RL78/G12

Serial Array Unit (CSI Master Communication)

R01AN1369EJ0100
Rev. 1.00
Mar.01, 2013

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/G12 units running in CSI slave mode are used as slave devices, it performs handshake processing using the BUSY signal.

Target Device

RL78/G12

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.

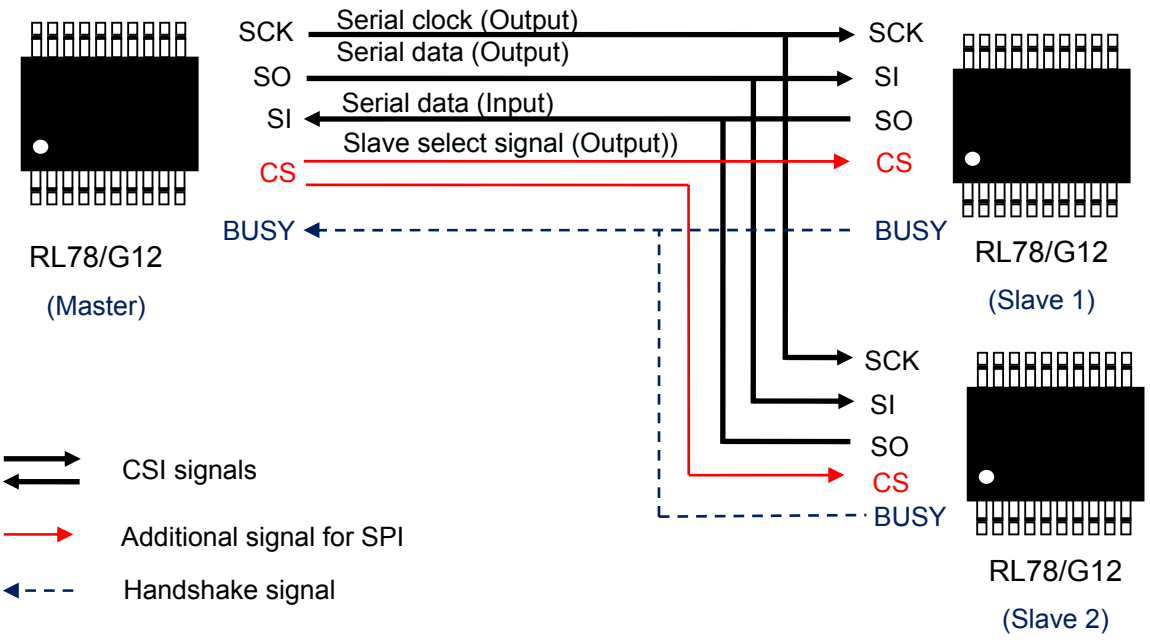
Contents

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/G12 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.

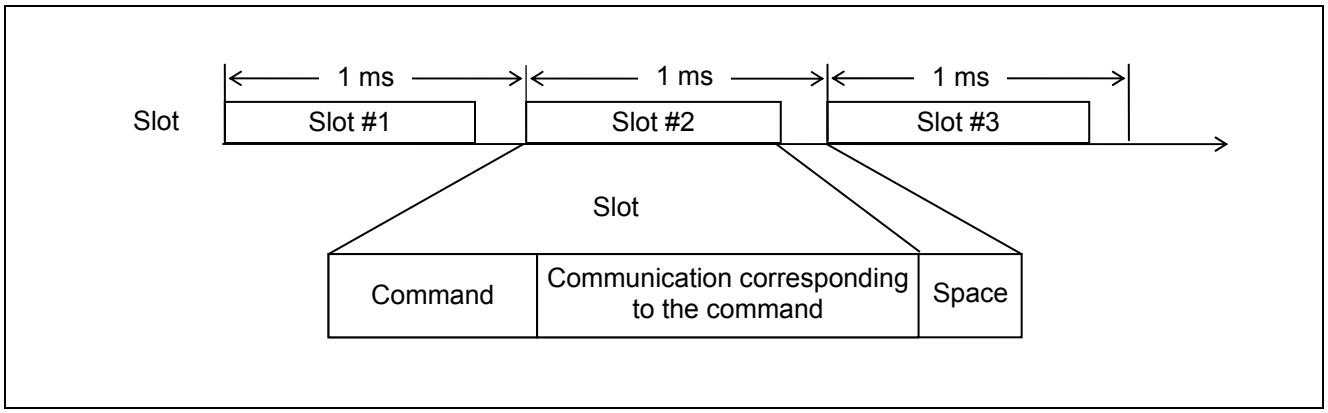


Figure 1.2 Outline of Slots

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 basic communication processing consists of subroutines that run based on interrupts. The CSI channel to be used can be changed easily by editing a header file.

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: 00/01/11/20
Port	P23 (CS1 signal output), P22 (CS2 signal output), P21 (BUSY signal input)

20/24-pin products: m = 0, 30-pin products: m = 0/1

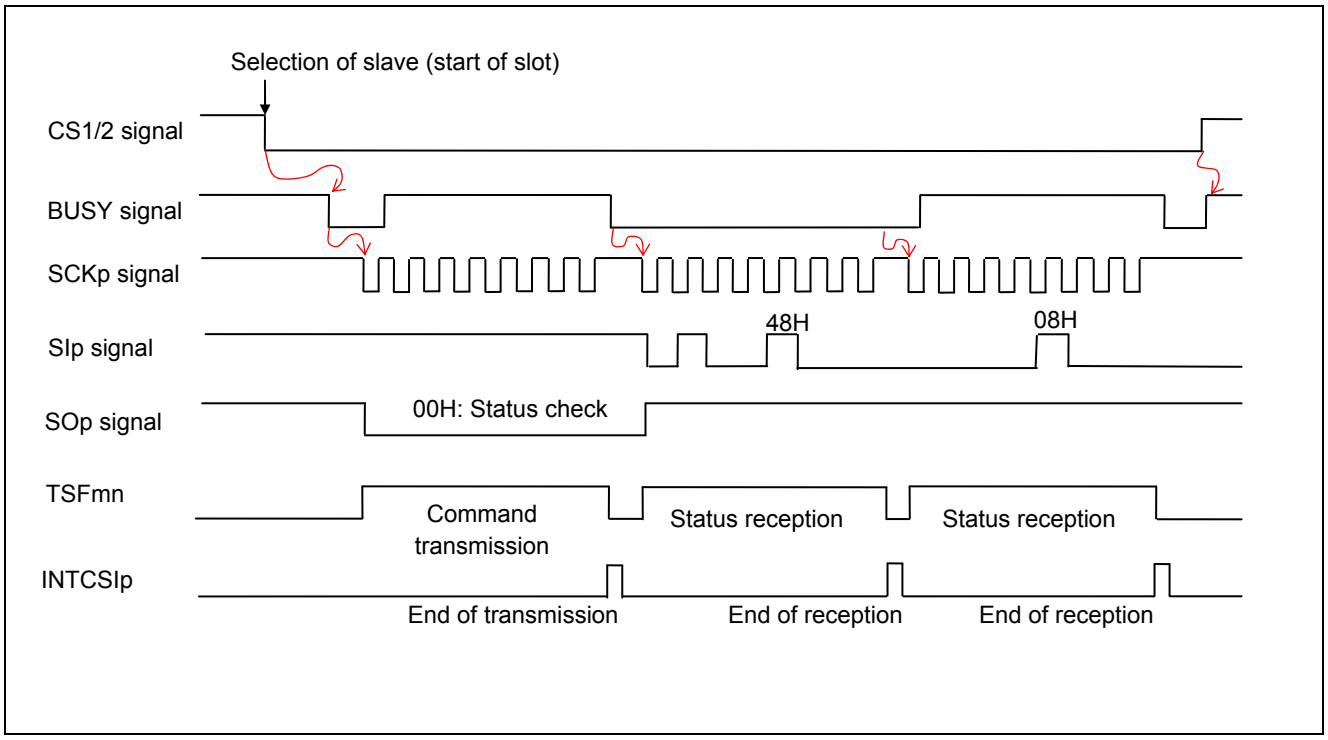


Figure 1.3 Timing Chart of Status Check Command

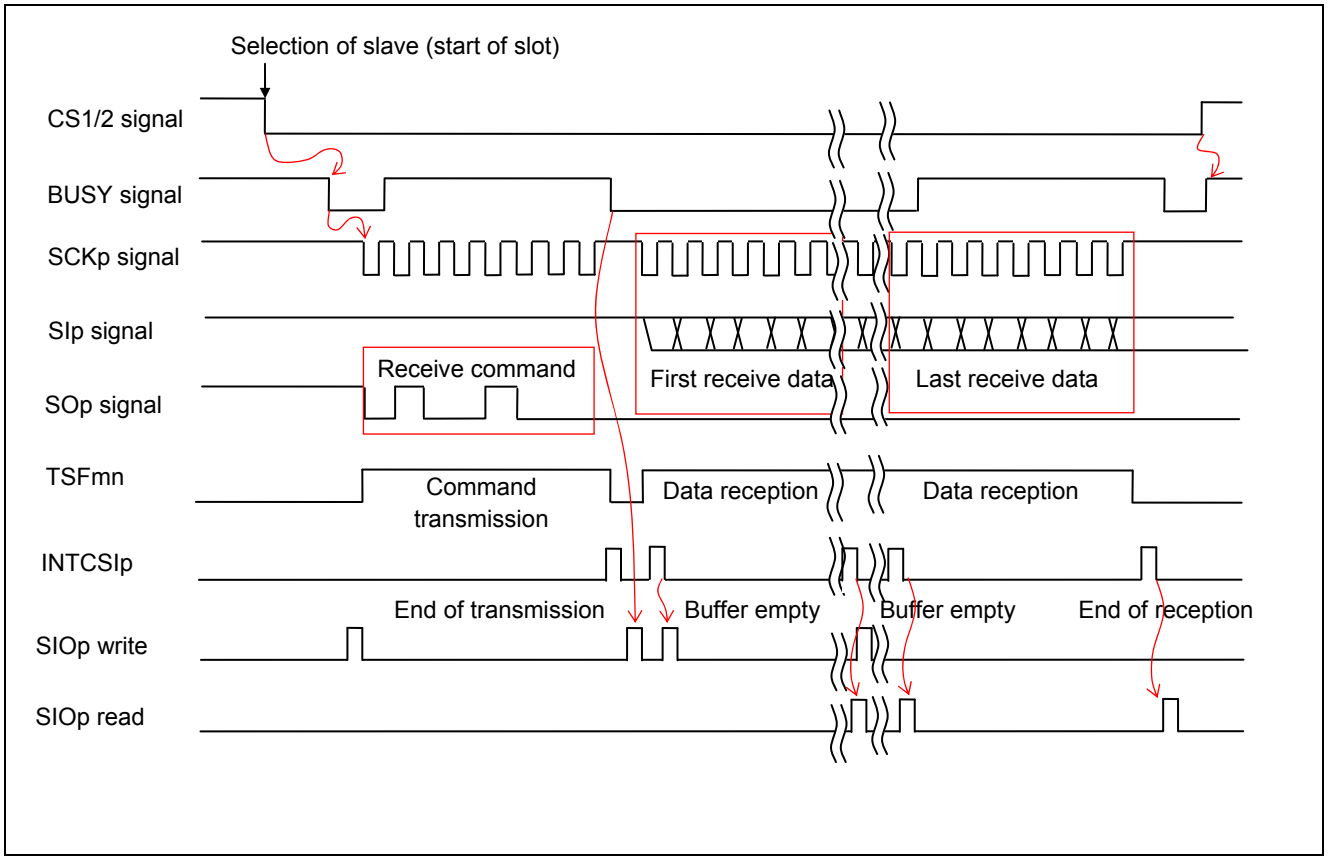


Figure 1.4 Timing Chart of Receive Command

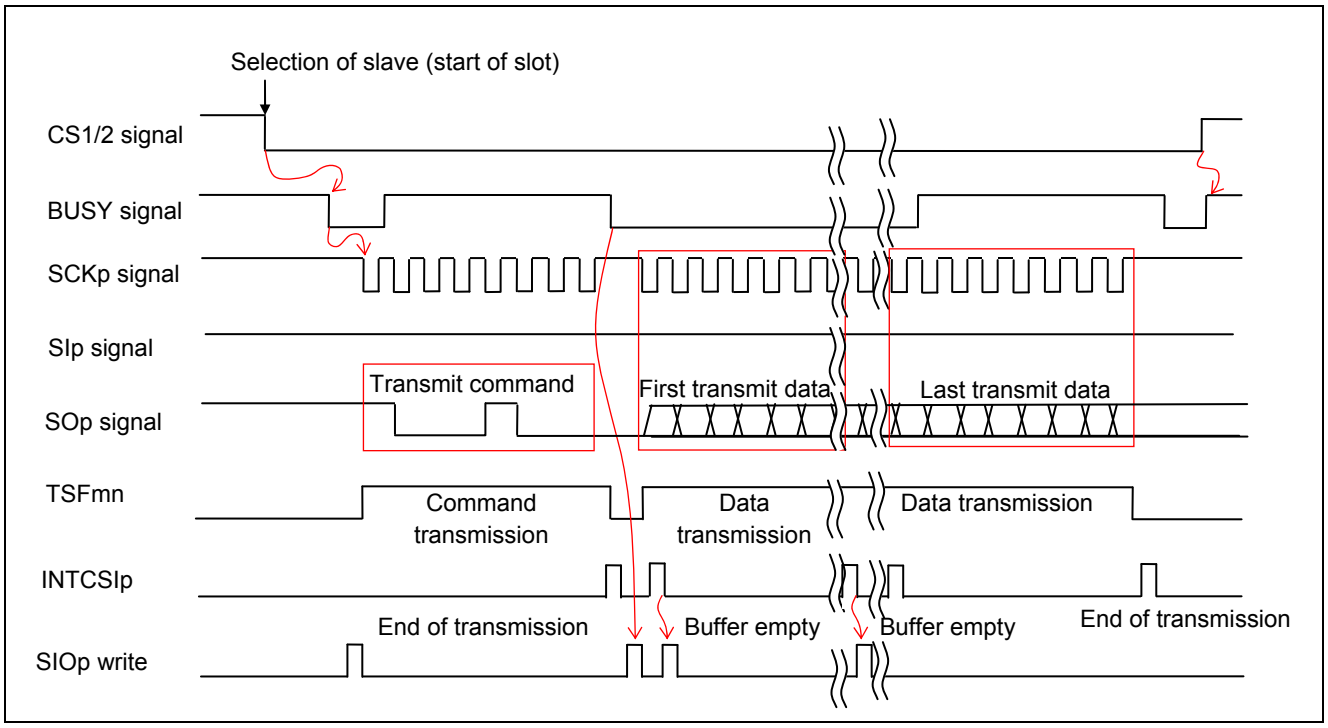


Figure 1.5 Timing Chart of Transmit Command

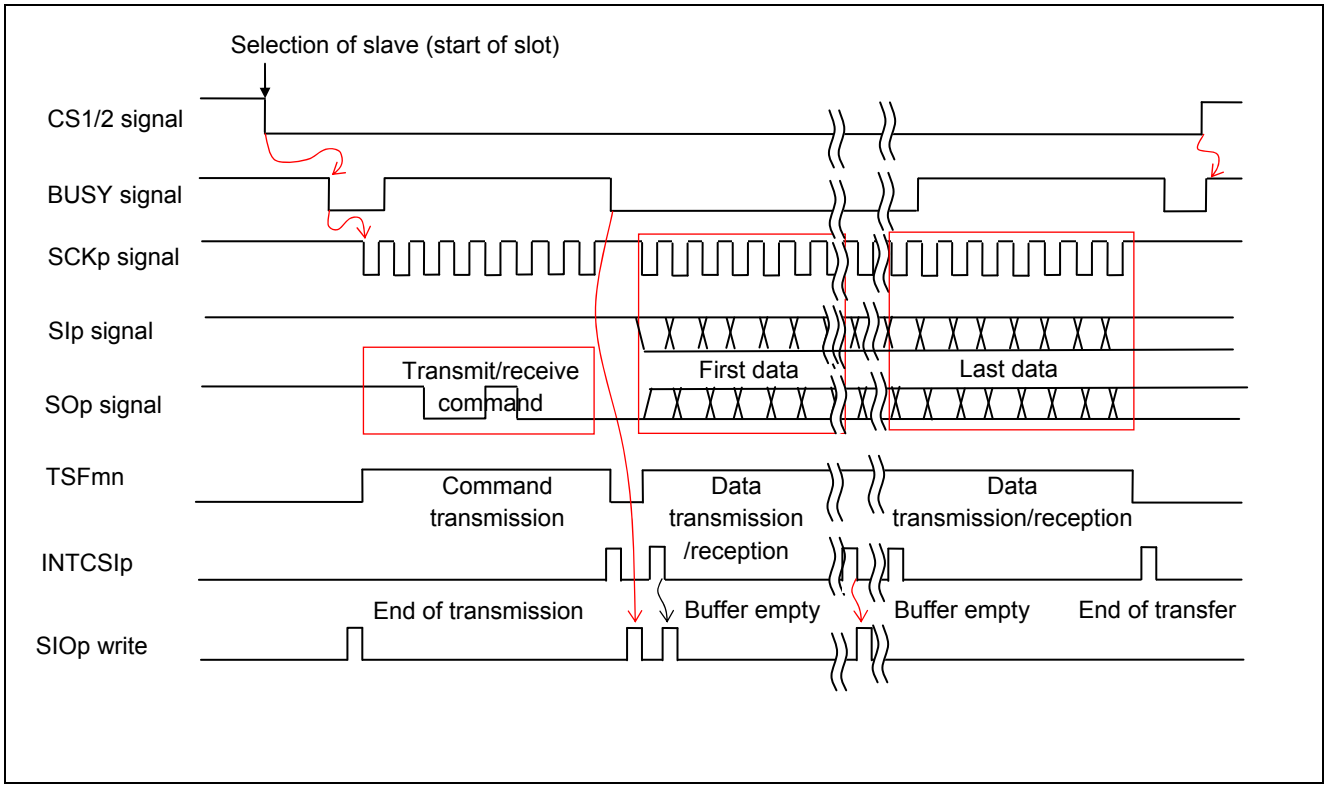


Figure 1.6 Timing Chart of Transmit/Receive Command

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/G12 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.

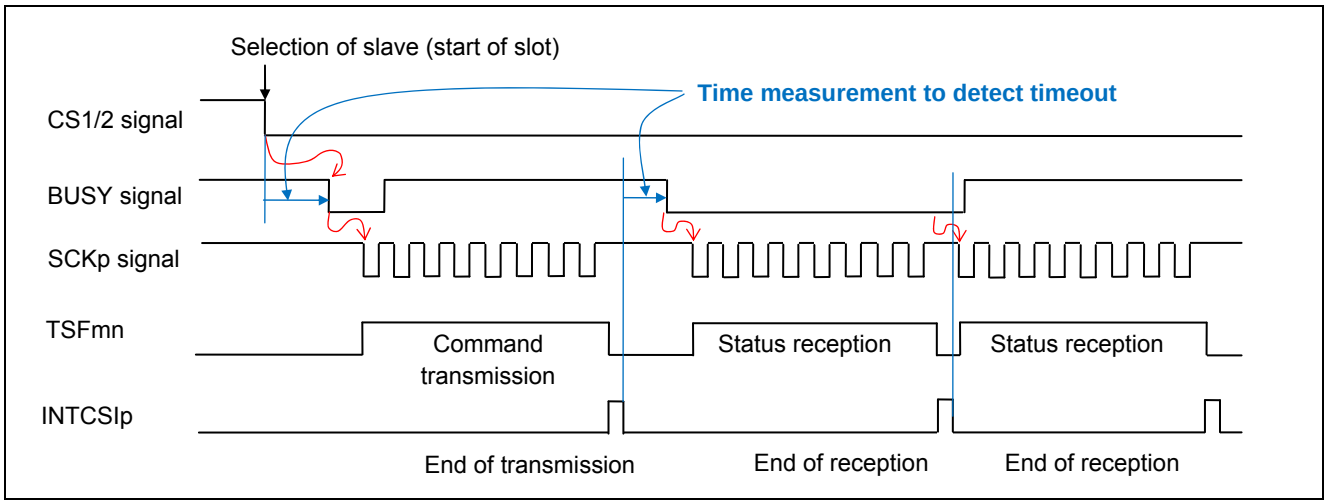


Figure 1.7 Handshaking Example

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/G12 (R5F1026A)
Operating frequency	• High-speed on-chip oscillator (HOCO) clock: 24 MHz • CPU/peripheral hardware clock: 24 MHz
Operating voltage	5.0 V (Operation is possible over a voltage range of 2.9 V to 5.5 V.) LVD operation (V_{LVI}): Reset mode which uses 2.81 V (2.76 V to 2.87 V)
Integrated development environment	CubeSuite+ V1.02.00 from Renesas Electronics Corp.
Assembler	RA78K0R V1.60 from Renesas Electronics Corp.
Board to be used	RL78/G12 target board (QB-R5F1026A-TB)

3. Related Application Notes

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

RL78/G12 Initialization (R01AN1030E) Application Note
RL78/G12 Serial Array Unit (CSI Slave Communication) (R01AN1370E) Application Note
RL78 Family CubeSuite+ Startup Guide (R01AN1232E) 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.

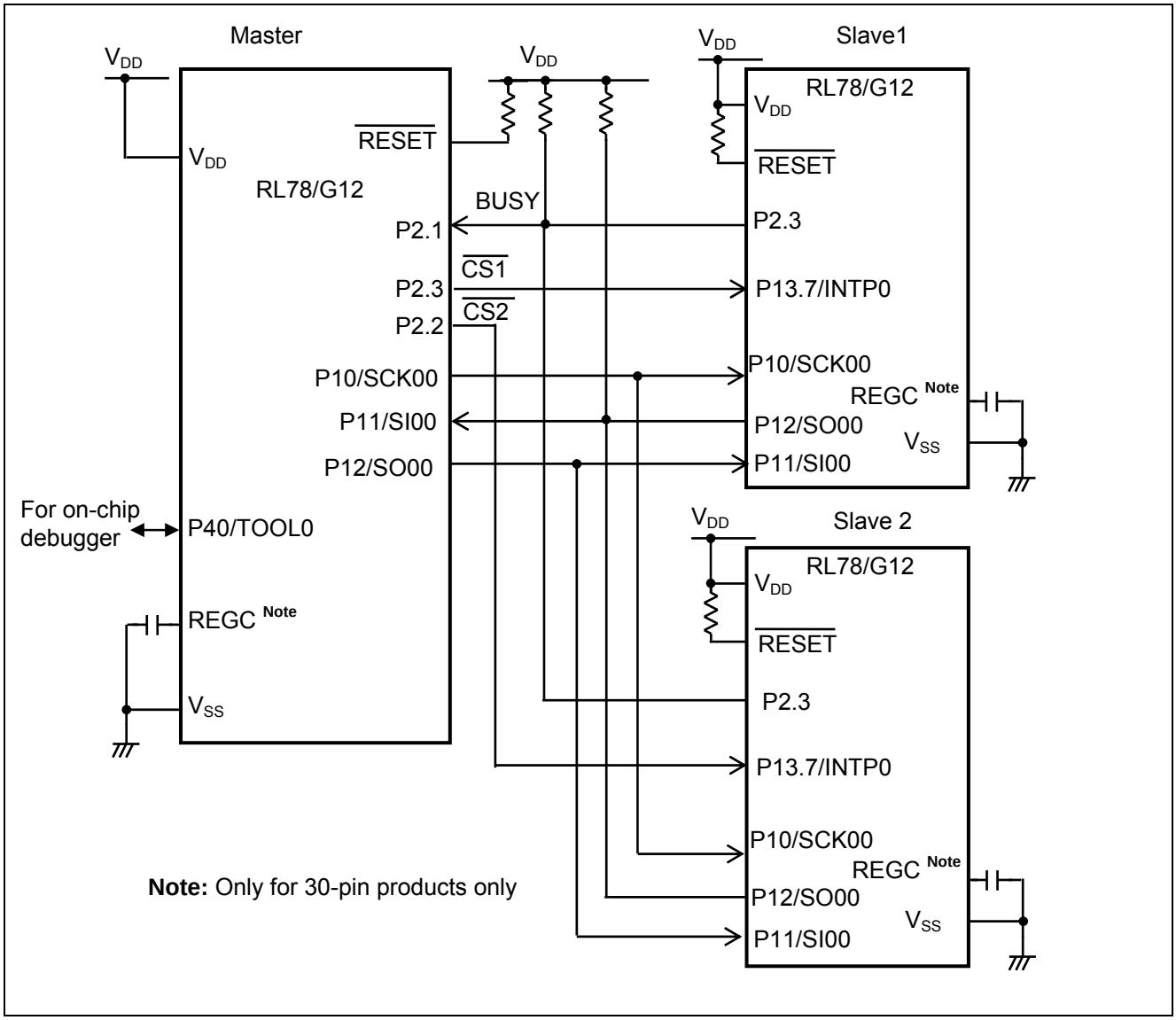


Figure 4.1 Hardware Configuration

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_{LVI}) that is specified as LVD.

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
P10/ANI16/PCLBUZ0/SCK00/SCL00 Note	Output	Serial clock output pin
P11/ANI17/SI00/RxD0/TOOLRxD/SDA00 Note	Input	Data receive pin
P12/ANI18/SO00/TxD0/TOOLTxD Note	Output	Data transmit pin
P21/ANI1/AVREFM (BUSY)	Input	BUSY signal input from the slave
P22/ANI2 (CS2)	Output	Slave 1 select signal
P23/ANI3 (CS1)	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: 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. 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	01101110B	Disables the watchdog timer. (Stops counting after the release from the reset state.)
000C1H	01111111B	LVD reset mode, 2.81 V (2.76 to 2.87 V)
000C2H	11100000B	HS mode, HOCO: 24 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	24000	RL78/G12 operating clock frequency in kHz (24 MHz)
BAUDRATE	↑	1000	Communication speed in kbps (1 Mbps)
DIVIDE	↑	CLKFREQ / BAUDRATE	Frequency division ratio necessary for attain the specified communication speed
SDRDATA	↑	(DIVIDE -1) * 200H	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
SAUmEN	↑	SAU0ENSAU0EN Note	SAU clock supply enable bit
SPSmL	↑	SPS0L Note	SAU prescaler setting register
SMRmn	↑	SMR00 Note	Channel mode setting register
SCRmn	↑	SCR00 Note	Channel communication operation setting register
SDRmn	↑	SDR00 Note	Channel serial data register
SIOp	↑	SIO00 Note	Lower 8 bits of channel serial data
SSRmnL	↑	SSR00L Note	Channel status register
SIRmnL	↑	SIR00L Note	Channel flag clear trigger register
SSmL	↑	SS0L Note	Channel start register
STmL	↑	ST0L Note	Channel stop register
TRGONn	↑	00000001B Note	Value for SSmL and STmL
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)
SOm	↑	SO0 Note	Channel output register
SOHIGH	↑	TRGONn	For setting value in channel output register
PM_CSIp	↑	PM1 Note	Port mode register
PM_SCKp	↑	PM1.0 Note	SCK signal port mode register
PM_SIp	↑	PM1.1 Note	SI signal port mode register
PM_SOp	↑	PM1.2 Note	SO signal port mode register

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

Constant	Defined In	Setting	Description
P_CSIp	DEV&CSI_CH .inc	P1	Output latch for the port to be used
P_SCKp	↑	P1.0	SCK signal port
P_SIp	↑	P1.1	SI signal port
P_SOp	↑	P1.2	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	P2.1	Port for checking the BUSY signal
CS1SIG	↑	P2.3	CS1 output port
CS2SIG	↑	P2.2	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
MSTRWOMD	↑	11000000B	Transmit/receive command
SELCS1SIG	↑	00000100B	Data for selecting slave 1
SELCS2SIG	↑	00001000B	Data for selecting slave 2
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
SINISAU	Initialize CSIp.
SINITAU	Initialize TM03 and TM03H.
SSTARTINTV	Start 1-ms interval timer.
SCHANGEDATA	Generate expected value from transmit data and next transmit data.
SCHKDTSUB	Compare receive data with expected value.
SETTRXPNTR	Set up pointer to data buffer that is determined by slave number.
SWAIT1MS	Wait for 1ms interval timing in HALT state.
SSLAVSEL	Output CS signal to slave identified by slave number.
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.
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.
SSEQRXSUB	Start continuous reception processing (receive number of characters specified in A register into buffer designated by HL).
SWAITSTREND	Wait for end of continuous transfer.
SSEQTXSUB	Start continuous transmission processing (send data designated by HL number of times specified in A register.
SSEQTRXSUB	Start continuous transmission/reception processing (HL: transmit pointer, DE: receive pointer, A: number of characters).
SSETENDINT	Set up transfer end interrupts.
SSETEMPTYINT	Set up buffer empty interrupts.
SCHNG2TRX	Stop operation temporarily and enable transmission/reception mode (transfer end interrupt).
SCHNG2TX	Stop operation temporarily and enable transmission mode (transfer end interrupt).
SCHNG2RX	Stop operation temporarily and enable reception mode (transfer end interrupt).
STARTCSIp	Start CSI.
STOPCSIp	Stop 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] 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] 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] 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

[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] 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] 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] 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] IINTCSIp

Synopsis	Start INTCSIp interrupt processing.
Explanation	This function is activated on an INTCSIp, switches the register bank to 1, and causes a branch to the address that is stored in the variable RCSISUBADDR.
Arguments	None
Return value	None
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.

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 (0FFH) 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] 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] 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] 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] 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] 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] 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] 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] STARTCSIp

Synopsis	Enable CSIp.
Explanation	This function enables the CSIp for operation.
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] 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.

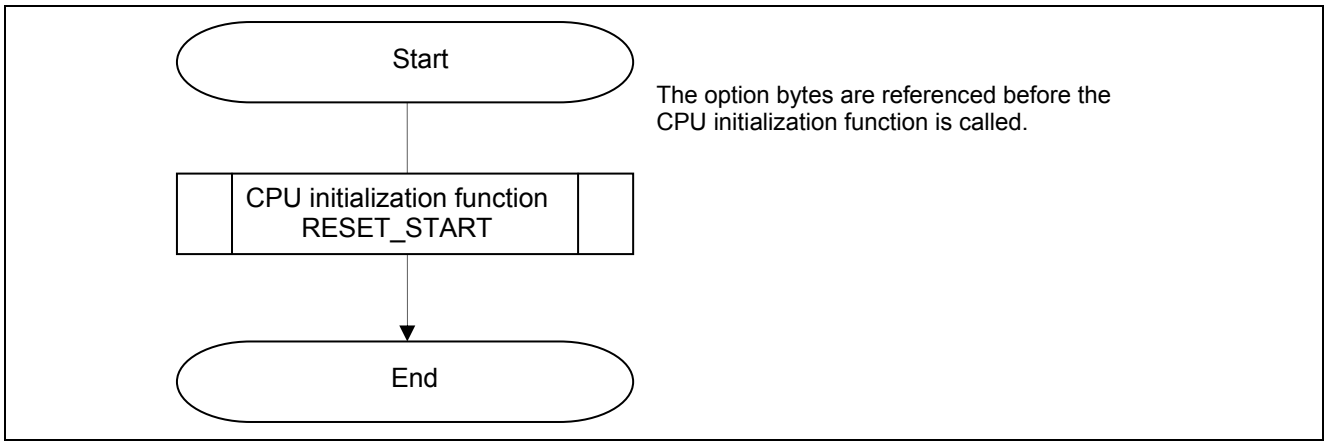


Figure 5.1 Overall Flow

5.7.1 CPU Initialization Function

Figure 5.2 shows the flowchart for the CPU initialization function.

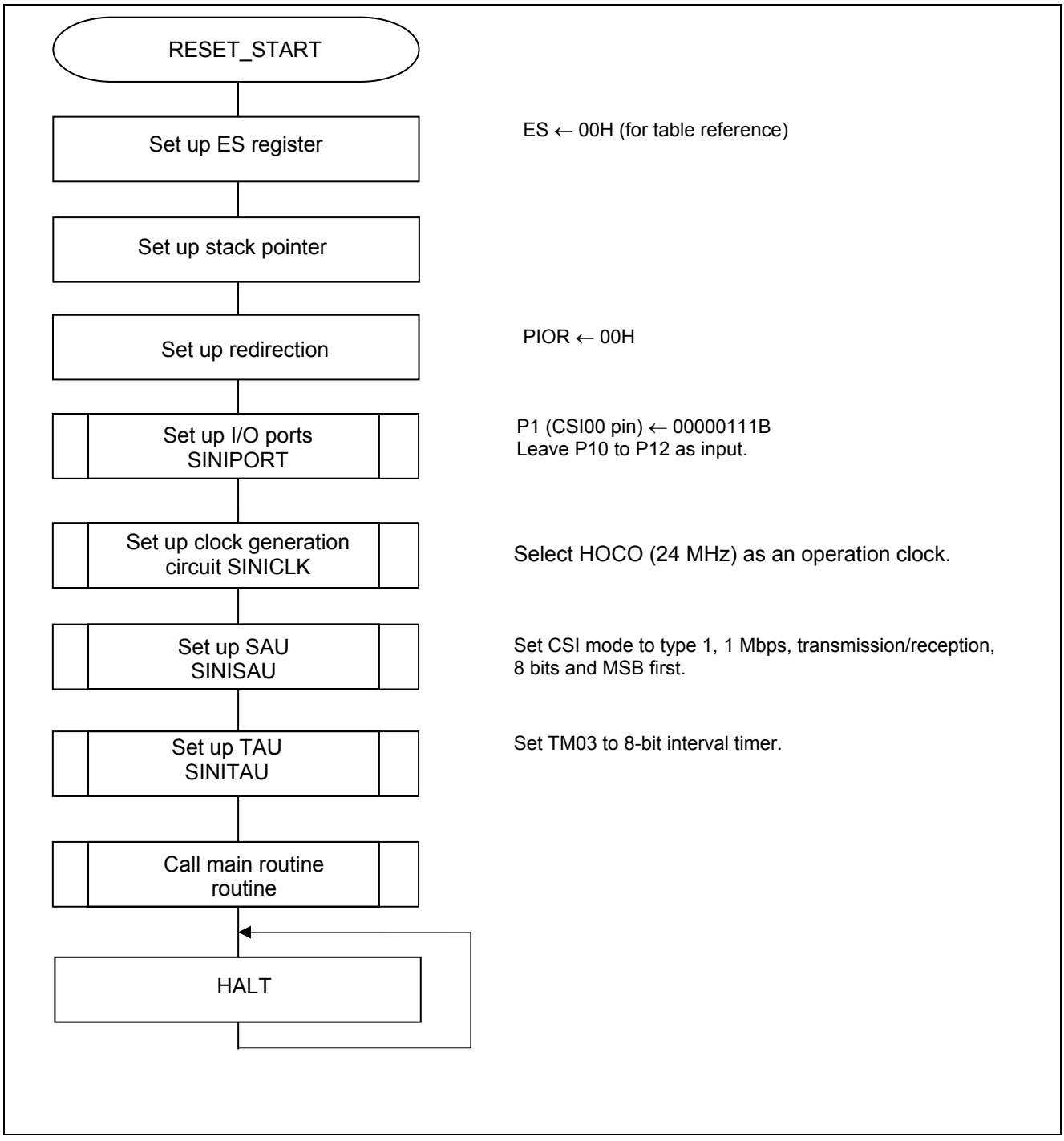


Figure 5.2 CPU Initialization Function

5.7.2 I/O Port Setup

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

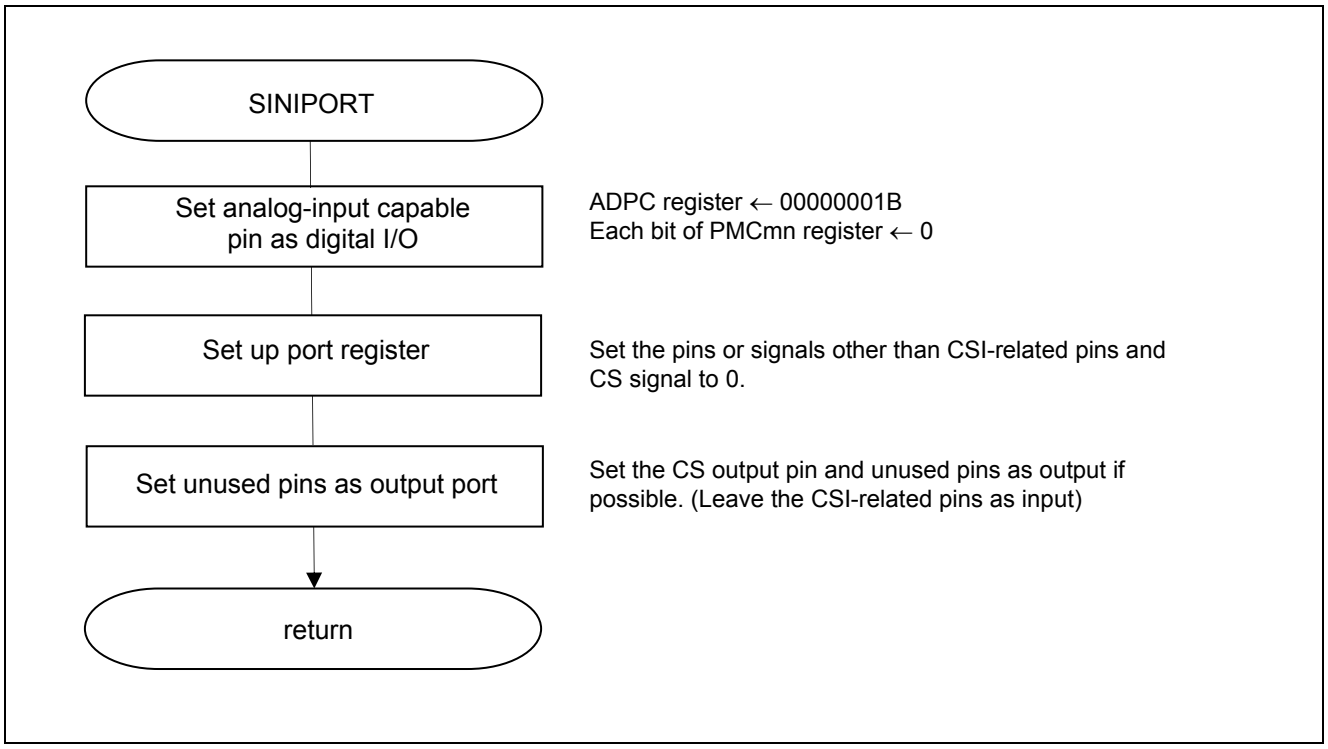


Figure 5.3 I/O Port Setup

Note: Refer to the section entitled "Flowcharts" in RL78/G12 Initialization Application Note (R01AN1030E) 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.

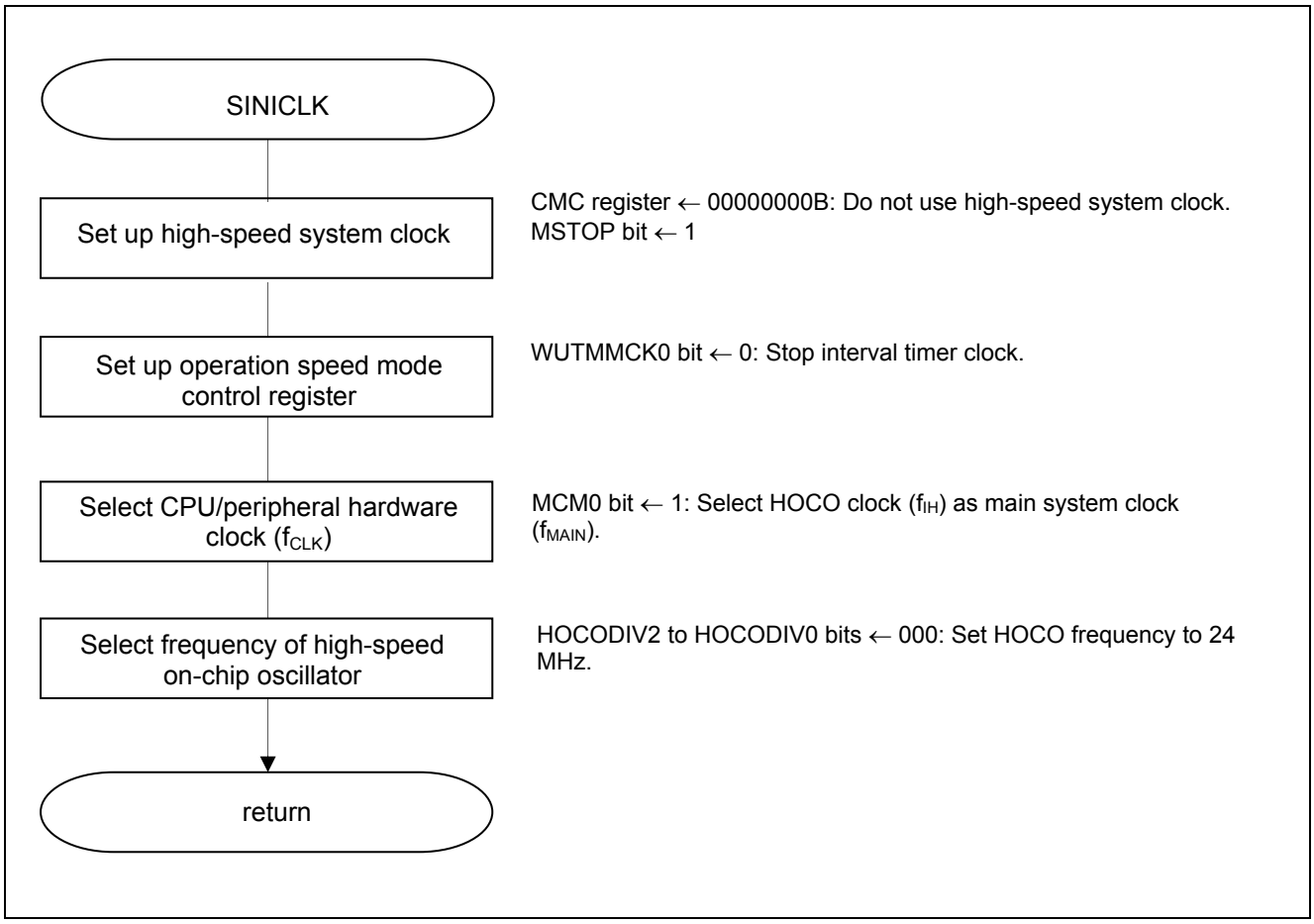


Figure 5.4 Clock Generator Circuit Setup

Caution: For details on the procedure for setting up the clock generation circuit (SINICLK), refer to the section entitled "Flowcharts" in RL78/G12 Initialization Application Note (R01AN1030E).

5.7.4 SAU Setup

Figure 5.5 shows the flowchart for SAU setup.

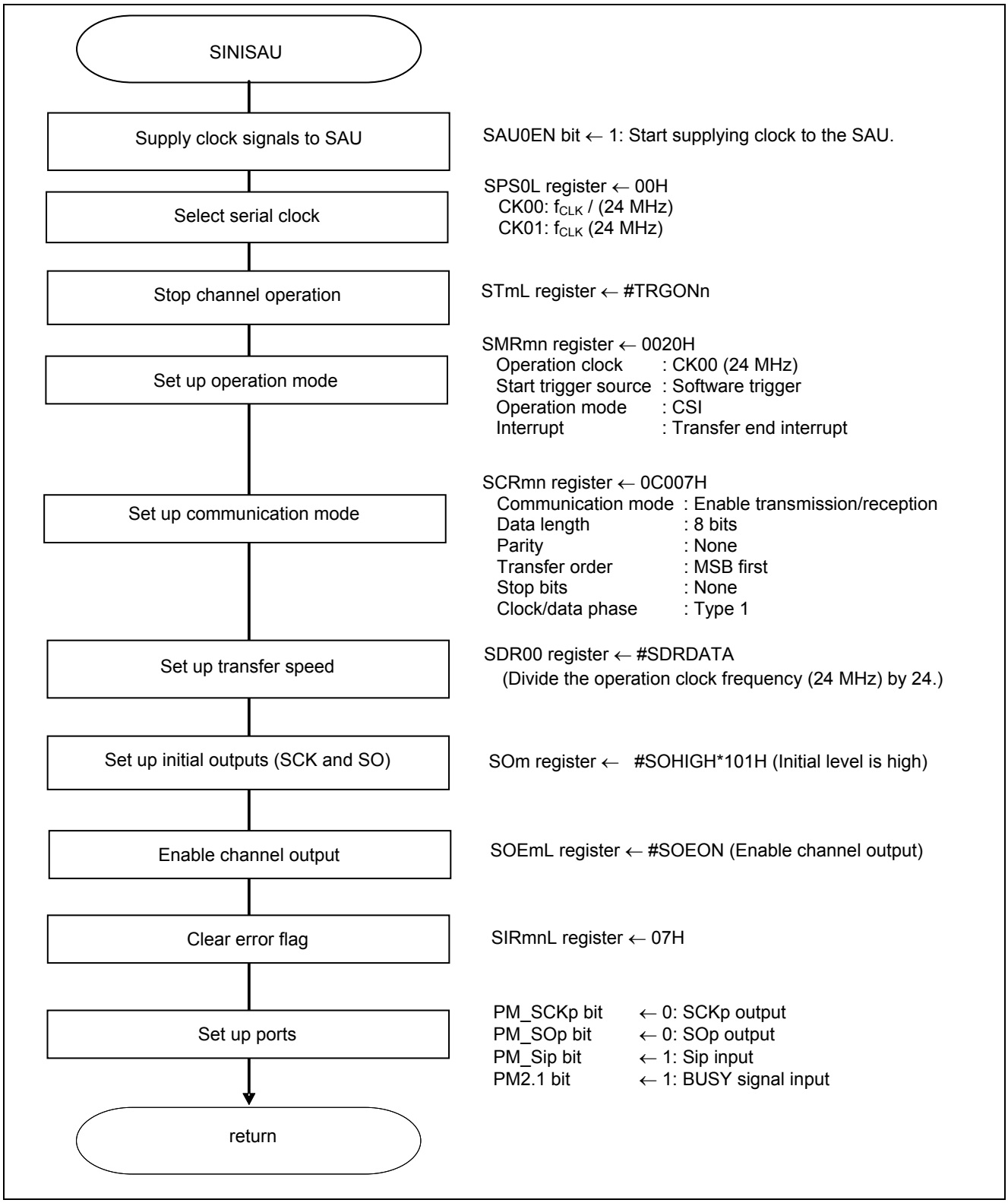


Figure 5.5 SAU Setup

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
TMKAE	0	ADCEN	IICA0EN	SAU1EN	SAU0EN	0	TAU0EN
x	0	x	x	**0/1**	**0/1**	0	x

Bits 3 and 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 (SPSm)
  Select an operation clock for SAU.

Symbol: SPSm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00
0	0	0	0	0	0	0	0	0	0	0	0	**0**	**1**	**0**	**0**

Bits 7 to 0

PRS mn3	PRS mn2	PRS mn1	PRS mn0	Selection of operation clock (CK0n) (n = 0 to 1)					
					f_{CLK} = 2 MHz	f_{CLK} = 5 MHz	f_{CLK} = 10 MHz	f_{CLK} = 20 MHz	f_{CLK} = 24 MHz
0	**0**	**0**	**0**	**f_{CLK}**	**2 MHz**	**5 MHz**	**10 MHz**	**20 MHz**	**24 MHz**
0	0	0	1	$f_{CLK}/2$	1 MHz	2.5 MHz	5 MHz	10 MHz	12 MHz
0	0	1	0	$f_{CLK}/2^2$	500 kHz	1.25 MHz	2.5 MHz	5 MHz	6 MHz
0	0	1	1	$f_{CLK}/2^3$	250 kHz	625 kHz	1.25 MHz	2.5 MHz	3 MHz
0	1	0	0	$f_{CLK}/2^4$	125 kHz	313 kHz	625 kHz	1.25 MHz	1.5 MHz
0	1	0	1	$f_{CLK}/2^5$	62.5 kHz	156 kHz	313 KHz	625 KHz	750 kHz
0	1	1	0	$f_{CLK}/2^6$	31.3 kHz	78.1 kHz	156 kHz	313 kHz	375 kHz
0	1	1	1	$f_{CLK}/2^7$	15.6 kHz	39.1 kHz	78.1 kHz	156 kHz	188 kHz
1	0	0	0	$f_{CLK}/2^8$	7.81 kHz	19.5 kHz	39.1 kHz	78.1 kHz	93.8 kHz
1	0	0	1	$f_{CLK}/2^9$	3.91 kHz	9.77 kHz	19.5 kHz	39.1 kHz	46.9 kHz
1	0	1	0	$f_{CLK}/2^{10}$	1.95 kHz	4.88 kHz	9.77 kHz	19.5 kHz	23.4 kHz
1	0	1	1	$f_{CLK}/2^{11}$	977 Hz	2.44 kHz	4.88 kHz	9.77 kHz	11.7 kHz
1	1	0	0	$f_{CLK}/2^{12}$	488 Hz	1.22 kHz	2.44 kHz	4.88 kHz	5.86 kHz
1	1	0	1	$f_{CLK}/2^{13}$	244 Hz	610 Hz	1.22 kHz	2.44 kHz	2.93 kHz
1	1	1	0	$f_{CLK}/2^{14}$	122 Hz	305 Hz	610 Hz	1.22 kHz	1.46 kHz
1	1	1	1	$f_{CLK}/2^{15}$	61 Hz	153 Hz	305 Hz	610 Hz	732 Hz

Caution: For details on the register setup procedures, refer to RL78/G12 User's Manual: Hardware.

Setting up channel operation mode

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

Symbol: SMRmn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS mn	CCS mn	0	0	0	0	0	STS mn	0	0	1	0	0	MD mn2	MD mn1	MD mn0
0	**0**	0	0	0	0	0	**0**	0	0	1	0	0	**0**	**0**	**0**

Bit 15

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 14

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 8

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

Bits 2 and 1

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

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

Caution: For details on the register setup procedures, refer to RL78/G12 User's Manual: Hardware.

Setting up channel communication mode

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

Symbol: SCRmn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXE mn	RXE mn	DAP mn	CKP mn	0	EOC mn	PTC mn1	PTC mn0	DIR mn	0	SLC mn1	SLC mn0	0	1	DLS mn1	DLS mn0
1	**1**	**0**	**0**	0	**0**	**0**	**0**	**0**	0	**0**	**0**	0	1	**1**	**1**

Bits 15 and 14

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

Bit 10

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

Bits 9 and 8

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

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

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

Caution: For details on the register setup procedures, refer to RL78/G12 User's Manual: Hardware.

Symbol: SCRmn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXE mn	RXE mn	DAP mn	CKP mn	0	EOC mn	PTC mn1	PTC mn0	DIR mn	0	SLC mn1	SLC mn0	0	1	DLS mn1	DLS mn0
1	**1**	**0**	**0**	0	**0**	**0**	**0**	**0**	0	**0**	**0**	0	1	**1**	**1**

Bits 1 and 0

DLSmn 1	DLSmn 0	Setting of data length in CSI mode
0	1	9-bit data length
1	0	7-bit data length
1	**1**	**8-bit data length**
Other than above		Setting prohibited

Setting up channel transfer clock

- Serial data register mn (SDRmn)
  Transfer clock frequency: f_{MCK}/24 (= 1 MHz)

Symbol: SDRmn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	**0**	**1**	**0**	**1**	**1**	**1**	0	x	x	x	x	x	x	x	x

Bits 15 to 9

SDRmn[15:9]							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**	**1**	**0**	**1**	**1**	**1**	**f_{MCK} /24**
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
1	1	1	1	1	1	0	f_{MCK} /254
1	1	1	1	1	1	1	f_{MCK} /256

Caution: For details on the register setup procedures, refer to RL78/G12 User's Manual: Hardware.

Setting initial output level

- Serial output register m (SOm)
  Initial output: 1

Symbol: SOm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	CKOm3	CKOm2	CKOm1	CKOm0	0	0	0	0	SOm3	SOm2	SOm1	SOm0
0	0	0	0	**0/1**	**0/1**	**0/1**	**0/1**	0	0	0	0	**0/1**	**0/1**	**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”.**

Enabling target channel data output

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

Symbol: SOEm

								SOEmL							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	SOEm3	SOEm2	SOEm1	SOEm0
0	0	0	0	0	0	0	0	0	0	0	0	**0/1**	**0/1**	**0/1**	**0/1**

Bit n

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

Caution: For details on the register setup procedures, refer to RL78/G12 User's Manual: Hardware.

Clearing the error flags

- Serial flag clear trigger register mn (SIRmn)
  Clear the error flags

Symbol: SIRmn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	FECTmn	PECTmn	OVCTmn
0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**

Bit 2

FECTmn	Clear trigger of framing error of channel n
0	Not cleared
1	**Clears the FEFmn bit of the SSRmn registers to 0.**

Bit 1

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

Bit 0

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

Configuring the interrupt mask

- Interrupt mask flag register 0H (MK0H)
  Disable interrupt processing.

Symbol: MK0H (20-/24- pin products)

7	6	5	4	3	2	1	0
TMMK01	TMMK00	IICAMK0	TMMK03H	TMMK01H	SREMK0	SRMK0 CSIMK01 IICMK01	STMK0 CSIMK00 IICMK00
x	x	x	x	x	x	**0/1**	**0/1**

CSIMK01	CSIMK00	Interrupt processing control
0	0	**Enables interrupt processing.**
1	**1**	**Disables interrupt processing.**

Caution: For details on the register setup procedures, refer to RL78/G12 User's Manual: Hardware.

Port setting (For CSI00)

- Port register 1 (P1)
- Port mode register 1 (PM1)
  Port setting for each of serial clock, transmit data and receive data.

Symbol: P1

7	6	5	4	3	2	1	0
P17	P16	P15	P14	P13	P12	P11	P10
x	x	x	x	x	**1**	**1**	**1**

Bit 2

P12 to P10	Output data control (in output mode)
0	0 is output
1	**1 is output**

Symbol: PM1

7	6	5	4	3	2	1	0
PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10
x	x	x	x	x	**0**	**1**	**0**

Bit 2

PM12	P12 pin I/O mode selection
0	**Output mode (output buffer on)**
1	Input mode (output buffer off)

Bit 1

PM11	P11 pin I/O mode selection
0	Output mode (output buffer on)
1	**Input mode (output buffer off)**

Bit 0

PM10	P10 pin I/O mode selection
0	**Output mode (output buffer on)**
1	Input mode (output buffer off)

Caution: For details on the register setup procedures, refer to RL78/G12 User's Manual: Hardware.

5.7.5 Timer Array Unit Setup

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

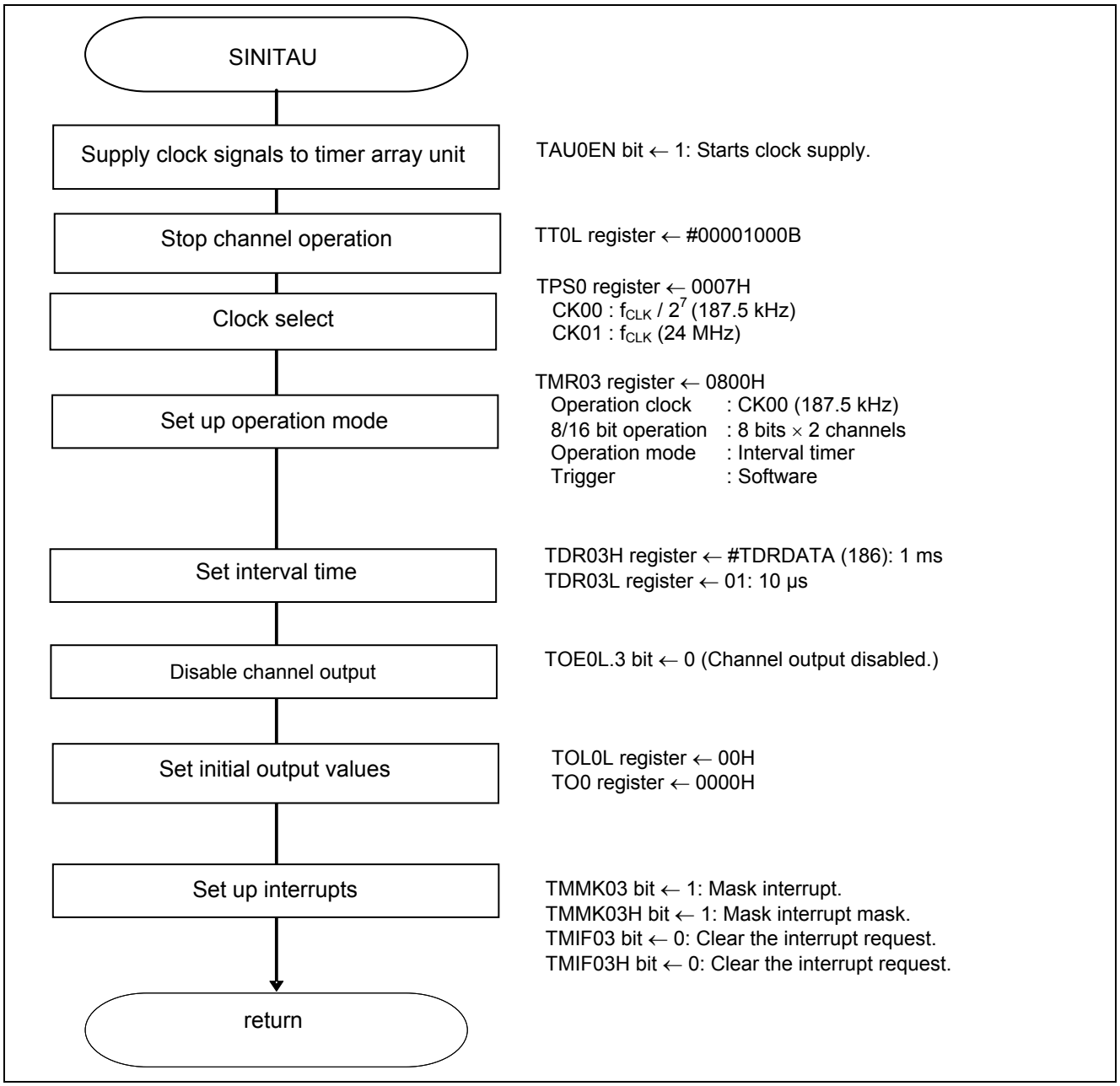


Figure 5.6 Timer Array Unit Setup

5.7.6 Main Processing

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

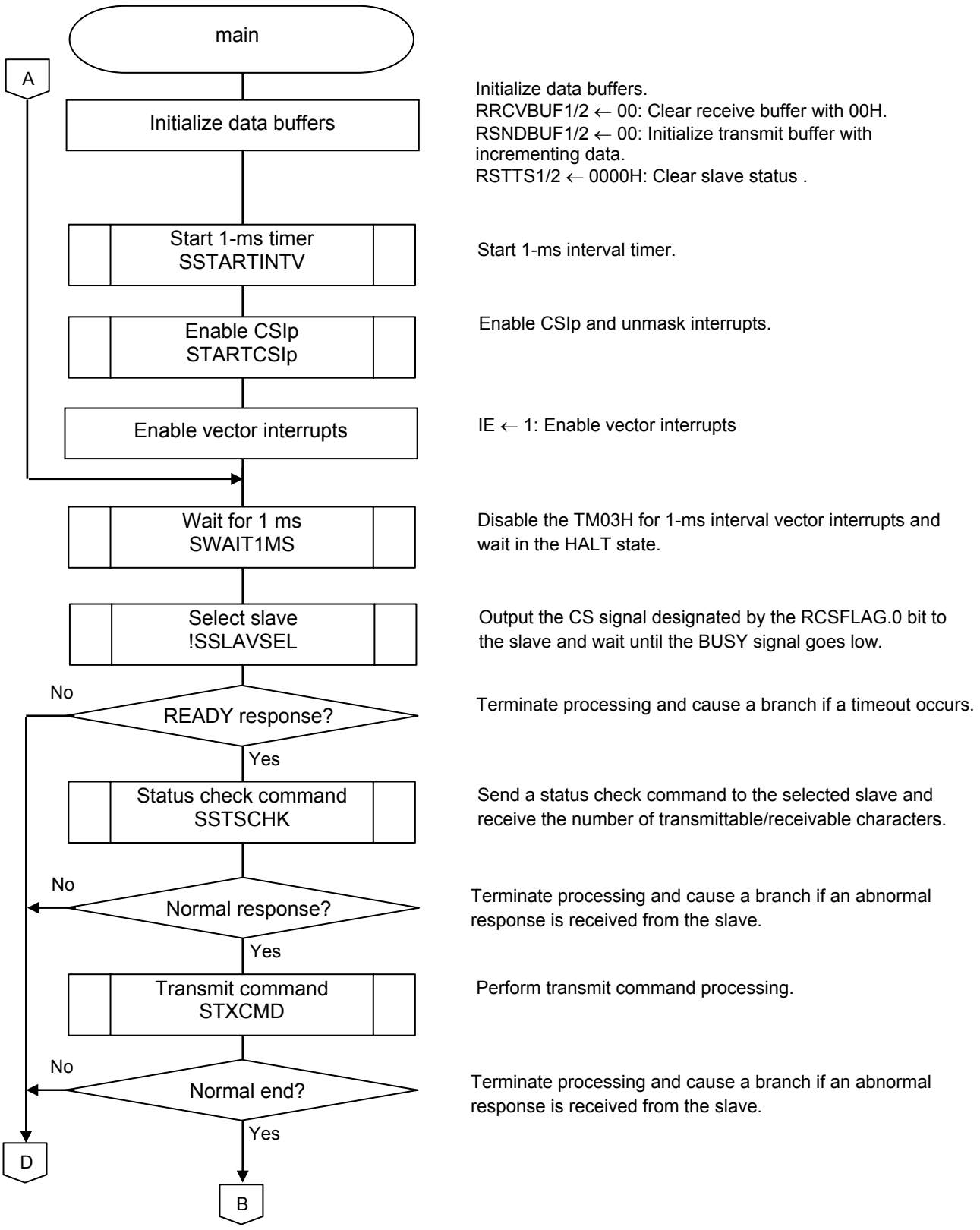


Figure 5.7 Main Processing (1/3)

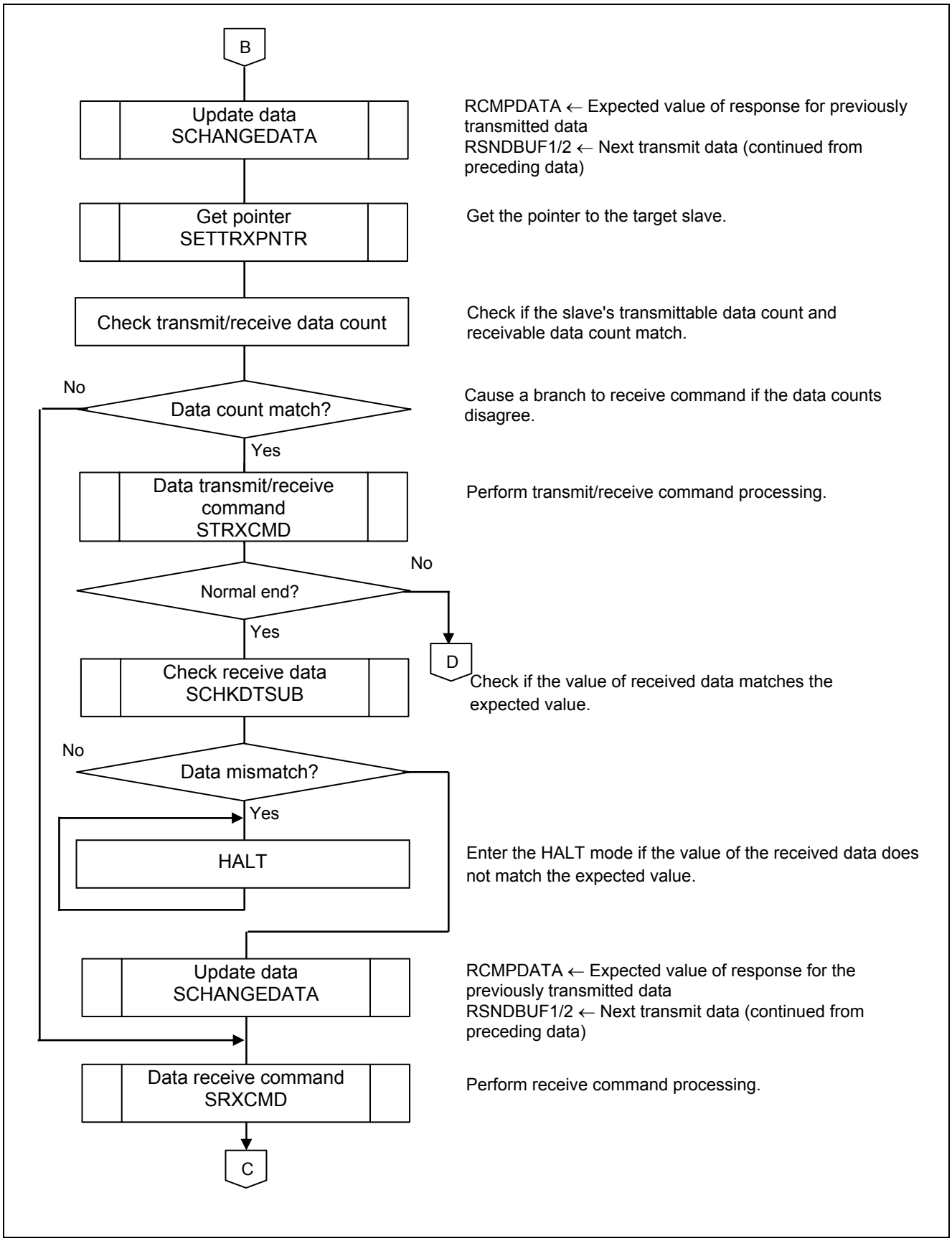
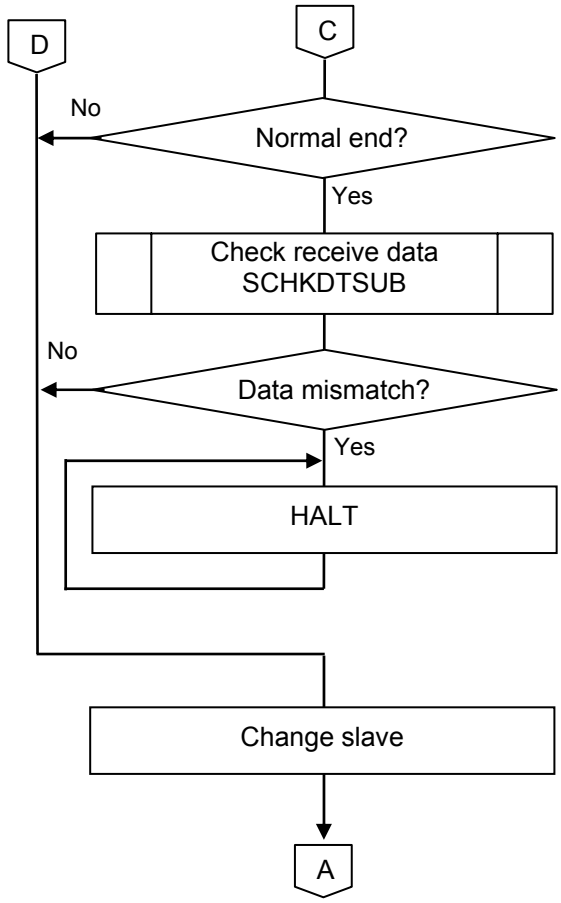


Figure 5.8 Main Processing (2/3)



Check if the value of received data matches the expected value.

Enter the HALT mode if the value of the received data does not match the expected value.

Invert the RCSFLAG.0 bit identifying the slave and continue processing starting at the status check step.

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.

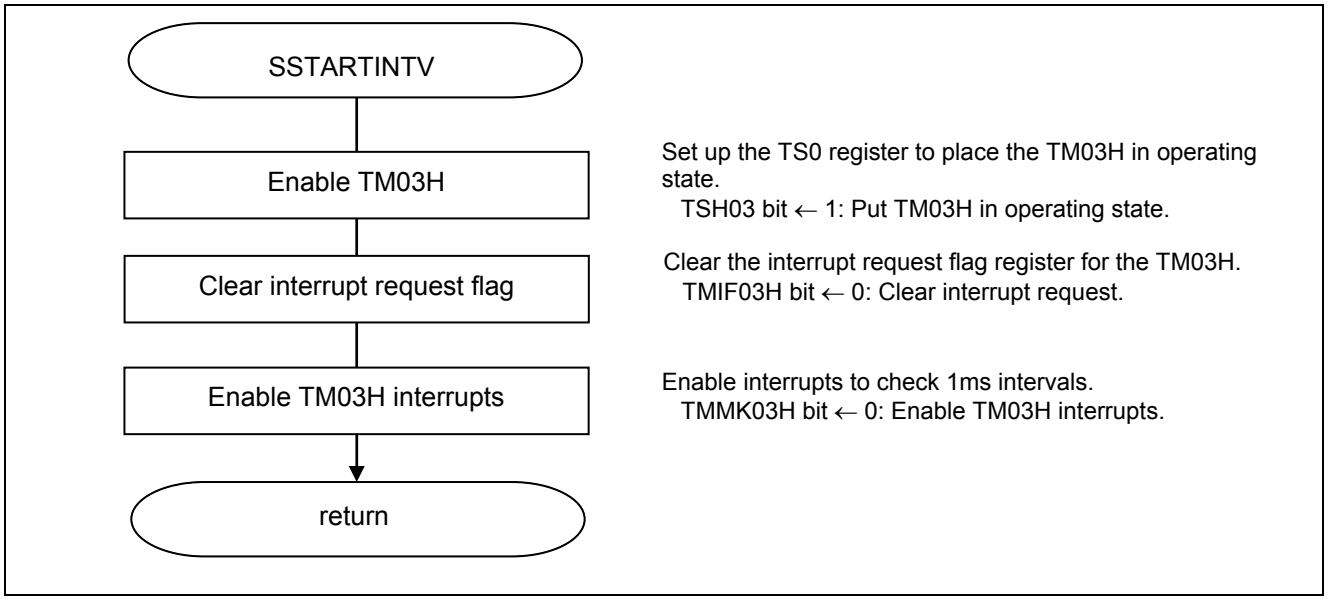


Figure 5.10 1-ms Interval Timer Startup Processing

Transiting to timer operating state (20-/24-pin products)

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

Symbol: TS0 (20-/24-pin products)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	TSH03	0	TSH01	0	0	0	0	0	TS03	TS02	TS01	TS00
0	0	0	0	**1**	0	x	0	0	0	0	0	**0**	x	x	x

Bit 11

TSH03	Operation start trigger of channel 03H
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/G12 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 (20-/24- pin products)

7	6	5	4	3	2	1	0
TMIF01	TMIF00	IICAIF0	TMIF03H	TMIF01H	SREIF0	SRIF0 CSIIF01 IICIF01	STIF0 CSIIF00 IICIF00
x	x	x	**0**	x	x	x	x

TMIF03H	Interrupt request flag
0	**No interrupt request signal is generated**
1	Interrupt request is generated, interrupt request status

Symbol: MK0H (20-/24- pin products)

7	6	5	4	3	2	1	0
TMMK01	TMMK00	IICAMK0	TMMK03H	TMMK01H	SREMK0	SRMK0 CSIMK01 IICMK01	STMK0 CSIMK00 IICMK00
x	x	x	**0**	x	x	x	x

TMMK03H	Interrupt processing control
0	**Enables interrupt processing.**
1	Disables interrupt processing.

Caution: For details on the register setup procedures, refer to RL78/G12 User's Manual: Hardware.

5.7.8 1-ms Interval Wait Function

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

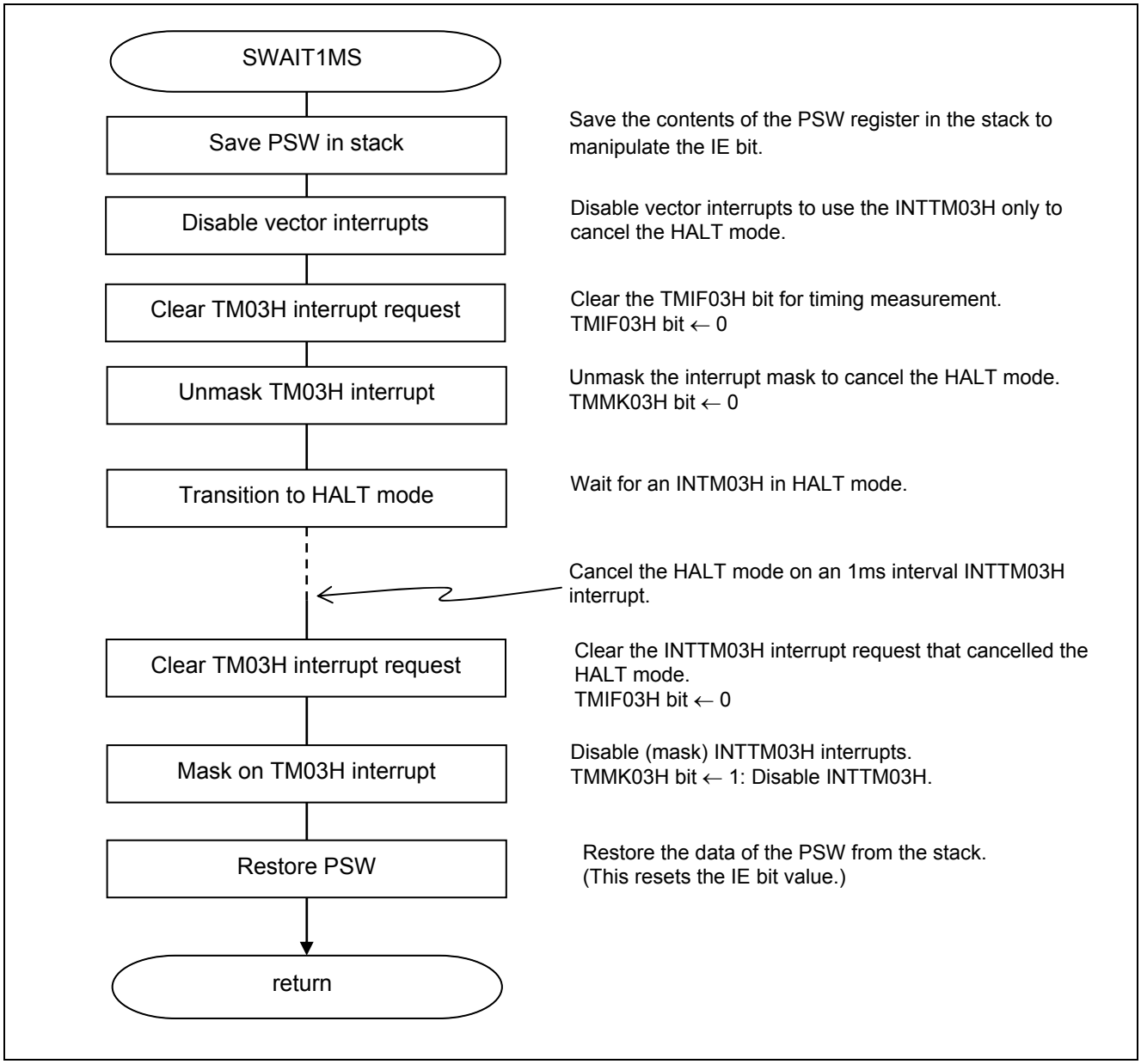


Figure 5.11 1-ms Interval Wait Function

Interrupt setting

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

Symbol: IF0H (20-/24- pin products)

7	6	5	4	3	2	1	0
TMIF01	TMIF00	IICAIF00	TMIF03H	TMIF01H	SREIF0	SRIF0 CSIIF01 IICIF01	STIF0 CSIIF00 IICIF00
x	x	x	**0**	x	x	x	x

Bit 4

TMIF03H	Interrupt request flag
0	**No interrupt request signal is generated**
1	Interrupt request is generated, interrupt request status

Symbol:: MK0H (20-/24- pin products)

7	6	5	4	3	2	1	0
TMMK01	TMMK00	IICAMK00	TMMK03H	TMMK01H	SREMK0	SRMK0 CSIMK01 IICMK01	STMK0 CSIMK00 IICMK00
x	x	x	**0/1**	x	x	x	x

Bit 4

TMMK03H	Interrupt processing control
0	**Enables interrupt processing.**
1	**Disables interrupt processing.**

Caution: For details on the register setup procedures, refer to RL78/G12 User's Manual: Hardware.

5.7.9 Slave Selection Processing

Figure 5.12 shows the flowchart for the slave selection processing.

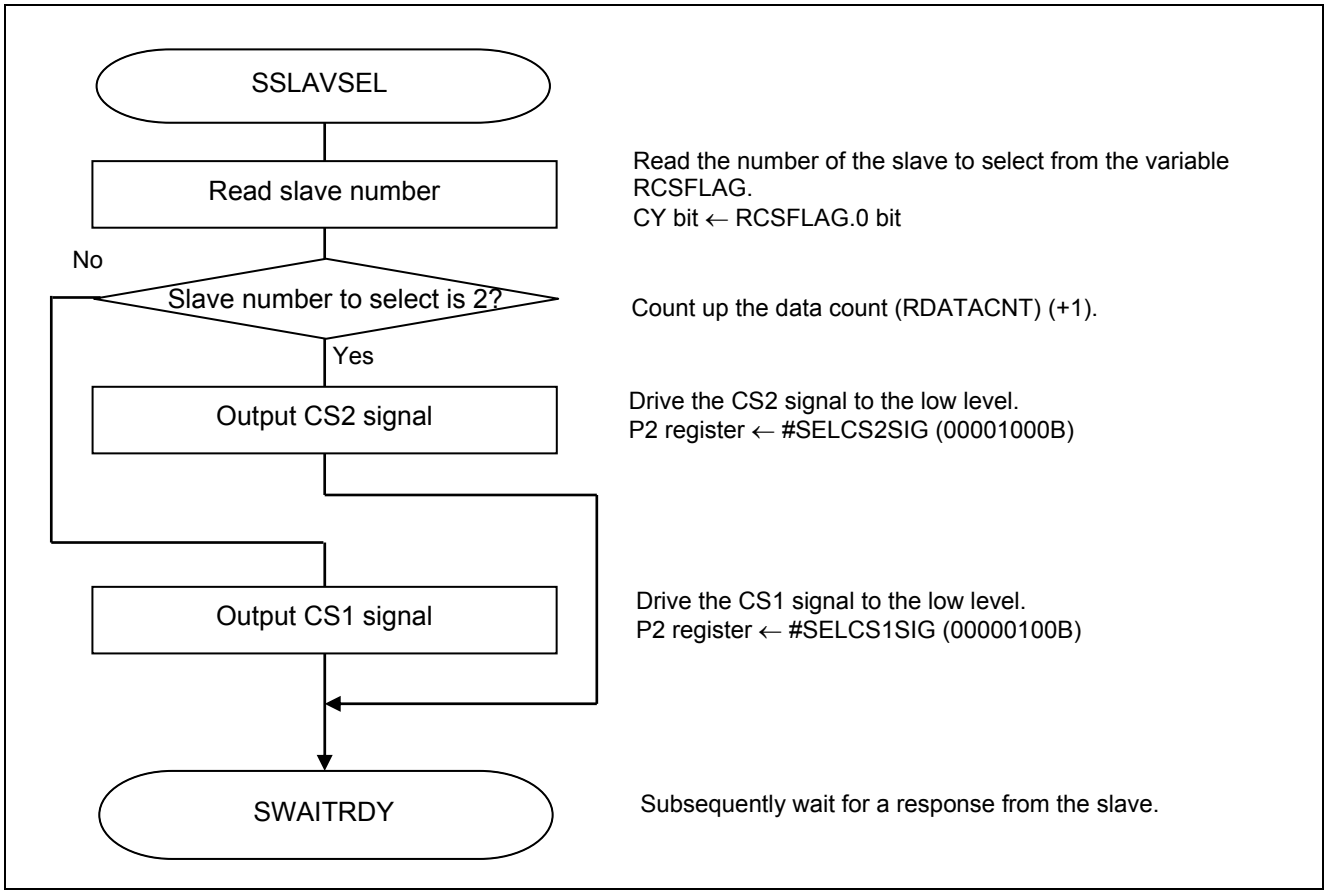


Figure 5.12 Slave Selection Processing

5.7.10 Slave Response Wait Processing

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

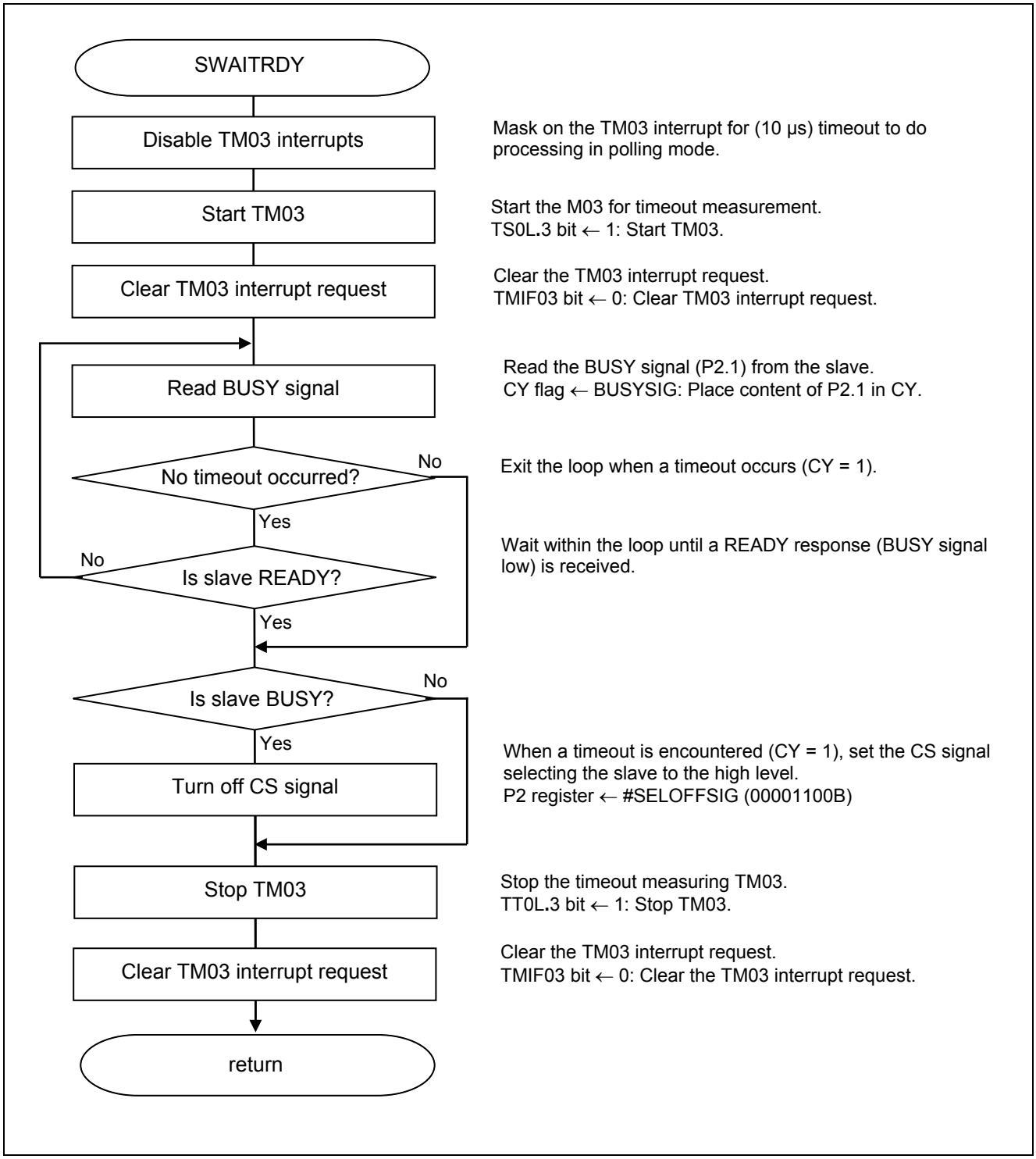


Figure 5.13 Slave Response Wait Processing

Transiting to timer operating state (20-/24-pin products)

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

Symbol: TS0 (20-/24-pin products)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	TSH03	0	TSH01	0	0	0	0	0	TS03	TS02	TS01	TS00
0	0	0	0	x	0	x	0	0	0	0	0	**1**	x	x	x

Bit 3

TS03	Operation enable (start) trigger of channel 03
0	No trigger operation
1	**TE03 is set to 1 and the count operation becomes enabled.**

Transiting to timer stopped state (20-/24-pin products)

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

Symbol: TT0 (20-/24-pin products)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	TTH03	0	TTH01	0	0	0	0	0	TT03	TT02	TT01	TT00
0	0	0	0	x	0	x	0	0	0	0	0	**1**	x	x	x

Bit 3

TT03	Operation stop trigger of channel 03
0	No trigger operation
1	**TE0n is cleared to 0. Operation is stopped (stop trigger is generated.**

Caution: For details on the register setup procedures, refer to RL78/G12 User's Manual: Hardware.

Interrupt setting (20-/24- pin products)

- Interrupt request flag register (IF1L)
  Clear the interrupt request flag
- Interrupt mask flag register (MK1L)
  Cancel interrupt mask

Symbol: IF1L (20-/24- pin products)

7	6	5	4	3	2	1	0
0	FLIF	MDIF	KRIF	TMKAIF	ADIF	TMIF03	TMIF02
0	x	x	x	x	x	**0**	x

Bit 1

TMIF03	Interrupt request flag
0	**No interrupt request signal is generated**
1	Interrupt request is generated, interrupt request status

Symbol: MK1L (20-/24- pin products)

7	6	5	4	3	2	1	0
1	FLMK	MDMK	KRMK	TMKAMK	ADMK	TMMK03	TMMK02
1	x	x	x	x	x	**1**	x

Bit 1

TMMK03	Interrupt processing control
0	Enables interrupt processing.
1	**Disables interrupt processing.**

Caution: For details on the register setup procedures, refer to RL78/G12 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.

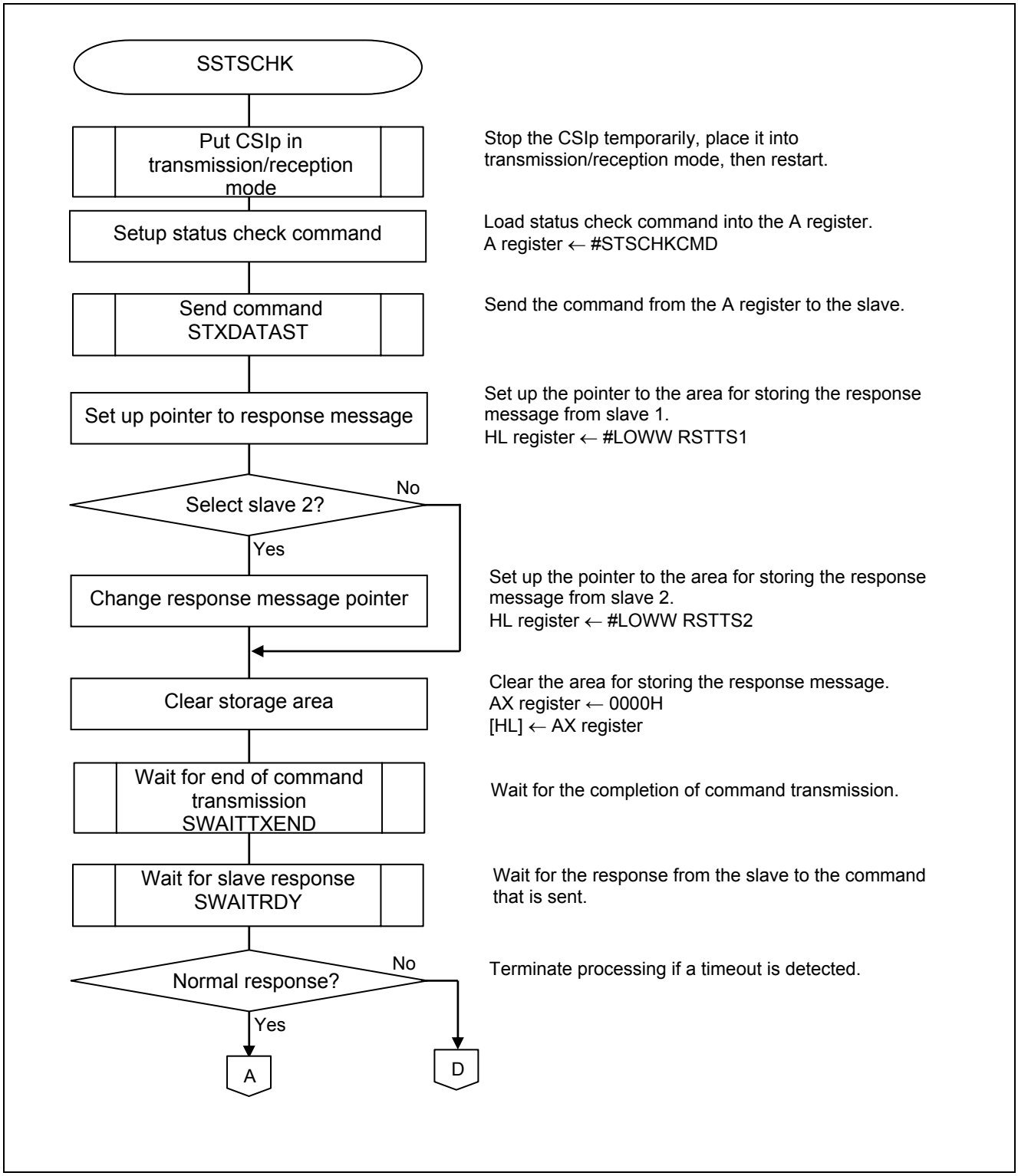
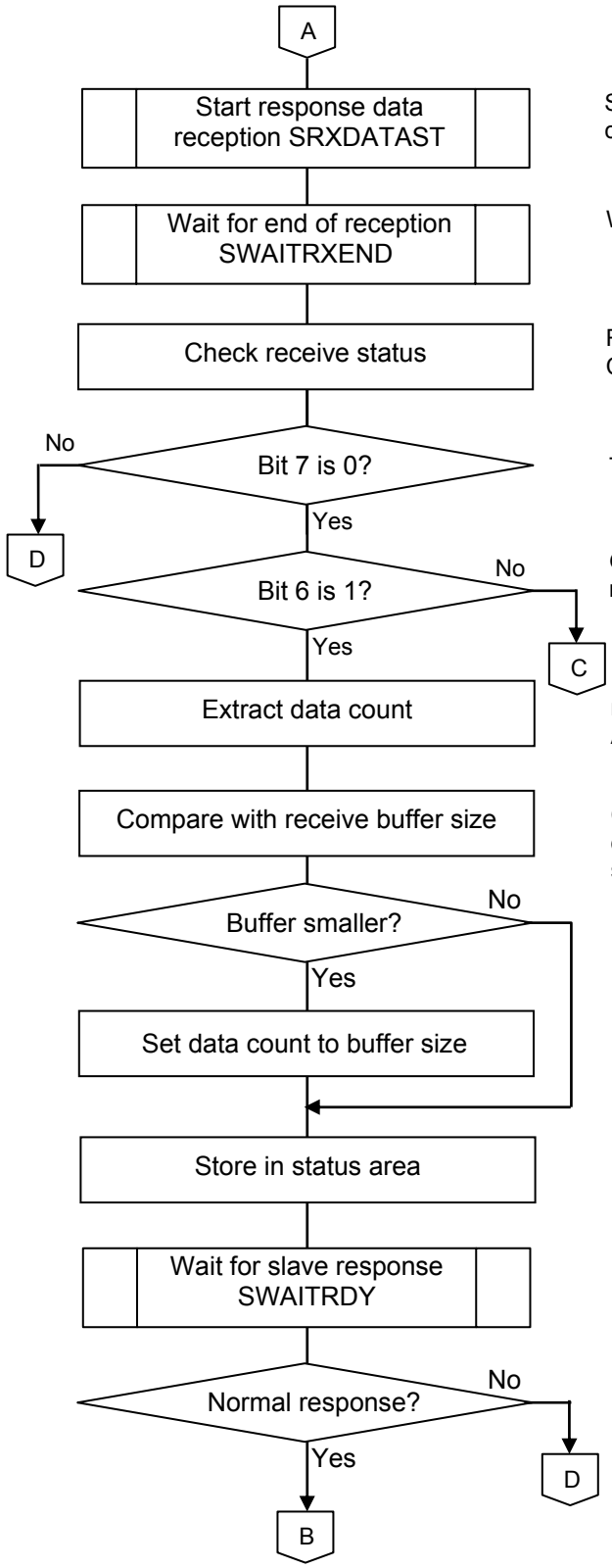


Figure 5.14 Status Check Processing (1/3)



Start the function to receive the response to the status check command.

Wait for the end of data transmission and get the status.

Read bit 7 of the received status.
CY bit ← A.7 bit (CY = 1 if error)

Terminate processing if bit 7 of the received status is 1.

Go to receivable data count processing of bit 6 of the received status is 0.

Extract the slave transmittable data count from the response message.
A register ← Slave transmittable data count

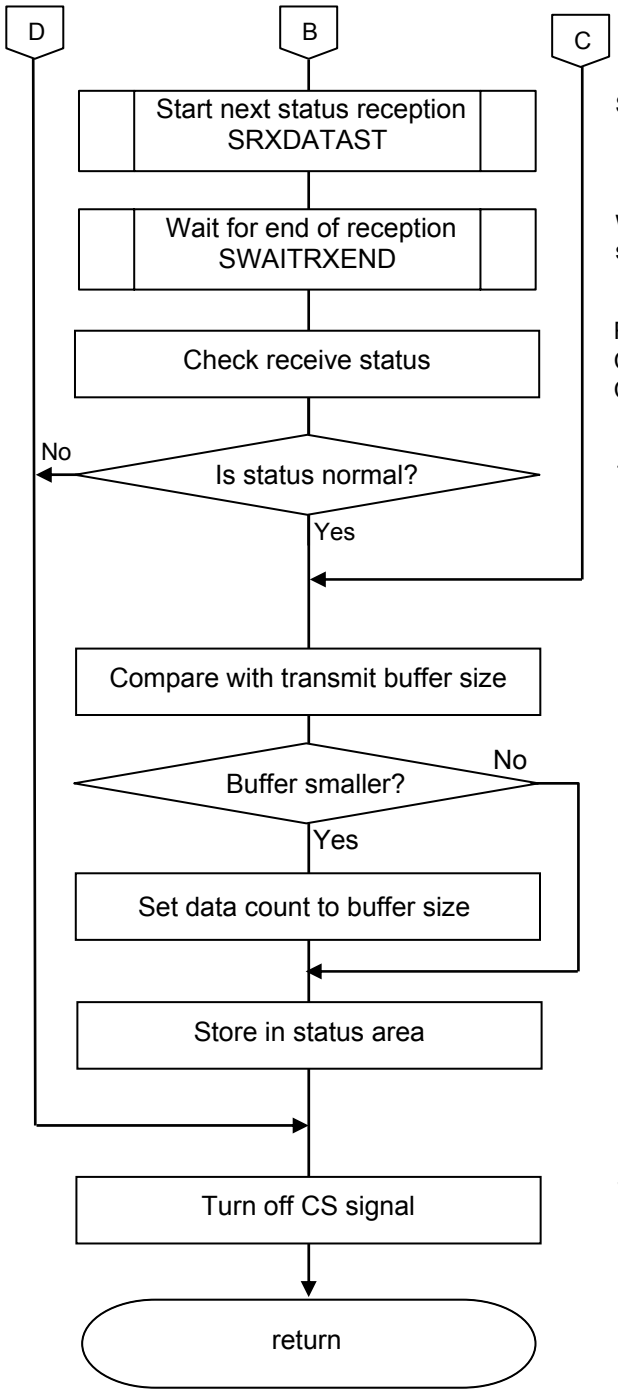
Compare the slave transmittable data count with the size of the receive buffer and place the smaller value in the status area.

```
CMP     A,      #CRXDTNO
SKC
MOV     A,      #CRXDTNO
MOV     [HL],   A
```

Wait for the slave to get ready for send the next response message.

Terminate processing if a timeout is detected.

Figure 5.15 Status Check Processing (2/3)



Start receive processing for the second status.

Wait for the end of data reception and get the received status.

Read bit 7 of the received status.
CY bit ← A.7 bit (CY = 1 if error)
CY bit ← CY bit | A.6 bit

Terminate processing if bits 6 and 7 of the received status is not 00.

Compare the slave receivable data count with the size of the transmit buffer and place the smaller value in the status area.

```
INCW    HL
CMP     A,      #CTXDTNO
SKC
MOV     A,      #CTXDTNO
MOV     [HL],   A
```

Set the CS signal selecting the slave to the high level.
P2 register ← #SELOFFSIG (00001100B)

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.

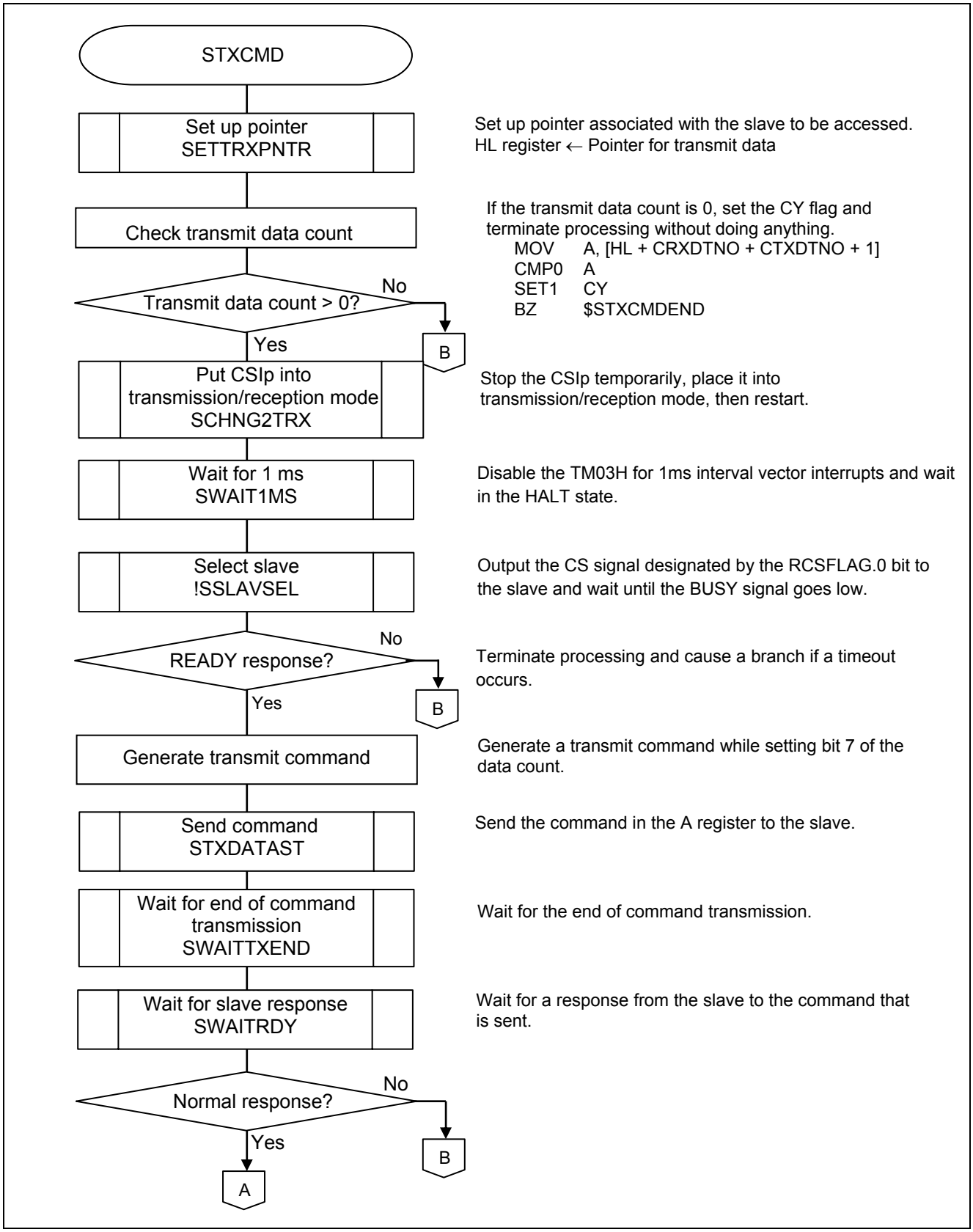


Figure 5.17 Continuous Data Transmission Processing (1/2)

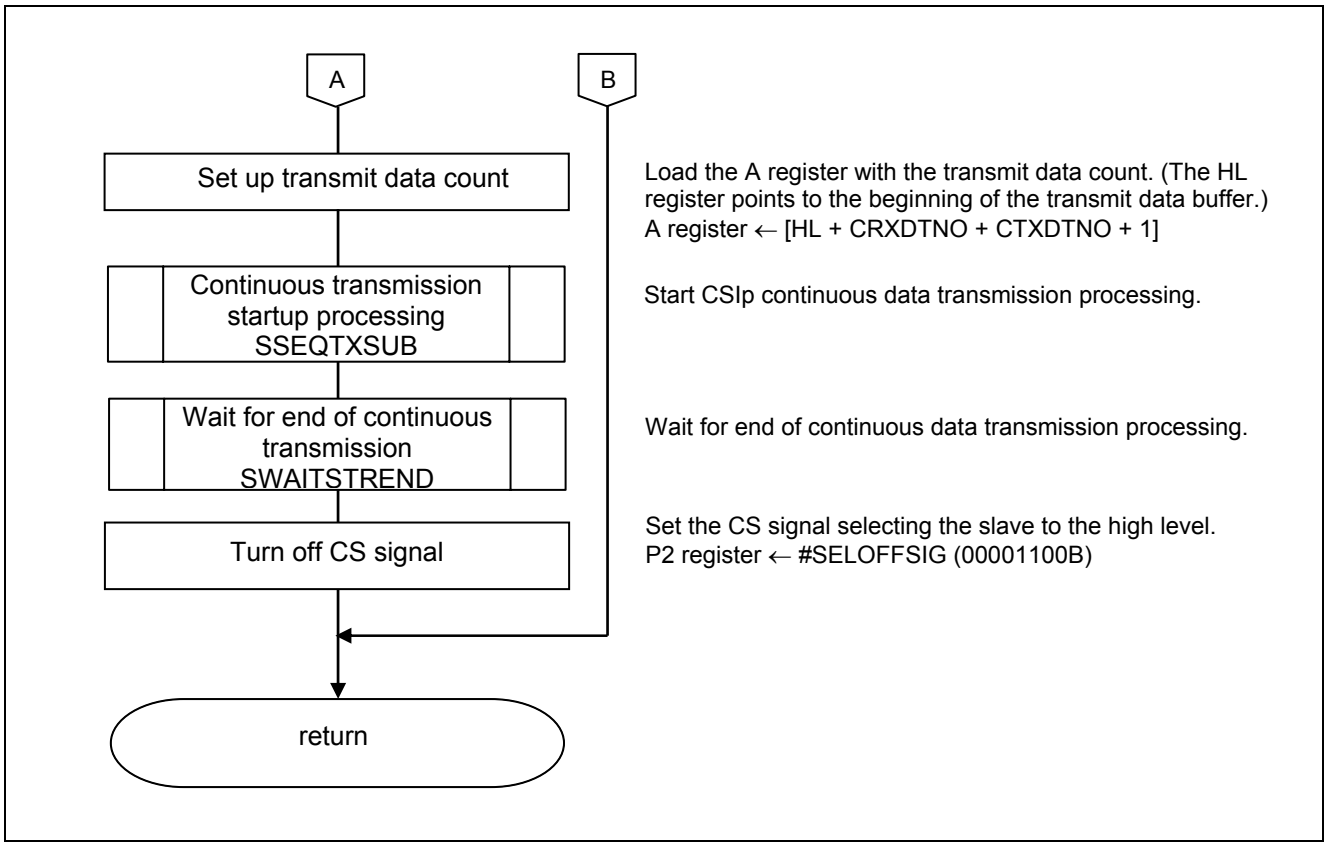
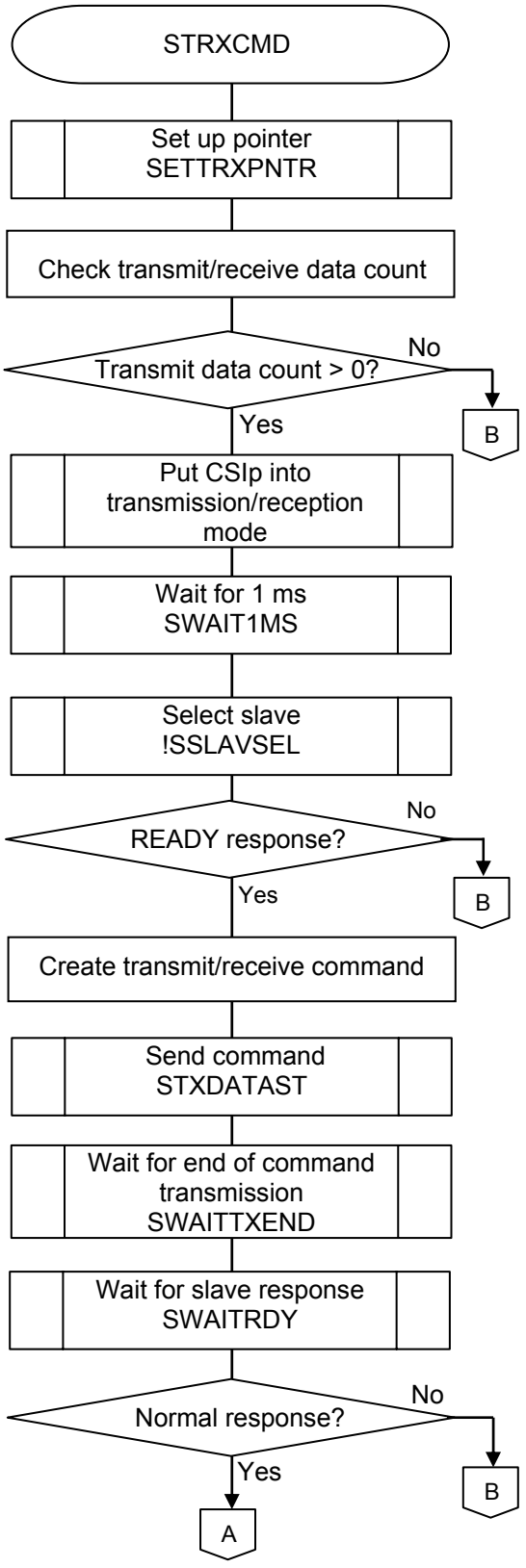


Figure 5.18 Continuous Data Transmission Processing (2/2)

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.



Set up pointer associated with the slave to be accessed.
HL register ← Pointer for transmit data
DE register ← Pointer for receive data

If the transmit/receive data count is 0, set the CY flag and terminate processing without doing anything.

```
MOV     A, [HL + CRXDTNO + CTXDTNO + 1]
CMP0    A
SET1    CY
BZ      $STRXCMDEND
```

Stop the CSIp temporarily, place it into transmission/reception mode, then restart.

Disable the TM03H for 1ms interval vector interrupts and wait in the HALT state.

Output the CS signal designated by the RCSFLAG.0 bit to the slave and wait until the BUSY signal goes low.

Terminate processing and cause a branch if a timeout occurs.

Generate a transmit/receive command while setting bits 7 and 6 of the data count.

Send the command in the A register to the slave.

Wait for the end of command transmission.

Wait for a response from the slave to the command that is sent.

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

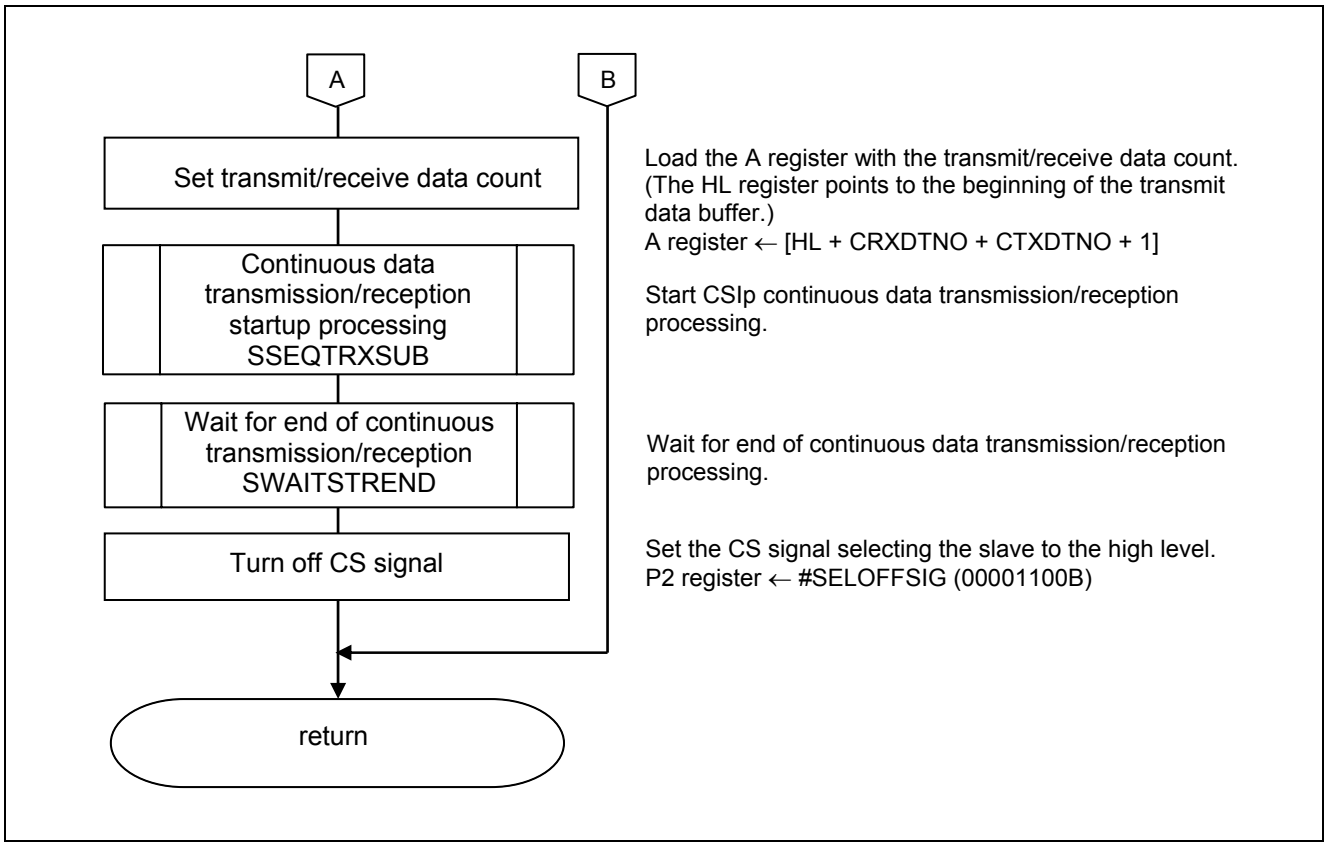


Figure 5.20 Continuous Data Transmission/Reception Processing (2/2)

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.

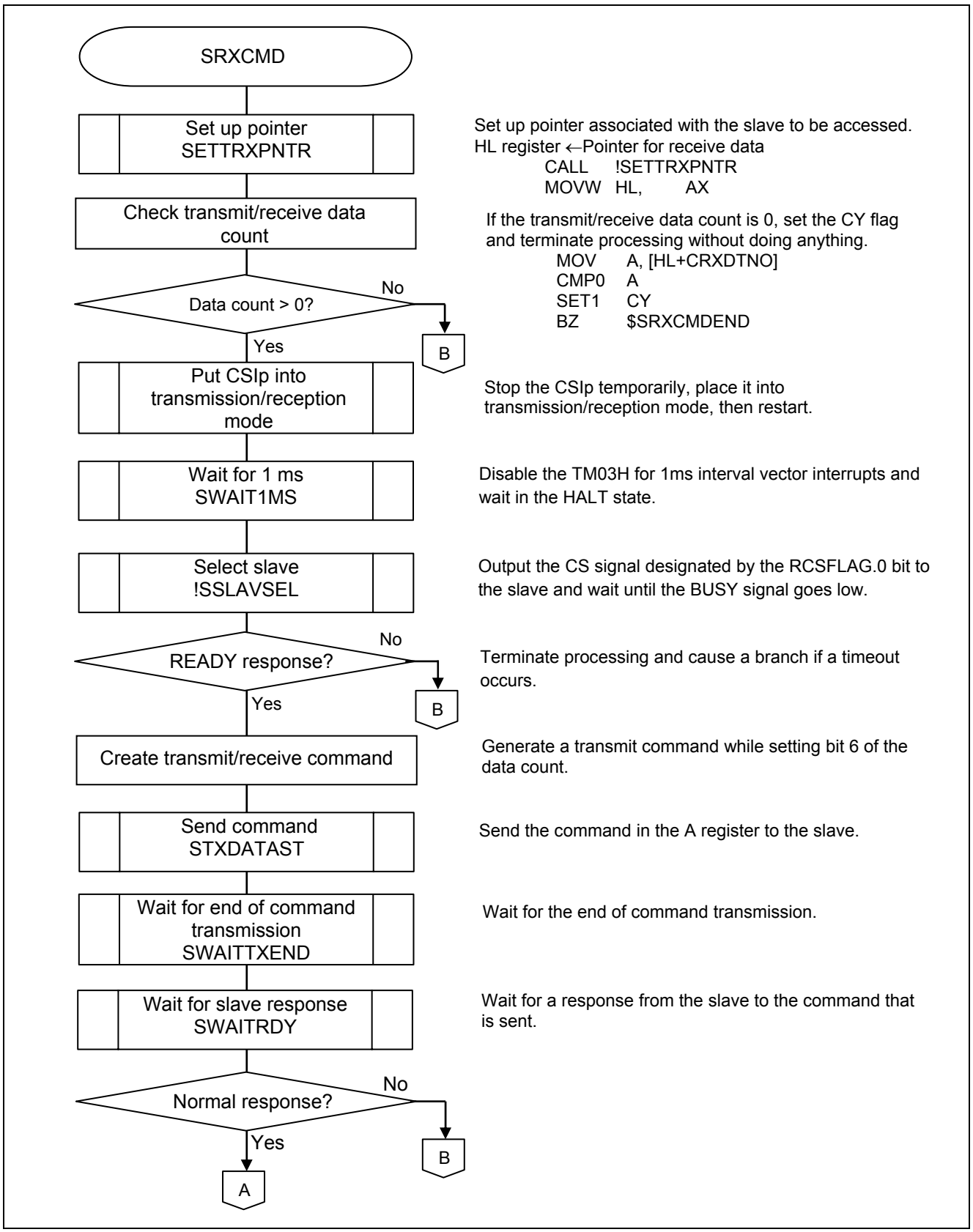


Figure 5.21 Continuous Data Reception Processing (1/2)

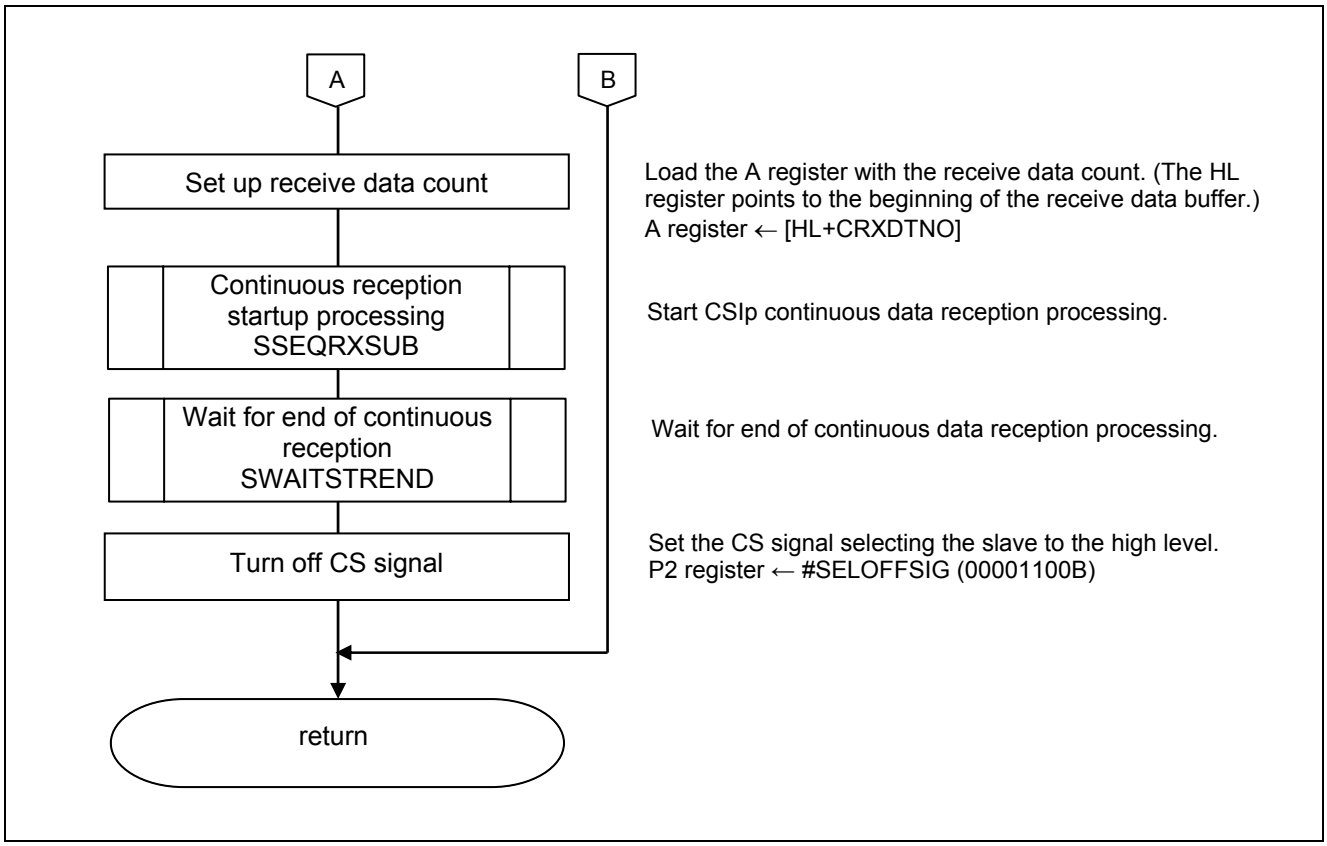
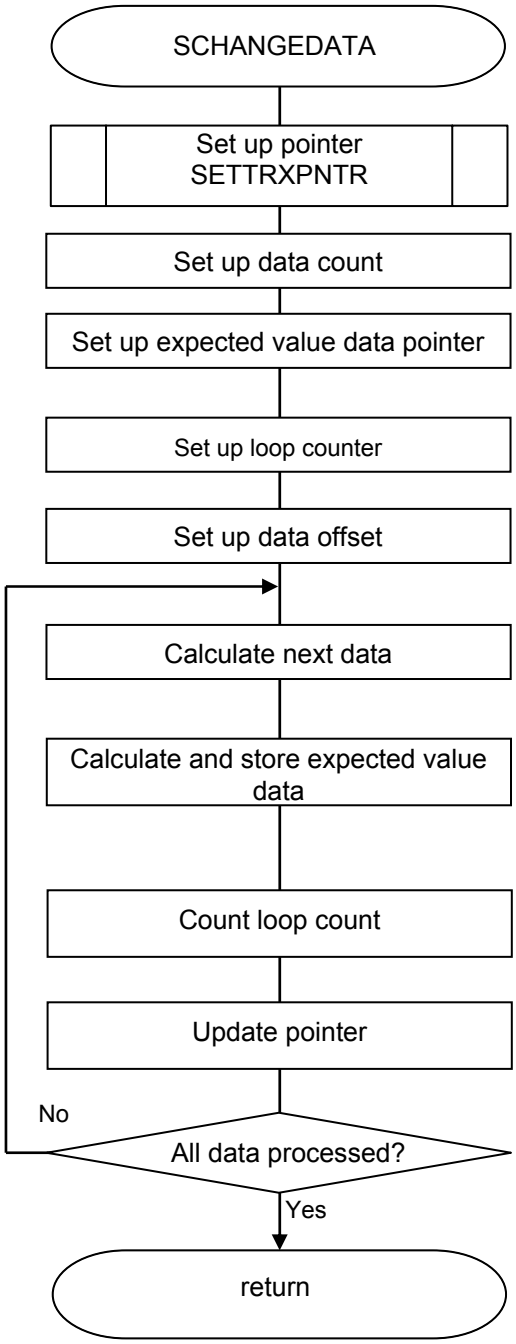


Figure 5.22 Continuous Data Reception Processing (2/2)

5.7.15 Data Update Processing

Figure 5.23 shows the flowchart for the data update processing.



Set up pointer associated with the slave to be accessed.
HL register ←Pointer for transmit data

Load the A register with the data count.
A register ← [HL + CRXDTNO + CTXDTNO + 1]

Load the DE register with the pointer to the area for storing the expected value data.
DE register ← #LOWW RCMPDATA

Load the B register with the loop count (data count).
B register ← A register

Load the X register with the offset to the next data.
A register ← A register

Add offset to the transmit data to calculate the next data.

```
MOV   A,   X
ADD   A,   [HL]
```

Take the complement of transmit data and select the expected value data.

```
XCH   A,   [HL]    : Store next data and read old data.
XOR   A,   #0FFH   : Calculate expected value data.
MOV   [DE], A      : Store expected value data.
```

Decrement loop counter by 1.

```
DEC   B
```

Update the data pointer.

```
INCW  HL
INCW  DE
```

Figure 5.23 Data Update Processing

5.7.16 Receive Data Check Processing

Figure 5.24 shows the flowchart for receive data check processing.

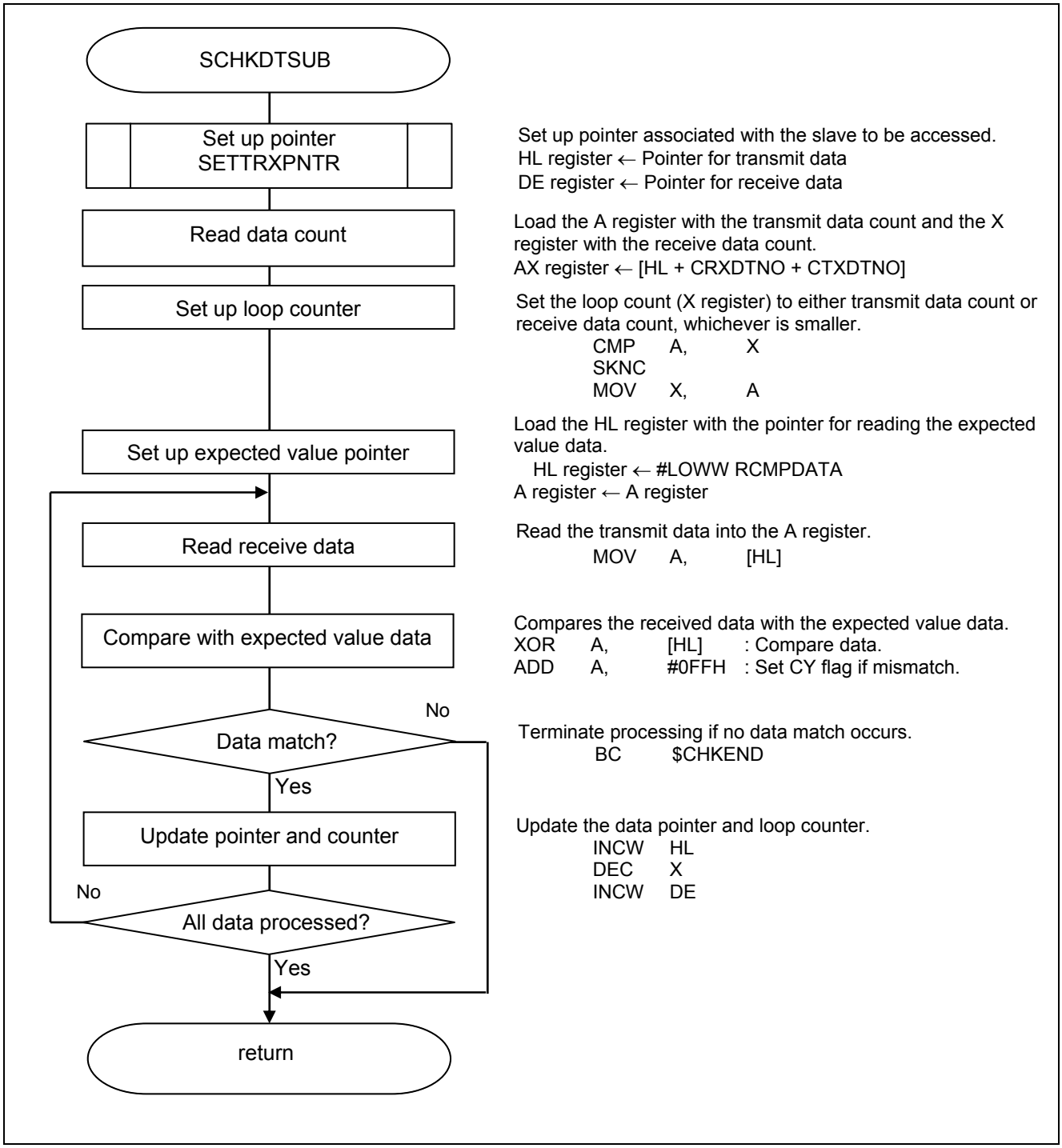


Figure 5.24 Receive Data Check Processing

5.7.17 Data Pointer Setup Processing

Figure 5.25 shows the flowchart for data pointer setup processing.

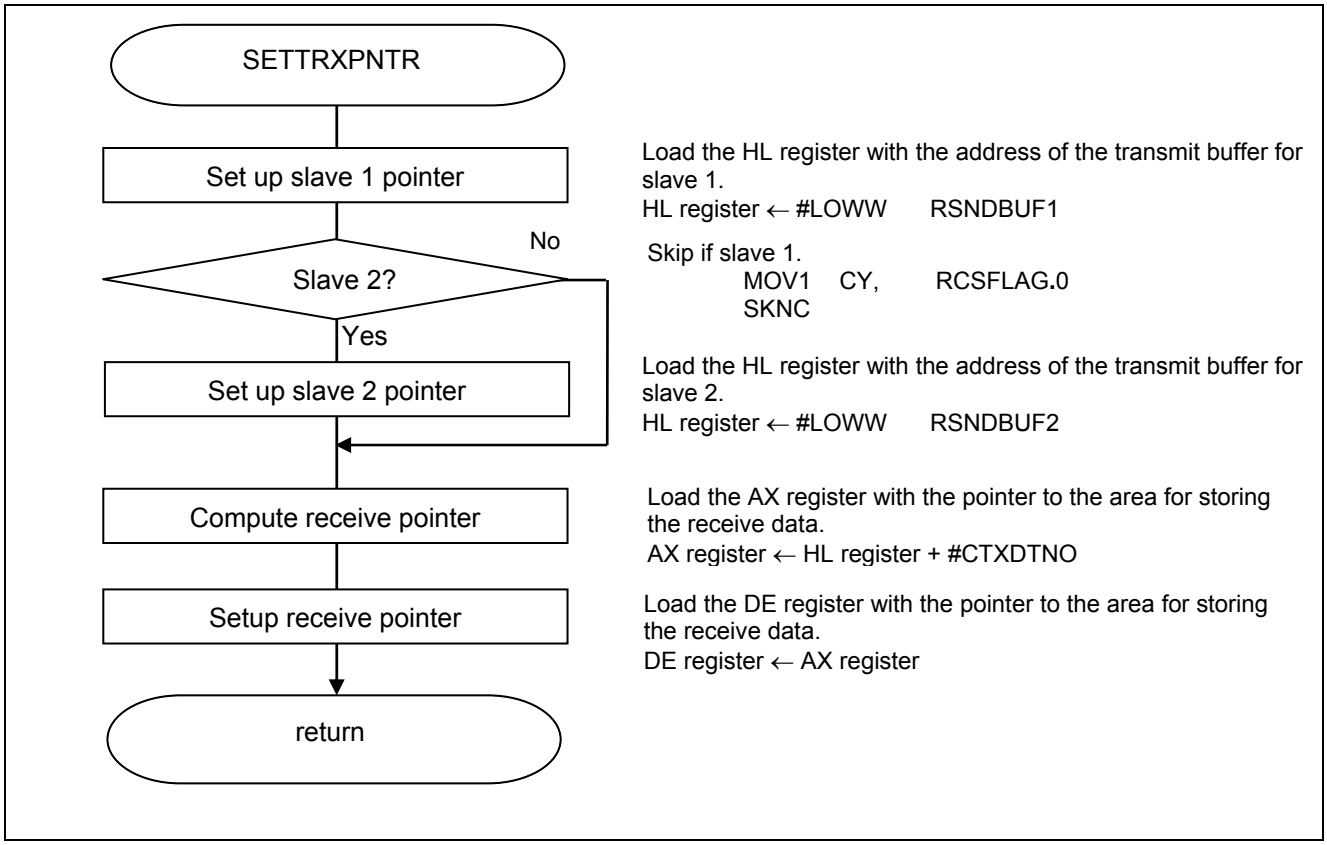


Figure 5.25 Data Pointer Setup Processing

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.

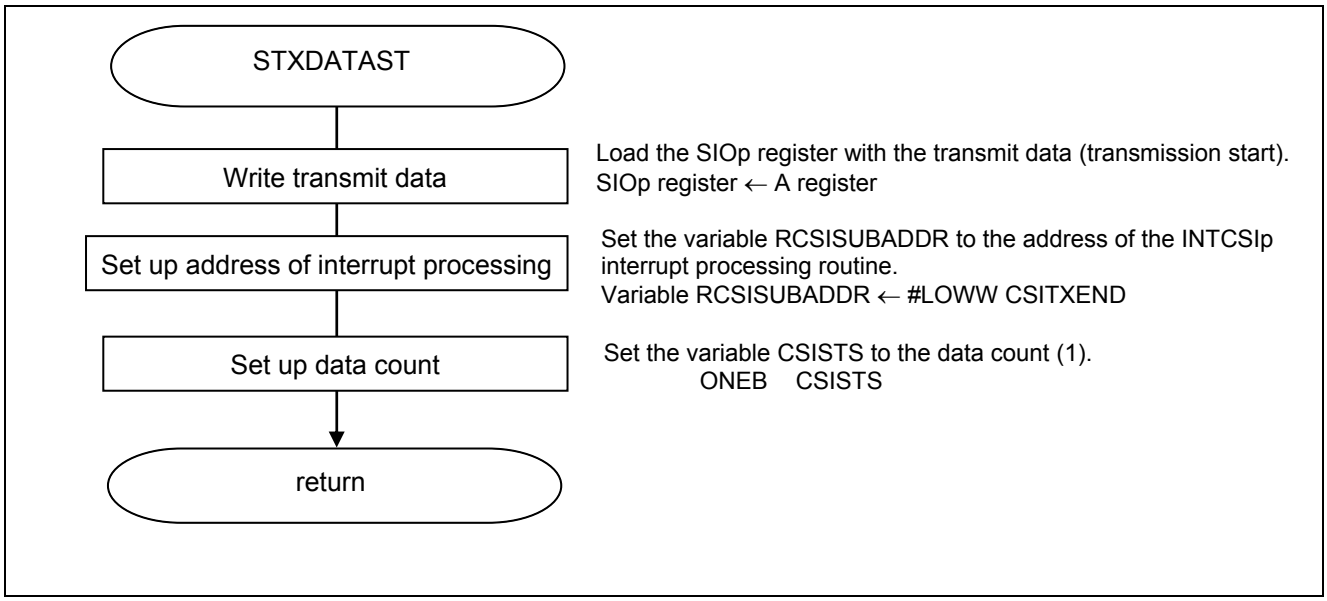


Figure 5.26 1-character Transmission Start Processing

5.7.19 1-character Transmission End Wait Processing

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

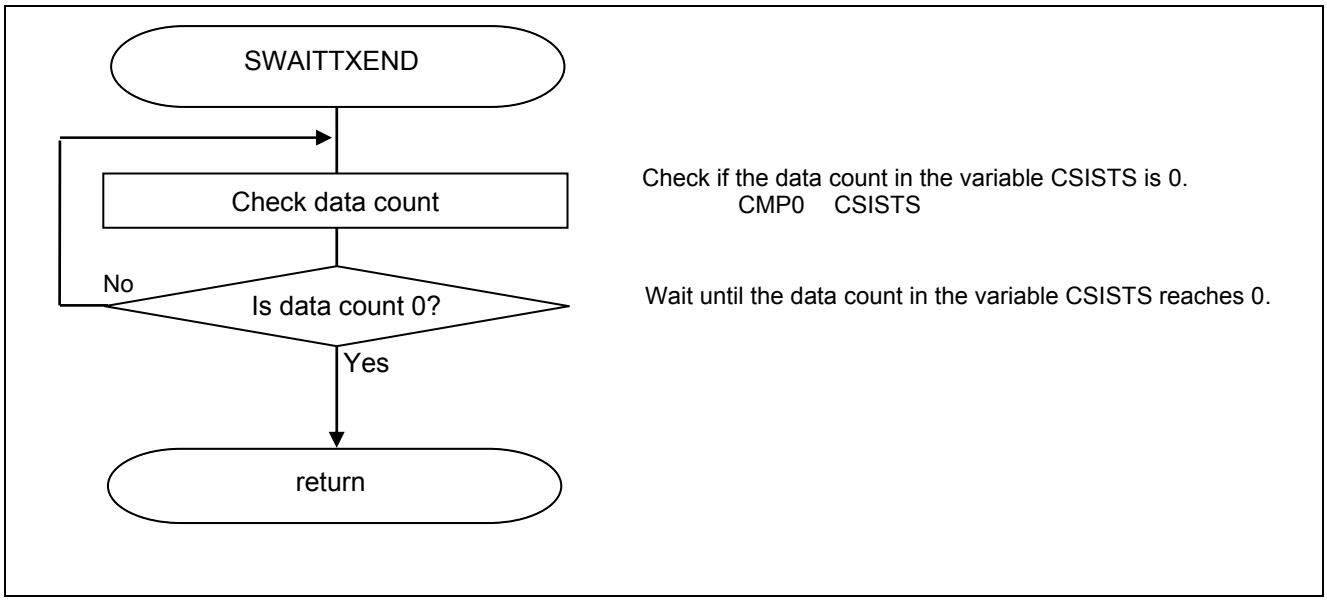


Figure 5.27 1-character Transmission End Wait Processing

5.7.20 1-character Reception Processing

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

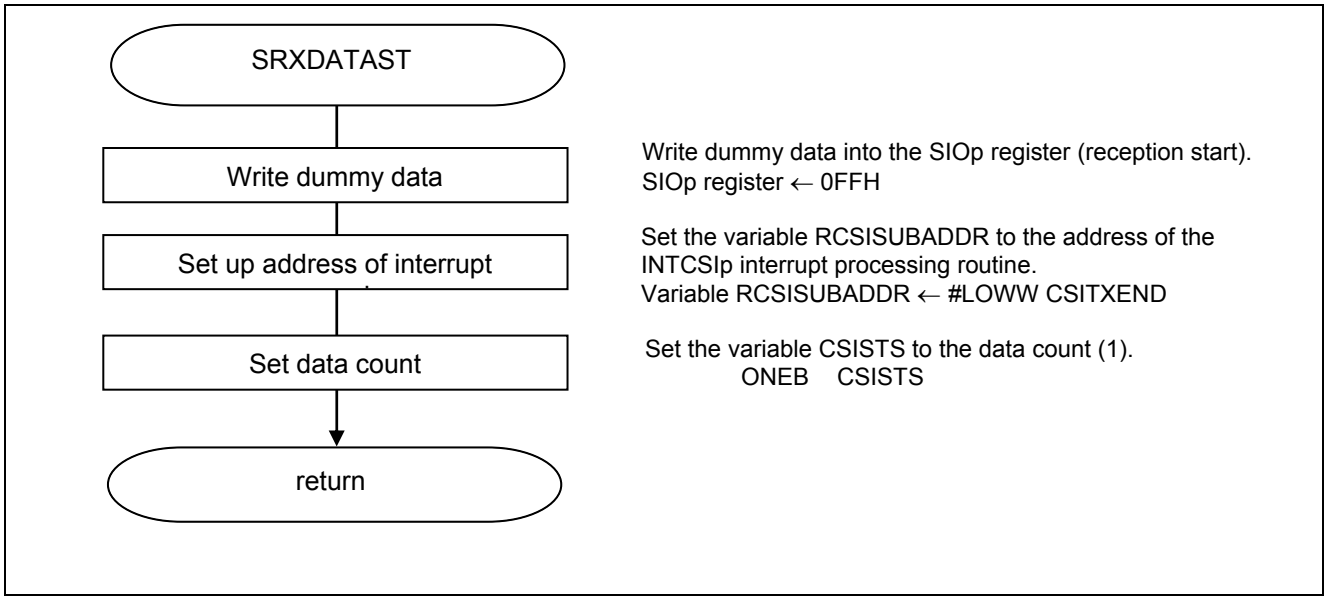


Figure 5.28 1-character Reception Processing

5.7.21 1-character Reception End Wait Processing

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

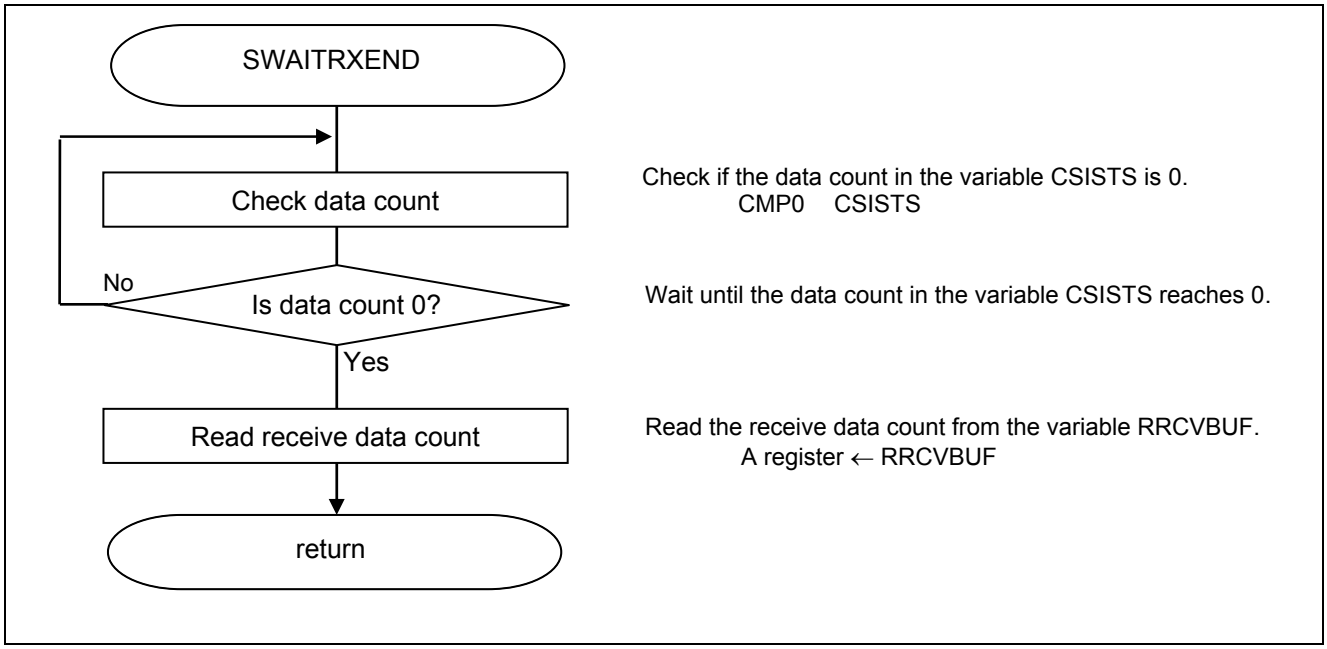


Figure 5.29 1-character Reception End Wait Processing

5.7.22 1-character Transfer State Check Processing

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

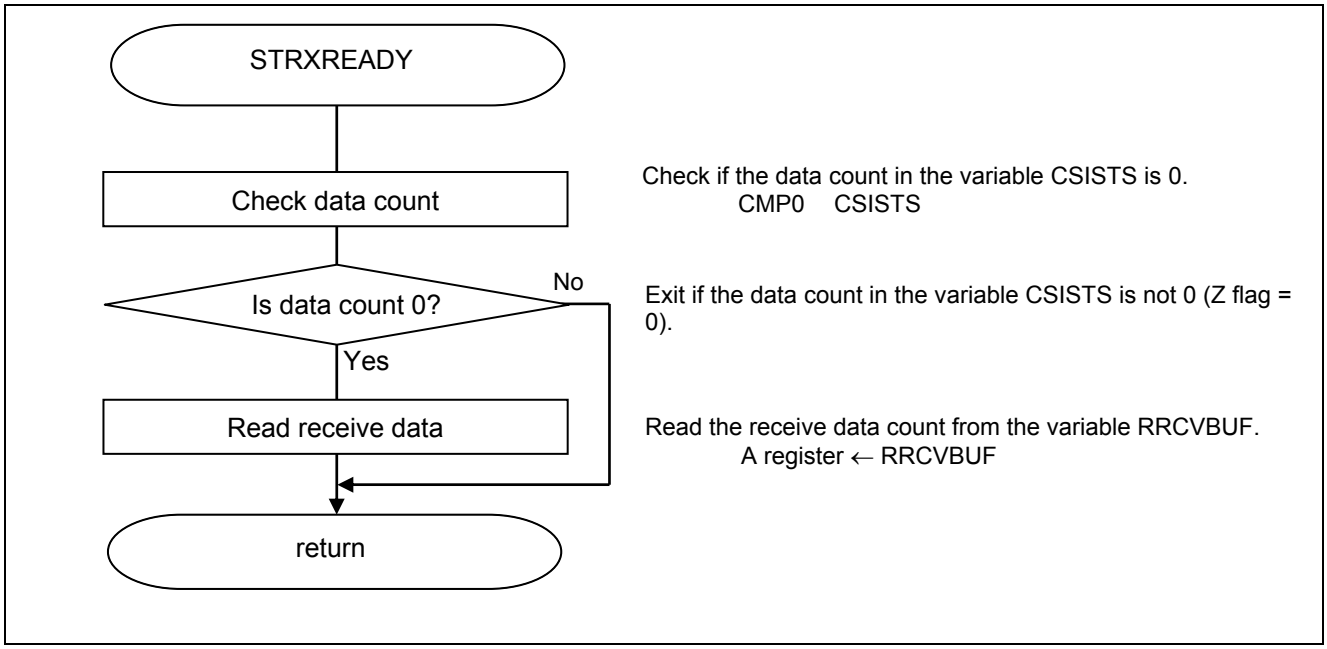


Figure 5.30 1-character Transfer State Check Processing

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.

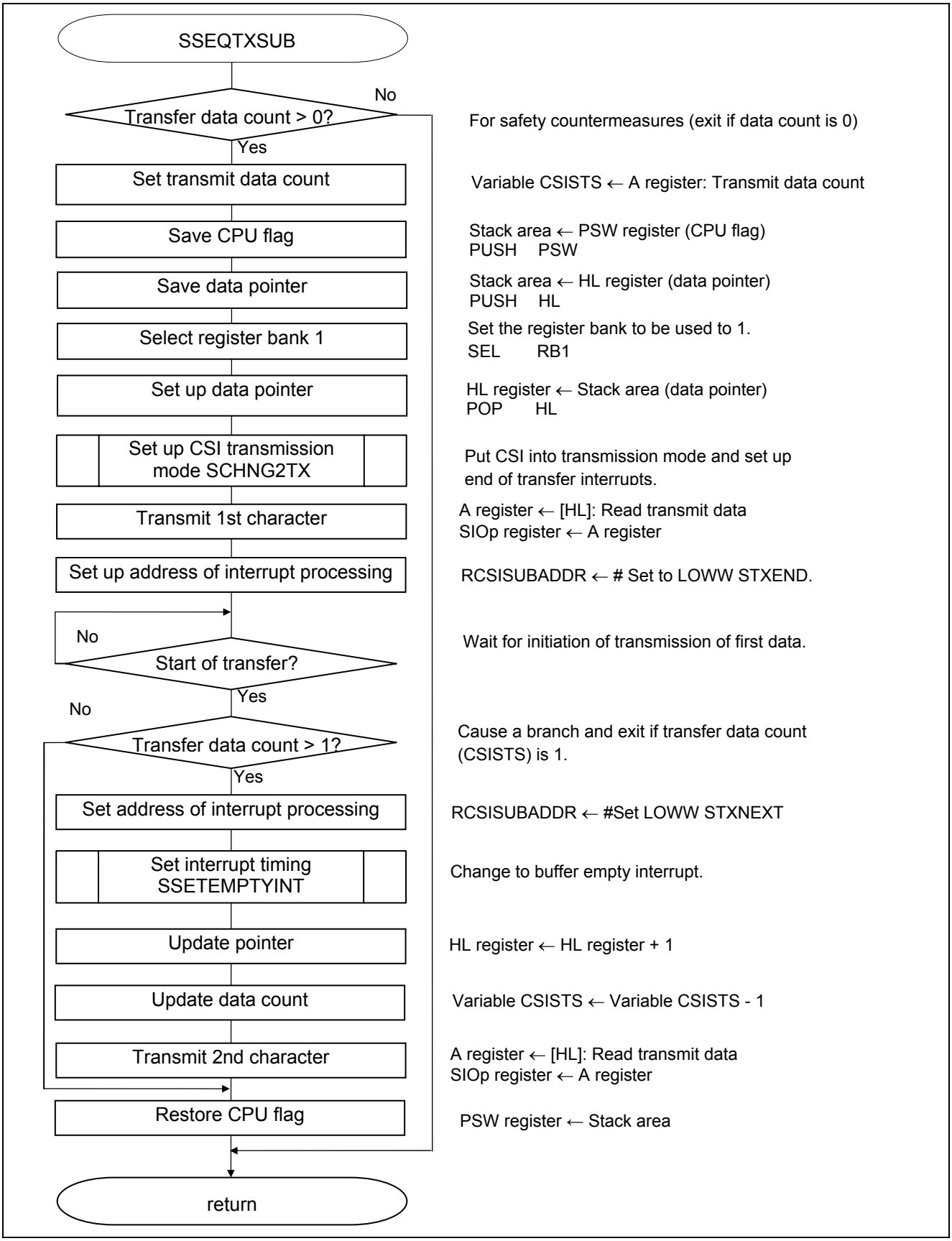


Figure 5.31 Continuous Transmission Start Processing

Checking communication status

- Serial status register mn (SSRmn/SSRmnL)
  Reads CSIp communication status.

Symbol: SSRmn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	TSF mn	BFF mn	0	0	FEF mn	PEF mn	OVF mn
0	0	0	0	0	0	0	0	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.**

Caution: For details on the register setup procedures, refer to RL78/G12 User's Manual: Hardware.

5.7.24 Continuous Reception Startup Processing

Figure 5.32 shows the flowchart for continuous reception startup processing.

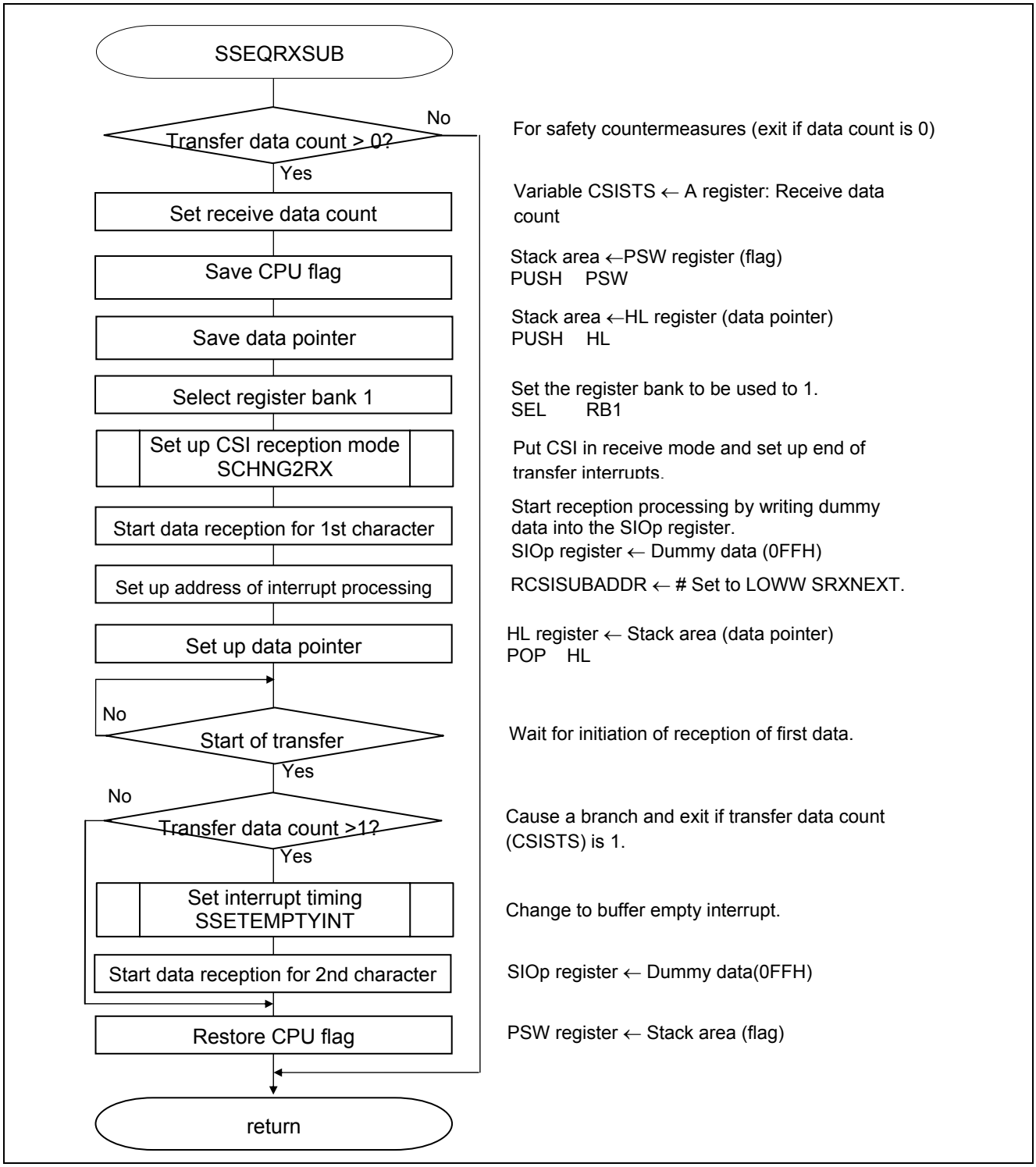


Figure 5.32 Continuous Reception Startup Processing

5.7.25 Continuous Transmission/Reception Startup Processing

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

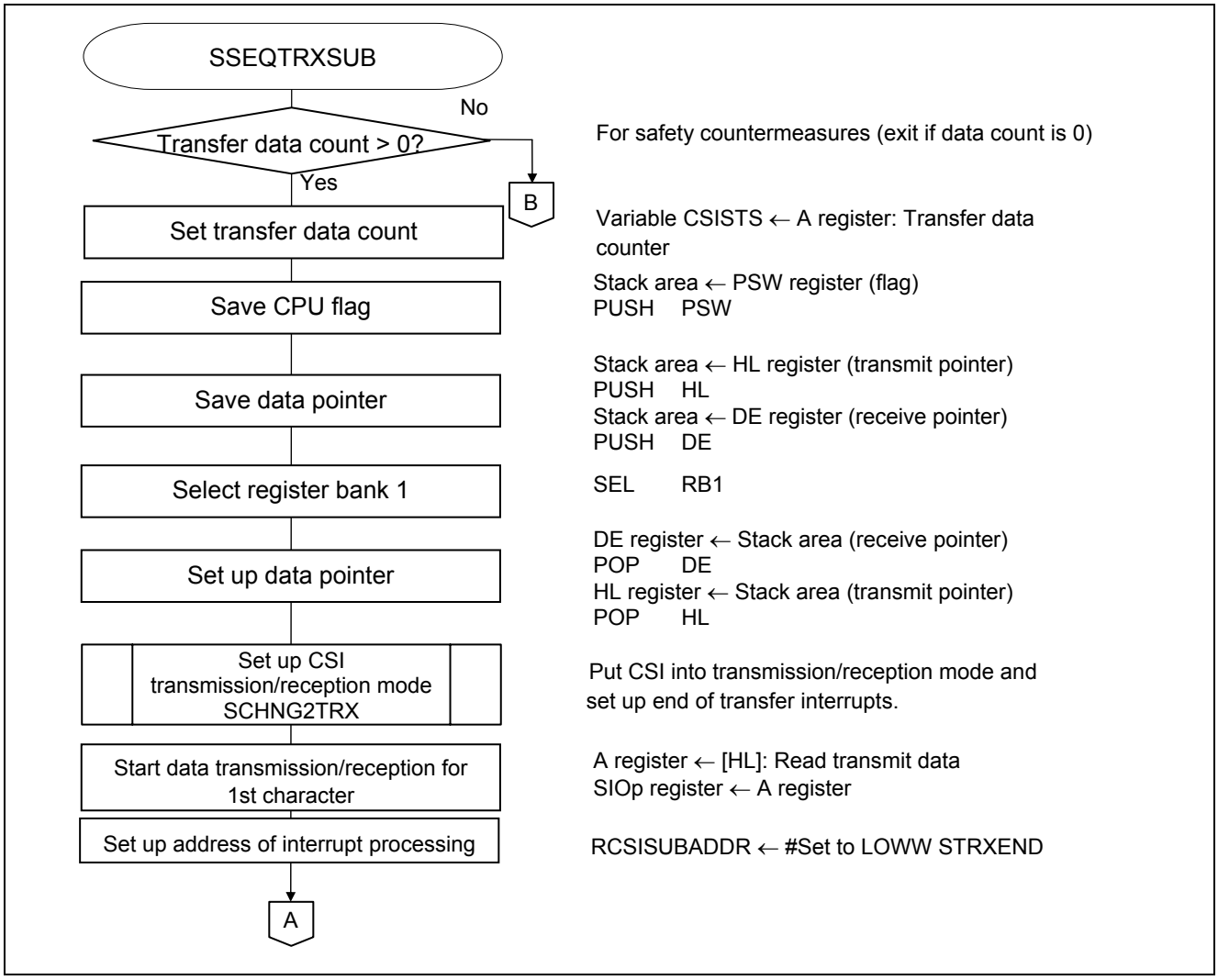


Figure 5.33 Continuous Transmission/Reception Startup Processing (1/2)

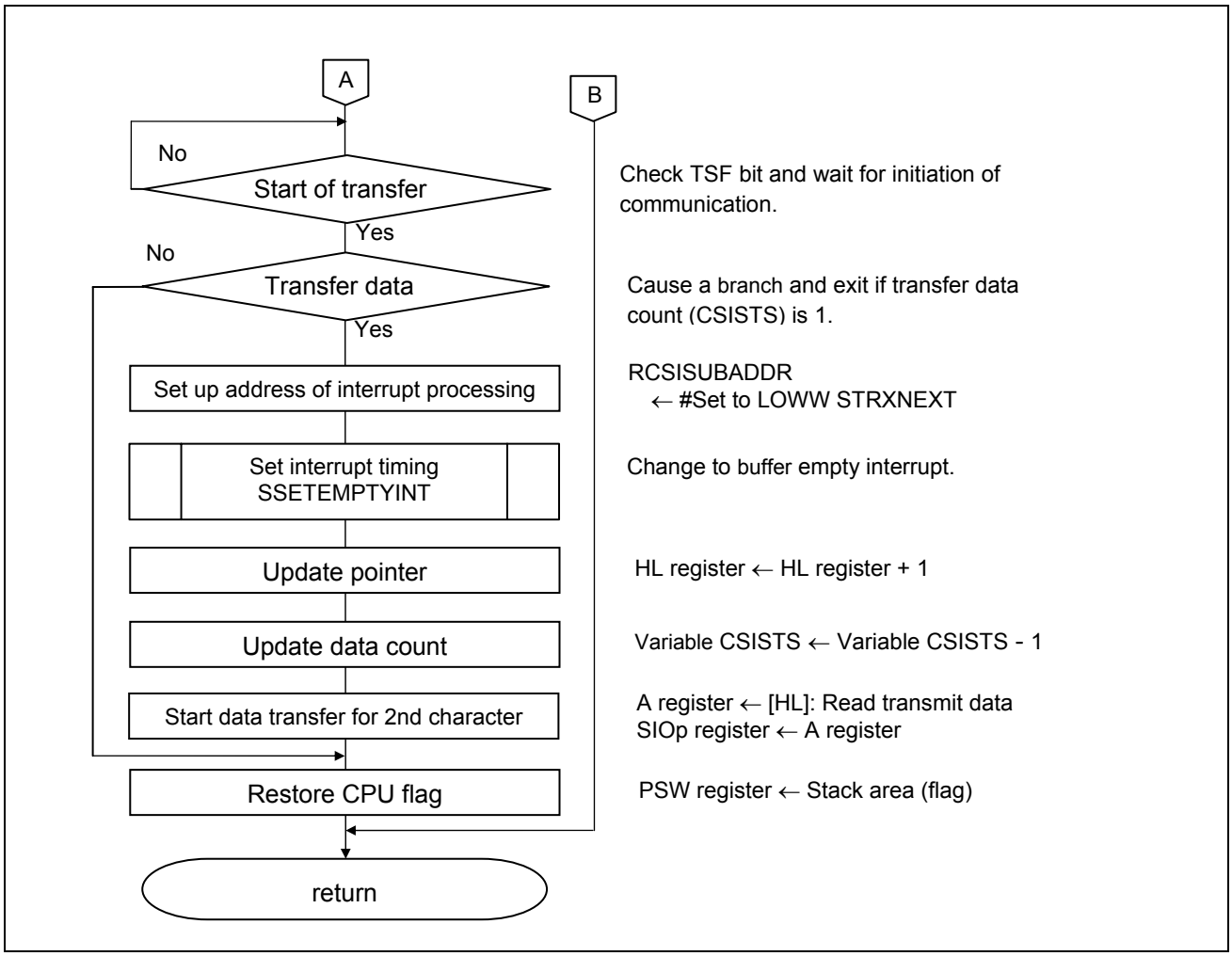


Figure 5.34 Continuous Transmission/Reception Startup Processing (2/2)

5.7.26 Continuous Transfer End Wait Processing

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

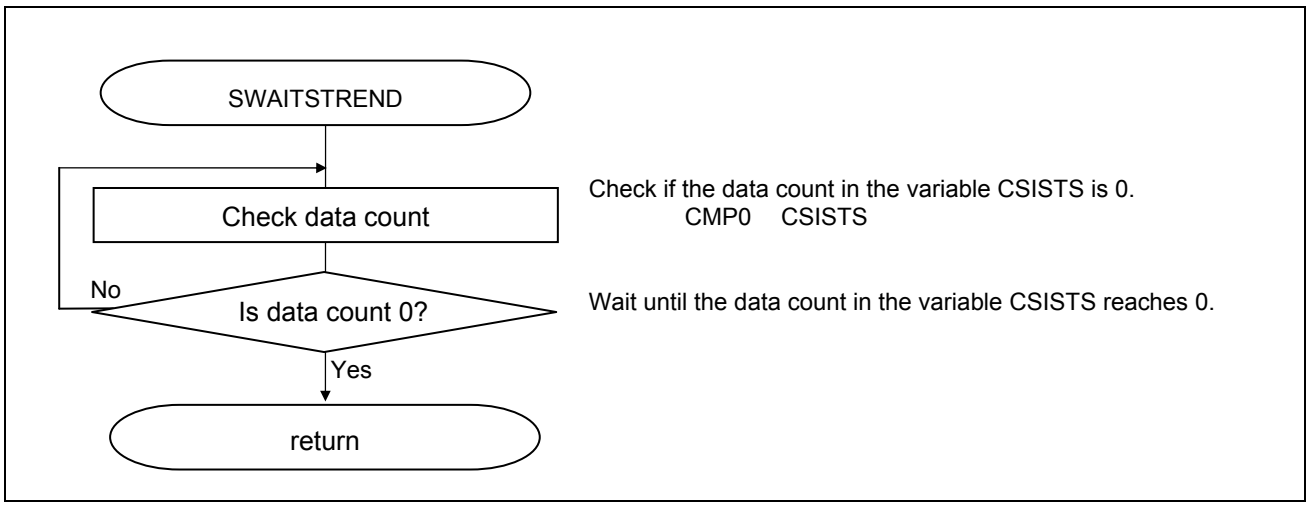


Figure 5.35 Continuous Transfer End Wait Processing

5.7.27 Transfer End Interrupt Setup Processing

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

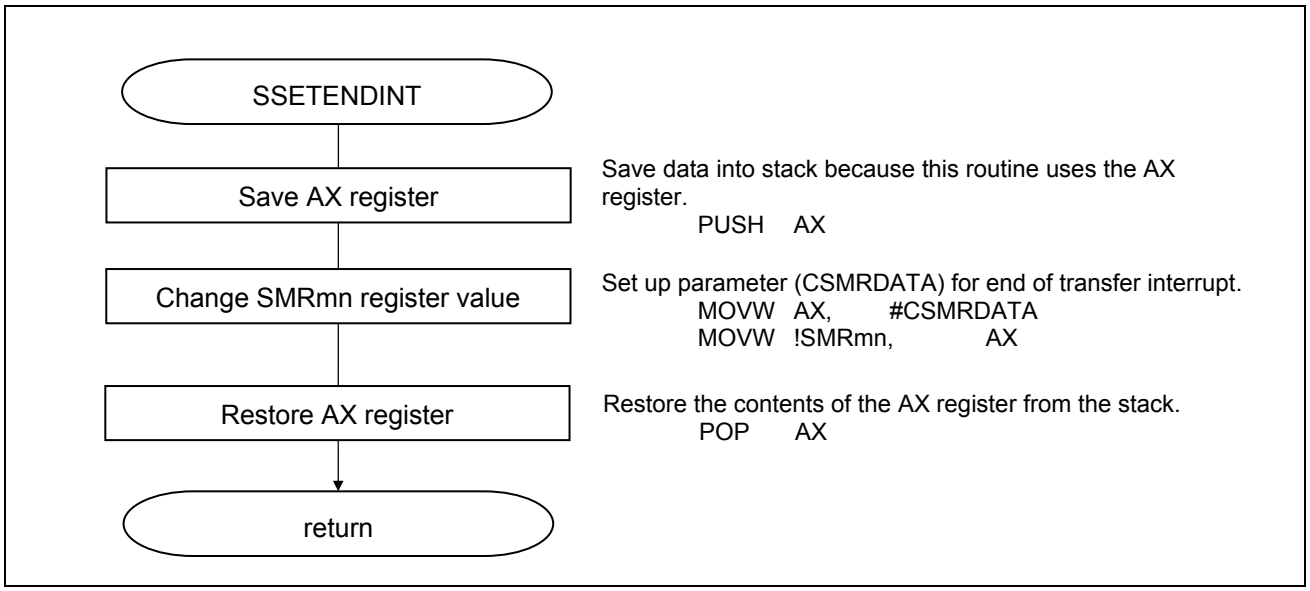


Figure 5.36 Transfer End Interrupt Setup Processing

Setting up the channel operating mode

- Serial mode register mn (SMRmn)
  Interrupt source and end of transfer interrupt

Symbol: SMRmn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS mn	CCS mn	0	0	0	0	0	STS mn	0	0	1	0	0	MD mn2	MD mn1	MD mn0
0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	**0**

Bit 0

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

Caution: For details on the register setup procedures, refer to RL78/G12 User's Manual: Hardware.

5.7.28 Buffer Empty Interrupt Setup Processing

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

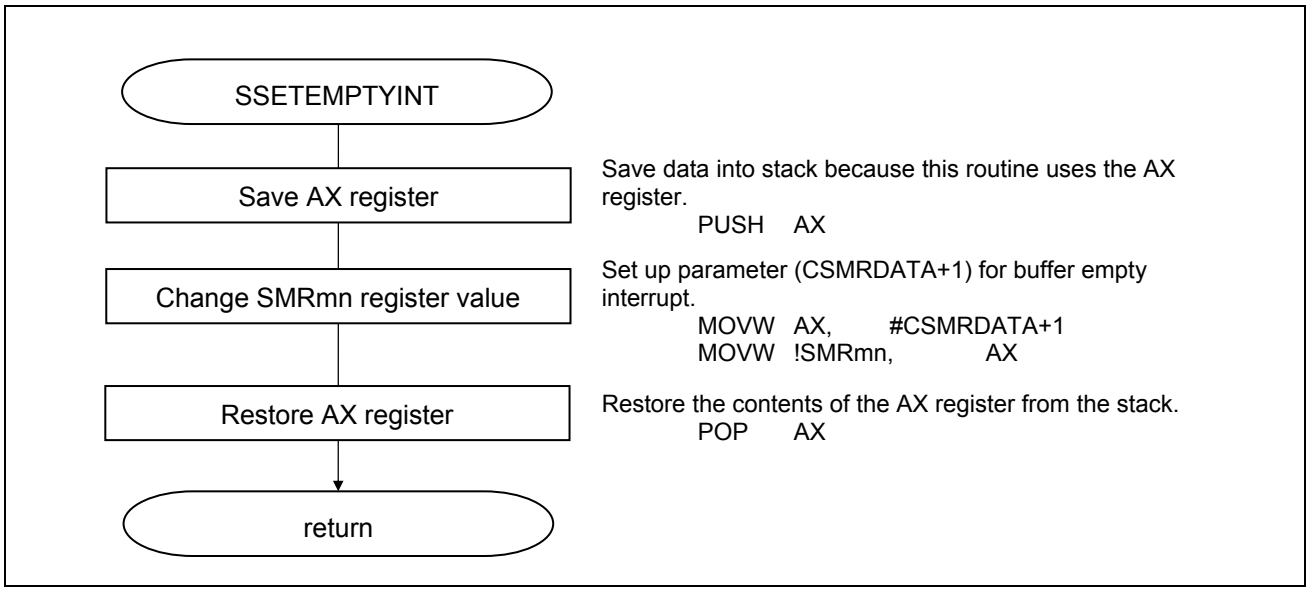


Figure 5.37 Buffer Empty Interrupt Setup Processing

Setting up the channel operating mode

- Serial mode register mn (SMRmn)
  Interrupt source and buffer empty interrupt

Symbol: SMRmn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS mn	CCS mn	0	0	0	0	0	STS mn	0	0	1	0	0	MD mn2	MD mn1	MD mn0
0	**0**	0	0	0	0	0	**0**	0	0	1	0	0	**0**	**0**	**1**

Bit 0

MDmn0	Channel n Interrupt Source Select
0	**Transfer end interrupt**
1	**Buffer empty interrupt**

Caution: For details on the register setup procedures, refer to RL78/G12 User's Manual: Hardware.

5.7.29 Transmission Mode Setup Processing

Figure 5.38 shows the flowchart for transmission mode setup processing.

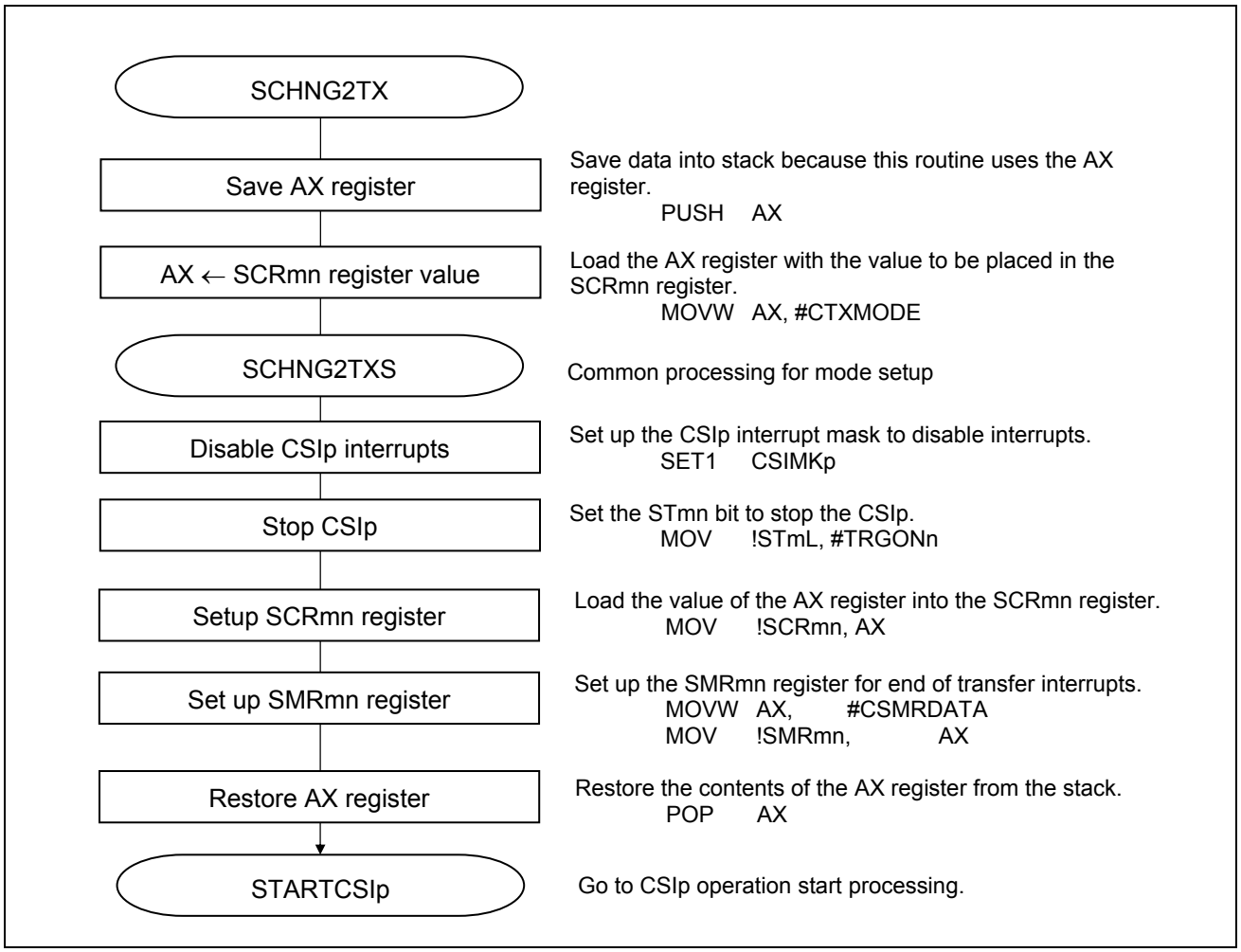


Figure 5.38 Transmission Mode Setup Processing

Interrupt setting (20-/24- pin products)

- Interrupt request flag register (MK0H)
  Interrupt mask setting

Symbol: MK0H (20-/24- pin products)

7	6	5	4	3	2	1	0
TMMK01	TMMK00	IICAMK0	TMMK 03H	TMMK 01H	SREMK0	SRMK0 CSIMK01 IICMK01	STMK0 CSIMK00 IICMK00
x	x	x	x	x	x	**0/1**	**0/1**

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

Caution: For details on the register setup procedures, refer to RL78/G12 User's Manual: Hardware.

Transiting to communication stopped state

- Serial channel startup register m (STm/STmL)
  Stop operation.

Symbol: STm

								SSmL							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	SSm3 Note	SSm2 Note	SSm1	SSm0
0	0	0	0	0	0	0	0	0	0	0	0	**0/1**	**0/1**	**0/1**	**0/1**

Note: 30-pin products only

Bits 3 to 0

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

Setting up the channel operating mode

- Serial mode register mn (SMRmn)
  Interrupt source and end of transmit interrupt

Symbol: SMRmn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS mn	CCS mn	0	0	0	0	0	STS mn	0	0	1	0	0	MD mn2	MD mn1	MD mn0
0	**0**	0	0	0	0	0	**0**	0	0	1	0	0	**0**	**0**	**0**

Bit 0

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

Setting up channel communication mode

- Serial communication operation register mn (SCRmn)
  Operating mode

Symbol: SCRmn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXE mn	RXE mn	DAP mn	CKP mn	0	EOC mn	PTC mn1	PTC mn0	DIR mn	0	SLC mn1	SLC mn0	0	1	DLS mn1	DLS mn0
0/1	**0/1**	**0**	**0**	0	**0**	**0**	**0**	**0**	0	**0**	**0**	0	1	**1**	**1**

Bits 15 and 14

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

Caution: For details on the register setup procedures, refer to RL78/G12 User's Manual: Hardware.

5.7.30 Reception Mode Setup Processing

Figure 5.39 shows the flowchart for reception mode setup processing.

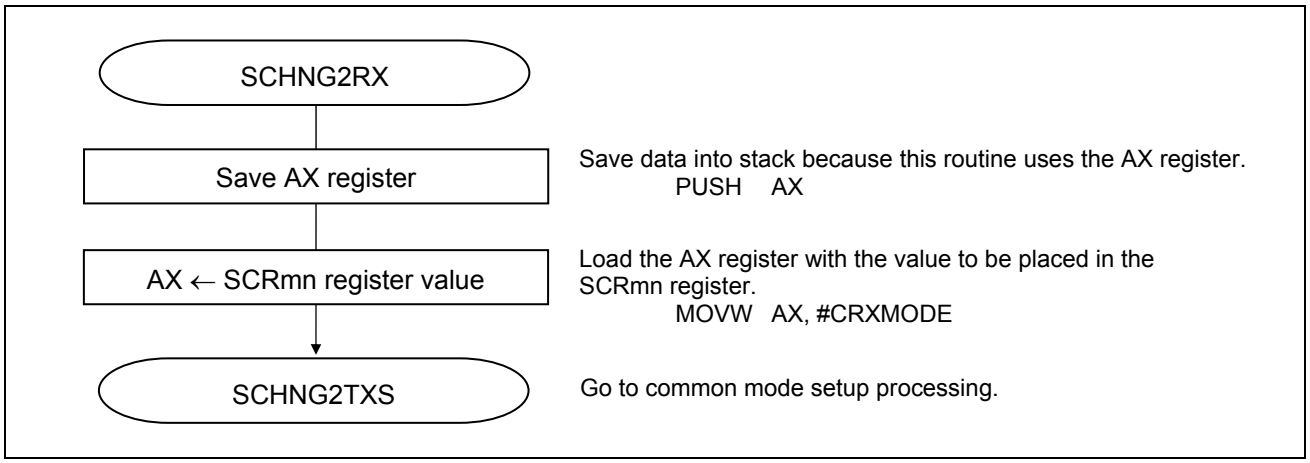


Figure 5.39 Reception Mode Setup Processing

5.7.31 Transmission/Reception Mode Setup Processing

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

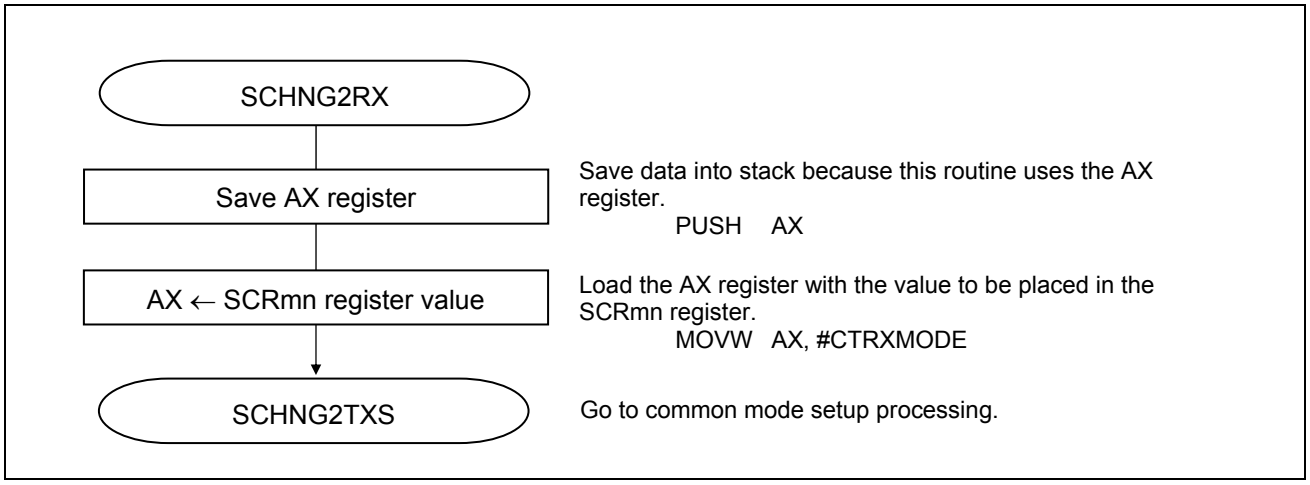


Figure 5.40 Transmission/Reception Mode Setup Processing

5.7.32 CSIp Communication Enable Processing

Figure 5.41 shows the flowchart for CSIp communication enable processing.

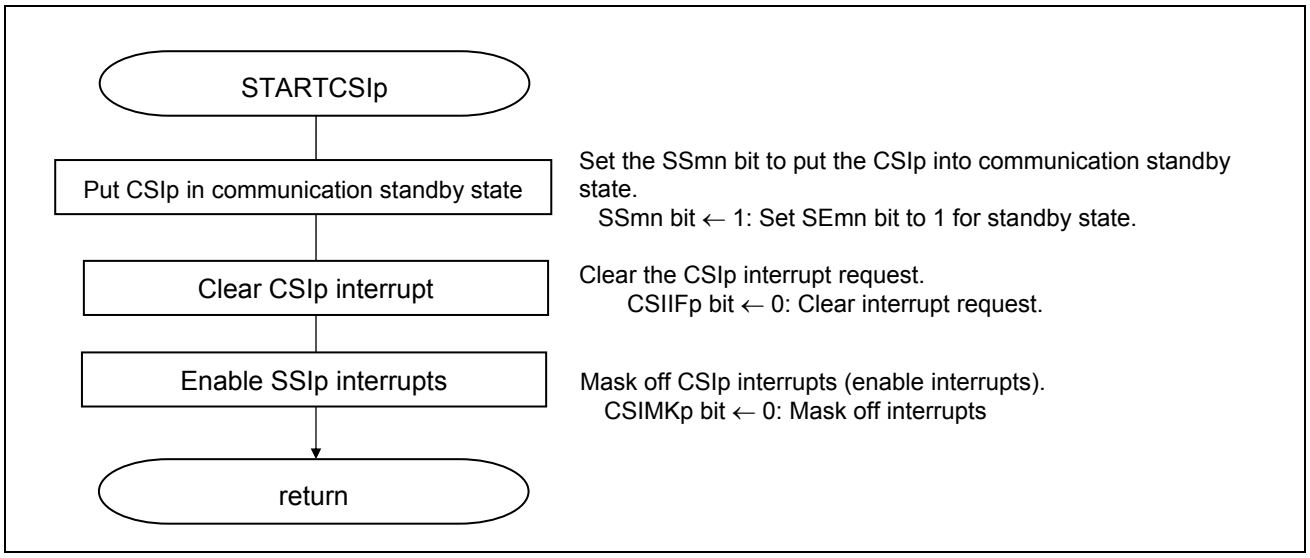


Figure 5.41 CSIp Communication Enable Processing

Transiting to communication standby state

- Serial channel startup register m (SSm/SSmL)
  Start operation.

Symbol: SSm

								SSmL							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	SSm3 Note	SSm2 Note	SSm1	SSm0
0	0	0	0	0	0	0	0	0	0	0	0	**0/1**	**0/1**	**0/1**	**0/1**

Note: 30-pin products only

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.**

Caution: For details on the register setup procedures, refer to RL78/G12 User's Manual: Hardware.

Interrupt setting (20-/24- pin products)

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

Symbol: IF0H (20-/24- pin products)

7	6	5	4	3	2	1	0
TMIF01	TMIF00	IICAIF0	TMIF03H	TMIF01H	SREIF0	SRIF0 CSIIF01 IICIF01	STIF0 CSIIF00 IICIF00
x	x	x	x	x	**0**	**0**	**0**

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

Symbol MK0H (20-/24- pin products)

7	6	5	4	3	2	1	0
TMMK01	TMMK00	IICAMK0	TMMK 03H	TMMK 01H	SREMK0	SRMK0 CSIMK01 IICMK01	STMK0 CSIMK00 IICMK00
x	x	x	x	x	x	**0/1**	**0/1**

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

Caution: For details on the register setup procedures, refer to RL78/G12 User's Manual: Hardware.

5.7.33 CSIp Communication Termination Processing

Figure 5.42 shows the flowchart for CSIp communication termination processing.

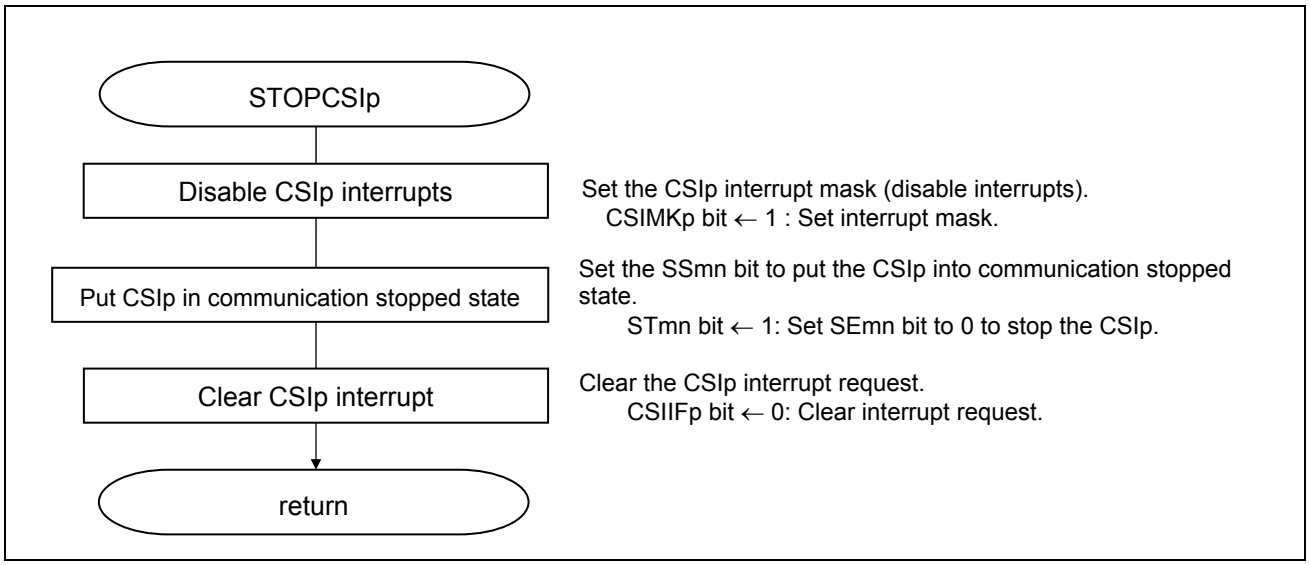


Figure 5.42 CSIp Communication Termination Processing

Transiting to communication stopped state

- Serial channel startup register m (STm/STmL)
  Stop operation.

Symbol: STm

								SSmL							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	SSm3 Note	SSm2 Note	SSm1	SSm0
0	0	0	0	0	0	0	0	0	0	0	0	**0/1**	**0/1**	**0/1**	**0/1**

Note: 30-pin products only

Bits 3 to 0

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

Caution: For details on the register setup procedures, refer to RL78/G12 User's Manual: Hardware.

Interrupt setting (20-/24- pin products)

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

Symbol: IF0H (20-/24- pin products)

7	6	5	4	3	2	1	0
TMIF01	TMIF00	IICAIF0	TMIF03H	TMIF01H	SREIF0	SRIF0 CSIIF01 IICIF01	STIF0 CSIIF00 IICIF00
x	x	x	x	x	**0**	**0**	**0**

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

Symbol: MK0H (20-/24- pin products)

7	6	5	4	3	2	1	0
TMMK01	TMMK00	IICAMK0	TMMK 03H	TMMK 01H	SREMK0	SRMK0 CSIMK01 IICMK01	STMK0 CSIMK00 IICMK00
x	x	x	x	x	x	**0/1**	**0/1**

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

Caution: For details on the register setup procedures, refer to RL78/G12 User's Manual: Hardware.

5.7.34 CSIp Interrupt Startup Processing

Figure 5.43 shows the flowchart for CSIp interrupt startup processing.

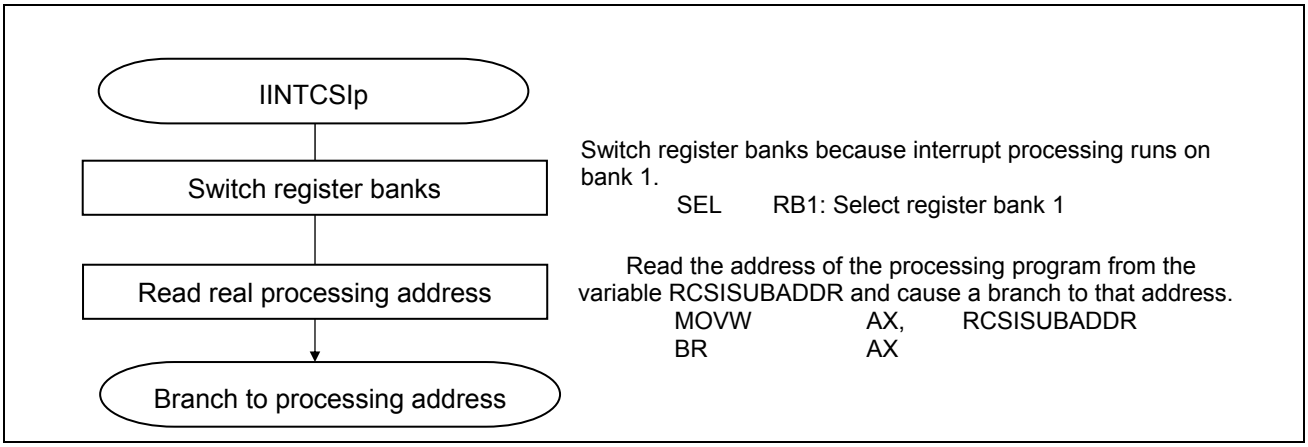


Figure 5.43 CSIp Interrupt Startup Processing

5.7.35 1-character Transfer End Interrupt Processing

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

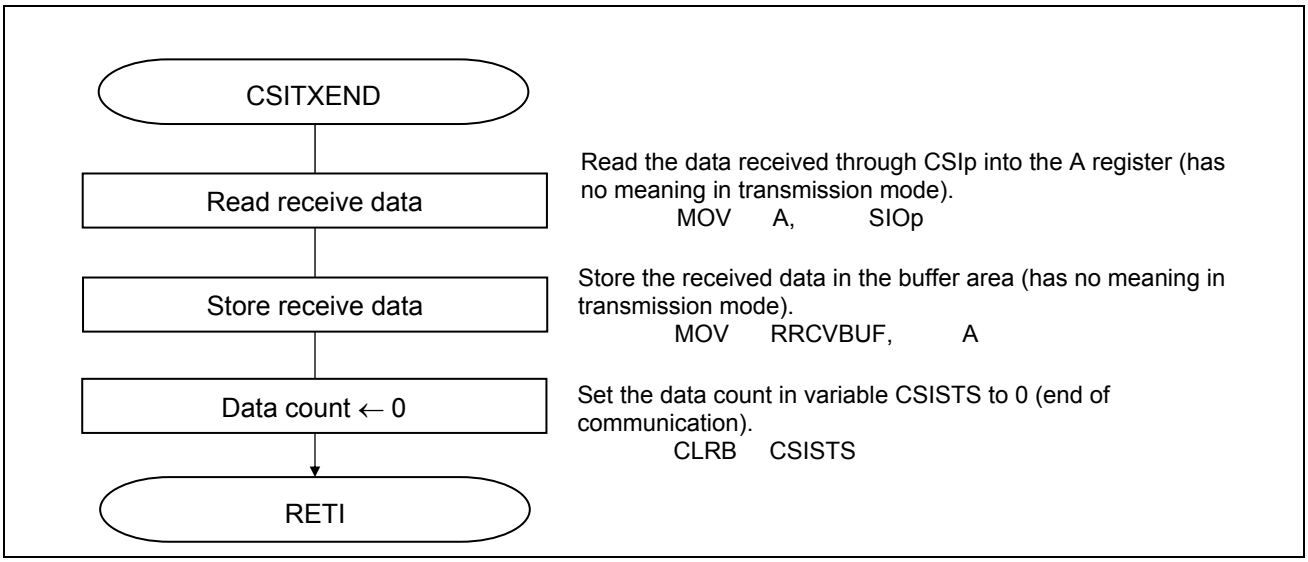


Figure 5.44 1-character Transfer End Interrupt Processing

5.7.36 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.

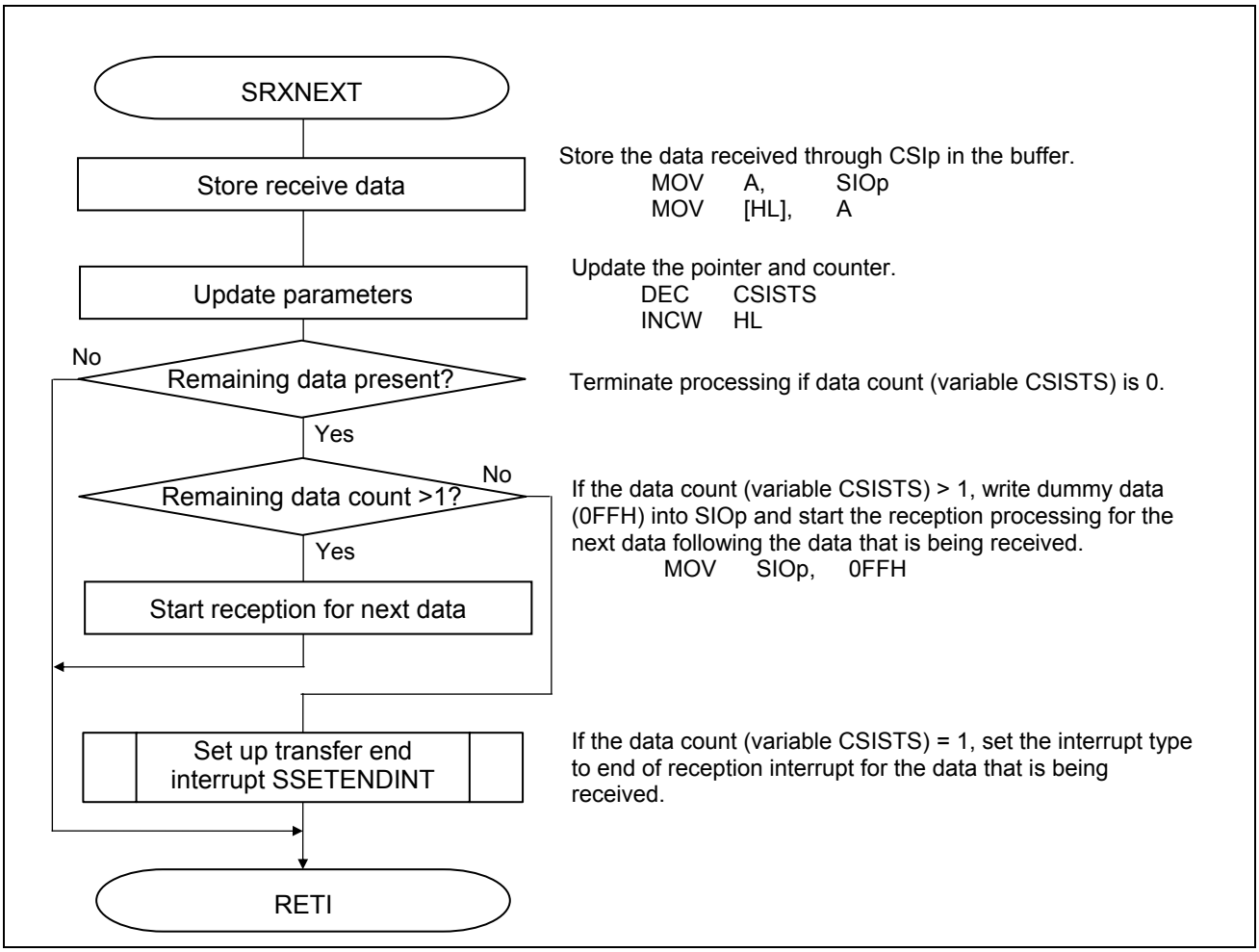


Figure 5.45 1-character Transfer End Interrupt Processing (Continuous Reception Mode)

5.7.37 Buffer Empty Interrupt Processing in Continuous Transmission Mode

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

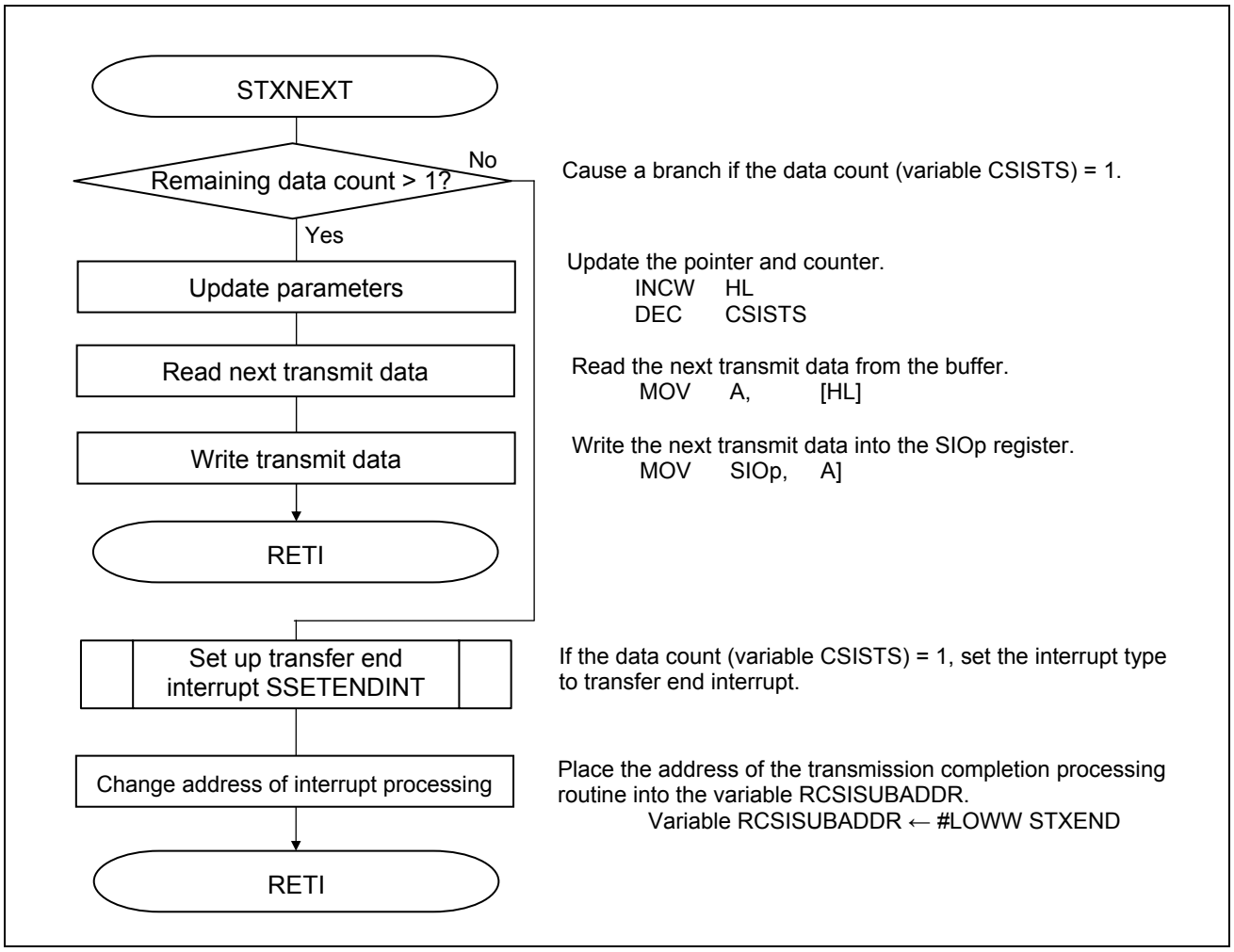


Figure 5.46 Buffer Empty Interrupt Processing (Continuous Transmission Mode)

5.7.38 Transmission End Interrupt Processing in Continuous Transmission Mode

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

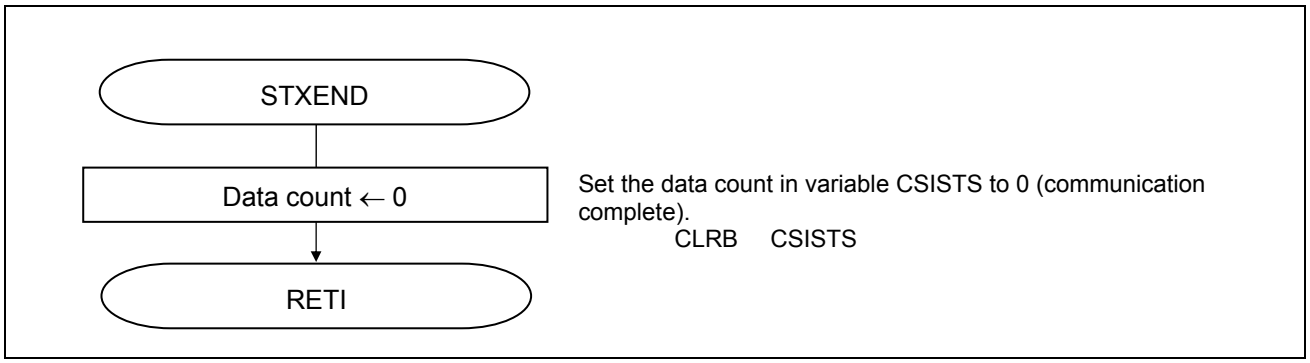


Figure 5.47 Transmission End Interrupt Processing (Continuous Transmission Mode)

5.7.39 Buffer Empty Interrupt Processing in Continuous Transmission Mode

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

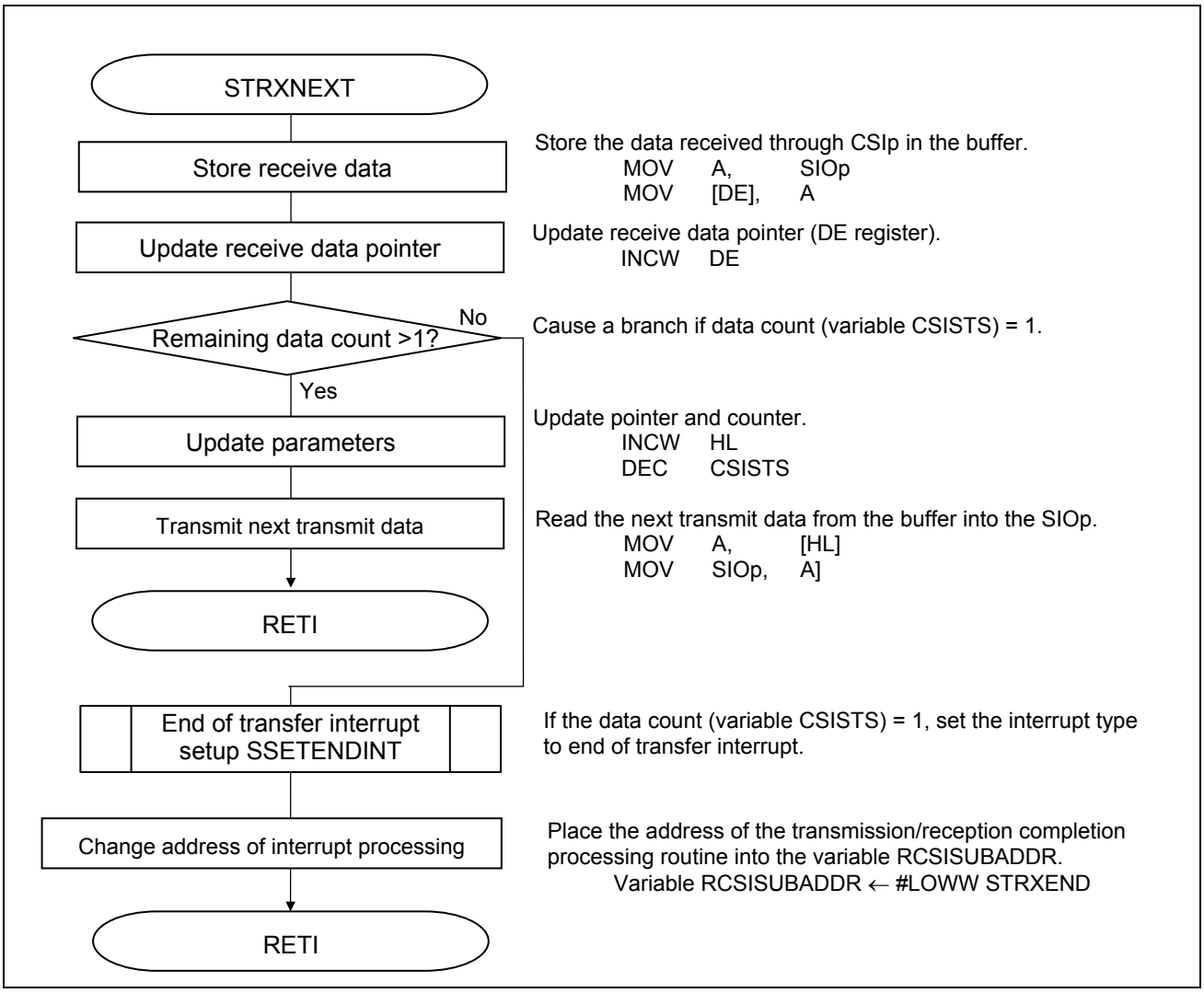


Figure 5.48 Buffer Empty Interrupt Processing (Continuous Transmission Mode)

5.7.40 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.

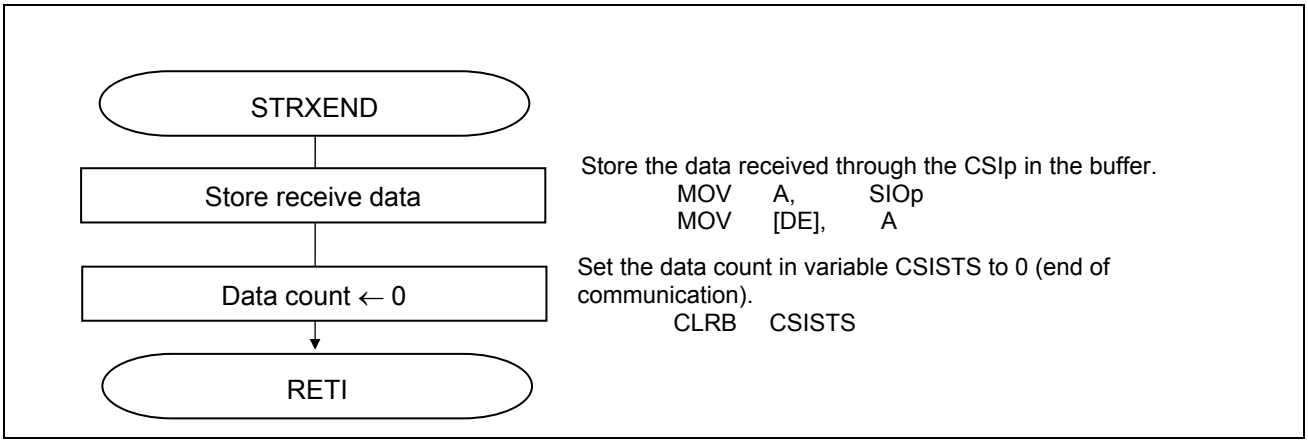


Figure 5.49 Transfer End Interrupt Processing (Continuous Transmission/Reception Mode)

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 24000 (24 MHz).
- CSI communication speed (BAUDRATE) in kbps: The initial value is 1000 (1 Mbps).
- Microcontroller to be used: The initial value is R5F1026.
- CSI channel to be used: The initial value is CSI00.

6.3 Changing the Transfer Rate

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

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

CLKFREQ     EQU    24000          ; 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 CubeSuite+ and specify the desired device in the project. For details, refer to RL78 Family CubeSuite+ Startup Guide (R01AN1232E) Application Note.

The microcontroller to be used is defined as shown below. Only the line that has no leading semicolon (';') is valid. To change the device to be used, append a semicolon to the beginning of the currently valid line and delete the leading semicolon at the beginning of the line defining the desired device.

```
;*********************************************************************
;
;   device select
;
;*********************************************************************
$SET( R5F1026 )    ; 20 pins with data flash memory

;$SET( R5F1036 )   ; 20 pins without data flash memory
;$SET( R5F1027 )   ; 24 pins with data flash memory
;$SET( R5F1037 )   ; 24 pins without data flash memory
;$SET( R5F102A )   ; 30 pins with data flash memory
;$SET( R5F103A )   ; 30 pins without data flash memory
```

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( R5F1026 : R5F1027 )
;=====================================================
;   for R5F1026 and R5F1027
;   select CSI00 or CSI01
;=====================================================
$SET( CSI00 )       ; CSI00 is selected
;$SET( CSI01 )      ; CSI01 is not selected now
```

Definition for 20-/24-pin products with on-chip data flash ROM

```
$ELSEIF( R5F1036 : R5F1037 : R5F103A )
;=====================================================
;   for R5F1036 , R5F1037 and R5F103A
;   CSI00 only
;=====================================================
$SET( CSI00 )       ; CSI00 is selected
```

Definition for products without on-chip data flash ROM

```
$ELSE
;=====================================================
;   for R5F102A
;   select CSI00 , CSI11 or CSI20
;=====================================================
$SET( CSI00 )       ; CSI00 is selected
;$SET( CSI11 )      ; CSI11 is not selected now
;$SET( CSI20 )      ; CSI20 is not selected now
$ENDIF
```

Definition for 30-pin products with on-chip data flash ROM

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 )
SAUmEN        EQU    SAU0EN        ; Peripheral enable register
SPSmL         EQU    SPS0L         ; Serial clock select register
SMRmn         EQU    SMR00         ; Serial mode register
SCRmn         EQU    SCR00         ; Serial communication operation setting register
SDRmn         EQU    SDR00         ; Serial data register
SIOp          EQU    SIO00         ; Serial data register (lower 8 bit)
SSRmnL        EQU    SSR00L        ; Serial status register
SIRmnL        EQU    SIR00L        ; Serial flag clear trigger register
SSmL          EQU    SS0L          ; Serial channel start register
STmL          EQU    ST0L          ; Serial channel stop register
TRGONn        EQU    00000001B     ; for trigger SS00/ST00
SOEmL         EQU    SOE0L         ; Serial output enable register
SOEON         EQU    TRGONn        ; for turn on SOE00
SOEOFF        EQU    11111110B     ; for turn off SOE00
SOm           EQU    SO0           ; Serial output register
SOHIGH        EQU    TRGONn        ; for set SO bit
PM_CSIp       EQU    PM1           ; port mode register for CSI
PM_SCKp       EQU    PM1.0         ; port mode register bit for SCK
PM_SIp        EQU    PM1.1         ; port mode register bit for SI
PM_SOp        EQU    PM1.2         ; port mode register bit for SO
P_CSIp        EQU    P1            ; port register for CSI
P_SCKp        EQU    P1.0          ; port register for SCK
P_SIp         EQU    P1.1          ; port register for SI
P_SOp         EQU    P1.2          ; 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/G12 User's Manual: Hardware (R01UH0200E)

RL78 Family User's Manual: Software (R01US0015E)

RL78 Family CubeSuite+ Startup Guide (R01AN1232E)

RL78/G12 Serial Array Unit (CSI Slave Communication) (R01AN1370E)

(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.)

Website and Support

Renesas Electronics Website

- http://www.renesas.com/index.jsp

Inquiries

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

Revision Record	RL78/G12 Serial Array Unit (CSI Master Communication)

Rev.	Date	Description	
		Page	Summary
1.00	Mar.01, 2013	—	First edition issued

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this manual, refer to the relevant sections of the manual. If the descriptions under General Precautions in the Handling of MPU/MCU Products and in the body of the manual differ from each other, the description in the body of the manual takes precedence.

1. Handling of Unused Pins

- Handle unused pins in accord 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 unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

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

3. Prohibition of Access to Reserved Addresses

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

4. Clock Signals

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

5. Differences between Products

- Before changing from one product to another, i.e. to one with a different part number, confirm that the change will not lead to problems.
  - The characteristics of MPU/MCU in the same group but having different part numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different part numbers, implement a system-evaluation test for each of the products.

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 of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of 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. 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 should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended 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; and industrial robots etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.
   Renesas Electronics products are neither intended nor 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 damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, 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, please evaluate the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may 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 should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

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

RENESAS

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

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

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

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

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

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

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

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 2.2