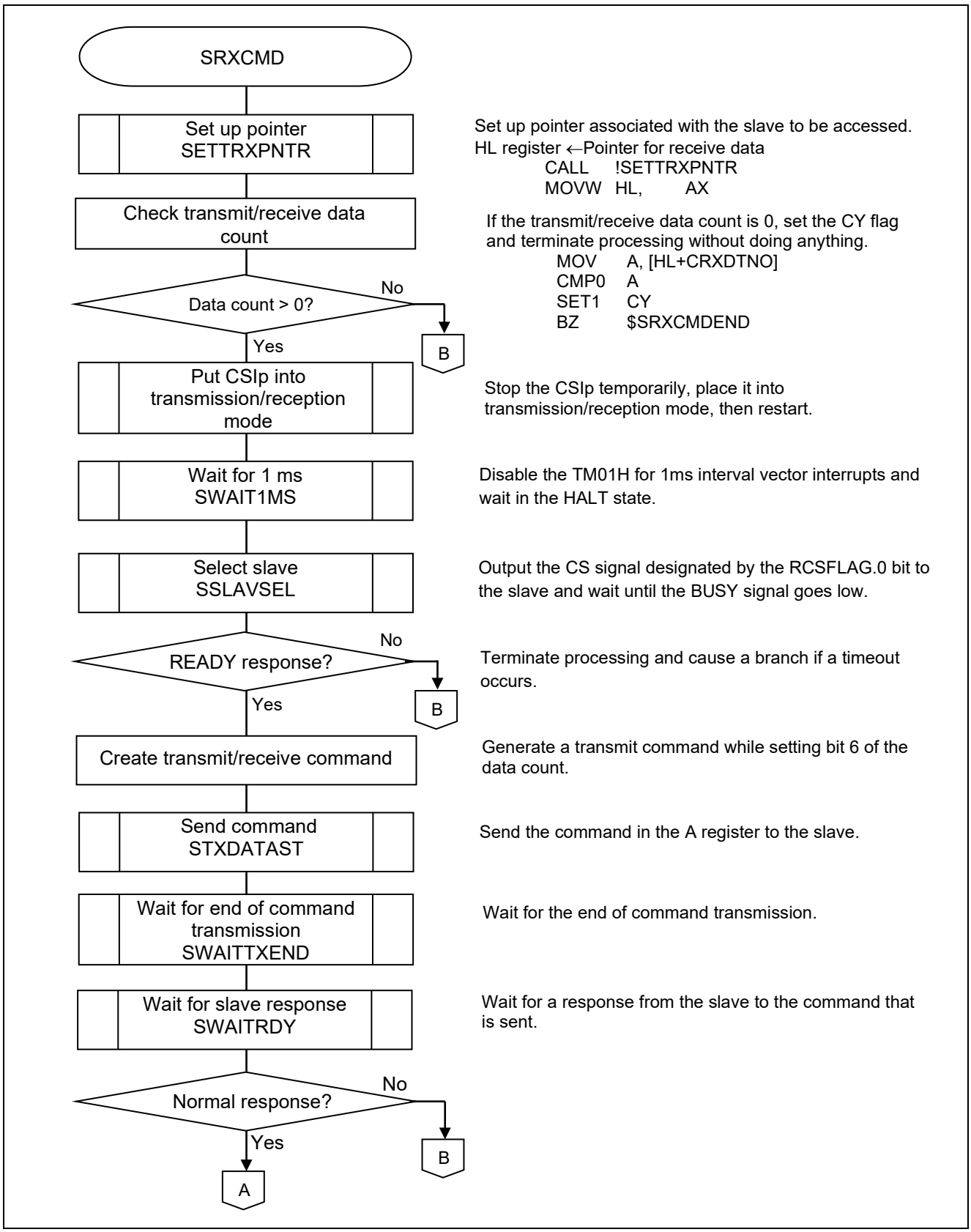


Figure 5.21 Continuous Data Reception Processing (1/2)

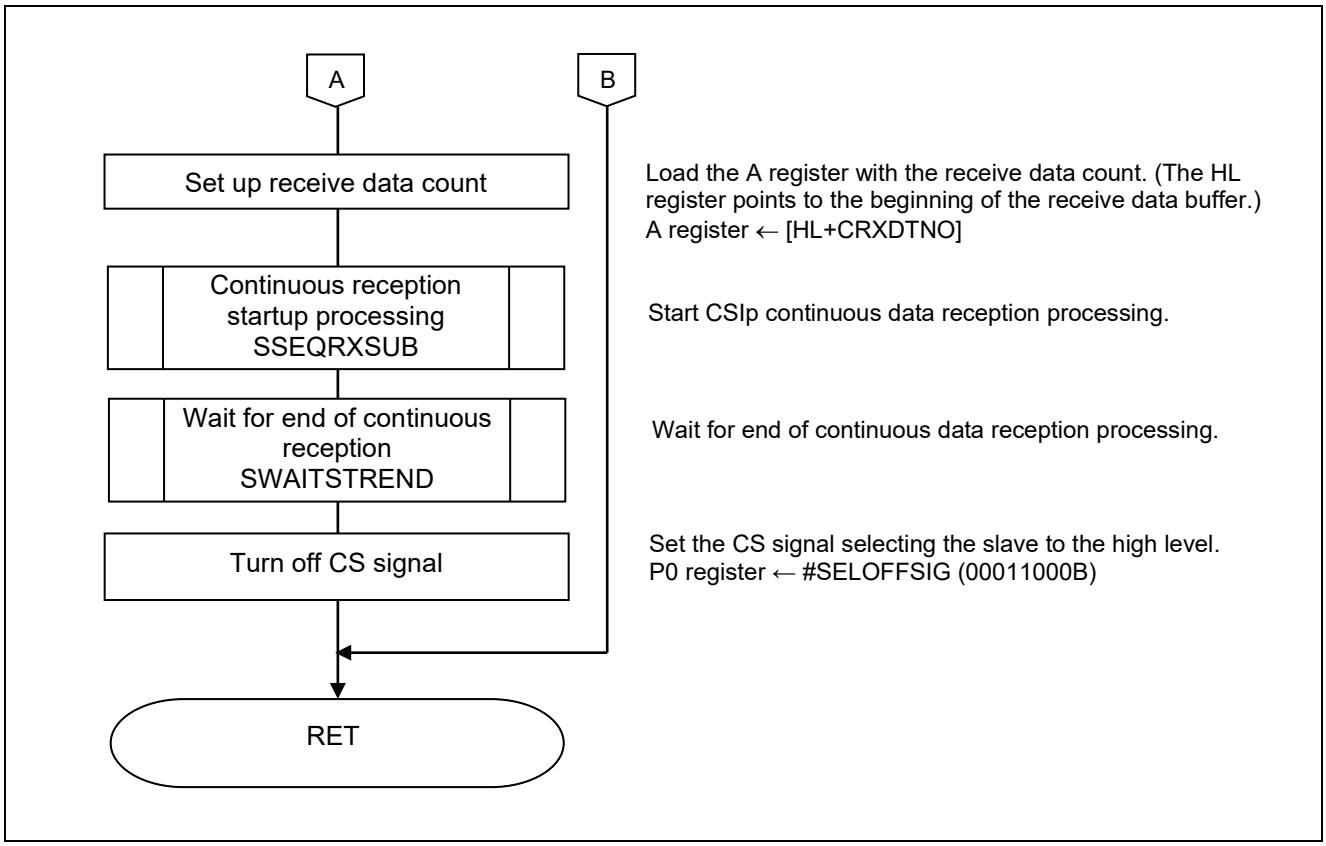


Figure 5.22 Continuous Data Reception Processing (2/2)

5.7.15 Data Update Processing

Figure 5.23 shows the flowchart for the data update processing.

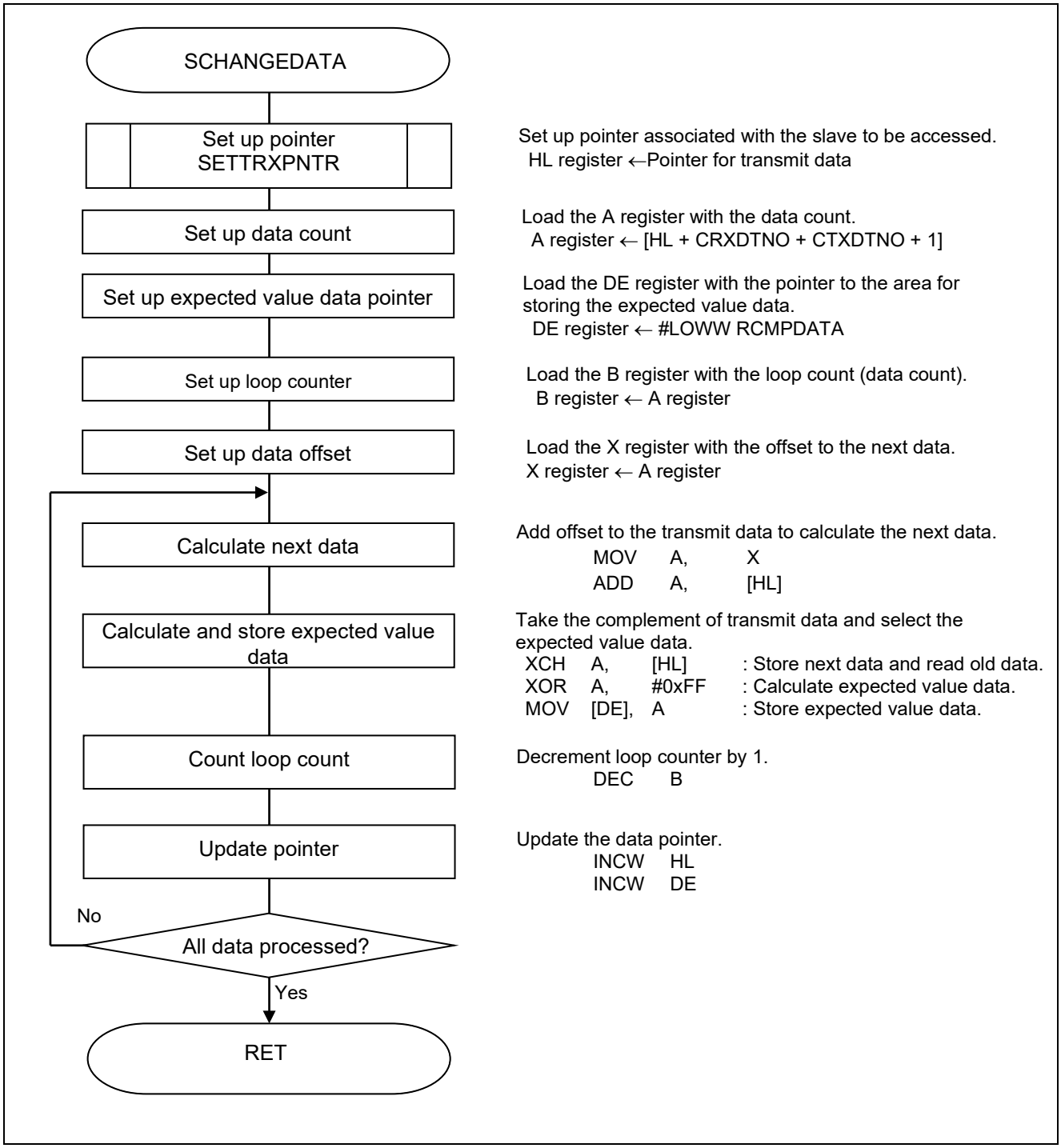


Figure 5.23 Data Update Processing

5.7.16 Receive Data Check Processing

Figure 5.24 shows the flowchart for receive data check processing.

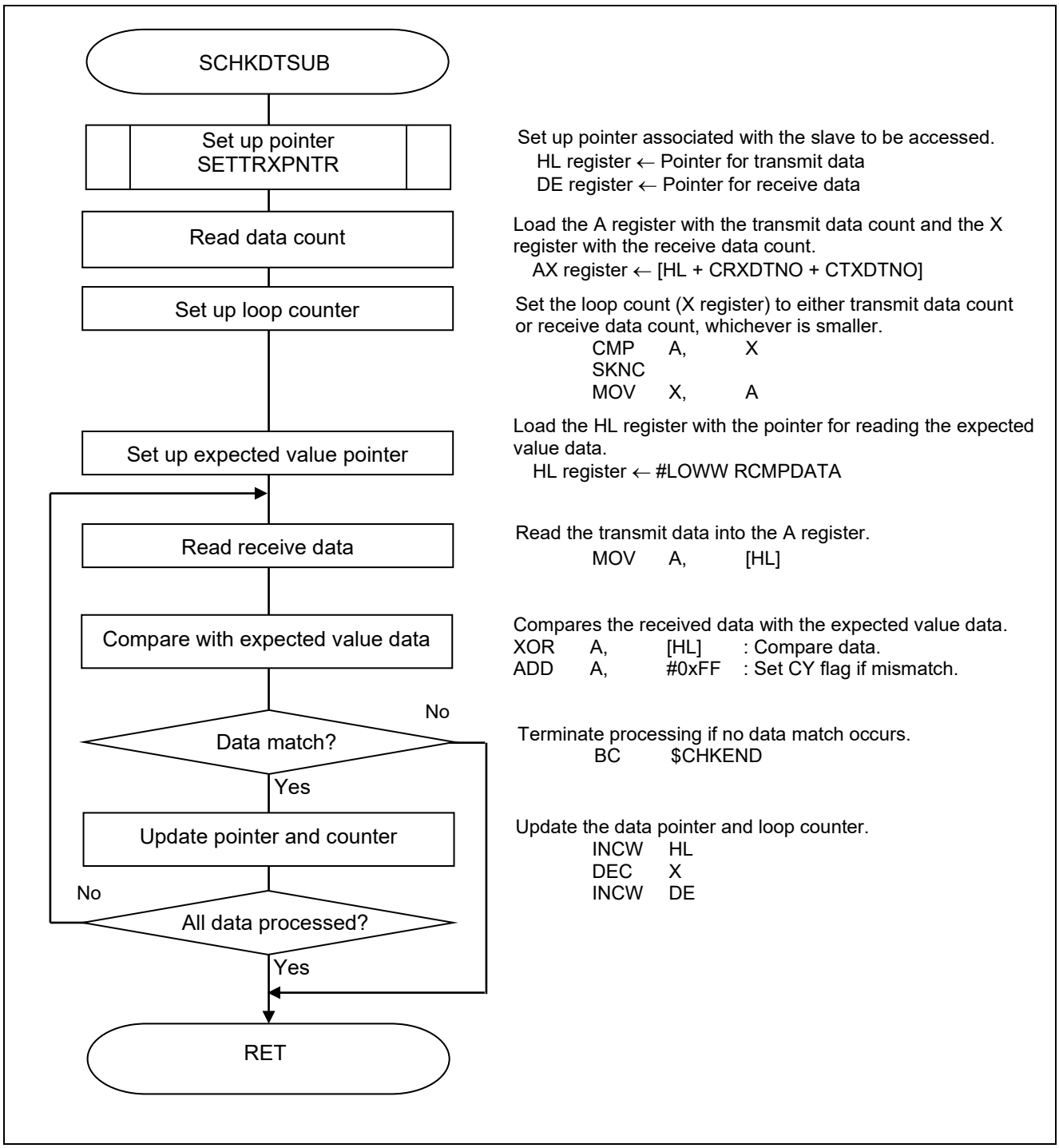


Figure 5.24 Receive Data Check Processing

5.7.17 Data Pointer Setup Processing

Figure 5.25 shows the flowchart for data pointer setup processing.

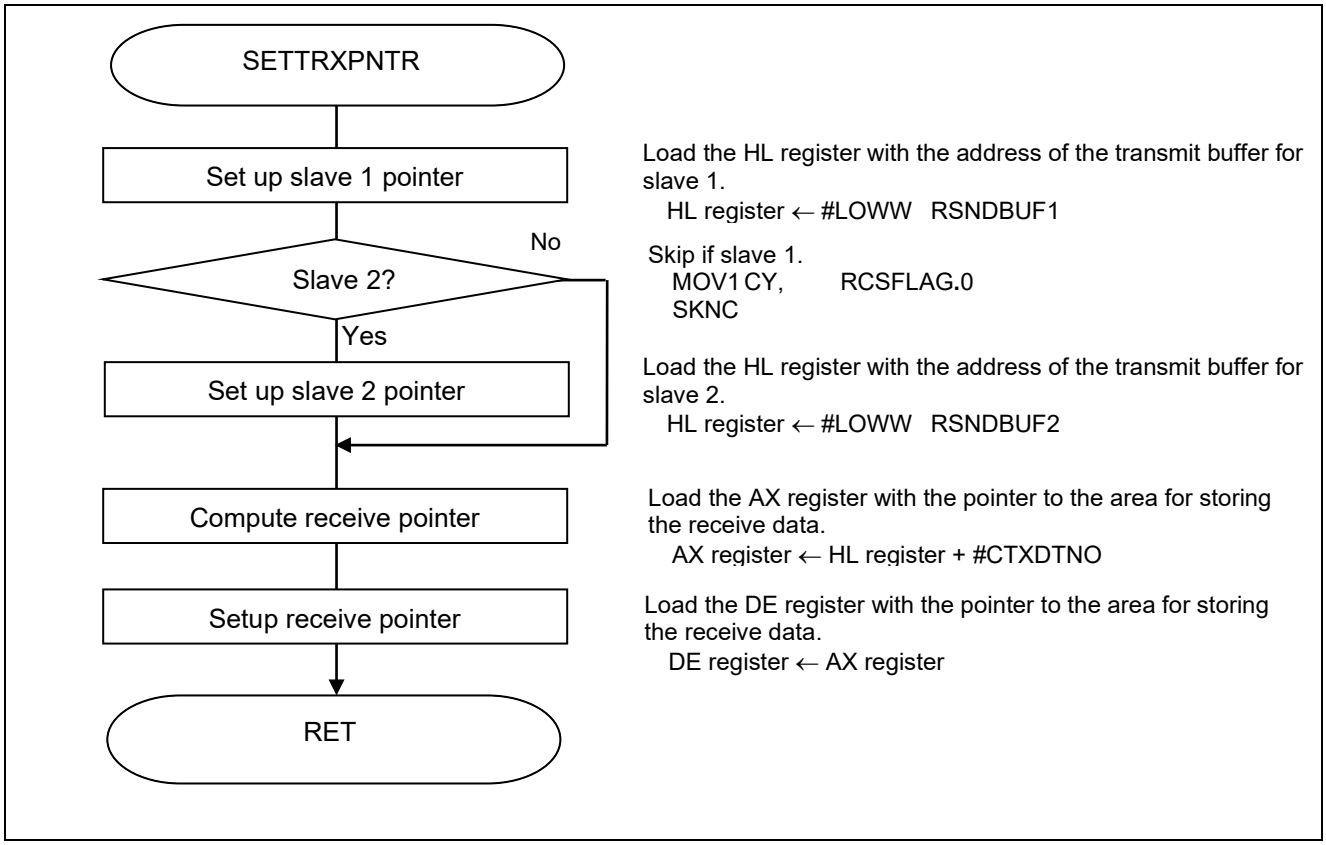


Figure 5.25 Data Pointer Setup Processing

Given below is a collection of subroutines that are used to perform basic character-based communication processing. Two functions, for start and wait processing, are used in pair. Data is exchanged through the A register. It is necessary to establish the direction of communication (SCHNG2TX: master transmission, SCHNG2RX: master reception, SCHNG2TRX: master transmission and reception) before using the CSIp.

5.7.18 1-character Transmission Start Processing

Figure 5.26 shows the flowchart for 1-character transmission start processing.

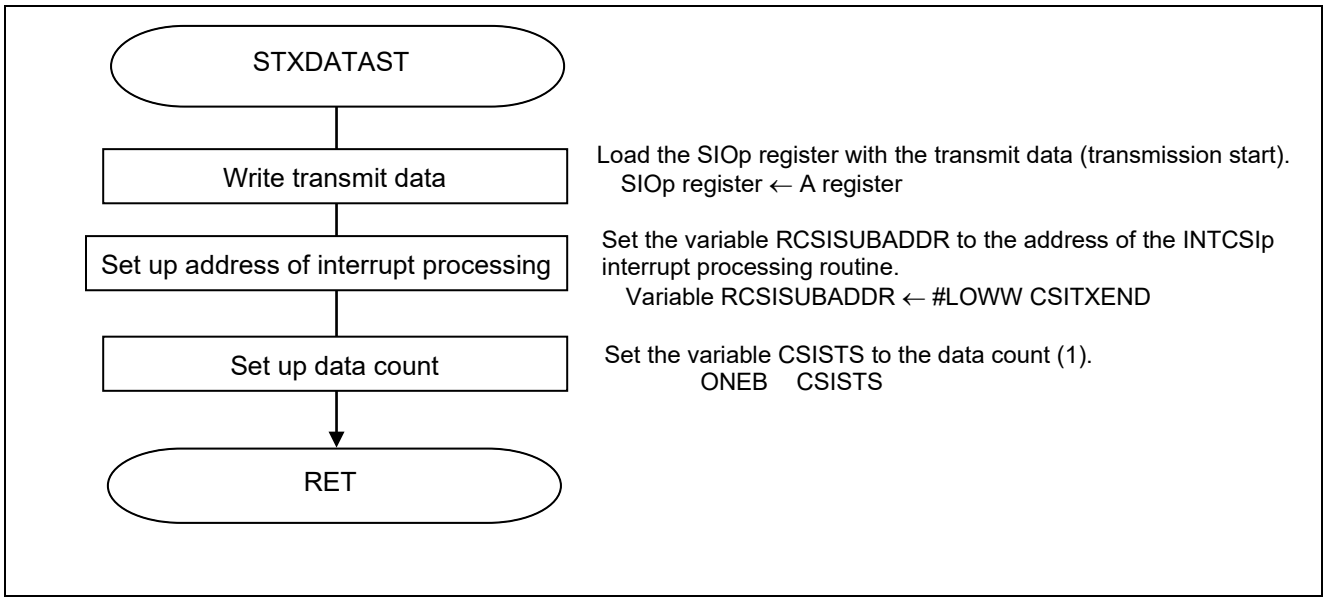


Figure 5.26 1-character Transmission Start Processing

5.7.19 1-character Transmission End Wait Processing

Figure 5.27 shows the flowchart for 1-character transmission end wait processing.

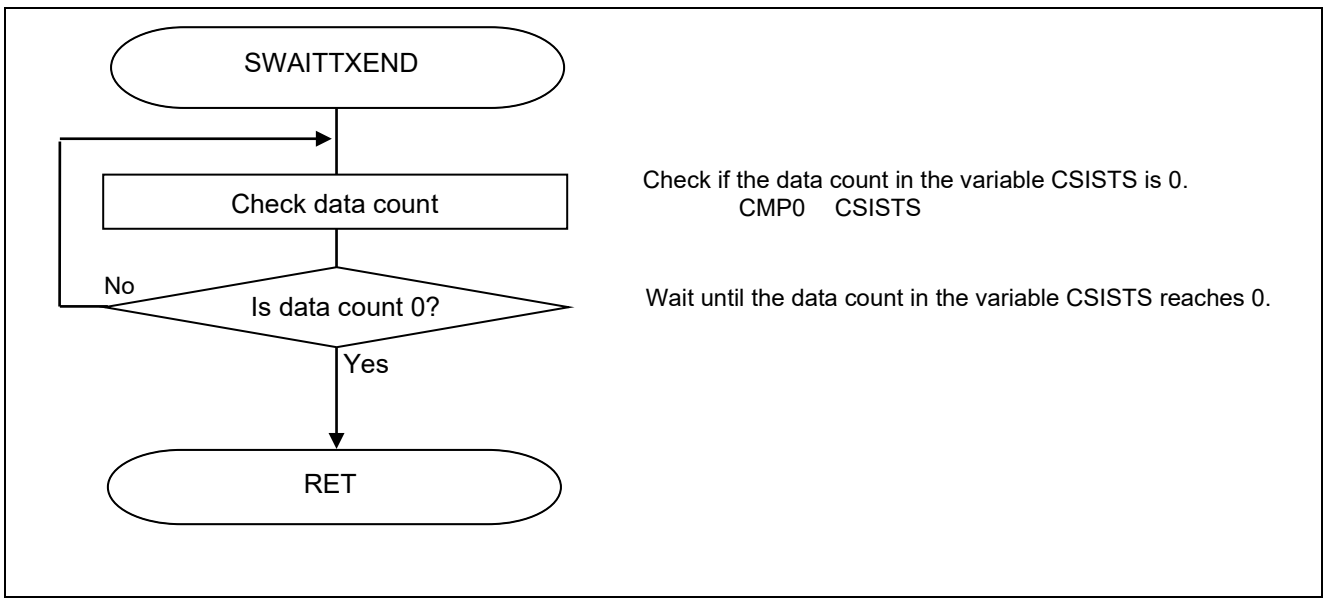


Figure 5.27 1-character Transmission End Wait Processing

5.7.20 1-character Reception Processing

Figure 5.28 shows the flowchart for 1-character reception processing.

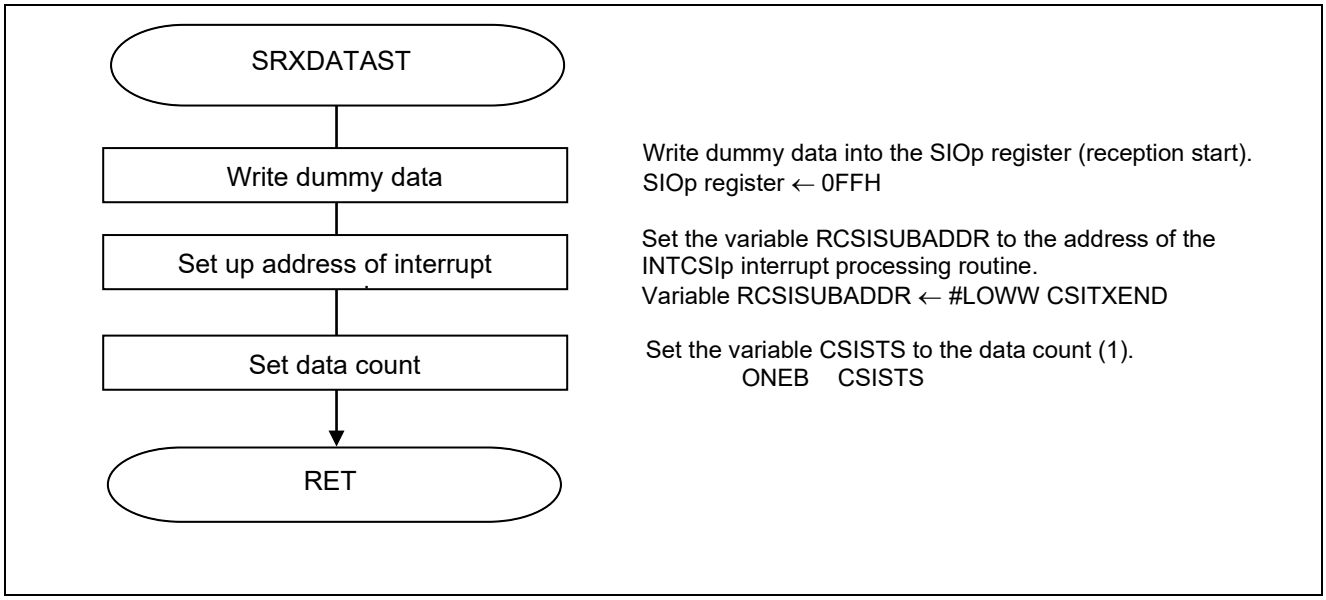


Figure 5.28 1-character Reception Processing

5.7.21 1-character Reception End Wait Processing

Figure 5.29 shows the flowchart for 1-character reception end wait processing.

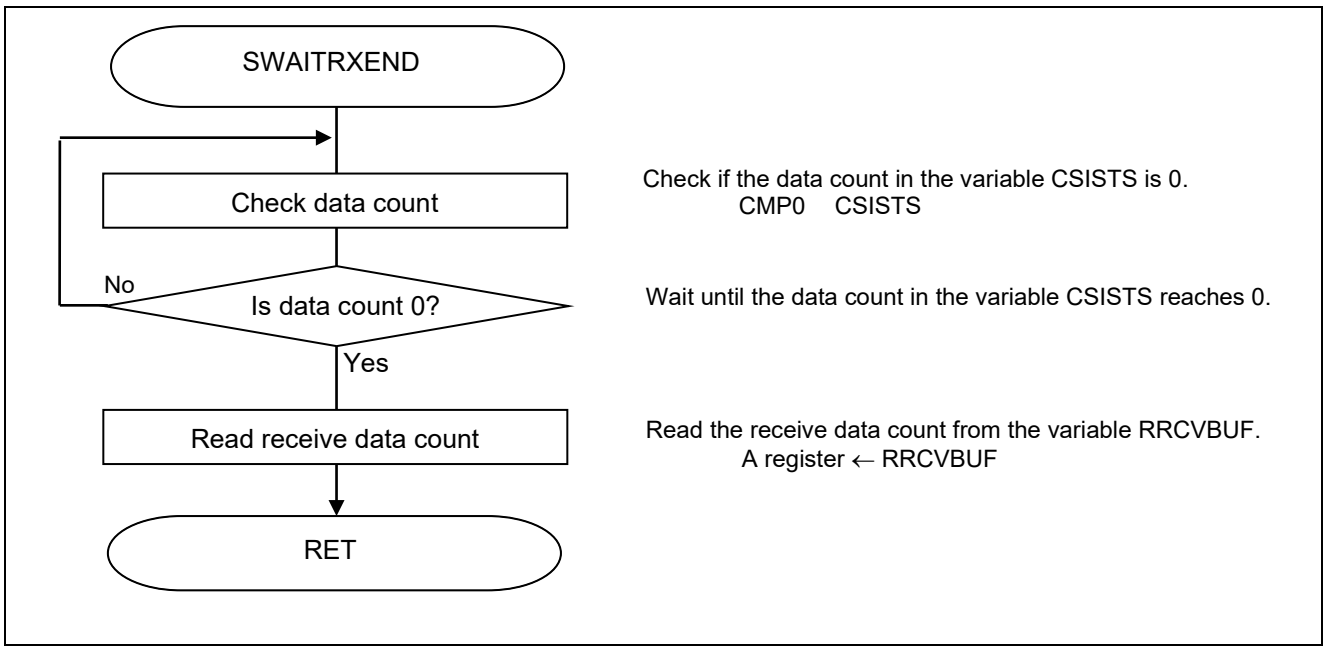


Figure 5.29 1-character Reception End Wait Processing

5.7.22 1-character Transfer State Check Processing

Figure 5.30 shows the flowchart for 1-character transfer state check processing.

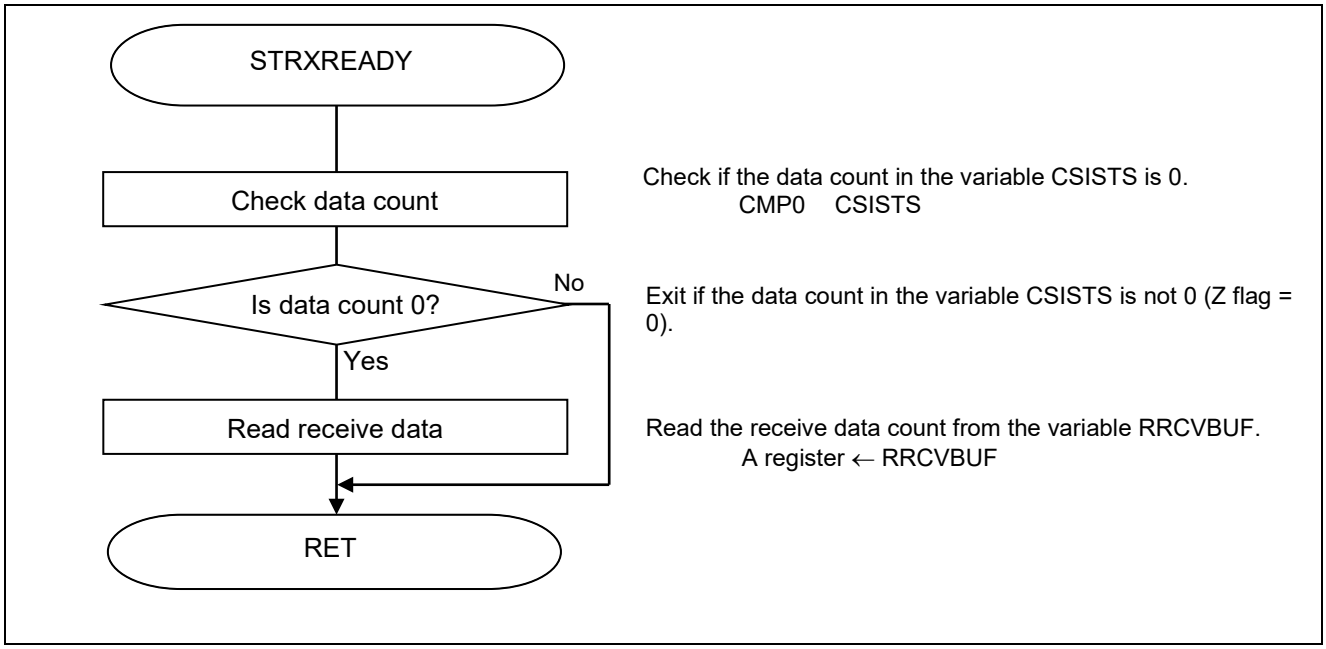


Figure 5.30 1-character Transfer State Check Processing

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 = Number of 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)

5.7.23 Continuous Transmission Start Processing

Figure 5.31 shows the flowchart for continuous transmission start processing.

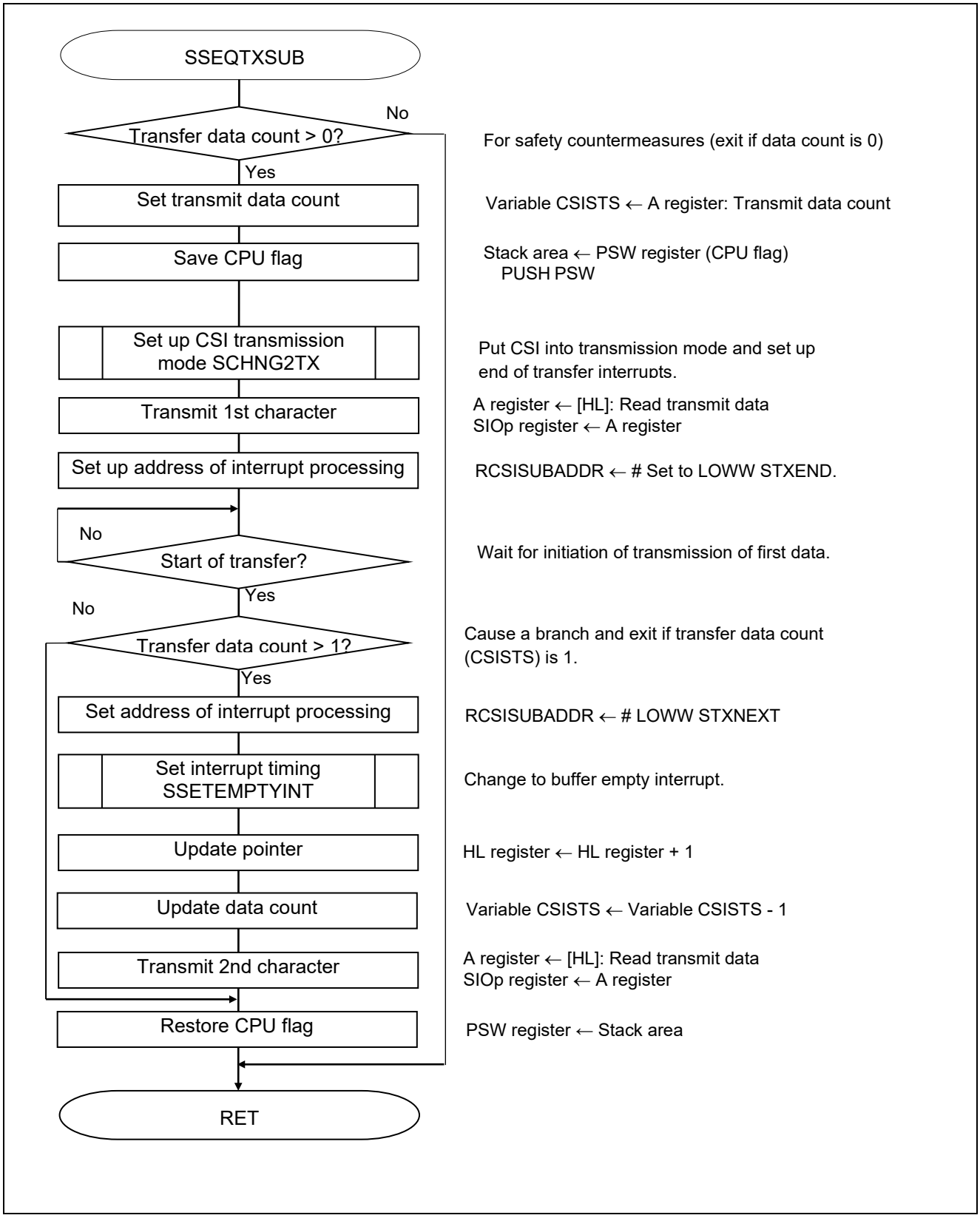


Figure 5.31 Continuous Transmission Start Processing

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

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

Remark: n : channel number (n=0, 1)
Note: SSR01 register only
Caution: For details on the register setup procedures, refer to RL78/G10 User's Manual: Hardware.

5.7.24 Continuous Reception Startup Processing

Figure 5.32 shows the flowchart for continuous reception startup processing.

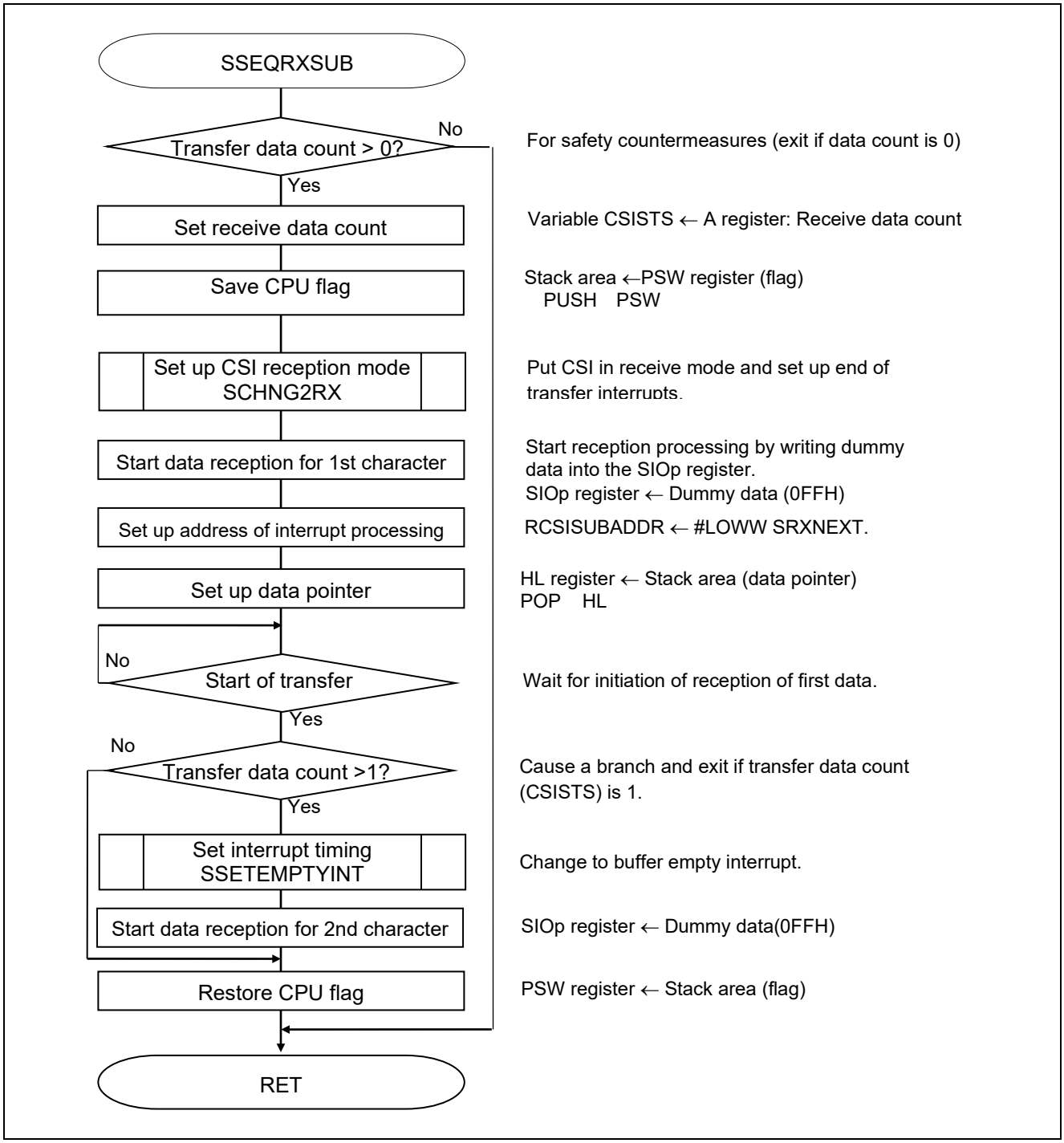


Figure 5.32 Continuous Reception Startup Processing

5.7.25 Continuous Transmission/Reception Startup Processing

Figures 5.33 and 5.34 show the flowcharts for continuous transmission/reception startup processing.

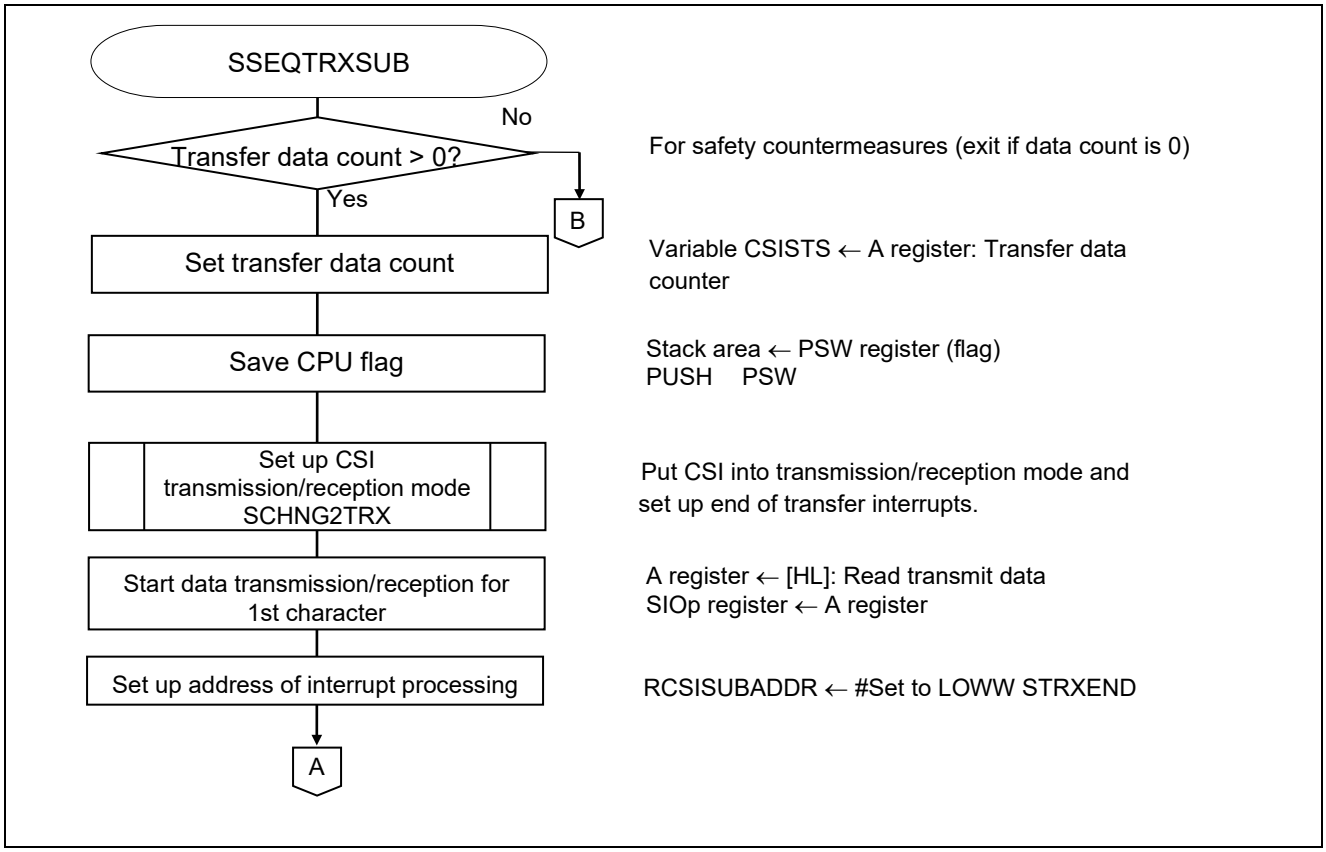


Figure 5.33 Continuous Transmission/Reception Startup Processing (1/2)

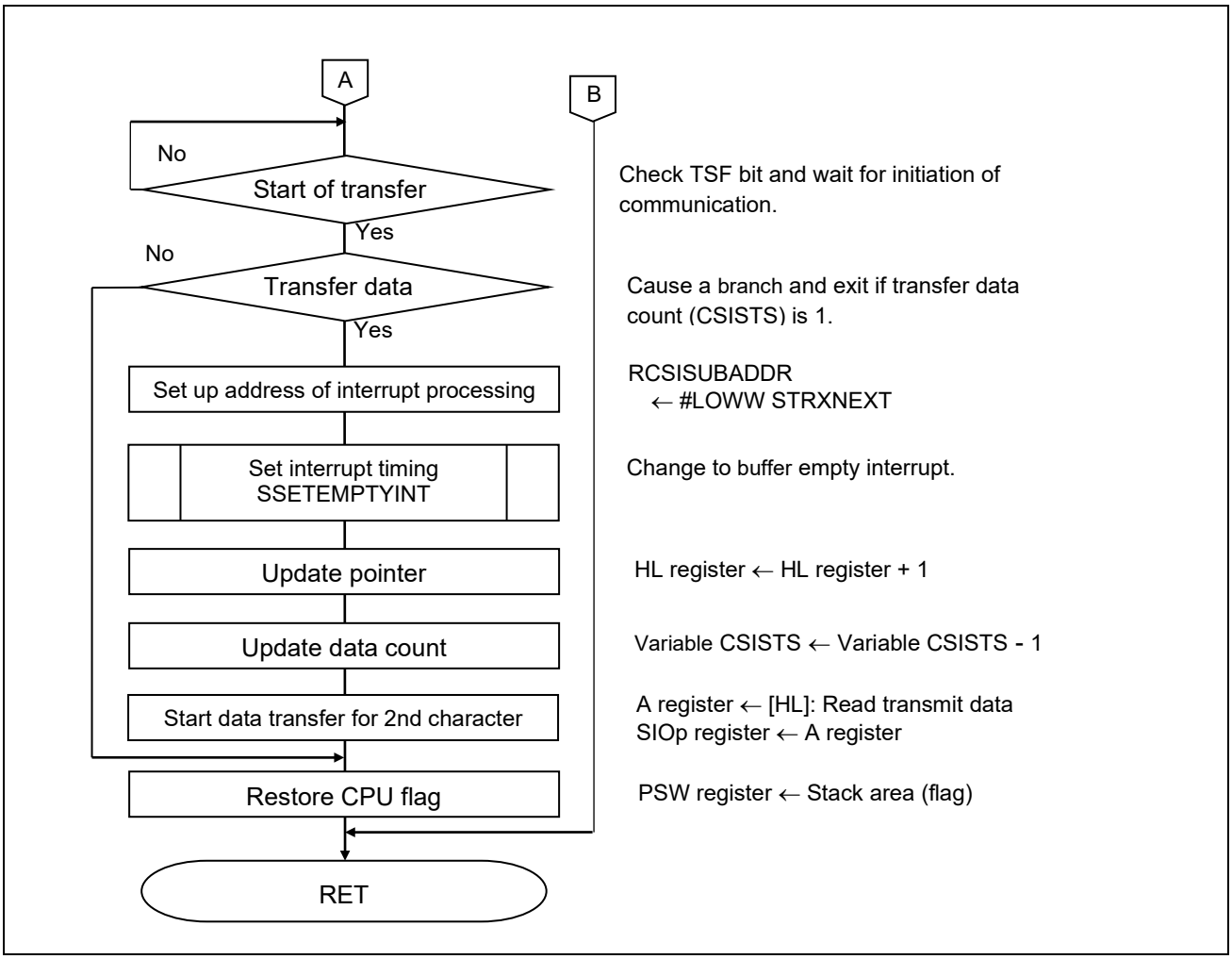


Figure 5.34 Continuous Transmission/Reception Startup Processing (2/2)

5.7.26 Continuous Transfer End Wait Processing

Figure 5.35 shows the flowchart for continuous transfer end wait processing.

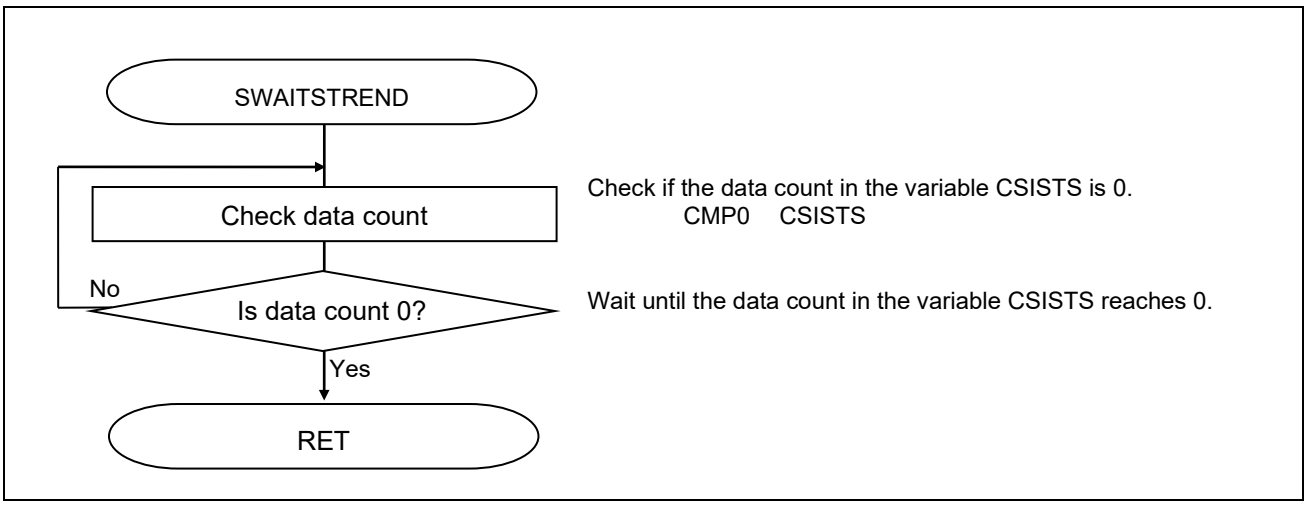


Figure 5.35 Continuous Transfer End Wait Processing

5.7.27 Transfer End Interrupt Setup Processing

Figure 5.36 shows the flowchart for transfer end interrupt setup processing.

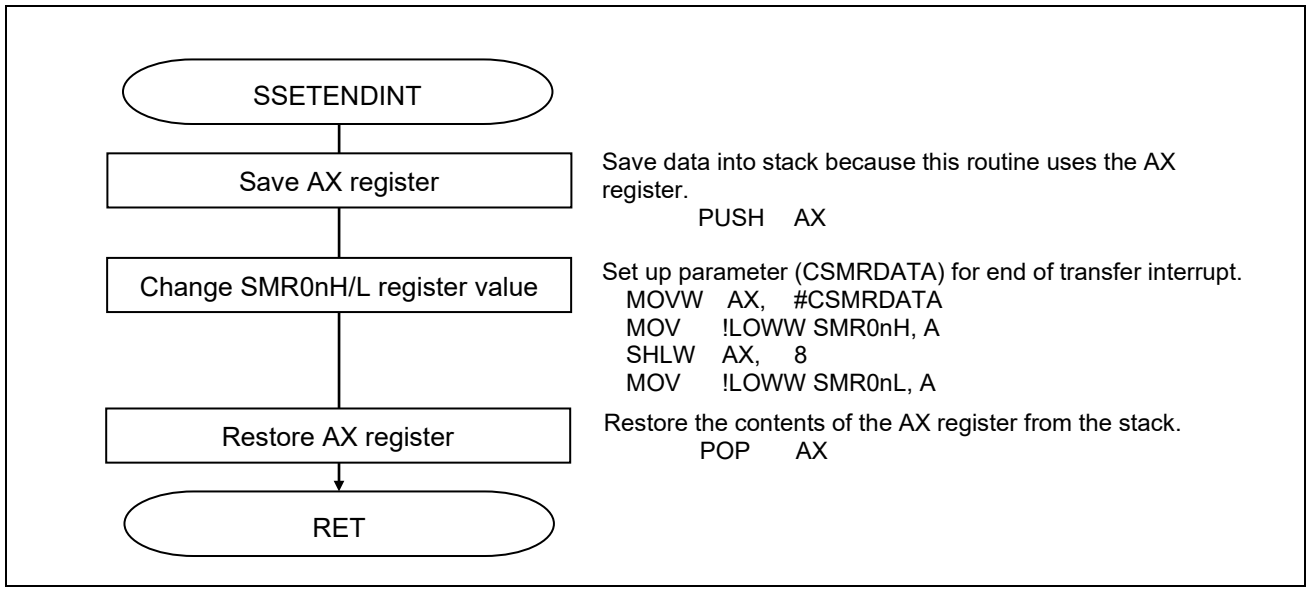


Figure 5.36 Transfer End Interrupt Setup Processing

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 (SMR0nH)

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 only

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

5.7.28 Buffer Empty Interrupt Setup Processing

Figure 5.37 shows the flowchart for buffer empty interrupt setup processing.

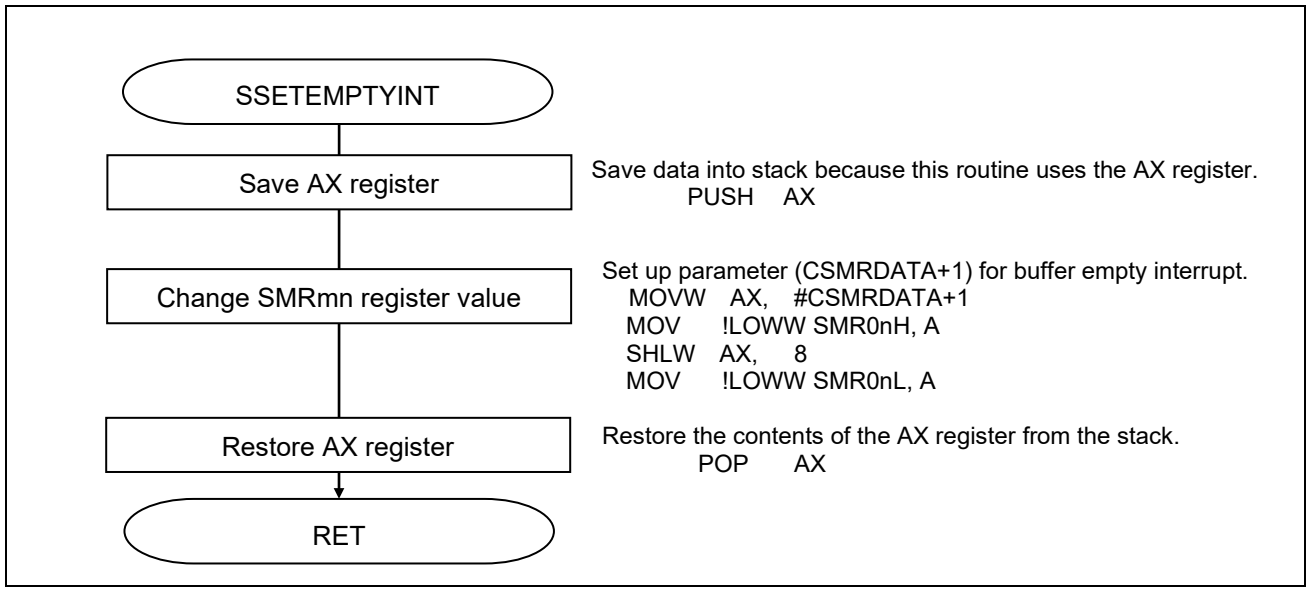


Figure 5.37 Buffer Empty Interrupt Setup Processing

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 (SMR0nH)

MDmn0	Channel n Interrupt Source Select
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.

5.7.29 Transmission Mode Setup Processing

Figure 5.38 shows the flowchart for transmission mode setup processing.

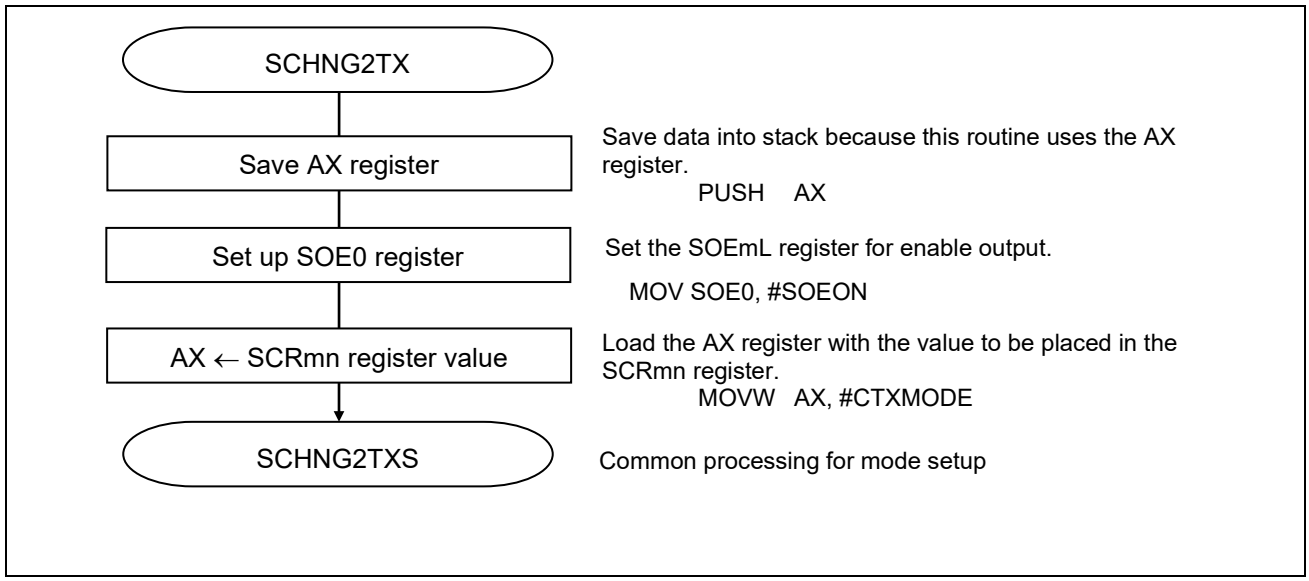


Figure 5.38 Transmission Mode Setup Processing

5.7.30 Reception Mode Setup Processing

Figure 5.39 shows the flowchart for reception mode setup processing.

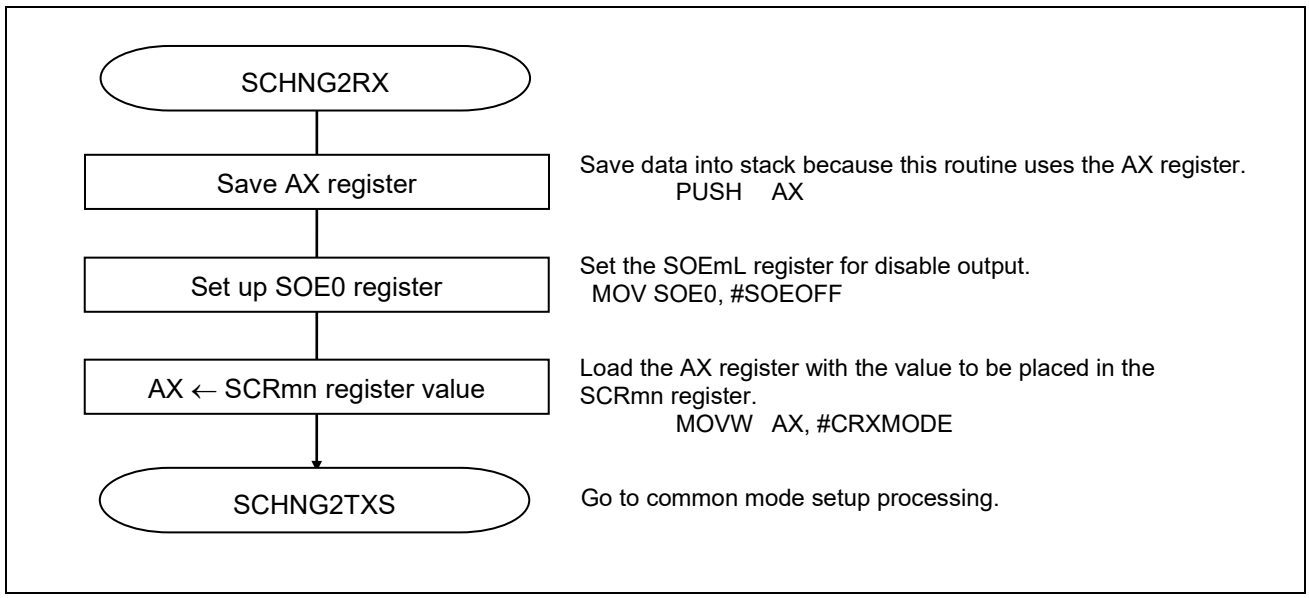


Figure 5.39 Reception Mode Setup Processing

5.7.31 Transmission/Reception Mode Setup Processing

Figure 5.40 shows the flowchart for transmission/reception mode setup processing.

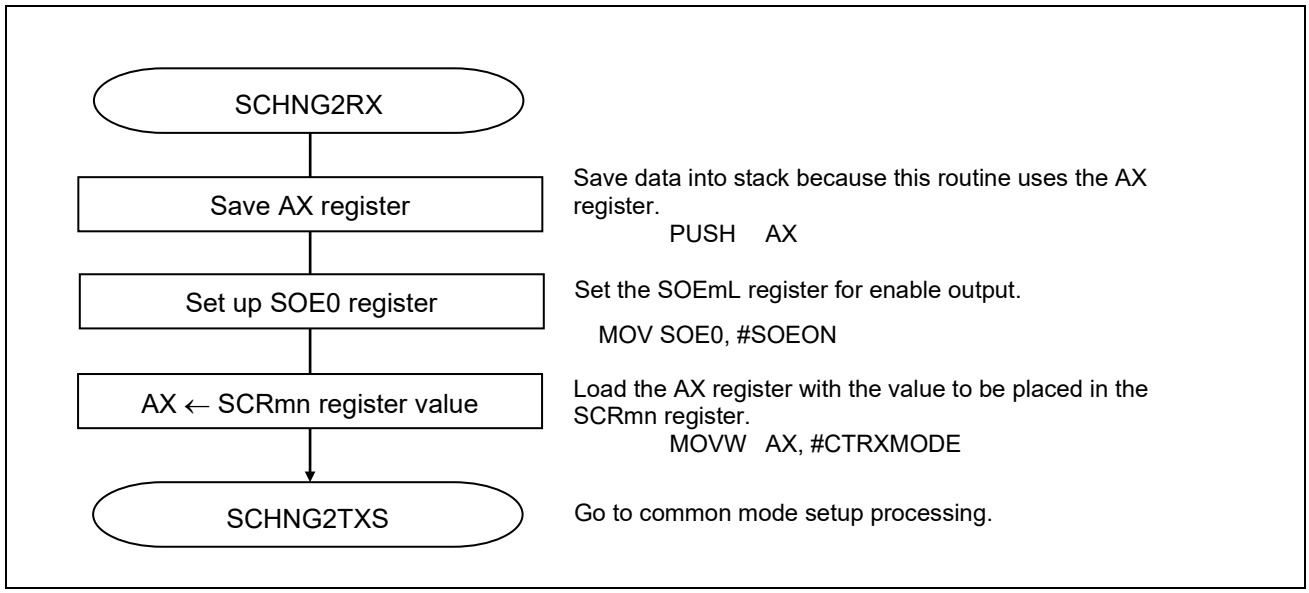


Figure 5.40 Transmission/Reception Mode Setup Processing

5.7.32 Common Processing for Mode Setup

Figure 5.41 shows the flowchart for common processing for mode setup.

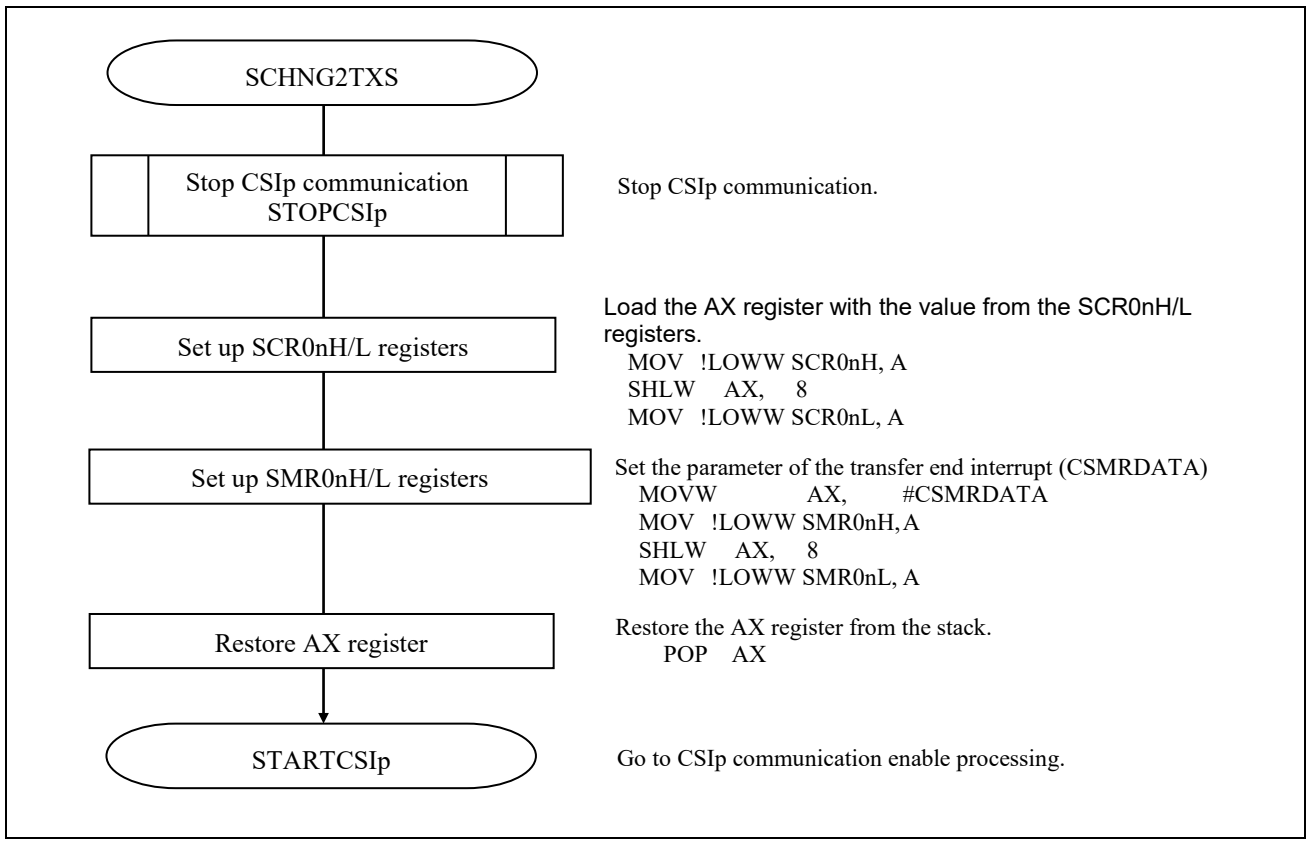


Figure 5.41 Common Processing for Mode Setup

Interrupt setting

- Interrupt mask flag register (MK0H)

Interrupt mask setting

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 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	**0/1**	**0/1**

Bit n

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

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	Disable communication
0	**1**	**Reception only**
1	**0**	**Transmission only**
1	**1**	**Transmission/reception**

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

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

Symbol: ST0

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

Bit n

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

Setting up the channel operating mode

- Serial mode register 0n (SMR0nH, SMR0nL)

Interrupt source and end of transmit 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)

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

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

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

5.7.33 CSIp Communication Enable Processing

Figure 5.41 shows the flowchart for CSIp communication enable processing.

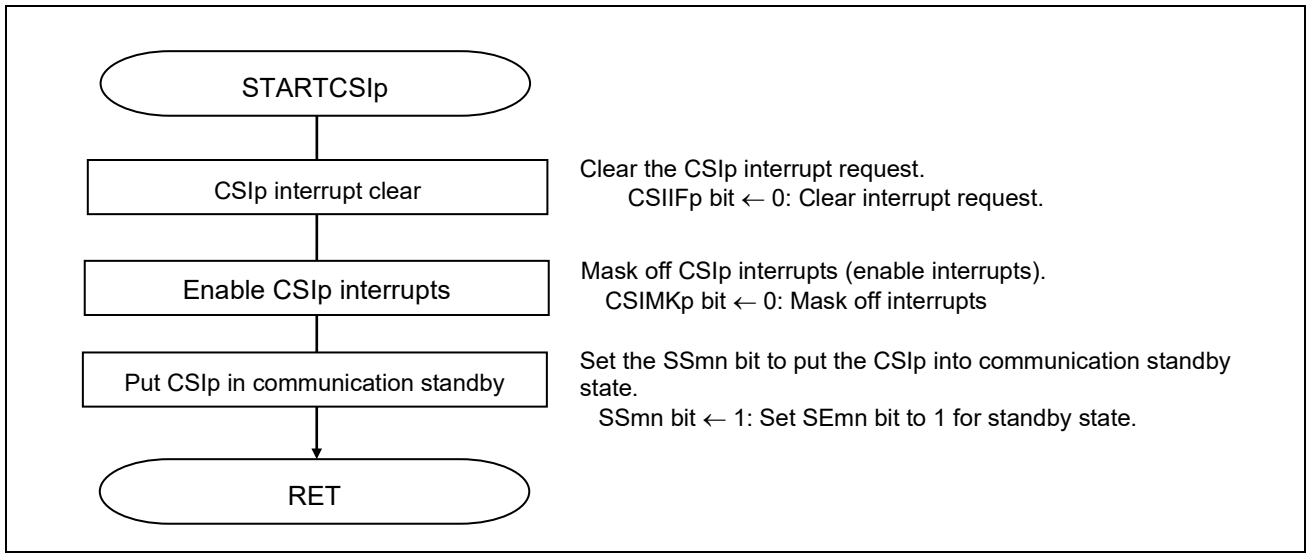


Figure 5.41 CSIp Communication Enable Processing

Transiting to communication standby state

- Serial channel startup register 0 (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	**0/1**	**0/1**

Bits 3 to 0

SSmn	Operation start trigger of channel n
0	No trigger operation
1	**Sets the SEmn bit to 1 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	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	**0**	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.

5.7.34 CSIp Communication Termination Processing

Figure 5.42 shows the flowchart for CSIp communication termination processing.

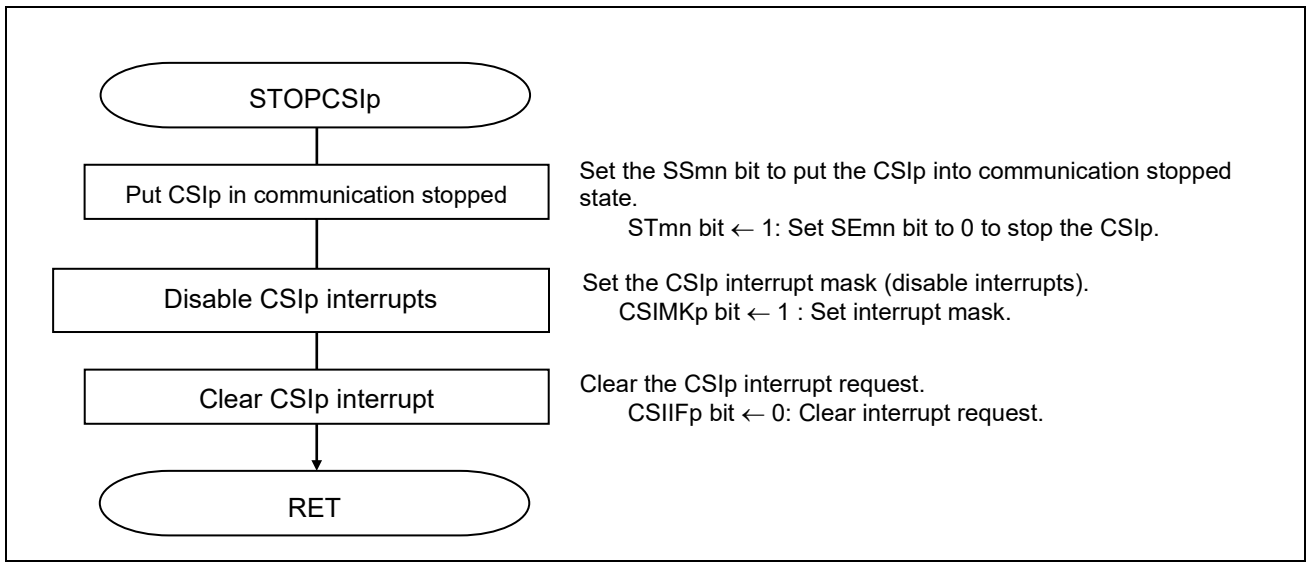


Figure 5.42 CSIp Communication Termination Processing

Transiting to communication stopped state

- 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	**0/1**	**0/1**

Bits n

ST0n	Operation start trigger of channel n
0	No trigger operation
1	**Sets the SEmn 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)
  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	**0**	x	x	x

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.

5.7.35 CSIp Interrupt Startup Processing

Figure 5.43 shows the flowchart for CSIp interrupt startup processing.

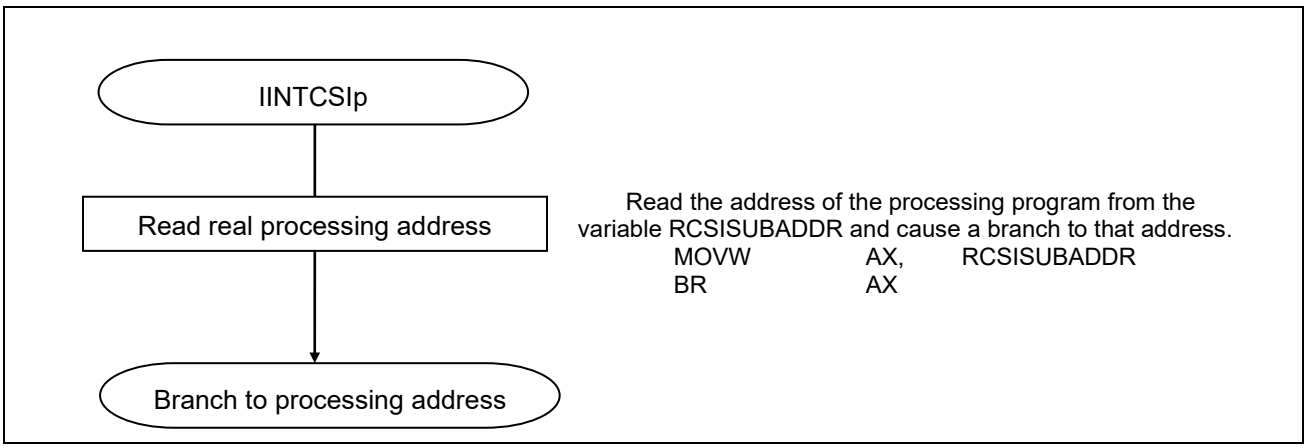


Figure 5.43 CSIp Interrupt Startup Processing

5.7.36 1-character Transfer End Interrupt Processing

Figure 5.44 shows the flowchart for 1-character transfer end interrupt processing.

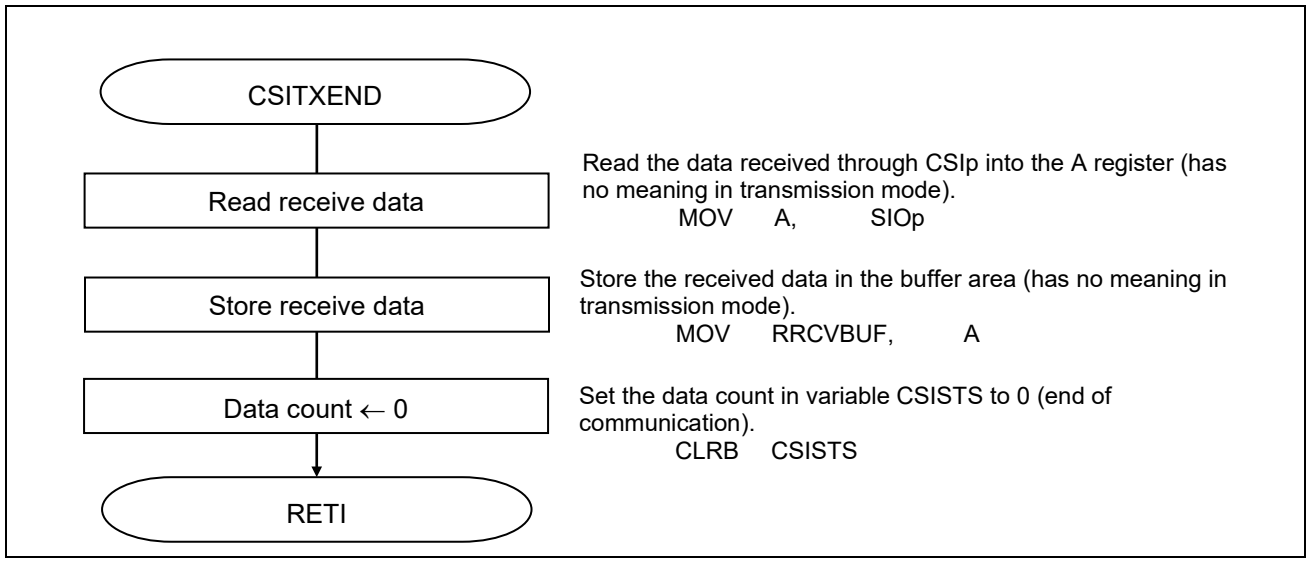


Figure 5.44 1-character Transfer End Interrupt Processing

5.7.37 1-character Transfer End Interrupt Processing in Continuous Reception Mode

Figure 5.45 shows the flowchart for 1-character transfer end interrupt processing in continuous reception mode.

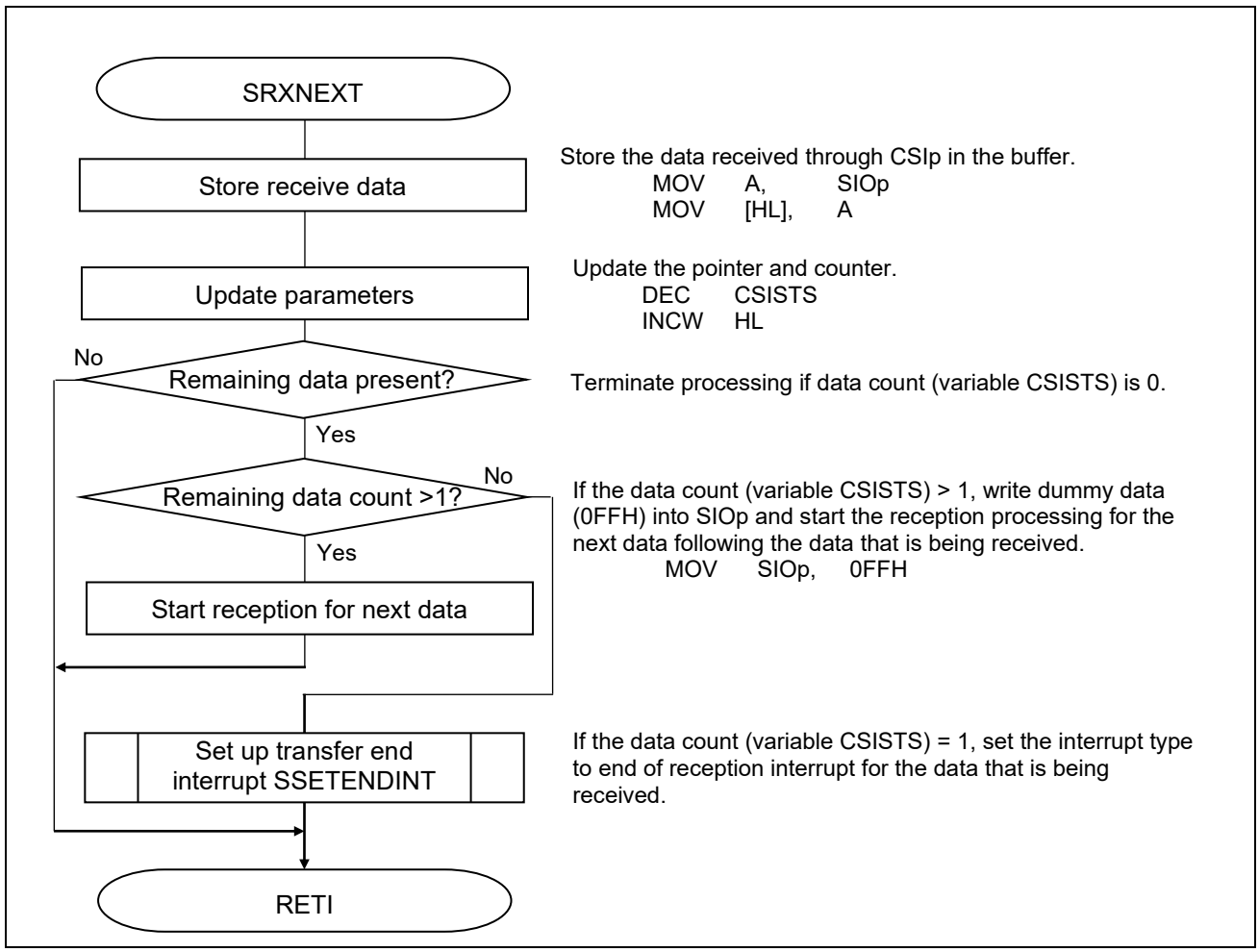


Figure 5.45 1-character Transfer End Interrupt Processing (Continuous Reception Mode)

5.7.38 Buffer Empty Interrupt Processing in Continuous Transmission Mode

Figure 5.46 shows the flowchart for buffer empty interrupt processing in continuous transmission mode.

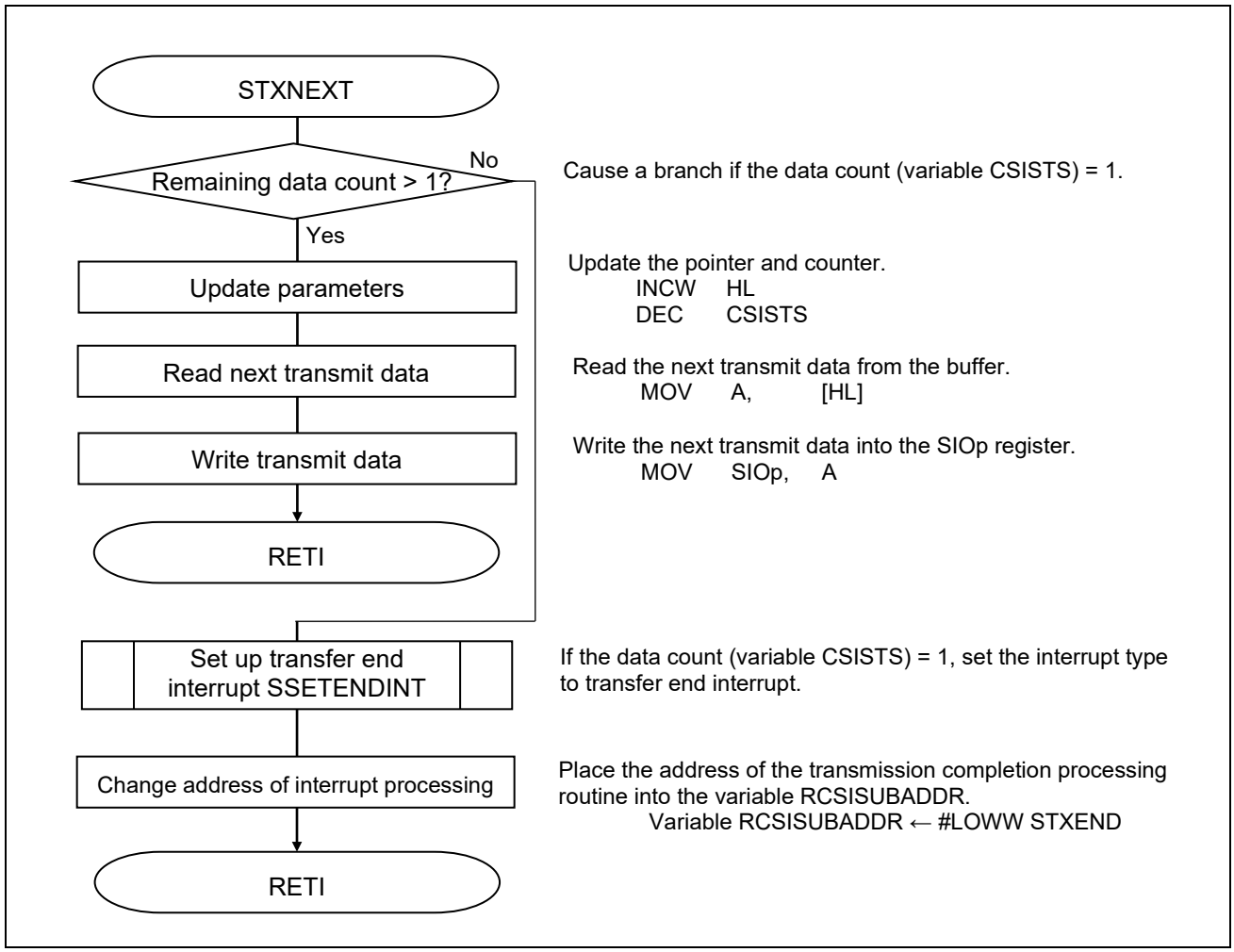


Figure 5.46 Buffer Empty Interrupt Processing (Continuous Transmission Mode)

5.7.39 Transmission End Interrupt Processing in Continuous Transmission Mode

Figure 5.47 shows the flowchart for transmission end interrupt processing in continuous transmission mode.

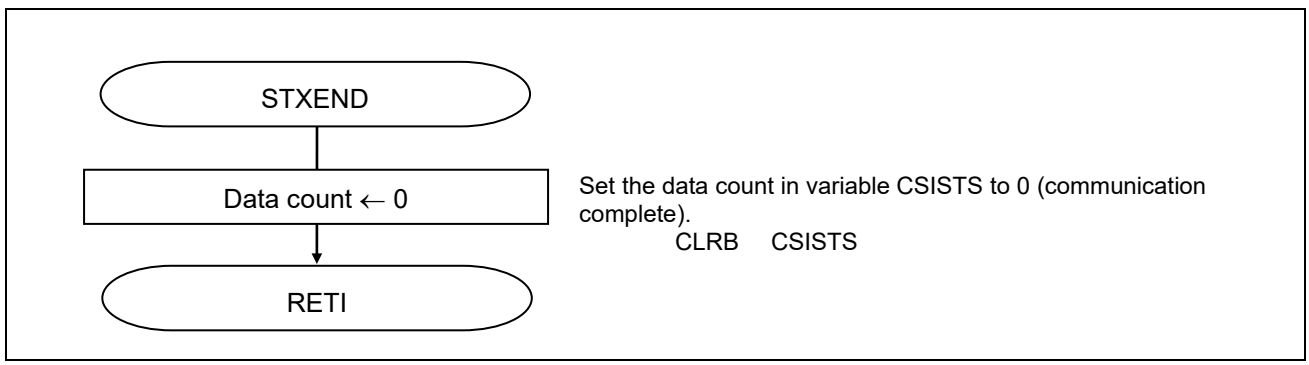


Figure 5.47 Transmission End Interrupt Processing (Continuous Transmission Mode)

5.7.40 Buffer Empty Interrupt Processing in Continuous Transmission Mode

Figure 5.48 shows the flowchart for buffer empty interrupt processing in continuous transmission mode.

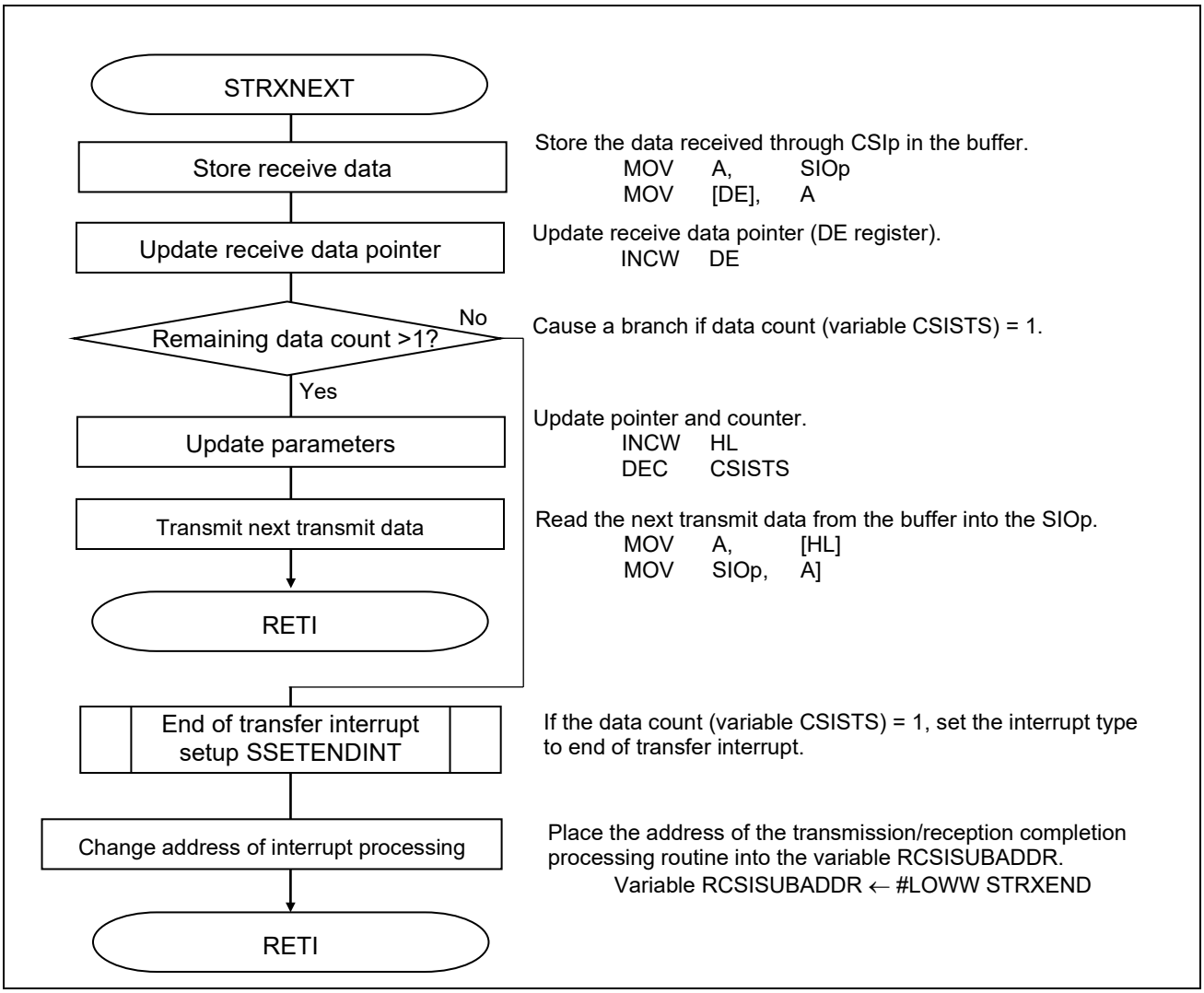


Figure 5.48 Buffer Empty Interrupt Processing (Continuous Transmission Mode)

5.7.41 Transfer End Interrupt Processing in Continuous Transmission/Reception Mode

Figure 5.49 shows the flowchart for transfer end interrupt processing in continuous transmission/reception mode.

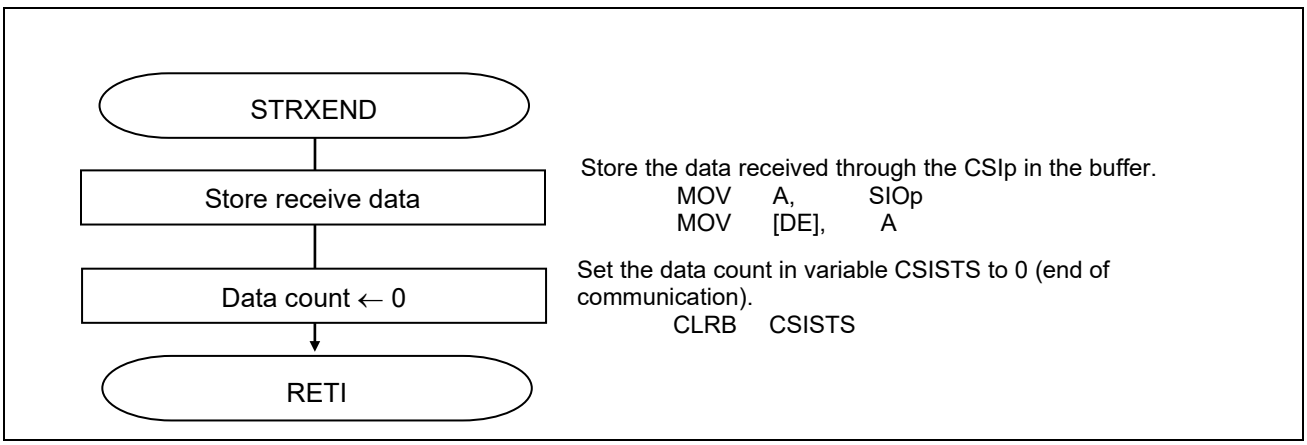


Figure 5.49 Transfer End Interrupt Processing (Continuous Transmission/Reception Mode)

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 (20 MHz).
- CSI communication speed (BAUDRATE) in kbps: The initial value is 1000 (1 Mbps).
- 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 20 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 Slave Communication) (R01AN1461E)

(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	RL78/G10 Serial Array Unit (CSI Master Communication) CC-RL

Rev.	Date	Description	
		Page	Summary
1.00	Nov. 27, 2015	—	First edition issued
1.10	June. 24. 2022	8	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.