Renesas Electronics America Inc - M30260F3AGP#U3A Datasheet

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	Not For New Designs
Core Processor	M16C/60
Core Size	16-Bit
Speed	20MHz
Connectivity	I²C, IEBus, SIO, UART/USART
Peripherals	DMA, PWM, Voltage Detect, WDT
Number of I/O	39
Program Memory Size	24KB (24K x 8)
Program Memory Type	FLASH
EEPROM Size	4K x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 12x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	48-LQFP
Supplier Device Package	48-LQFP (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/renesas-electronics-america/m30260f3agp-u3a

How to Use This Manual

1. Purpose and Target Readers

This manual is designed to provide the user with an understanding of the hardware functions and electrical characteristics of the MCU. It is intended for users designing application systems incorporating the MCU. A basic knowledge of electric circuits, logical circuits, and MCUs is necessary in order to use this manual.
The manual comprises an overview of the product; descriptions of the CPU, system control functions, peripheral functions, and electrical characteristics; and usage notes.

Particular attention should be paid to the precautionary notes when using the manual. These notes occur within the body of the text, at the end of each section, and in the Usage Notes section.

The revision history summarizes the locations of revisions and additions. It does not list all revisions. Refer to the text of the manual for details.

The following documents apply to the M16C/26A Group (M16C/26A, M16C/26B, and M16C/26T). Make sure to refer to the latest versions of these documents. The newest versions of the documents listed may be obtained from the Renesas Technology Web site.

Document Type	Description	Document Title	Document No.
Hardware manual	Hardware specifications (pin assignments, memory maps, peripheral function specifications, electrical characteristics, timing charts) and operation description Note: Refer to the application notes for details on using peripheral functions.	M16C/26A Group (M16C/26A, M16C/26B, M16C/26T) Hardware Manual	This hardware manual
Software manual	Description of CPU instruction set	M16C/60, M16C/20, M16C/Tiny Series Software Manual	REJ09B0137
Application note	Information on using peripheral functions and application examples Sample programs Information on writing programs in assembly language and C	Available from Renesas Technology Web site.	
Renesas technical update	Product specifications, updates on documents, etc.		

4. List of Abbreviations and Acronyms

Abbreviation	Full Form
ACIA	Asynchronous Communication Interface Adapter
bps	bits per second
CRC	Cyclic Redundancy Check
DMA	Direct Memory Access
DMAC	Direct Memory Access Controller
GSM	Global System for Mobile Communications
Hi-Z	High Impedance
IEBus	Inter Equipment bus
I/O	Input/Output
IrDA	Infrared Data Association
LSB	Least Significant Bit
MSB	Most Significant Bit
NC	Non-Connection
PLL	Phase Locked Loop
PWM	Pulse Width Modulation
SFR	Special Function Registers
SIM	Subscriber Identity Module
UART	Universal Asynchronous Receiver/Transmitter
VCO	Voltage Controlled Oscillator

4. Special Function Registers (SFRs)

Table 4.1 SFR Information(1)[1]

Address	Register	Symbol	After reset
000016			
000116			
000216			
000316			
000416	Processor mode register 0	PM0	0016
000516	Processor mode register 1	PM1	000010002
000616	System clock control register 0	CM0	010010002[5] 011010002(M16C/26T)
000716	System clock control register 1	CM1	001000002
000816			
000916	Address match interrupt enable register	AIER	XXXXXX002
000A16	Protect register	PRCR	XX0000002
000B16			
000C16	Oscillation stop detection register[2]	CM2	0X0000002
000D16			
000E16	Watchdog timer start register	WDTS	XX16
000F16	Watchdog timer control register	WDC	00XXXXXX2
001016	Address match interrupt register 0	RMAD0	0016
001116			0016
001216			X016
001316			
001416	Address match interrupt register 1	RMAD1	0016
001516			0016
001616			X016
001716			
001816			
001916	Voltage detection register 1 [3, 4]	VCR1	000010002
001A16	Voltage detection register 2 [3, 4]	VCR2	0016
001B16			
001C16	PLL control register 0	PLC0	0001X0102
001D16			
001E16	Processor mode register 2	PM2	XXX000002
001F16	Low voltage detection interrupt register[4]	D4INT	0016
002016	DMA0 source pointer	SAR0	XX16
002116			XX16
002216			XX16
002316			
002416	DMA0 destination pointer	DAR0	XX16
002516			XX16
002616			XX16
002716			
002816	DMA0 transfer counter	TCR0	XX16
002916			XX16
002A16			
002B16			
002C16	DMA0 control register	DM0CON	00000X002
002D16			
002E16			
002F16			
003016	DMA1 source pointer	SAR1	XX16
003116			XX16
003216			XX16
003316			
003416	DMA1 destination pointer	DAR1	XX16
003516			XX16
003616			XX16
003716			
003816	DMA1 transfer counter	TCR1	XX16
003916			XX16
003A16			
003B16			
003C16	DMA1 control register	DM1CON	00000X002
003D16			
003E16			
003F16			

NOTES:
1. The blank spaces are reserved. No access is allowed.
2. Bits CM27, CM21, and CM20 do not change at oscillation stop detection reset.
3. The VCR1 and VCR2 registers do not change at software reset, watchdog timer reset, and oscillation stop detection reset.
4. Registers VCR1, VCR2, and D4INT cannot be used in M16C/26T.
5. M16C/26A, M16C/26B

X : Undefined

6. Processor Mode

The microcomputer supports single-chip mode only. **Figures 6.1** and **6.2** show the associated registers.

Processor Mode Register 0 (1)

b7	b6	b5	b4	b3	b2	b1	b0
0	0	0	0		0	0	0

Symbol	Address	After Reset
PM0	0004$_{16}$	00$_{16}$

Bit Symbol	Bit Name	Function	RW
$\overline{\text{(b2-b0)}}$	Reserved bit	Set to "0"	RW
PM03	Software reset bit	The microcomputer is reset when this bit is set to "1". When read, its content is "0".	RW
$\overline{\text{(b7-b4)}}$	Reserved bit	Set to "0"	RW

NOTES:
1. Rewrite the PM0 register after the PRC1 bit in the PRCR register is set to "1" (write enable).

Processor Mode Register 1 (1)

b7	b6	b5	b4	b3	b2	b1	b0
	0	0	0	1		0	

Symbol	Address	After Reset
PM1	0005$_{16}$	00001000$_2$

Bit Symbol	Bit Name	Function	RW
PM10	Flash data block access bit (2)	0: Disabled 1: Enabled (3)	RW
$\overline{\text{(b1)}}$	Reserved bit	Set to "0"	RW
PM12	Watchdog timer function select bit	0 : Watchdog timer interrupt 1 : Watchdog timer reset (4)	RW
$\overline{\text{(b3)}}$	Reserved bit	Set to "1"	RW
$\overline{\text{(b6-b4)}}$	Reserved bit	Set to "0"	RW
PM17	Wait bit (5)	0 : No wait state 1 : Wait state (1 wait)	RW

NOTES:
1. Rewrite the PM1 register after the PRC1 bit in the PRCR register is set to "1" (write enable).
2. To access the two 2K-byte data spaces in data block A and data block B, set the PM10 bit to "1". The PM10 bit is not available in mask version.
3. When the FMR01 bit in the FMR0 register is set to "1" (enables CPU rewrite mode), the PM10 bit is automatically set to "1".
4. Set the PM12 bit to "1" by program. (Writing "0" by program has no effect)
5. When the PM17 bit is set to "1" (wait state), one wait is inserted when accessing the internal RAM or the internal ROM.

Figure 6.1 PM0 Register, PM1 Register

7.7 System Clock Protective Function

When the main clock is selected for the CPU clock source, this function protects the clock from modifications in order to prevent the CPU clock from becoming halted by run-away.

If the PM21 bit in the PM2 register is set to “1” (clock modification disabled), the following bits are protected against writes:

- CM02, CM05, and CM07 bits in CM0 register
- CM10, CM11 bits in CM1 register
- CM20 bit in CM2 register
- All bits in PLC0 register

Before the system clock protective function can be used, the following register settings must be made while the CM05 bit in the CM0 register is “0” (main clock oscillating) and CM07 bit is “0” (main clock selected for the CPU clock source):

(1) Set the PRC1 bit in the PRCR register to “1” (enable writes to PM2 register).

(2) Set the PM21 bit in the PM2 register to “1” (disable clock modification).

(3) Set the PRC1 bit in the PRCR register to “0” (disable writes to PM2 register).

Do not execute the WAIT instruction when the PM21 bit is set to “1”.

7.8 Oscillation Stop and Re-oscillation Detect Function

The oscillation stop and re-oscillation detect function allows the detection of main clock oscillation stop and reoscillation. At oscillation stop or re-oscillation detection, reset or oscillation stop, re-oscillation detection interrupt are generated. Depending on the CM27 bit in the CM2 register. The oscillation stop detection function can be enabled and disabled by the CM20 bit in the CM2 register. Table 7.8.1 lists a specification overview of the oscillation stop and re-oscillation detect function.

Table 7.8.1. Specification Overview of Oscillation Stop and Re-oscillation Detect Function

Item	Specification
Oscillation stop detectable clock and frequency bandwidth	$f(X_{IN}) \geq 2$ MHz
Enabling condition for oscillation stop, re-oscillation detection function	Set the CM20 bit to “1”(enable)
Operation at oscillation stop, re-oscillation detection	•Reset occurs (when the CM27 bit is set to "0") •Oscillation stop, re-oscillation detection interrupt occurs(when the CM27 bit is set to "1")

9.3 Interrupt Control

The following describes how to enable/disable the maskable interrupts, and how to set the priority in which order they are accepted. What is explained here does not apply to nonmaskable interrupts.

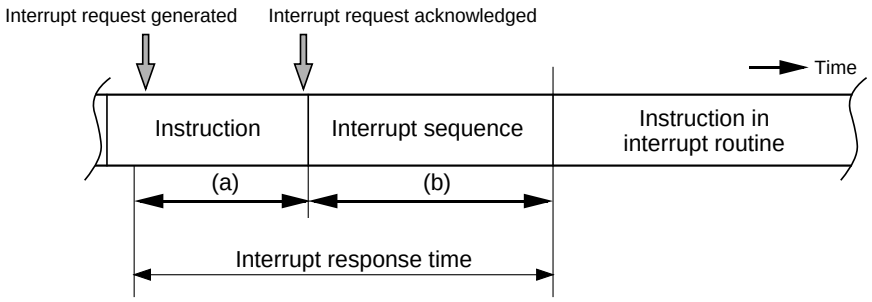
Use the I flag in the FLG register, IPL, and the ILVL2 to ILVL0 bits in the each interrupt control register to enable/disable the maskable interrupts. Whether an interrupt is requested is indicated by the IR bit in each interrupt control register.

Figure 9.3.1 shows the interrupt control registers.

Figure 9.3.2 shows the IFSR, IFSR2A registers.

9.4.1 Interrupt Response Time

Figure 9.4.1.1 shows the interrupt response time. The interrupt response or interrupt acknowledge time denotes the time from when an interrupt request is generated till when the first instruction in the interrupt routine is executed. Specifically, it consists of the time from when an interrupt request is generated till when the instruction then executing is completed ((a) in Figure 9.4.1.1) and the time during which the interrupt sequence is executed ((b) in Figure 9.4.1.1).



(a) The time from when an interrupt request is generated till when the instruction then executing is completed. The length of this time varies with the instruction being executed. The DIVX instruction requires the longest time, which is equal to 30 cycles (without wait state, the divisor being a register).

(b) The time during which the interrupt sequence is executed. For details, see the table below. Note, however, that the values in this table must be increased 2 cycles for the $\overline{\text{DBC}}$ interrupt and 1 cycle for the address match and single-step interrupts.

Interrupt vector address	SP value	Without wait
Even	Even	18 cycles
Even	Odd	19 cycles
Odd	Even	19 cycles
Odd	Odd	20 cycles

Figure 9.4.1.1. Interrupt response time

9.4.2 Variation of IPL when Interrupt Request is Accepted

When a maskable interrupt request is accepted, the interrupt priority level of the accepted interrupt is set in the IPL.

When a software interrupt or special interrupt request is accepted, one of the interrupt priority levels listed in Table 9.4.2.1 is set in the IPL. Shown in Table 9.4.2.1 are the IPL values of software and special interrupts when they are accepted.

Table 9.4.2.1. IPL Level That is Set to IPL When A Software or Special Interrupt Is Accepted

Interrupt sources	Level that is set to IPL
Watchdog timer, $\overline{\text{NMI}}$, Oscillation stop and re-oscillation detection, voltage down detection	7
Software, address match, $\overline{\text{DBC}}$, single-step	Not changed

9.4.3 Saving Registers

In the interrupt sequence, the FLG register and PC are saved to the stack.

At this time, the 4 high-order bits of the PC and the 4 high-order (IPL) and 8 low-order bits in the FLG register, 16 bits in total, are saved to the stack first. Next, the 16 low-order bits of the PC are saved. Figure 9.4.3.1 shows the stack status before and after an interrupt request is accepted.

The other necessary registers must be saved in a program at the beginning of the interrupt routine. Use the PUSHM instruction, and all registers except SP can be saved with a single instruction.

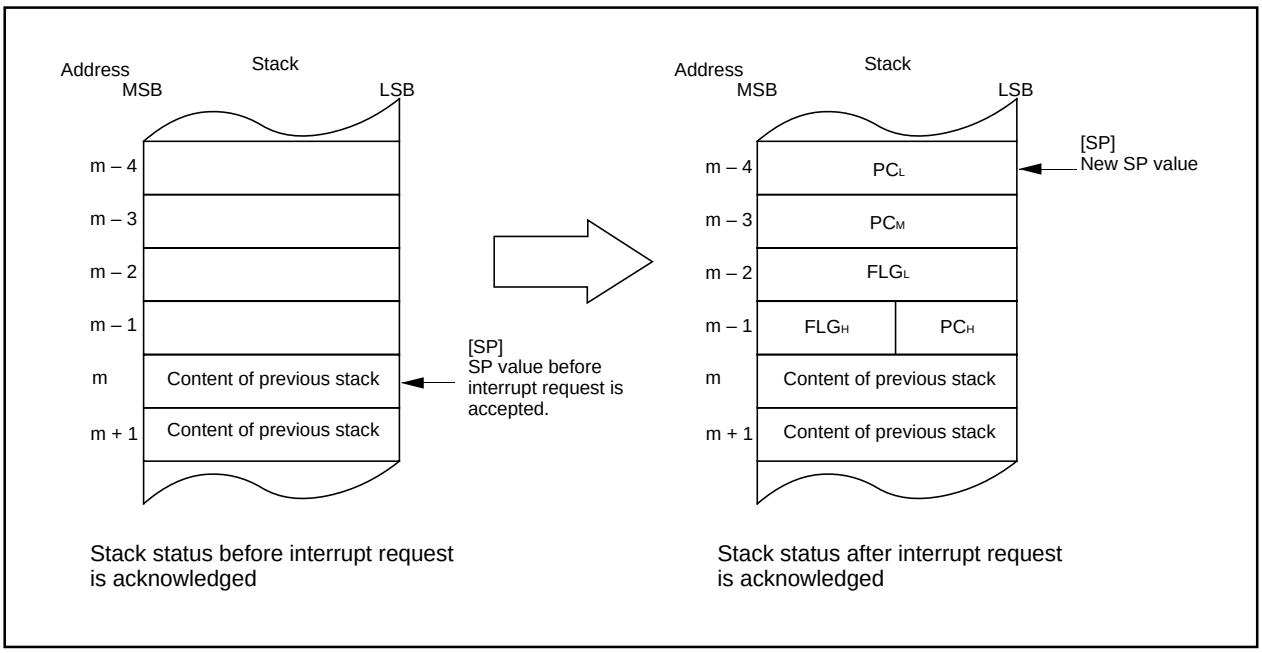


Figure 9.4.3.1. Stack Status Before and After Acceptance of Interrupt Request

11.5 Channel Priority and DMA Transfer Timing

If both DMA0 and DMA1 are enabled and DMA transfer request signals from DMA0 and DMA1 are detected active in the same sampling period (one period from a falling edge to the next falling edge of CPU clock), the DMAS bit on each channel is set to “1” (DMA requested) at the same time. In this case, the DMA requests are arbitrated according to the channel priority, DMA0 > DMA1. The following describes DMAC operation when DMA0 and DMA1 requests are detected active in the same sampling period. Figure 11.5.1 shows an example of DMA transfer effected by external factors.

DMA0 request having priority is received first to start a transfer when a DMA0 request and DMA1 request are generated simultanelously. After one DMA0 transfer is completed, a bus arbitration is returned to the CPU. When the CPU has completed one bus access, a DMA1 transfer starts. After one DMA1 transfer is completed, the bus arbitration is again returned to the CPU.

In addition, DMA requsts cannot be counted up since each channel has one DMAS bit. Therefore, when DMA requests, as DMA1 in Figure 11.5.1, occurs more than one time, the DAMS bit is set to "0" as soon as getting the bus arbitration. The bus arbitration is returned to the CPU when one transfer is completed.

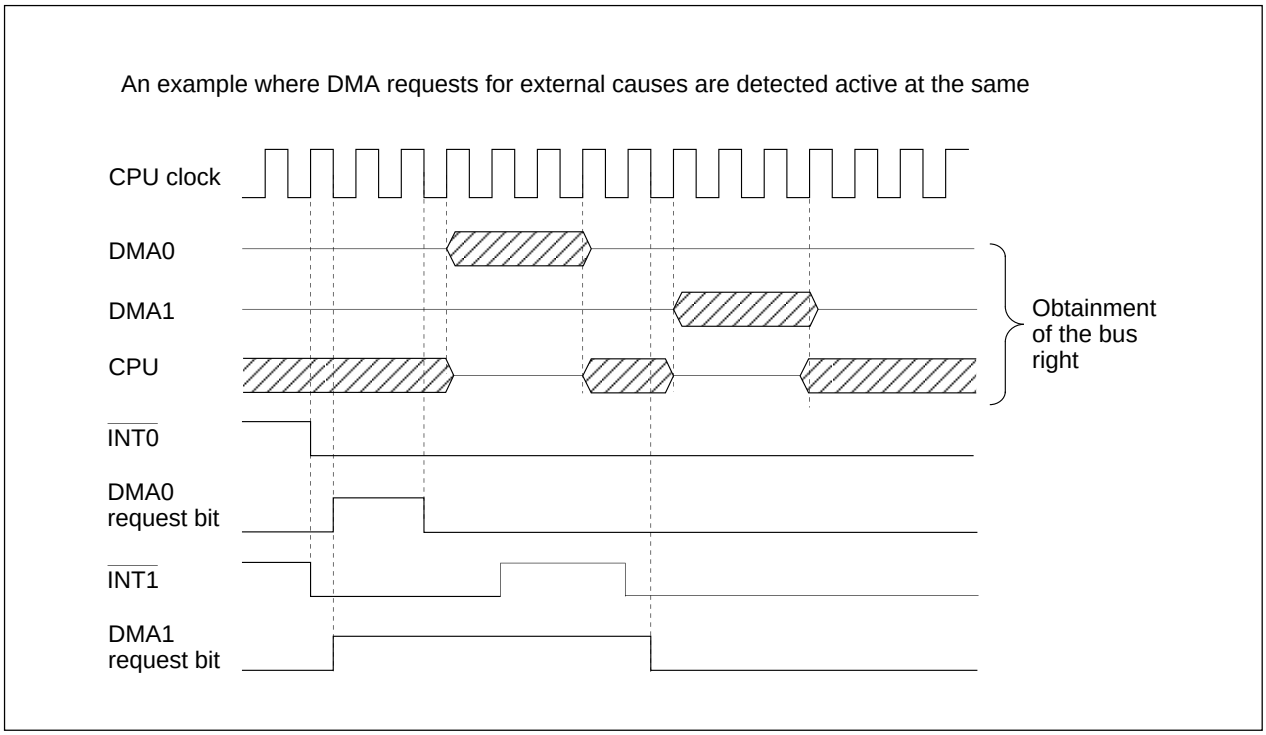


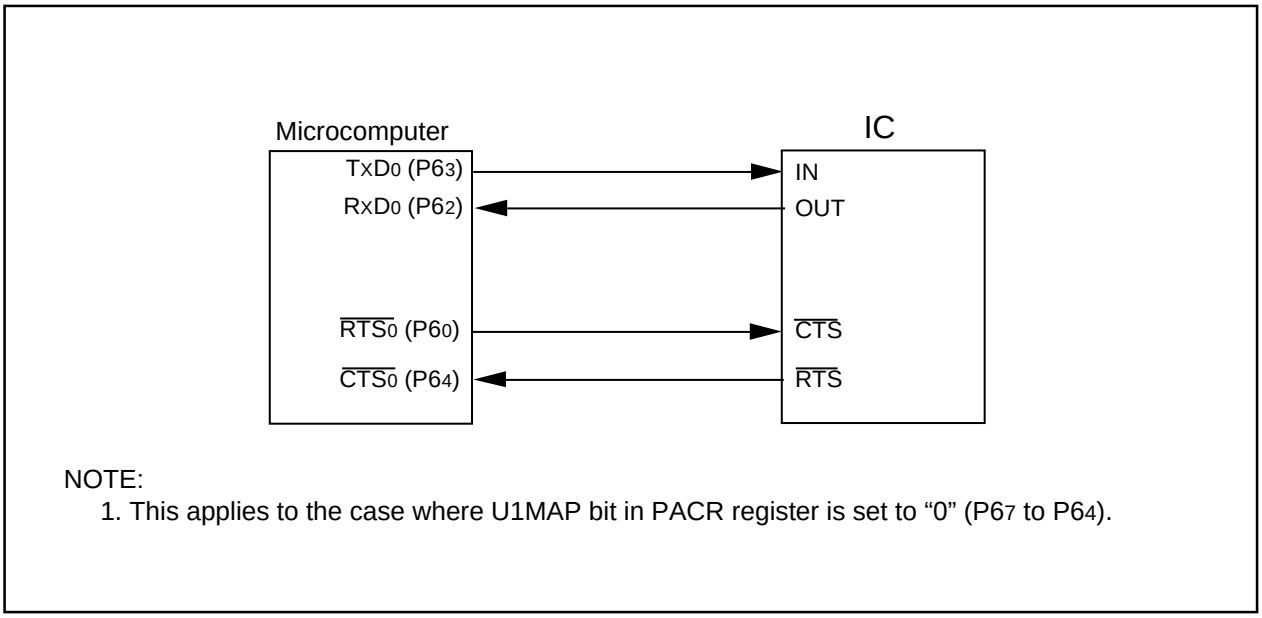
Figure 11.5.1 DMA Transfer by External Factors

13.1.2.6. $\overline{\text{CTS}}/\overline{\text{RTS}}$ Separate Function (UART0)

This function separates $\overline{\text{CTS}}_0/\overline{\text{RTS}}_0$, outputs $\overline{\text{RTS}}_0$ from the $P6_0$ pin, and accepts as input the $\overline{\text{CTS}}_0$ from the $P6_4$ pin. To use this function, set the register bits as shown below.

- Set the CRD bit in the U0C0 register to "0" (enables UART0 $\overline{\text{CTS}}/\overline{\text{RTS}}$)
- Set the CRS bit in the U0C0 register to "1"(outputs UART0 $\overline{\text{RTS}}$)
- Set the CRD bit in the U1C0 register to "0" (enables UART1 $\overline{\text{CTS}}/\overline{\text{RTS}}$)
- Set the CRS bit in the U1C0 register to "0" (inputs UART1 $\overline{\text{CTS}}$)
- Set the RCSP bit in the UCON register to "1" (inputs $\overline{\text{CTS}}_0$ from the $P6_4$ pin)
- Set the CLKMD1 bit in the UCON register to "0" ($CLKS_1$ not used)

Note that when using the $\overline{\text{CTS}}/\overline{\text{RTS}}$ separate function, UART1 $\overline{\text{CTS}}/\overline{\text{RTS}}$ separate function cannot be used.



NOTE:
1. This applies to the case where U1MAP bit in PACR register is set to “0” ($P6_7$ to $P6_4$).

Figure 13.1.2.6.1. $\overline{\text{CTS}}/\overline{\text{RTS}}$ Separate Function

Figure 13.1.6.2 shows the example of connecting the SIM interface. Connect TxD2 and RxD2 and apply pull-up.

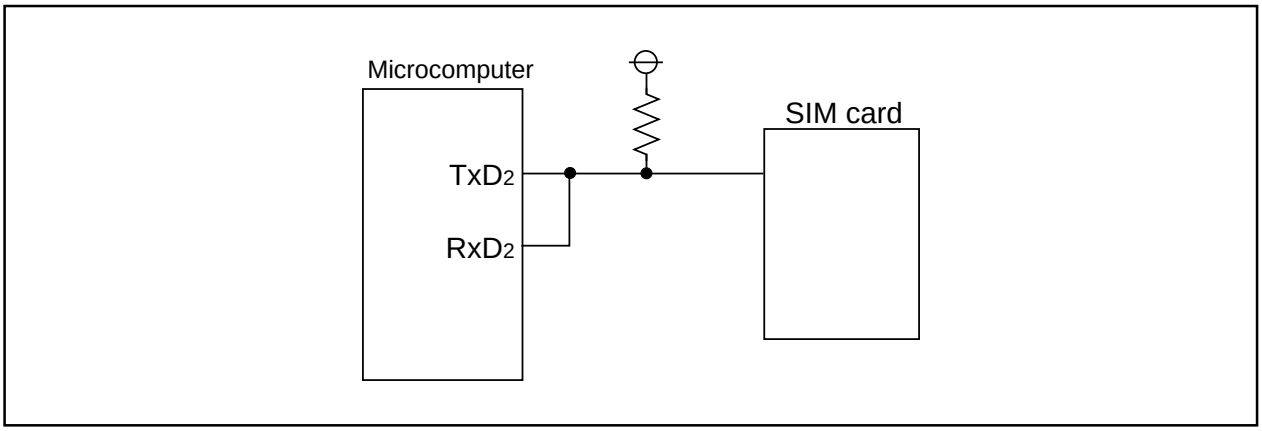


Figure 13.1.6.2. SIM Interface Connection

13.1.6.1 Parity Error Signal Output

The parity error signal is enabled by setting the U2ERE bit in the U2C1 register' to "1".

• When receiving

The parity error signal is output when a parity error is detected while receiving data. This is achieved by pulling the TxD2 output low with the timing shown in Figure 13.1.6.1.1. If the R2RB register is read while outputting a parity error signal, the PER bit is cleared to "0" and at the same time the TxD2 output is returned high.

• When transmitting

A transmission-finished interrupt request is generated at the falling edge of the transfer clock pulse that immediately follows the stop bit. Therefore, whether a parity signal has been returned can be determined by reading the port that shares the RxD2 pin in a transmission-finished interrupt service routine.

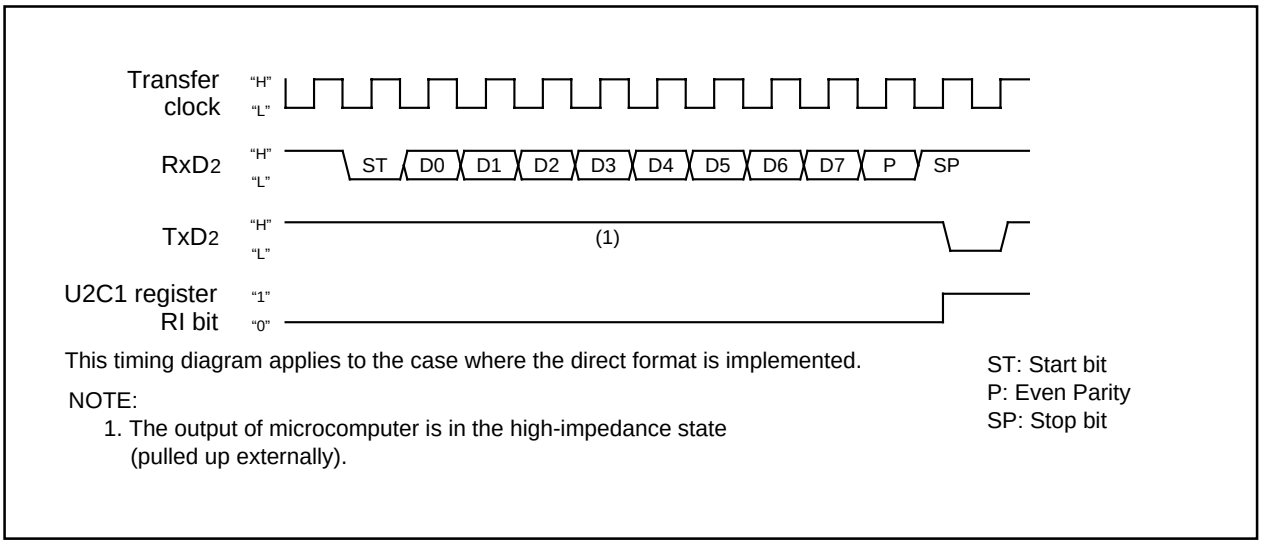


Figure 13.1.6.1.1. Parity Error Signal Output Timing

14.1.2 Repeat mode

In repeat mode, analog voltage applied to a selected pin is repeatedly converted to a digital code. Table 14.1.2.1 shows the repeat mode specifications. Figure 14.1.2.1 shows the operation example in repeat mode. Figure 14.1.2.2 shows the ADCON0 to ADCON2 registers in repeat mode.

Table 14.1.2.1 Repeat Mode Specifications

Item	Specification
Function	The CH2 to CH0 bits in the ADCON0 register and the ADGSEL1 to ADGSEL0 bits in the ADCON2 register select pins. Analog voltage applied to a selected pin is repeatedly converted to a digital code
A/D Conversion Start Condition	• When the TRG bit in the ADCON0 register is “0” (software trigger) Set the ADST bit in the ADCON0 register to “1” (A/D conversion started) • When the TRG bit in the ADCON0 register is “1” (hardware trigger) The $\overline{\text{ADTRG}}$ pin input changes state from “H” to “L” after setting the ADST bit to “1” (A/D conversion started)
A/D Conversion Stop Condition	Set the ADST bit to “0” (A/D conversion halted)
Interrupt Request Generation Timing	None generated
Analog Input Pin	Select one pin from AN_0 to AN_7, AN_{30} to AN_{32} and AN_{24}
Readout of A/D Conversion Result	Readout one of the AD0 to AD7 registers that corresponds to the selected pin

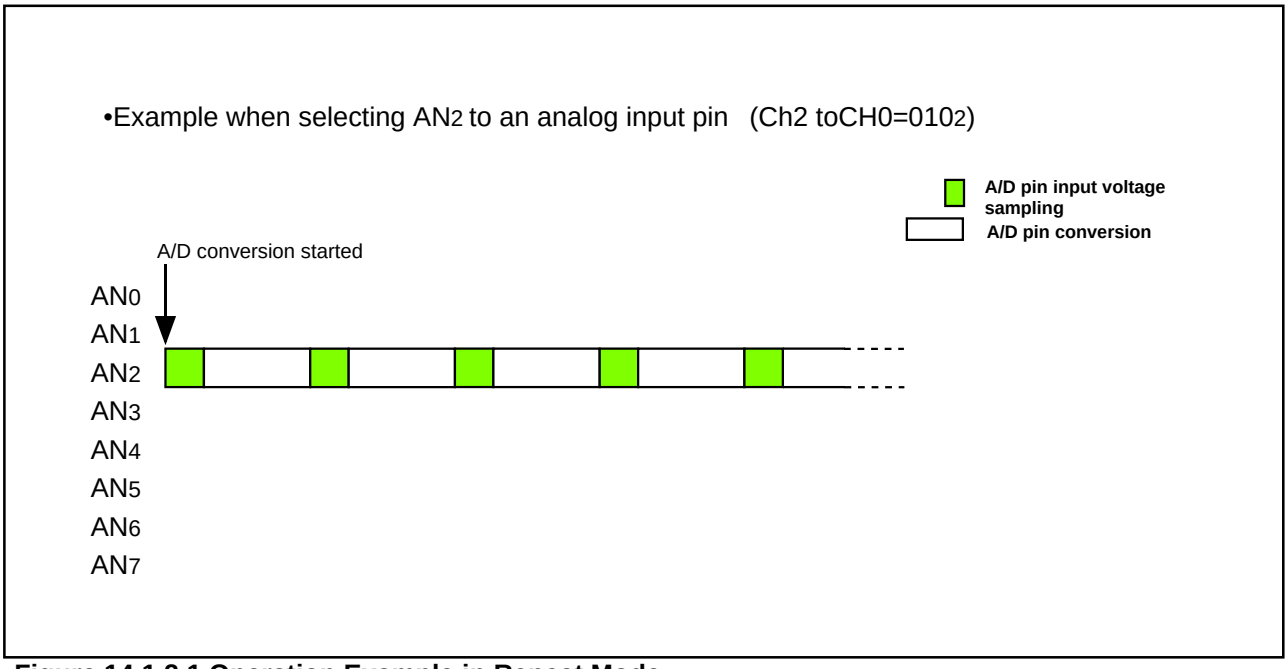
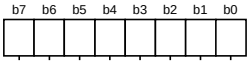


Figure 14.1.2.1 Operation Example in Repeat Mode

Port Pi register (i=6 to 8 and 10) (1)

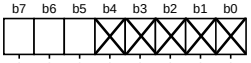


Symbol	Address	After reset
P6 to P8	03EC16, 03ED16, 03F016	Indeterminate
P10	03F416	Indeterminate

Bit symbol	Bit name	Function	RW
Pi_0	Port Pi0 bit	The pin level on any I/O port which is set for input mode can be read by reading the corresponding bit in this register. The pin level on any I/O port which is set for output mode can be controlled by writing to the corresponding bit in this register 0 : "L" level 1 : "H" level (1) (i = 6 to 8 and 10)	RW
Pi_1	Port Pi1 bit		RW
Pi_2	Port Pi2 bit		RW
Pi_3	Port Pi3 bit		RW
Pi_4	Port Pi4 bit		RW
Pi_5	Port Pi5 bit		RW
Pi_6	Port Pi6 bit		RW
Pi_7	Port Pi7 bit		RW

NOTE:
1. Ports must be enabled using the PACR register.
In 48-pin package, set PACR2, PACR1, PACR0 to "100$_2$"
In 42-pin package, set PACR2, PACR1, PACR0 to "001$_2$"

Port P1 register (1)

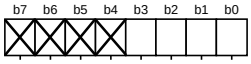


Symbol	Address	After reset
P1	03E116	Indeterminate

Bit symbol	Bit name	Function	RW
$\overline{\text{(b4-b0)}}$	Nothing is assigned. In an attempt to write to this bit, write "0". The value, if read, turns out to be indeterminate.		—
P1_5	Port P15 bit	The pin level on any I/O port which is set for input mode can be read by reading the corresponding bit in this register. The pin level on any I/O port which is set for output mode can be controlled by writing to the corresponding bit in this register 0 : "L" level 1 : "H" level	RW
P1_6	Port P16 bit		RW
P1_7	Port P17 bit		RW

NOTE:
1. Ports must be enabled using the PACR register.
In 48-pin package, set PACR2, PACR1, PACR0 to "100$_2$"
In 42-pin package, set PACR2, PACR1, PACR0 to "001$_2$"

Port P9 register (1)



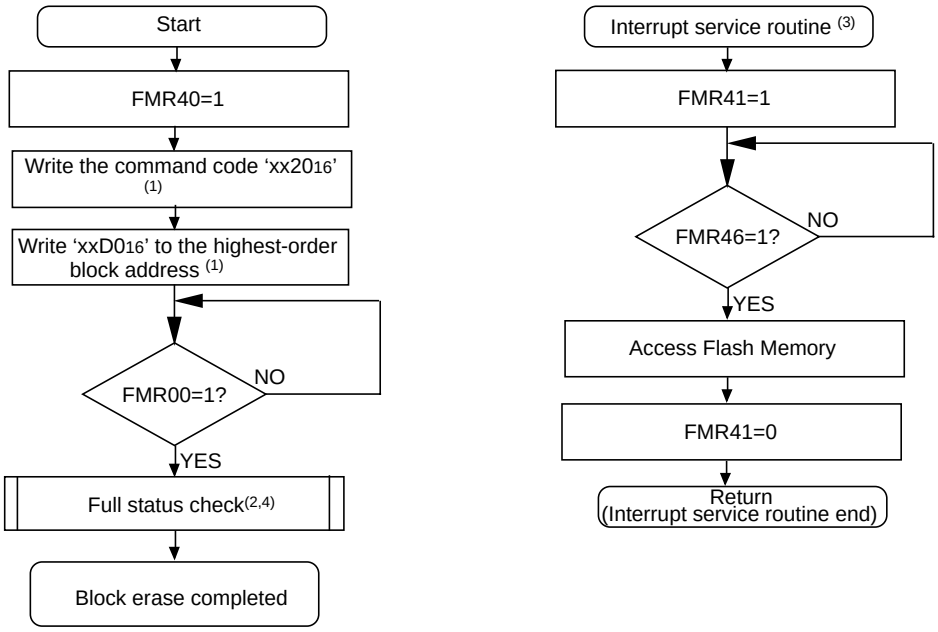
Symbol	Address	After reset
P9	03F116	Indeterminate

Bit symbol	Bit name	Function	RW
P9_0	Port P90 bit	The pin level on any I/O port which is set for input mode can be read by reading the corresponding bit in this register. The pin level on any I/O port which is set for output mode can be controlled by writing to the corresponding bit in this register 0 : "L" level 1 : "H" level	RW
P9_1	Port P91 bit		RW
P9_2	Port P92 bit		RW
P9_3	Port P93 bit		RW
$\overline{\text{(b7-b4)}}$	Nothing is assigned. In an attempt to write to this bit, write "0". The value, if read, turns out to be indeterminate.		—

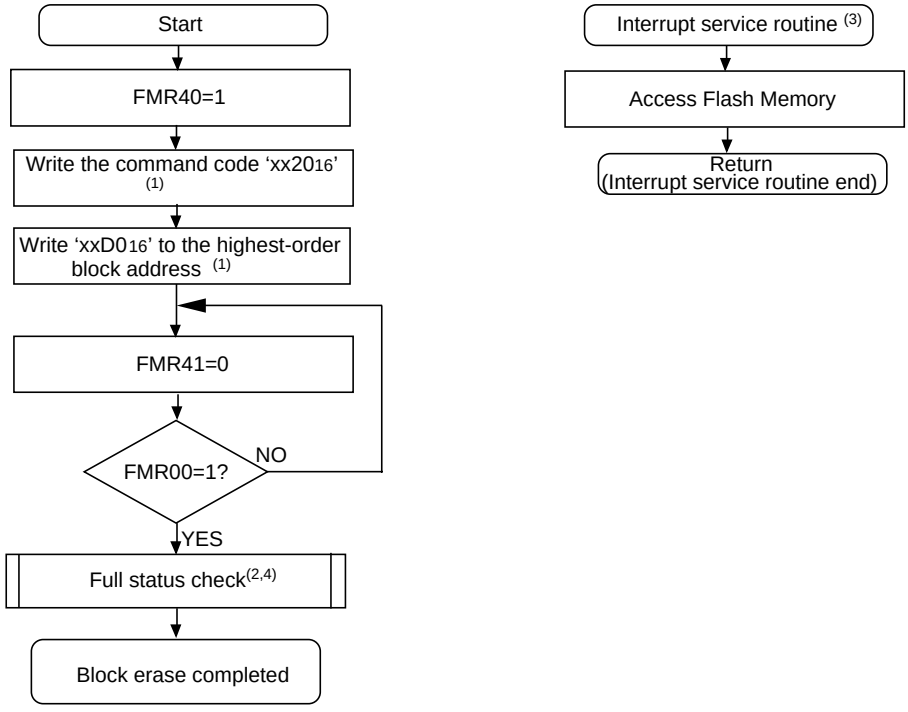
NOTE:
1. Ports must be enabled using the PACR register.
In 48-pin package, set PACR2, PACR1, PACR0 to "100$_2$"
In 42-pin package, set PACR2, PACR1, PACR0 to "001$_2$"

Figure 16.2.1. P1, P6, P7, P8, P9, and P10 Registers

(EW0 mode)



(EW1 mode)

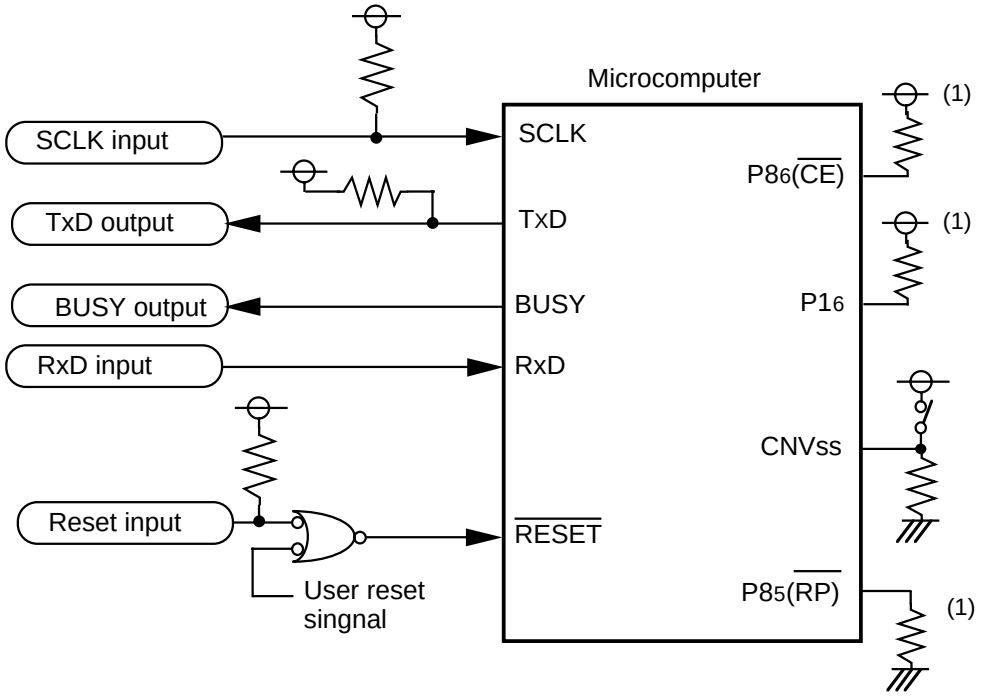


NOTES:
1. Write the command code and data to even address.
2. Execute the clear status register command and block erase command at least 3 times until an erase error is not generated when an erase error is generated.
3. In EW0 mode, allocate an interrupt vector table of an interrupt, to be used, to a RAM area.
4. Refer to "**Figure 17.8.4.1 Full Status Check and Handling Procedure for Each Error**."

Figure 17.7.5.2. Block Erase Command (at use erase suspend)

17.9.2 Example of Circuit Application in Standard Serial I/O Mode

Figure 17.9.2.1 shows an example of a circuit application in standard serial I/O mode 1 and Figure 17.9.2.2 shows an example of a circuit application in standard serial I/O mode 2. Refer to the user's manual for a serial writer to handle pins controlled by the serial writer.



(1) Controlling pins and external circuits vary with the serial programmer. For more information, refer to the user's manual included with the serial programmer.
(2) In this example, a selector controls the input voltage applied to CNVss to switch between single-chip mode and standard serial I/O mode.
(3) In standard serial input/output mode 1, if the user reset signal becomes “L” while the microcomputer is communicating with the serial programmer, break the connection between the user reset signal and the RESET pin using a jumper switch.

NOTE:
1. Set following either or both
   - Connect the CE pin to VCC
   - Connect the RP pin to VSS and the P16 pin to VCC

Figure 17.9.2.1. Circuit Application in Standard Serial I/O Mode 1

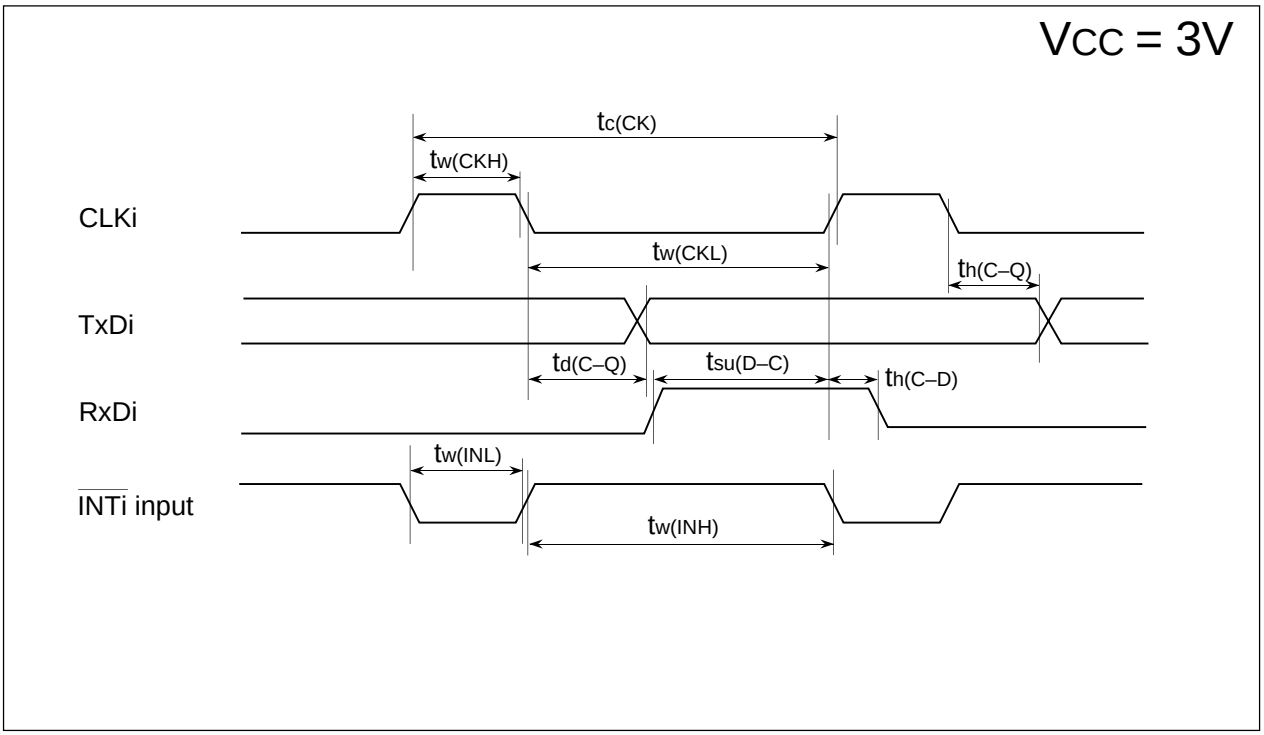


Figure 18.4. Timing Diagram (2)

19.5.6 Rewrite the Interrupt Control Register

(1) The interrupt control register for any interrupt should be modified in places where no requests for that interrupt may occur. Otherwise, disable the interrupt before rewriting the interrupt control register.

(2) To rewrite the interrupt control register for any interrupt after disabling that interrupt, be careful with the instruction to be used.

Changing any bit other than the IR bit

If while executing an instruction, a request for an interrupt controlled by the register being modified occurs, the IR bit in the register may not be set to “1” (interrupt requested), with the result that the interrupt request is ignored. If such a situation presents a problem, use the instructions shown below to modify the register.

Usable instructions: AND, OR, BCLR, BSET

Changing the IR bit

Depending on the instruction used, the IR bit may not always be cleared to “0” (interrupt not requested). Therefore, be sure to use the MOV instruction to clear the IR bit.

(3) When using the I flag to disable an interrupt, refer to the sample program fragments shown below as you set the I flag. (Refer to (2) for details about rewrite the interrupt control registers in the sample program fragments.)

Examples 1 through 3 show how to prevent the I flag from being set to “1” (interrupts enabled) before the interrupt control register is rewritten, due to the internal bus and the instruction queue buffer timing.

Example 1: Using the NOP instruction to keep the program waiting until the interrupt control register is modified

```
INT_SWITCH1:
        FCLR    I               ; Disable interrupts
        AND.B   #00h, 0055h     ;Set the TA0IC register to 00₁₆
        NOP                     ;
        NOP
        FSET    I               ; Enable interrupts
```

The number of NOP instruction is as follows.
PM20 = 1 (1 wait) : 2, PM20 = 0 (2 waits): 3

Example 2:Using the dummy read to keep the FSET instruction waiting

```
INT_SWITCH2:
        FCLR    I               ; Disable interrupts
        AND.B   #00h, 0055h     ; Set the TA0IC register to 00₁₆
        MOV.W   MEM, R0         ; Dummy read
        FSET    I               ; Enable interrupts
```

Example 3:Using the POPC instruction to changing the I flag

```
INT_SWITCH3:
        PUSHC   FLG
        FCLR    I               ; Disable interrupts
        AND.B   #00h, 0055h     ; Set the TA0IC register to 00₁₆
        POPC    FLG             ; Enable interrupts
```

19.5.7 Watchdog Timer Interrupt

Initialize the watchdog timer after the watchdog timer interrupt occurs.

19.6 DMAC

19.6.1 Write to DMAE Bit in DMiCON Register

When both of the conditions below are met, follow the steps below.

Conditions

- The DMAE bit is set to “1” again while it remains set (DMAi is in an active state).
- A DMA request may occur simultaneously when the DMAE bit is being written.

Step 1: Write “1” to the DMAE bit and DMAS bit in DMiCON register simultaneously(*1).
Step 2: Make sure that the DMAi is in an initial state(*2) in a program.
If the DMAi is not in an initial state, the above steps should be repeated.

Notes:

1. The DMAS bit remains unchanged even if “1” is written. However, if “0” is written to this bit, it is set to “0” (DMA not requested). In order to prevent the DMAS bit from being modified to “0”, “1” should be written to the DMAS bit when “1” is written to the DMAE bit. In this way the state of the DMAS bit immediately before being written can be maintained.
   Similarly, when writing to the DMAE bit with a read-modify-write instruction, “1” should be written to the DMAS bit in order to maintain a DMA request which is generated during execution.

2. Read the TCRi register to verify whether the DMAi is in an initial state. If the read value is equal to a value which was written to the TCRi register before DMA transfer start, the DMAi is in an initial state. (If a DMA request occurs after writing to the DMAE bit, the value written to the TCRi register is “1”.) If the read value is a value in the middle of transfer, the DMAi is not in an initial state.

Appendix 2.2 Differences between M16C/26A Group and M16C/26 Group

Item	M16C/26A Group	M16C/26 Group
Clock Generation Circuit	4 circuits (Main clock oscillation circuit, Sub clock oscillation circuit, on-chip oscillator, PLL frequency synthesizer)	3 circuits (Main clock oscillation circuit, Sub clock oscillation circuit, on-chip oscillator)
System Clock Source After Reset (Initial value of the CM21 bit in the CM2 register)	On-chip oscillator (Initial value "1" of CM21 bit)	Main clock (Initial value "0" of CM21 bit)
On-chip Oscillator Clock	Selectable (8MHz/1MHz/500KHz)	Fixed (1MHz)
PACR2 to PACR0 in the PACR register	Necessary to set after reset 48pin:"100$_2$", 42pin:"001$_2$"	No PACR register
IFSR20 bit in the IFSR2A register	Necessary to set to "1" after reset	No IFSR2A register
External Interrupt	8 causes ($\overline{INT2}$ added)	7 causes
13 pin (48-pin version) Function	P8$_4$/$\overline{INT2}$/ZP	IV$_{CC}$
P7$_0$, P7$_1$	N-ch open drain output and CMOS output are selectable by S/W	N-ch open drain output
A/D Input Pin (48-pin version)	12 channels	8 channels
A/D operation Mode	8 modes (single, repeat, single sweep, repeat sweep mode 0, repeat sweep mode 1, simultaneous sampling, delayed trigger mode 0, delayed trigger mode 1) 1 shunt current measurement function is available	5 modes (single, repeat, single sweep, repeat sweep mode 0, repeat sweep mode 1)
Timer B Operation Mode	5 modes (timer, event counter, pulse periods measurement, pulse width measurment, A/D trigger) 1 shunt current measurement function is available	4 modes (timer, event counter, pulse periods measurement, pulse width measurment)
CRC Calculation	Available (compatible to CRC-CCITT and CRC-16 methods)	Not available
Three-phase motor Control	•Waveform output/Switching port output by software is enabled •Position data retention function	•Waveform output/Switching port output by software is disabled •No position data retention function
Digital Debounce Function	This function is in the $\overline{NMI}$/$\overline{SD}$ pin and $\overline{INT5}$ pin	Not available
3 pin (48-pin version) function	P9$_0$/CLK$_{OUT}$/TB0$_{IN}$/AN3$_0$ (CLK$_{OUT}$: f1, f8, f32, and f$_C$ output)	P9$_0$/TB0$_{IN}$
UART1 Compatible pin	Switching to P6$_4$ to P6$_7$ or P7$_0$ to P7$_3$ is enabled	P6$_4$ to P6$_7$
Flash Memory Protect Function	Protection to blocks 0, 1 by FMR02 bit Protection to the blocks 0 to 3 by FMR16 bit	Protection to blocks 0,1 by FMR02 bit
Package	PLQP0048KB-A(48P6Q), PRSP0042GA-B(42P2R)	PLQP0048KB-A(48P6Q)

NOTE:

1. Since the emulator between the M16C/26A Group and M16C/29 Group are the same, all functions of M16C/29 are built in the emulator. When evaluating M16C/26A Group, do not access to the SFR which is not built in M16C/26A Group. Refer to Hardware Manual about detail and electrical characteristics.

Register Index