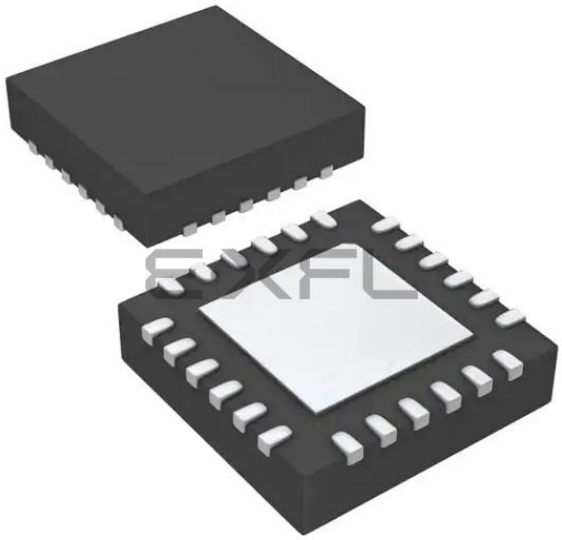


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	RL78
Core Size	16-Bit
Speed	24MHz
Connectivity	CSI, I²C, UART/USART
Peripherals	DMA, LVD, POR, PWM, WDT
Number of I/O	18
Program Memory Size	4KB (4K x 8)
Program Memory Type	FLASH
EEPROM Size	2K x 8
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	A/D 11x8/10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	24-WFQFN Exposed Pad
Supplier Device Package	24-HWQFN (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/renesas-electronics-america/r5f10277dna-w0

CHAPTER 1 OUTLINE

1.1 Differences between the R5F102 Products and the R5F103 Products

The following are differences between the R5F102 products and the R5F103 products.

- Whether the data flash memory is mounted or not
- High-speed on-chip oscillator oscillation frequency accuracy
- Number of channels in serial interface
- Whether the DMA function is mounted or not
- Whether a part of the safety functions are mounted or not

1.1.1 Data Flash

The data flash memory of 2 KB is mounted on the R5F102 products, but not on the R5F103 products.

Product	Data Flash
R5F102 products R5F1026A, R5F1027A, R5F102AA, R5F10269, R5F10279, R5F102A9, R5F10268, R5F10278, R5F102A8, R5F10267, R5F10277, R5F102A7, R5F10266 Note	2KB
R5F103 products R5F1036A, R5F1037A, R5F103AA, R5F10369, R5F10379, R5F103A9, R5F10368, R5F10378 R5F103A8, R5F10367, R5F10377, R5F103A7, R5F10366	Not mounted

Note The RAM in the R5F10266 has capacity as small as 256 bytes. Depending on the customer's program specification, the stack area to execute the data flash library may not be kept and data may not be written to or erased from the data flash memory.

Caution When the flash memory is rewritten via a user program, the code flash area and RAM area are used because each library is used. When using the library, refer to RL78 Family Flash Self Programming Library Type01 User's Manual and RL78 Family Data Flash Library Type04 User's Manual.

1.1.2 On-chip oscillator characteristics

(1) High-speed on-chip oscillator oscillation frequency of the R5F102 products

Oscillator	Condition	MIN	MAX	Unit
High-speed on-chip oscillator oscillation frequency accuracy	T_A = -20 to +85 °C	-1.0	+1.0	%
	T_A = -40 to -20 °C	-1.5	+1.5	
	T_A = +85 to +105 °C	-2.0	+2.0	

(2) High-speed on-chip oscillator oscillation frequency of the R5F103 products

Oscillator	Condition	MIN	MAX	Unit
High-speed on-chip oscillator oscillation frequency accuracy	T_A = -40 to + 85 °C	-5.0	+5.0	%

1.1.3 Peripheral Functions

The following are differences in peripheral functions between the R5F102 products and the R5F103 products.

RL78/G12		R5F102 product		R5F103 product	
		20, 24 pin product	30 pin product	20, 24 pin product	30 pin product
Serial interface	UART	1 channel	3 channels	1 channel	
	CSI	2 channels	3 channels	1 channel	
	Simplified I^2C	2 channels	3 channels	None	
DMA function		2 channels		None	
Safety function	CRC operation	Yes		None	
	RAM guard	Yes		None	
	SFR guard	Yes		None	

Table 4-6. Concept of Basic Settings

Output Function of Used Pin	Output Function of Unused Alternate Function		
	Port output function	SAU output function	Output function for other than SAU
Port output function	–	Output is high (1)	Output is low (0)
Output function for SAU	High (1)	–	Output is low (0)
Output function for other than SAU	Low (0)	Output is high (1)	Output is low (0)Note

Note Because more than one output function other than SAU might be assigned to a single pin, the output of unused alternate functions must be set to low level (0). For details about the setting method, see **4.5.2 Register settings for alternate function whose output function is not used**.

4.5.2 Register settings for alternate function whose output function is not used

If the output of an alternate function is not used, specify the settings as described below. If the output of the peripheral function is still the target of the peripheral I/O redirection function, the output can be switched to another pin by setting the peripheral I/O redirection register (PIOR). This allows usage of the port function or other alternate functions assigned to the target pin.

(1) SOp = 1 and TxDq = 1 (settings when the serial output (SOp/TxDq) of SAU is not used)

If the serial output (SOp/TxDq) is not used, such as a case in which only the serial input of SAU is used, set the bit in the serial output enable register m (SOEm) that corresponds to the unused output to 0 (output disabled) and set the SOmn bit in the serial output register m (SOm) to 1 (high). These settings are the same as the initial state.

(2) SCKp = 1, SDAr = 1, and SCLr = 1 (settings when channel n in SAU is not used)

If SAU is not used, set the bit n (SEmn) in the serial channel enable status register m (SEm) to 0 (operation stopped state), set the bit in the serial output enable register m (SOEm) that corresponds to the unused output to 0 (output disabled), and set the SOmn bit and CKOmn bit in the serial output register m (SOm) to 1 (high). These settings are the same as the initial state.

(3) TOmn = 0 (settings when the output of channel n in TAU is not used)

If the TOmn output of TAU is not used, set the bit in the timer output enable register 0 (TOE0) that corresponds to the unused output to 0 (output disabled) and set the bit in the timer output register 0 (TO0) to 0 (low). These settings are the same as the initial state.

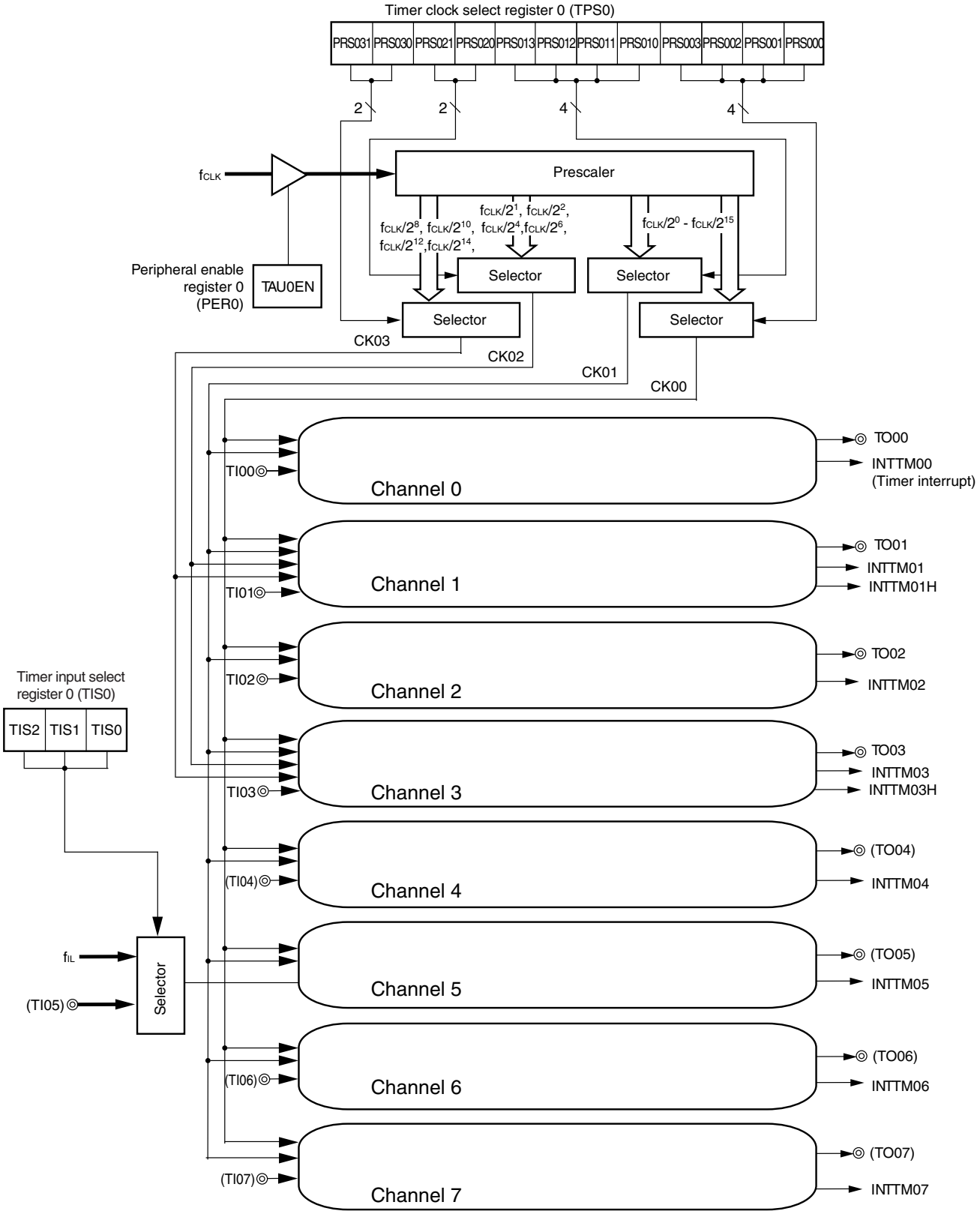
(4) SDAAn = 0 and SCLAn = 0 (setting when IICA is not used)

If IICA is not used, set the IICEn bit in the IICA control register n0 (IICCTLn0) to 0 (operation stopped). This setting is the same as the initial state.

(5) PCLBUZn = 0 (setting when clock output or buzzer output is not used)

If the clock output or buzzer output is not used, set the PCLOEn bit in the clock output selection register n (CKSn) to 0 (output disabled). This setting is the same as the initial state.

Figure 6-2. Entire Configuration of Timer Array Unit (30-pin products)



Remark Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR).

Cautions 1. Be sure to clear bits 15, 14, 11, 10 to "0".

2. If f_{CLK} (undivided) is selected as the operation clock (CKmk) and TDRnm is set to 0000H (n = 0 or 1, m = 0 to 7), interrupt requests output from timer array units cannot be used.

Remarks 1. f_{CLK}: CPU/peripheral hardware clock frequency

2. The above selected clock , but a signal which becomes high level for one period of f_{CLK} from its rising edge (m = 2 to 15). For details, see **6.5.1 Count clock (f_{TCLK}).**

By using channels 1 and 3 in the 8-bit timer mode and specifying CK02 or CK03 as the operation clock, the interval times shown in **Table 6-4** can be achieved by using the interval timer function.

Table 6-4. Interval Times Available for Operation Clock CKS02 or CKS03

Clock		Interval time (f_{CLK} = 20 MHz)[Note]			
		10 μs	100 μs	1 ms	10 ms
CK02	$f_{CLK}/2$	√	–	–	–
	$f_{CLK}/2^2$	√	–	–	–
	$f_{CLK}/2^4$	√	√	–	–
	$f_{CLK}/2^6$	–	√	√	–
CK03	$f_{CLK}/2^8$	–	√	√	–
	$f_{CLK}/2^{10}$	–	–	√	–
	$f_{CLK}/2^{12}$	–	–	–	√
	$f_{CLK}/2^{14}$	–	–	–	–

Note The margin is within 4 %.

Remarks 1. f_{CLK}: CPU/peripheral hardware clock frequency

2. For details of a signal of $f_{CLK}/2^n$ selected with the TPSm register, see **6.5.1 Count clock (f_{TCLK})**.

6.3.3 Timer mode register 0n (TMR0n)

The TMR0n register sets an operation mode of channel n. This register is used to select the operation clock (f_{MCK}), select the count clock, select the master/slave, select the 16 or 8-bit timer (only for channels 1 and 3), specify the start trigger and capture trigger, select the valid edge of the timer input, and specify the operation mode (interval, capture, event counter, one-count, or capture and one-count).

Rewriting the TMR0n register is prohibited when the register is in operation (when TE0n = 1). However, bits 7 and 6 (CIS0n1, CIS0n0) can be rewritten even while the register is operating with some functions (when TE0n = 1) (for details, see **6.8 Independent Channel Operation Function of Timer Array Unit** and **6.9 Simultaneous Channel Operation Function of Timer Array Unit)**.

The TMR0n register can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0000H.

Caution The bits mounted depend on the channels in the bit 11 of TMR0n register.

TMR02, TMR04, TMR06: MASTER0n bit (n = 2, 4, 6)

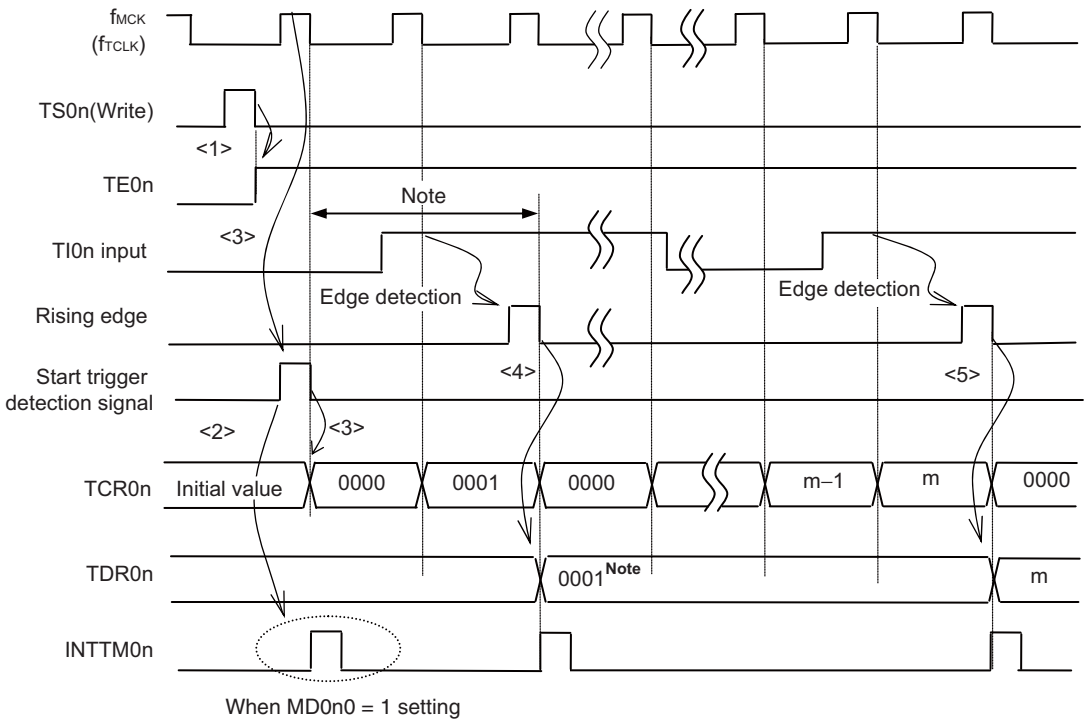
TMR01, TMR03: SPLIT0n bit (n = 1, 3)

TMR00, TMR05, TMR07: Fixed to 0

(3) Capture mode operation (input pulse interval measurement)

<1> Operation is enabled (TE0n = 1) by writing 1 to the TS0n bit.
<2> Timer count register 0n (TCR0n) holds the initial value until count clock generation.
<3> A start trigger is generated at the first count clock after operation is enabled. And the value of 0000H is loaded to the TCR0n register and counting starts in the capture mode. (When the MD0n0 bit is set to 1, INTTM0n is generated by the start trigger.)
<4> On detection of the valid edge of the TI0n input, the value of the TCR0n register is captured to timer data register 0n (TDR0n) and INTTM0n is generated. However, this capture value is nomeaning. The TCR0n register keeps on counting from 0000H.
<5> On next detection of the valid edge of the TI0n input, the value of the TCR0n register is captured to timer data register 0n (TDR0n) and INTTM0n is generated.

Figure 6-24. Operation Timing (In Capture Mode: Input Pulse Interval Measurement)



Note If a clock has been input to TI0n (the trigger exists) when capturing starts, counting starts when a trigger is detected, even if no edge is detected. Therefore, the first captured value (<4>) does not determine a pulse interval (in the above figure, 0001 just indicates two clock cycles but does not determine the pulse interval) and so the user can ignore it.

Caution In the first cycle operation of count clock after writing the TS0n bit, an error at a maximum of one clock is generated since count start delays until count clock has been generated. When the information on count start timing is necessary, an interrupt can be generated at count start by setting MD0n0 = 1.

Remark The timing is shown in **Figure 6-24** indicates while the noise filter is not used. By making the noise filter on-state, the edge detection becomes 2 fMCK cycles (it sums up to 3 to 4 cycles) later than the normal cycle of TI0n input.
The error per one period occurs be the asynchronous between the period of the TI0n input and that of the count clock (fMCK).

7.3.3 Interval timer control register (ITMC)

This register is used to set up the starting and stopping of the 12-bit interval timer operation and to specify the timer compare value.

The ITMC register can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0FFFH.

Figure 7-4. Format of Interval Timer Control Register (ITMC)

Address: FFF90H After reset: 0FFFH R/W

Symbol	15	14	13	12	11 to 0
ITMC	RINTE	0	0	0	ITCMP11 to ITCMP0

RINTE	12-bit interval timer operation control
0	Count operation stopped (count clear)
1	Count operation started

ITCMP11 to ITCMP0	Specification of the 12-bit interval timer compare value
001H	These bits generate an interrupt at the fixed cycle (count clock cycles x (ITCMP setting + 1)).
• • •	
FFFH1	
Example interrupt cycles when 001H or FFFH is specified for ITCMP11 to ITCMP0 • ITCMP11 to ITCMP0 = 001H, count clock: when f_{IL} = 15 kHz 1/15 [kHz] × (1 + 1) ÷ 0.1333 [ms] = 133.3 [μs] • ITCMP11 to ITCMP0 = FFFH, count clock: when f_{IL} = 15 kHz 1/15 [kHz] × (4095 + 1) ÷ 273 [ms]	

Cautions 1. When RINTE bit is changed from 0 to 1, set WUTMMCK0 bit of OSMC register to 1 before the change so that the operation clock is established.

2. Before changing the RINTE bit from 1 to 0, use the interrupt mask flag register to disable the INTIT interrupt servicing. When operation starts (0 to 1) again, clear the TMKAIF flag, and then enable the interrupt servicing.

3. The value read from the RINTE bit is applied one count clock cycle after setting the RINTE bit.

4. When setting the RINTE bit after returned from standby mode and entering standby mode again, confirm that the written value of the RINTE bit is reflected, or wait that more than one clock of the count clock has elapsed after returned from standby mode. Then enter standby mode.

5. Only change the setting of the ITCMP11 to ITCMP0 bits when RINTE = 0.
However, it is possible to change the settings of the ITCMP11 to ITCMP0 bits at the same time as when changing RINTE from 0 to 1 or 1 to 0.

10.6.3 Software trigger mode (scan mode, sequential conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> After the software counts up to the stabilization wait time (1 μs), the ADCS bit of the ADM0 register is set to 1 to perform A/D conversion on the four analog input channels specified by scan 0 to scan 3, which are specified by the analog input channel specification register (ADS). A/D conversion is performed on the analog input channels in order, starting with that specified by scan 0.

<3> A/D conversion is sequentially performed on the four analog input channels, the conversion results are stored in the A/D conversion result register (ADCR, ADCRH) each time conversion ends, and the A/D conversion end interrupt request signal (INTAD) is generated. After A/D conversion of the four channels ends, the A/D conversion of the channel following the specified channel automatically starts (until all four channels are finished).

<4> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.

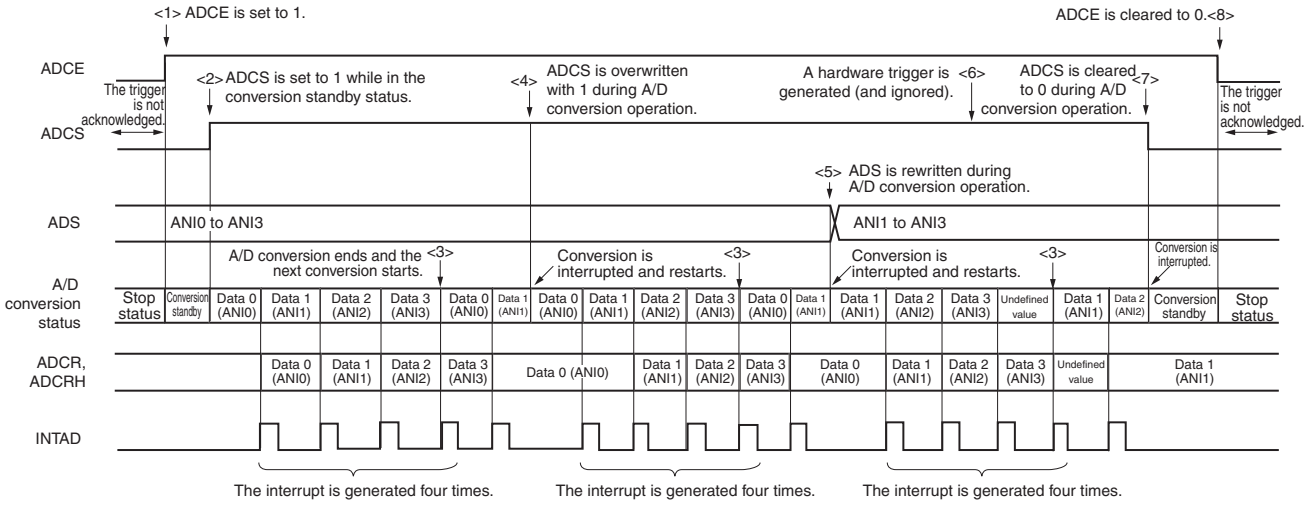
<5> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register. The partially converted data is discarded.

<6> Even if a hardware trigger is input during conversion operation, A/D conversion does not start.

<7> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status.

<8> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start.

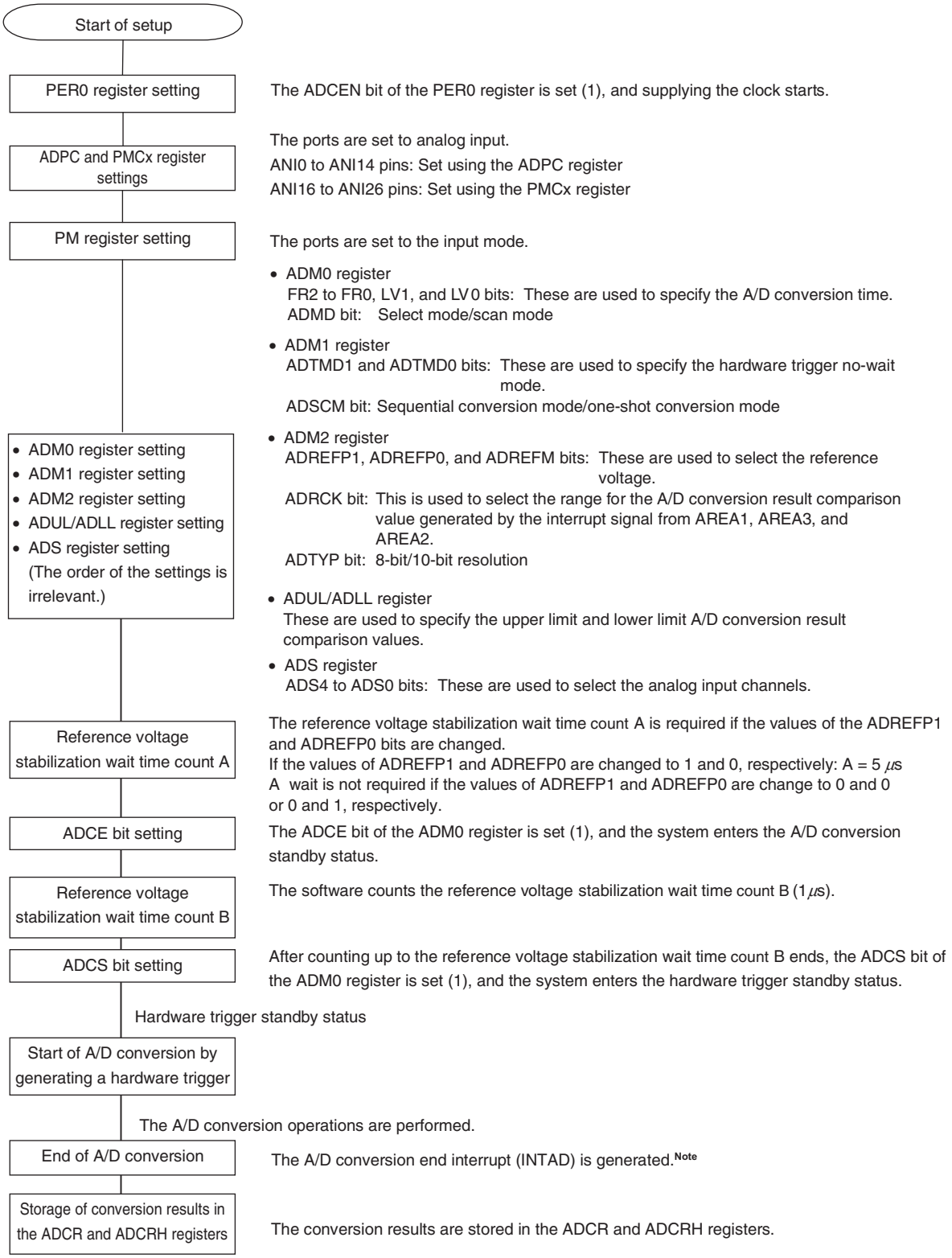
Figure 10-19. Example of Software Trigger Mode (Scan Mode, Sequential Conversion Mode) Operation Timing



10.7.2 Setting up hardware trigger no-wait mode

Figure 10-30. Setting up Hardware Trigger No-Wait Mode

<R>



Note Depending on the settings of the ADRCK bit and ADUL/ADLL register, there is a possibility of no interrupt signal being generated. In this case, the results are not stored in the ADCR and ADCRH registers.

Figure 11-9. Format of Serial Communication Operation Setting Register mn (SCRmn) (2/2)

Address: F0118H, F0119H (SCR00), F011EH, F011FH (SCR03) After reset: 0087H R/W
F0158H, F0159H (SCR10), F015AH, F015BH (SCR11)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn	RXEmn	DAPmn	CKPmn	0	EOCmn	PTCmn1	PTCmn0	DIRmn	0	SLCmn1[Note 1]	SLCmn0	0	1	DLSmn1[Note 2]	DLSmn0

PTCmn1	PTCmn0	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[Note 3].	No parity judgment
1	0	Outputs even parity.	Judged as even parity.
1	1	Outputs odd parity.	Judges as odd parity.
Be sure to set PTCmn1, PTCmn0 = 0, 0 in the CSI mode and simplified I^2C mode.			

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.
Be sure to clear DIRmn = 0 in the simplified I^2C mode.	

SLCmn1[Note 1]	SLCmn0	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 (mn = 00, 02, 10 only)
1	1	Setting prohibited
When the transfer end interrupt is selected, the interrupt is generated when all stop bits have been completely transferred. Set the stop bit length to 1 bit (SLCmn1, SLCmn0 = 0, 1) during UART reception and in the simplified I^2C mode. Set no stop bit (SLCmn1, SLCmn0 = 0, 0) in the CSI mode. Set the stop bit length to 1 bit (SLCmn1, SLCmn0 = 0, 1) or 2 bits (SLCmn1, SLCmn0 = 1, 0) during UART transmission.		

DLSmn1[Note 2]	DLSmn0	Setting of data length in CSI and UART modes
0	1	9-bit data length (stored in bits 0 to 8 of the SDR00 and SDR01 registers) (settable in UART0 mode only)
1	0	7-bit data length (stored in bits 0 to 6 of the SDRmn register)
1	1	8-bit data length (stored in bits 0 to 7 of the SDRmn register)
Other than above		Setting prohibited
Be sure to set DLSmn1, DLSmn0 = 1, 1 in the simplified I^2C mode.		

Notes 1. Provided in the SCR00, SCR02, and SCR10 registers only.
2. Provided in the SCR00 and SCR01 registers only. Others are fixed to 1.
3. 0 is always added regardless of the data contents.

Caution Be sure to clear bits 3, 6, and 11 to "0" (also clear bit 5 of the SCR01, SCR03, or SCR11 register to 0). Be sure to set bit 2 to "1".

Remark m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 11, 20)

Caution When using a serial array unit for UART, both the transmitter side (even-numbered channel) and the receiver side (odd-numbered channel) can only be used for UART.

UART performs the following two types of communication operations.

- UART transmission (See **11.6.1**.)
- UART reception (See **11.6.2**.)

11.6.1 UART transmission

UART transmission is an operation to transmit data from the RL78 microcontroller to another device asynchronously (start-stop synchronization).

Of the two channels used for UART, the even-numbered channel is used for UART transmission.

UART	UART0	UART1	UART2
Target channel	Channel 0 of SAU0	Channel 2 of SAU0	Channel 0 of SAU1
Pins used	TxD0	TxD1	TxD2
Interrupt	INTST0	INTST1	INTST2
	Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected.		
Error detection flag	None		
Transfer data length	7, 8, or 9 bits[Note 1]		
Transfer rate[Note 2]	Max. f_{MCK}/6 [bps] (SDRmn[15:9] = 2 or greater), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps][Note]		
Data phase	Non-inverted output (default: high level) Inverted output (default: low level)		
Parity bit	The following selectable • No parity bit • Appending 0 parity • Appending even parity • Appending odd parity		
Stop bit	The following selectable • Appending 1 bit • Appending 2 bits		
Data direction	MSB or LSB first		

Notes 1. Only UART0 can be specified for the 9-bit data length.

2. Use this operation within a range that satisfies the conditions above and the peripheral function characteristics in the electrical specifications (see **CHAPTER 28 ELECTRICAL SPECIFICATIONS (A, D: T_A = -40 to +85°C)** or **CHAPTER 29 ELECTRICAL SPECIFICATIONS (G: T_A = -40 to +105°C)**).

Remarks 1. f_{MCK}: Operation clock frequency of target channel
f_{CLK}: System clock frequency

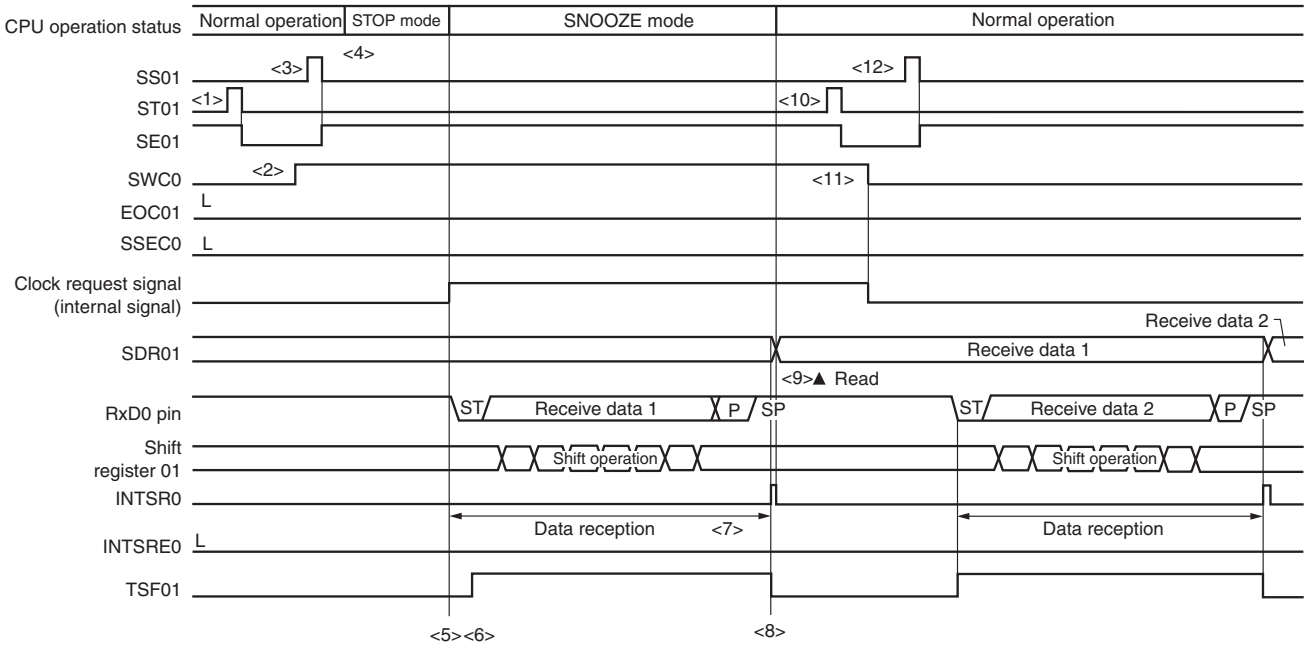
2. m: Unit number (m = 0, 1) n: Channel number (n = 0, 2), mn = 00, 02, 10

(1) SNOOZE mode operation (EOC01 = 0, SSEC0 = 0/1)

Because of the setting of EOC01 = 0, even though a communication error occurs, an error interrupt (INTSRE0) is not generated, regardless of the setting of the SSEC0 bit. A transfer end interrupt (INTSR0) will be generated.

Figure 11-90. Timing Chart of SNOOZE Mode Operation (EOC01 = 0, SSEC0 = 0/1)

<R>



Note Read the received data when SWC0 = 1.

Caution Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, be sure to set the ST00 bit to 1 (clear the SE00 bit to stop the operation).
And after completion the receive operation, also clearing SWC0 bit to 0 (SNOOZE mode release).

Remark <1> to <12> in the figure correspond to <1> to <12> in **Figure 11-92 Flowchart of SNOOZE Mode Operation (EOC01 = 0, SSEC0 = 0/1 or EOC01 = 1, SSEC0 = 0)**.

11.7 Operation of Simplified I²C (IIC00, IIC01, IIC11, IIC20) Communication

This is a clocked communication function to communicate with two or more devices by using two lines: serial clock (SCL) and serial data (SDA). This communication function is designed to execute single communication with devices such as EEPROM, flash memory, and A/D converter, and therefore, can be used only by the master.

Operate the control registers by software for setting the start and stop conditions while observing the specifications of the I²C bus line.

[Data transmission/reception]

- Master transmission, master reception (only master function with a single master)
- ACK output function[Note] and ACK detection function
- Data length of 8 bits
  (When an address is transmitted, the address is specified by the higher 7 bits, and the least significant bit is used for R/W control.)
- Generation of start condition and stop condition for software

[Interrupt function]

- Transfer end interrupt

[Error detection flag]

- Overrun error
- ACK error

* [Functions not supported by simplified I²C]

- Slave transmission, slave reception
- Multi master function (Arbitration loss detection function)
- Wait detection function

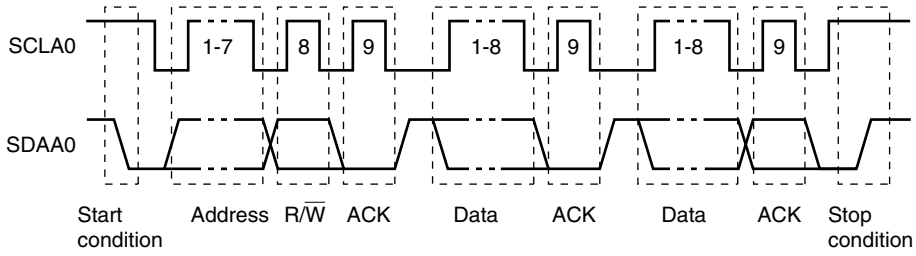
Note When receiving the last data, ACK will not be output if 0 is written to the SOEmn (SOEm register) bit and serial communication data output is stopped. See the processing flow in **11.7.3 (2)** for details.

Remark m: Unit number, n: Channel number (mn = 00, 01, 03, 10)

12.5 I²C Bus Definitions and Control Methods

The following section describes the I²C bus's serial data communication format and the signals used by the I²C bus. Figure 12-14 shows the transfer timing for the "start condition", "address", "data", and "stop condition" output via the I²C bus's serial data bus.

Figure 12-14. I²C Bus Serial Data Transfer Timing



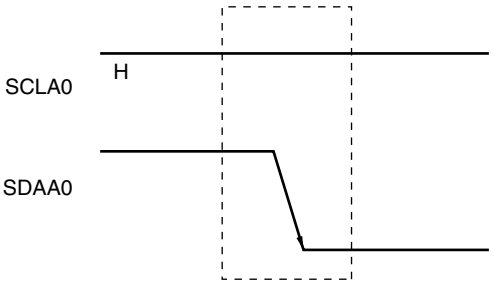
The master device generates the start condition, slave address, and stop condition.
The acknowledge (ACK) can be generated by either the master or slave device (normally, it is output by the device that receives 8-bit data).

The serial clock (SCLA0) is continuously output by the master device. However, in the slave device, the SCLA0 pin low level period can be extended and a wait can be inserted.

12.5.1 Start conditions

A start condition is met when the SCLA0 pin is at high level and the SDAA0 pin changes from high level to low level. The start conditions for the SCLA0 pin and SDAA0 pin are signals that the master device generates to the slave device when starting a serial transfer. When the device is used as a slave, start conditions can be detected.

Figure 12-15. Start Conditions



A start condition is output when bit 1 (STT0) of IICA control register 00 (IICCTL00) is set (1) after a stop condition has been detected (SPD0: Bit 0 of the IICA status register 0 (IICS0) = 1). When a start condition is detected, bit 1 (STD0) of the IICS0 register is set (1).

13.2.3 Multiplication/division data register C (MDCL, MDCH)

The MDCH and MDCL registers are used to store the accumulated result while in the multiply-accumulator mode or the remainder of the operation result while in the division mode. These registers are not used while in the multiplication mode.

The MDCH and MDCL registers can be set by a 16-bit manipulation instruction.

Reset signal generation clears these registers to 0000H.

Figure 13-4. Format of Multiplication/Division Data Register C (MDCH, MDCL)

Address: F00E0H, F00E1H, F00E2H, F00E3H After reset: 0000H, 0000H R/W

Symbol F00E3H F00E2H

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

Symbol F00E1H F00E0H

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

Cautions 1. The MDCH and MDCL registers values read during division operation processing (when the multiplication/division control register (MDUC) value is 81H or C1H) will not be guaranteed.

2. During multiply-accumulator processing, do not use software to rewrite the values of the MDCH and MDCL registers. If this is done, the operation result will be undefined.

3. The data is in the two's complement format in the multiply-accumulator mode (signed).

Table 13-4. Functions of MDCH and MDCL Registers During Operation Execution

Operation Mode	Setting	Operation Result
Multiplication mode (unsigned or signed)	–	–
Multiply-accumulator mode (unsigned)	MDCH: Initial accumulated value (unsigned) (higher 16 bits) MDCL: Initial accumulated value (unsigned) (lower 16 bits)	MDCH: accumulated value (unsigned) (higher 16 bits) MDCL: accumulated value (unsigned) (lower 16 bits)
Multiply-accumulator mode (signed)	MDCH: Initial accumulated value (signed) (higher 16 bits) MDCL: Initial accumulated value (signed) (lower 16 bits)	MDCH: accumulated value (signed) (higher 16 bits) MDCL: accumulated value (signed) (lower 16 bits)
Division mode (unsigned)	–	MDCH: Remainder (unsigned) (higher 16 bits) MDCL: Remainder (unsigned) (lower 16 bits)

Table 15-4. Flags Corresponding to Interrupt Request Sources (30-pin products) (2/2)

Interrupt Source	Interrupt Request Flag	Register	Interrupt Mask Flag	Register	Priority Specification Flag	Register	R5F102Ax products	R5F103Ax products
INTTM05	TMIF05	IF2L	TMMK05	MK2L	TMPR005, TMPR105	PR02L, PR12L	✓	✓
INTTM06	TMIF06		TMMK06		TMPR006, TMPR106		✓	✓
INTTM07	TMIF07		TMMK07		TMPR007, TMPR107		✓	✓
INTMD	MDIF	IF2H	MDMK	MK2H	MDPR0, MDPR1	PR02H, PR12H	✓	✓
INTFL	FLIF		FLMK		FLPR0, FLPR1		✓	✓

Notes 1. If interrupt source INTST2, INTCSI20, or INTIIC20 occurs, bit 0 of the IF0H register is set to 1. In addition, bit 0 of the MK0H, PR00H, and PR10H registers corresponds to these three interrupt sources.

2. If interrupt source INTST0, INTCSI00, or INTIIC00 occurs, bit 5 of the IF0H register is set to 1. In addition, bit 5 of the MK0H, PR00H, and PR10H registers corresponds to these three interrupt sources.

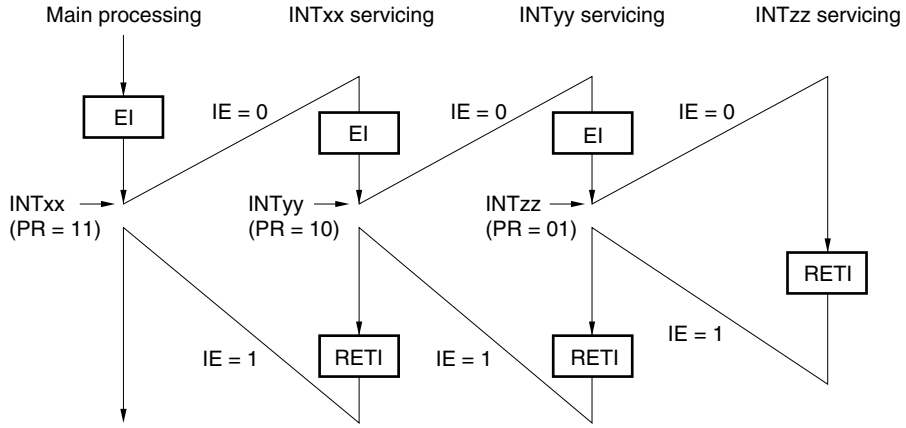
3. Do not use a UART0 reception error interrupt and an interrupt of channel 1 of TAU0 (at higher 8-bit timer operation) at the same time because they share flags for the interrupt request sources. If the UART0 reception error interrupt is not used (EOC01 = 0), UART0 and channel 1 of TAU0 (at higher 8-bit timer operation) can be used at the same time. If one of the interrupt sources INTSRE0 and INTTM01H is generated, bit 7 of the IF0H register is set to 1. Bit 7 of the MK0H, PR00H, and PR10H registers supports these two interrupt sources.

4. If interrupt source INTST1, INTCSI11, or INTIIC11 occurs, bit 1 of the IF1L register is set to 1. Bit 1 of the MK1L, PR01L, and PR11L registers corresponds to these three interrupt sources.

5. Do not use a UART1 reception error interrupt and an interrupt of channel 3 of TAU0 (at higher 8-bit timer operation) at the same time because they share flags for the interrupt request sources. If the UART1 reception error interrupt is not used (EOC01 = 0), UART1 and channel 3 of TAU0 (at higher 8-bit timer operation) can be used at the same time. If interrupt source INTTM03H is generated, bit 2 of the IF1L register is set to 1. Bit 2 of the MK1L, PR01L, and PR11L registers supports these two interrupt sources.

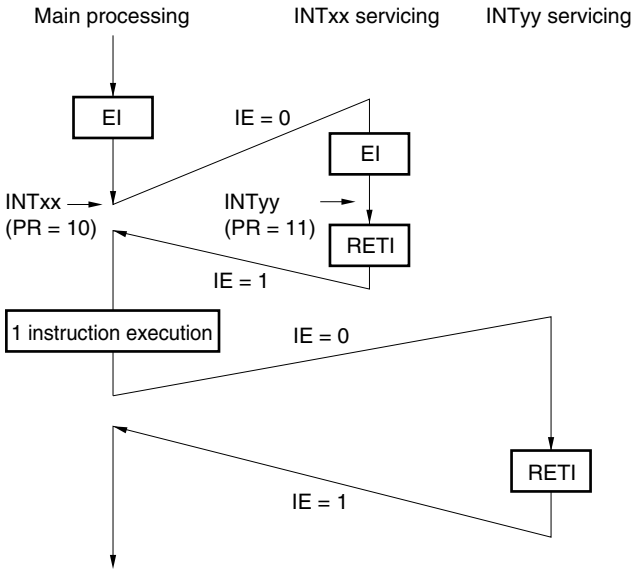
Figure 15-13. Examples of Multiple Interrupt Servicing (1/2)

Example 1. Multiple interrupt servicing occurs twice



During servicing of interrupt INTxx, two interrupt requests, INTyy and INTzz, are acknowledged, and multiple interrupt servicing takes place. Before each interrupt request is acknowledged, the EI instruction must always be issued to enable interrupt request acknowledgment.

Example 2. Multiple interrupt servicing does not occur due to priority control



Interrupt request INTyy issued during servicing of interrupt INTxx is not acknowledged because its priority is lower than that of INTxx, and multiple interrupt servicing does not take place. The INTyy interrupt request is held pending, and is acknowledged following execution of one main processing instruction.

PR = 00: Specify level 0 with ××PR1× = 0, ××PR0× = 0 (higher priority level)
PR = 01: Specify level 1 with ××PR1× = 0, ××PR0× = 1
PR = 10: Specify level 2 with ××PR1× = 1, ××PR0× = 0
PR = 11: Specify level 3 with ××PR1× = 1, ××PR0× = 1 (lower priority level)
IE = 0: Interrupt request acknowledgment is disabled
IE = 1: Interrupt request acknowledgment is enabled.

CHAPTER 18 RESET FUNCTION

The following seven operations are available to generate a reset signal.

(1) External reset input via $\overline{\text{RESET}}$ pin
(2) Internal reset by watchdog timer program loop detection
(3) Internal reset by comparison of supply voltage and detection voltage of power-on-reset (POR) circuit
(4) Internal reset by comparison of supply voltage of the voltage detector (LVD) and detection voltage
(5) Internal reset by execution of illegal instruction[Note]
(6) Internal reset by RAM parity error
(7) Internal reset by illegal-memory access

<R> External and internal resets start program execution from the address at 00000H and 00001H when the reset signal is generated.

A reset is effected when a low level is input to the $\overline{\text{RESET}}$ pin, the watchdog timer overflows, or by POR and LVD circuit voltage detection, execution of illegal instruction[Note], RAM parity error or illegal-memory access, and each item of hardware is set to the status shown in Table 18-1.

Note The illegal instruction is generated when instruction code FFH is executed.
Reset by the illegal instruction execution not issued by emulation with the in-circuit emulator or on-chip debug emulator.

Cautions 1. After power is turned on, P125 functions as the $\overline{\text{RESET}}$ pin (20- or 24-pin products only). Even if the internal reset signal is released by the power-on reset (POR), the reset state is retained while a low level is input to this pin.
When P125/KR1/SI01 is used, select the port function (PORTSELB = 0) by the option byte (000C1H) and release all reset sources.

2. For an external reset, input a low level for 10 μs or more to the $\overline{\text{RESET}}$ pin. To perform an external reset upon power application, input a low level to the RESET pin, turn power on, continue to input a low level on the pin for 10 us or more within the operating voltage range shown in 28.4 or 29.4 AC Characteristics, and then input a high level to the pin.

3. During reset input, the X1 clock, high-speed on-chip oscillator clock, and low-speed on-chip oscillator clock stop oscillating, and external main system clock input is invalid.

4. The port pins become the following state because each SFR and 2nd SFR are initialized after reset.
- P40: High-impedance during the external reset period or reset period by the POR. High level during other types of reset or after receiving a reset signal (connected to the internal pull-up resistor).
- P125: High level during the reset period or after receiving a reset signal (connected to the internal pull-up resistor).
- Ports other than P40 and p125: High-impedance during the reset period or after receiving a reset signal.

Remark V_{POR}: POR power supply rise detection voltage

24.8 Data Flash

24.8.1 Data flash overview

An overview of the data flash memory is provided below.

- The user program can rewrite the data flash memory by using the data flash library. For details, refer to **RL78 Family Data Flash Library User's Manual**.
- The data flash memory can also be rewritten to through serial programming using the dedicated flash memory programmer or an external device.
- The data flash can be erased in 1-block (1-Kbyte) units.
- The data flash can be accessed only in 8-bit units.
- The data flash can be directly read by CPU instructions.
- Instructions can be executed from the code flash memory while rewriting the data flash memory (that is, background operation (BGO) is supported).
- Because the data flash memory is an area exclusively used for data, it cannot be used to execute instructions.
- Accessing the data flash memory is not possible while rewriting the code flash memory (during self-programming).
- Manipulating the DFLCTL register is not possible while rewriting the data flash memory.
- Transition to the STOP mode is not possible while rewriting the data flash memory.

Cautions 1. The data flash memory is stopped after a reset is canceled. The data flash control register (DFLCTL) must be set up in order to use the data flash memory.

2. Interrupts are disabled during data flash rewrite for only the R5F10266. Execute the data flash library with the IE flag cleared (0) by the DI instruction.

3. The high-speed on-chip oscillator should be kept operating during data flash rewrite. If it is kept stopping, the high-speed on-chip oscillator clock should be operated (HIOSTOP = 0). The data flash library should be executed after 30 μs have elapsed.

Remark For rewriting the code flash memory via a user program, see **24.6 Self-Programming**.

24.8.2 Register controlling data flash memory

24.8.2.1 Data flash control register (DFLCTL)

This register is used to enable or disable accessing to the data flash.

The DFLCTL register is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets this register to 00H.

Figure 24-10. Format of Data Flash Control Register (DFLCTL)

Address: F0090H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	<0>
DFLCTL	0	0	0	0	0	0	0	DFLEN

DFLEN	Data flash access control
0	Disables data flash access
1	Enables data flash access

Caution Manipulating the DFLCTL register is not possible while rewriting the data flash memory.

27.1 Conventions Used in Operation List

27.1.1 Operand identifiers and specification methods

Operands are described in the “Operand” column of each instruction in accordance with the description method of the instruction operand identifier (refer to the assembler specifications for details). When there are two or more description methods, select one of them. Alphabetic letters in capitals and the symbols, #, !, !!, $, $!, [ ], and ES: are keywords and are described as they are. Each symbol has the following meaning.

- #: Immediate data specification
- !: 16-bit absolute address specification
- !!: 20-bit absolute address specification
- $: 8-bit relative address specification
- $!: 16-bit relative address specification
- [ ]: Indirect address specification
- ES:: Extension address specification

In the case of immediate data, describe an appropriate numeric value or a label. When using a label, be sure to describe the #, !, !!, $, $!, [ ], and ES: symbols.

For operand register identifiers, r and rp, either function names (X, A, C, etc.) or absolute names (names in parentheses in Table 27-1 below, R0, R1, R2, etc.) can be used for description.

Table 27-1. Operand Identifiers and Specification Methods

Identifier	Description Method
r	X (R0), A (R1), C (R2), B (R3), E (R4), D (R5), L (R6), H (R7)
rp	AX (RP0), BC (RP1), DE (RP2), HL (RP3)
sfr	Special-function register symbol (SFR symbol) FFF00H to FFFFFH
sfrp	Special-function register symbols (16-bit manipulatable SFR symbol. Even addresses only[Note]) FFF00H to FFFFFH
saddr	FFE20H to FFF1FH Immediate data or labels
saddrp	FFE20H to FF1FH Immediate data or labels (even addresses only[Note])
<R> addr20	00000H to FFFFFH Immediate data or labels
addr16	0000H to FFFFH Immediate data or labels (Only even addresses for 16-bit data transfer instructions[Note])
addr5	0080H to 00BFH Immediate data or labels (even addresses only[Note])
word	16-bit immediate data or label
byte	8-bit immediate data or label
bit	3-bit immediate data or label
RBn	RB0 to RB3

Note Bit 0 = 0 when an odd address is specified.

Remark The special function registers can be described to operand sfr as symbols. See **Table 3-6 SFR List** for the symbols of the special function registers. The extended special function registers can be described to operand !addr16 as symbols. See **Table 3-7 Extended SFR (2nd SFR) List** for the symbols of the extended special function registers.