To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. 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.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. 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.
5. When exporting the 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. You should not use Renesas Electronics products or the 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. 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.
6. 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.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. 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 categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. 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 an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   "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.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. 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.
9. 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 system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. 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.
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.

To all our customers

Regarding the change of names mentioned in the document, such as Mitsubishi Electric and Mitsubishi XX, to Renesas Technology Corp.

The semiconductor operations of Hitachi and Mitsubishi Electric were transferred to Renesas Technology Corporation on April 1st 2003. These operations include microcomputer, logic, analog and discrete devices, and memory chips other than DRAMs (flash memory, SRAMs etc.) Accordingly, although Mitsubishi Electric, Mitsubishi Electric Corporation, Mitsubishi Semiconductors, and other Mitsubishi brand names are mentioned in the document, these names have in fact all been changed to Renesas Technology Corp. Thank you for your understanding. Except for our corporate trademark, logo and corporate statement, no changes whatsoever have been made to the contents of the document, and these changes do not constitute any alteration to the contents of the document itself.

Note : Mitsubishi Electric will continue the business operations of high frequency & optical devices and power devices.

Renesas Technology Corp.
Customer Support Dept.
April 1, 2003

Usage Notes Reference Book

16

M16C/62 (M16C/62N, M3062GF8NFP/GP) Group

Usage Notes Reference Book

Renesas 16-BIT SINGLE-CHIP MICROCOMPUTER M16C FAMILY / M16C/60 SERIES

Renesas Electronics
www.renesas.com

2002.06

Keep safety first in your circuit designs!

- Mitsubishi Electric Corporation puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage. Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of non-flammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

- These materials are intended as a reference to assist our customers in the selection of the Mitsubishi semiconductor product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Mitsubishi Electric Corporation or a third party.
- Mitsubishi Electric Corporation assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
- All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Mitsubishi Electric Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Mitsubishi Electric Corporation or an authorized Mitsubishi Semiconductor product distributor for the latest product information before purchasing a product listed herein.
  The information described here may contain technical inaccuracies or typographical errors.
  Mitsubishi Electric Corporation assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
  Please also pay attention to information published by Mitsubishi Electric Corporation by various means, including the Mitsubishi Semiconductor home page (http://www.mitsubishichips.com).
- When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Mitsubishi Electric Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
- Mitsubishi Electric Corporation semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Mitsubishi Electric Corporation or an authorized Mitsubishi Semiconductor product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
- The prior written approval of Mitsubishi Electric Corporation is necessary to reprint or reproduce in whole or in part these materials.
- If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
  Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
- Please contact Mitsubishi Electric Corporation or an authorized Mitsubishi Semiconductor product distributor for further details on these materials or the products contained therein.

Preface

This book describes the M16C/62 (M16C/62N, M3062GF8NFP/GP) group's precautions for use, which contains paragraphs describing precautions of the user's manual and technical news relevant to these paragraphs. Please refer to this book when developing your systems. However, all of precautions are not contained in this book, please perform sufficient evaluation under systems development.

1.1 Precautions for Interrupts

1.1.1 Reading address 00000$_{16}$

When maskable interrupt is occurred, CPU reads the interrupt information (the interrupt number and interrupt request level) in the interrupt sequence.

The interrupt request bit of the certain interrupt written in address 00000$_{16}$ will then be set to "0".

Even if the address 00000$_{16}$ is read out by software, "0" is set to the enabled highest priority interrupt source request bit. Therefore interrupt can be canceled and unexpected interrupt can occur.

Do not read address 00000$_{16}$ by software.

1.1.2 Setting the stack pointer

The value of the stack pointer immediately after reset is initialized to 0000$_{16}$. Accepting an interrupt before setting a value in the stack pointer may become a factor of runaway. Be sure to set a value in the stack pointer before accepting an interrupt. When using the $\overline{\text{NMI}}$ interrupt, initialize the stack pointer at the beginning of a program. Concerning the first instruction immediately after reset, generating any interrupts including the $\overline{\text{NMI}}$ interrupt is prohibited.

1.1.3 The $\overline{\text{NMI}}$ interrupt

1. The $\overline{\text{NMI}}$ interrupt can not be disabled. Be sure to connect $\overline{\text{NMI}}$ pin to Vcc via a pull-up resistor if unused. Be sure to work on it.
2. The $\overline{\text{NMI}}$ pin also serves as P8$_5$, which is exclusively input. Reading the contents of the P8 register allows reading the pin value. Use the reading of this pin only for establishing the pin level at the time when the $\overline{\text{NMI}}$ interrupt is input.
3. Do not attempt to go into stop mode with the input to the $\overline{\text{NMI}}$ pin being in the "L" state. With the input to the $\overline{\text{NMI}}$ being in the "L" state, the CM10 is fixed to "0", so attempting to go into stop mode is turned down.
4. Do not attempt to go into wait mode with the input to the $\overline{\text{NMI}}$ pin being in the "L" state. With the input to the $\overline{\text{NMI}}$ pin being in the "L" state, the CPU stops but the oscillation does not stop, so no power is saved. In this instance, the CPU is returned to the normal state by a later interrupt.
5. Signals input to the $\overline{\text{NMI}}$ pin require "L" level and "H" level of 2 clock +300ns or more, from the operation clock of the CPU.

1.1.4 External interrupt

1. Either an "L" level or an "H" level of at least 250 ns width is necessary for the signal input to pins $\overline{\text{INT}_0}$ through $\overline{\text{INT}_5}$ regardless of the CPU operation clock(M16C/62N). Either an "L" level or an "H" level of at least 380 ns width is necessary for the signal input to pins $\overline{\text{INT}_0}$ through $\overline{\text{INT}_5}$ regardless of the CPU operation clock (M3062GF8NFP/GP).
2. When the polarity of the $\overline{\text{INT}_0}$ to $\overline{\text{INT}_5}$ pins is changed or the interrupt request cause of the software interrupt numbers 8 to 9 is changed, the interrupt request bit is sometimes set to "1". After these changes were made, set the interrupt request bit to "0". Figure 1.1.1 shows the procedure for changing the $\overline{\text{INT}}$ interrupt generate factor.

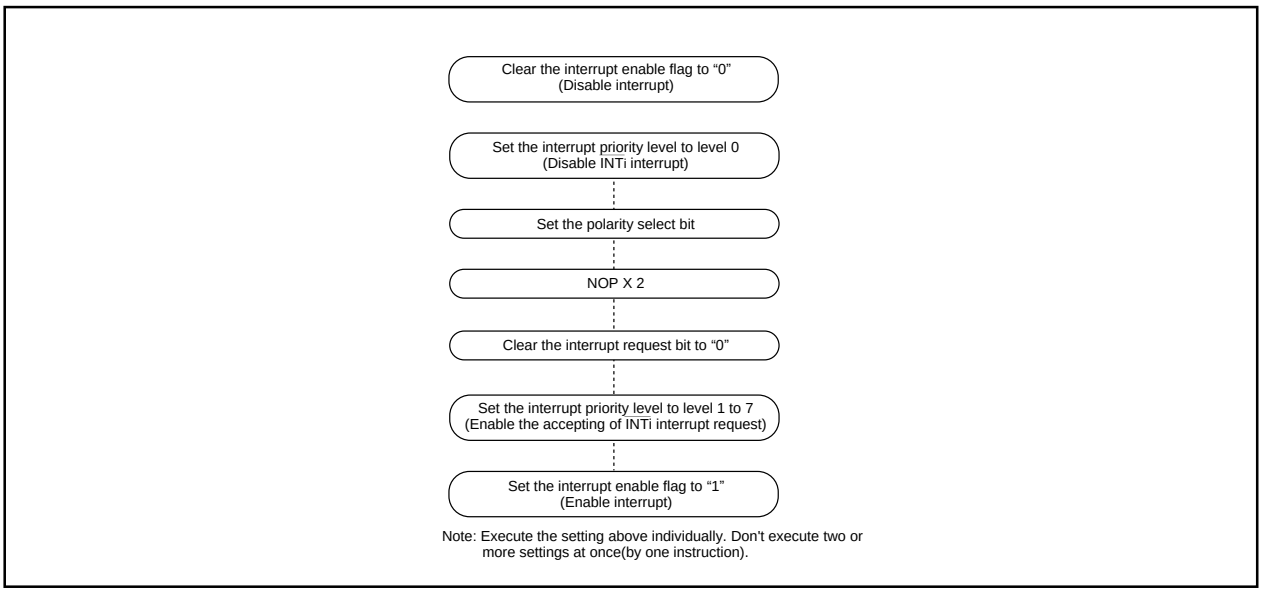


Figure 1.1.1. Switching condition of $\overline{\text{INT}}$ interrupt request

1.1.5 Watchdog timer interrupt (M16C/62N)

Write to the watchdog timer start register after the watchdog timer interrupt occurs (initialize watchdog timer).

1.1.6 Rewrite the interrupt control register

To rewrite the interrupt control register, do so at a point that does not generate the interrupt request for that register. If there is possibility of the interrupt request occur, rewrite the interrupt control register after the interrupt is disabled. The program examples are described as follow:

Example 1:

```
INT_SWITCH1:
    FCLR    I             ; Disable interrupts.
    AND.B   #00h, 0055h   ; Clear TA0IC int. priority level and int. request bit.
    NOP                   ; Four NOP instructions are required when using HOLD function.
    NOP
    FSET    I             ; Enable interrupts.
```

Example 2:

```
INT_SWITCH2:
    FCLR    I             ; Disable interrupts.
    AND.B   #00h, 0055h   ; Clear TA0IC int. priority level and int. request bit.
    MOV.W   MEM, R0       ; Dummy read.
    FSET    I             ; Enable interrupts.
```

Example 3:

```
INT_SWITCH3:
    PUSHC   FLG           ; Push Flag register onto stack
    FCLR    I             ; Disable interrupts.
    AND.B   #00h, 0055h   ; Clear TA0IC int. priority level and int. request bit.
    POPC    FLG           ; Enable interrupts.
```

The reason why two NOP instructions (four when using the HOLD function) or dummy read are inserted before FSET I in Examples 1 and 2 is to prevent the interrupt enable flag I from being set before the interrupt control register is rewritten due to effects of the instruction queue.

When changing an interrupt control register in a sate of interrupts being disabled, please read the following precautions on instructions used before changing the register.

(1) Changing a non-interrupt request bit

If an interrupt request for an interrupt control register is generated during an instruction to rewrite the register is being executed, there is a case that the interrupt request bit is not set and consequently the interrupt is ignored. This will depend on the instruction. If this creates problems, use the below instructions to change the register.

Instructions : AND, OR, BCLR, BSET

(2) Changing the interrupt request bit

When attempting to clear the interrupt request bit of an interrupt control register, the interrupt request bit is not cleared sometimes. This will depend on the instruction. If this creates problems, use the below instructions to change the register.

Instructions : MOV

1.2 Precautions for Protect

The write-enable bit of port 9 direction register and SI/Oi control register (i=3,4) goes to “0” when the next write instruction is executed after write-enabled state is readied. Make changes in input/output and SI/Oi control register (i=3,4) immediately after the instruction that sets “1” in the write-enable bit of port P9 direction register and SI/Oi control register (i=3,4)(avoid causing an interrupt). Also take measures to prevent DMA transfer from being executed.

Precautions for Timer A

1.3.1 Precautions for Timer A (timer mode)

1. To clear reset, the count start flag is set to “0”. Set a value in the timer Ai register, then set the flag to “1”.

2. Reading the timer Ai register while a count is in progress allows reading, with arbitrary timing, the value of the counter. Reading the timer Ai register with the reload timing shown in Figure 1.3.1 gets “$FFFF_{16}$”. Reading the timer Ai register after setting a value in the timer Ai register with a count halted but before the counter starts counting gets a proper value.

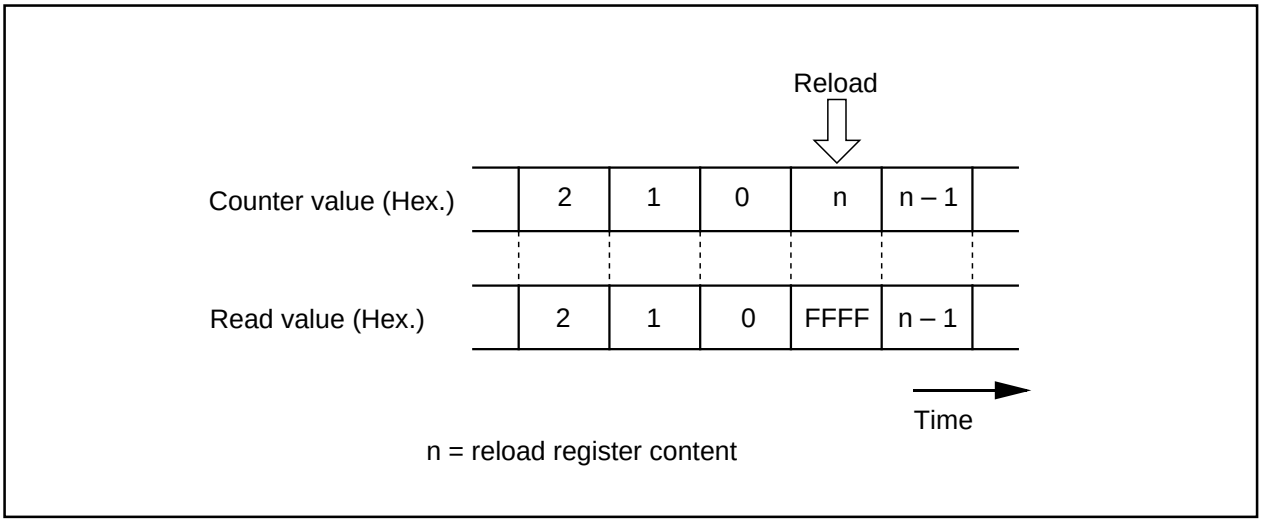


Figure 1.3.1. Reading timer Ai register

1.3.2 Precautions for Timer A (event counter mode)

1. To clear reset, the count start flag is set to “0”. Set a value in the timer Ai register, then set the flag to “1”.

2. Reading the timer Ai register while a count is in progress allows reading, with arbitrary timing, the value of the counter. Reading the timer Ai register with the reload timing shown in Figure 1.3.2 gets “$FFFF_{16}$” by underflow or “0000_{16}” by overflow. Reading the timer Ai register after setting a value in the timer Ai register with a count halted but before the counter starts counting gets a proper value.

3. Please note the standards for the differences between the 2 pulses used in the 2-phase pulse signals input signals to the TAiIN pin and TAiOUT pin (i = 2, 3, 4), as shown in Figure 1.3.3.

4. When free run type is selected, if count is stopped, set a value in the timer Ai register again.

5. In the case of using as “Free-Run type”, the timer register contents may be unknown when counting begins. If the timer register is set before counting has started, then the starting value will be unknown.
   - **In the case where the up/down count will not be changed.**
     Enable the “Reload” function and write to the timer register before counting begins. Rewrite the value to the timer register immediately after counting has started. If counting up, rewrite “0000_{16}” to the timer register. If counting down, rewrite “$FFFF_{16}$” to the timer register. This will cause the same operation as “Free-Run type” mode.
   - **In the case where the up/down count has changed.**
     First set to “Reload type” operation. Once the first counting pulse has occurred, the timer may be changed to “Free-Run type”.

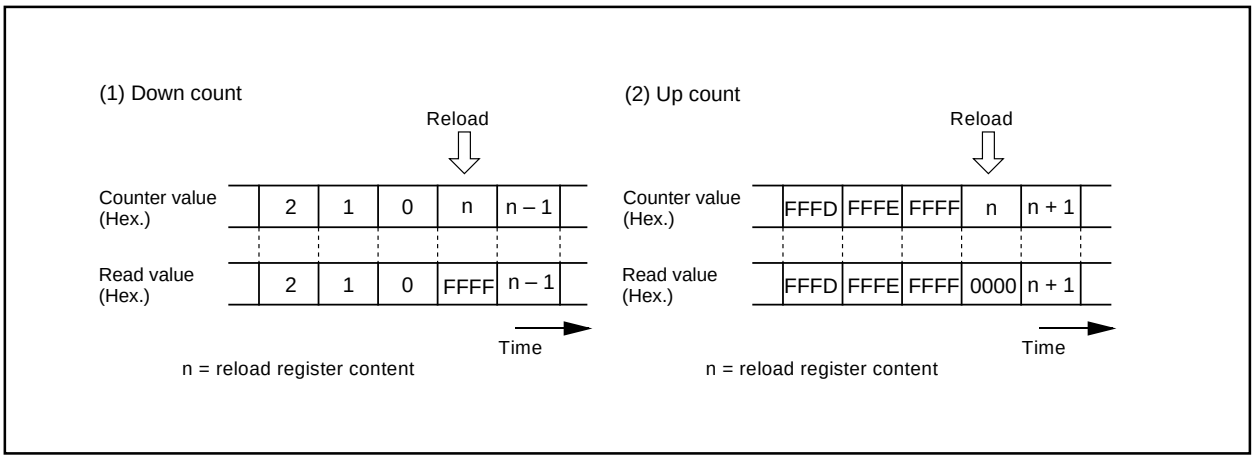


Figure 1.3.2. Reading timer Ai register

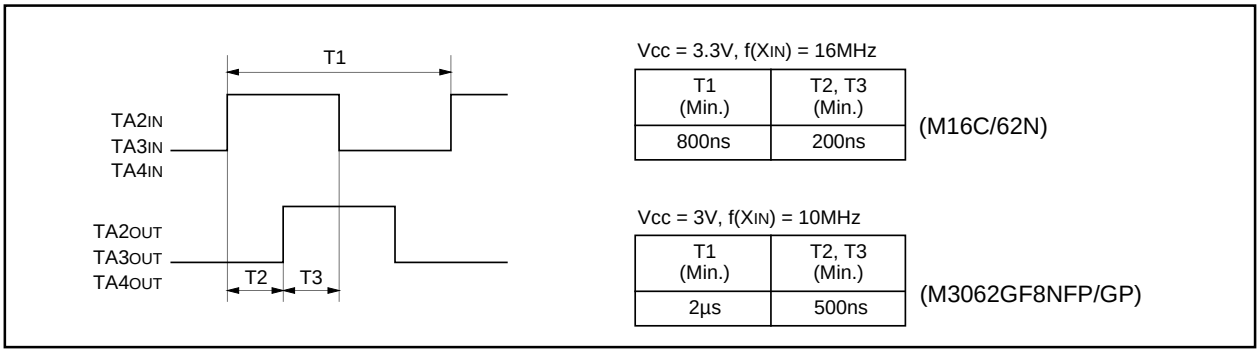


Figure 1.3.3. Standard of 2-phase pulses

1.3.3 Precautions for Timer A (one-shot timer mode)

1. At reset, the count start flag is set to “0”. Set a value in the timer Ai register, then set the flag to “1”.

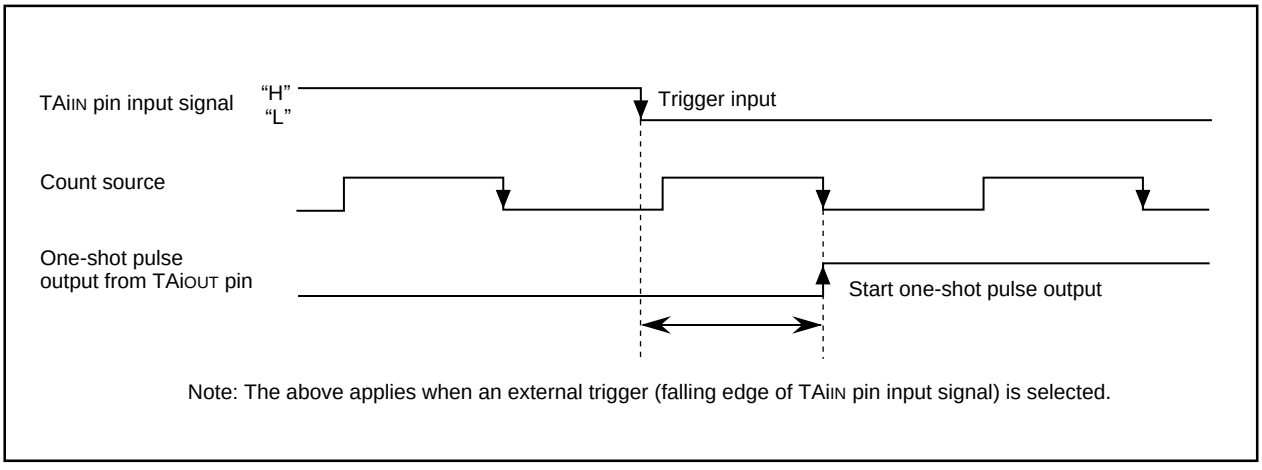
2. Setting the count start flag to “0” while a count is in progress causes as follows:
   • The counter stops counting and a content of reload register is reloaded.
   • The TAiOUT pin outputs “L” level.
   • The timer Ai interrupt request bit becomes “1” after 1 cycle of BCLK.

   Therefore, set the count start flag to “0” first and then execute two NOP instructions. After that set the timer Ai interrupt request bit to “0”.

3. The output from the one-shot timer synchronizes with the count source generated internally. Therefore, when an external trigger has been selected, a delay of one cycle of the maximum count source occurs between the trigger input to the TAiIN pin and the one-shot timer output (Figure 1.3.4).

4. The timer Ai interrupt request bit goes to “1” if the timer's operation mode is set using any of the following procedures:
   • Selecting one-shot timer mode after reset.
   • Changing operation mode from timer mode to one-shot timer mode.
   • Changing operation mode from event counter mode to one-shot timer mode.
   Therefore, to use timer Ai interrupt (interrupt request bit), set timer Ai interrupt request bit to “0” after the above listed changes have been made.

5. If a trigger occurs while a count is in progress, after the counter performs one down count following the reoccurrence of a trigger, the reload register contents are reloaded, and the count continues. To generate a trigger while a count is in progress, generate the second trigger after an elapse longer than one cycle of the timer's count source after the previous trigger occurred.



Note: The above applies when an external trigger (falling edge of TAiIN pin input signal) is selected.

Figure1.3.4. One-shot timer delay

1.3.4 Precautions for Timer A (pulse width modulation mode)

1. To clear reset, the count start flag is set to “0”. Set a value in the timer Ai register, then set the flag to “1”.

2. The timer Ai interrupt request bit becomes “1” if setting operation mode of the timer in compliance with any of the following procedures:
   • Selecting PWM mode after reset.
   • Changing operation mode from timer mode to PWM mode.
   • Changing operation mode from event counter mode to PWM mode.
   Therefore, to use timer Ai interrupt (interrupt request bit), set timer Ai interrupt request bit to “0” after the above listed changes have been made.

3. Setting the count start flag to “0” while PWM pulses are being output causes the counter to stop counting. If the TAiOUT pin is outputting an “H” level in this instance, the output level goes to “L”, and the timer Ai interrupt request bit goes to “1”. If the TAiOUT pin is outputting an “L” level in this instance, the level does not change, and the timer Ai interrupt request bit does not becomes “1”.

1.4 Precautions for Timer B

1.4.1 Precautions for Timer B (timer mode, event counter mode)

1. To clear reset, the count start flag is set to “0”. Set a value in the timer Bi register, then set the flag to “1”.

2. Reading the timer Bi register while a count is in progress allows reading, with arbitrary timing, the value of the counter. Reading the timer Bi register with the reload timing shown in Figure 1.4.1 gets “$FFFF_{16}$”. Reading the timer Bi register after setting a value in the timer Bi register with a count halted but before the counter starts counting gets a proper value.

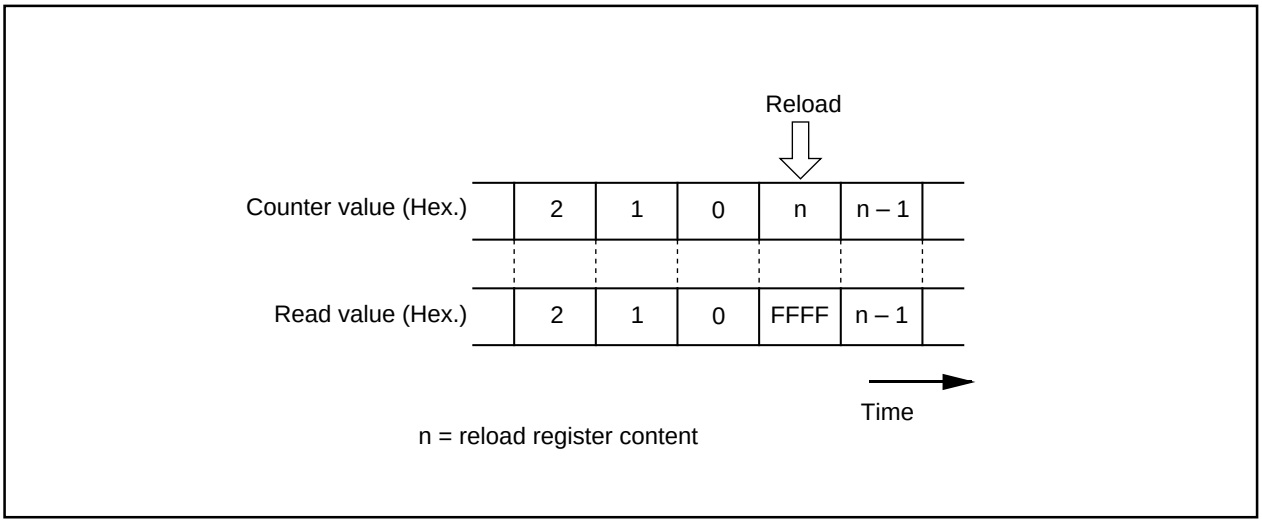


Figure 1.4.1. Reading timer Bi register

1.4.2 Precautions for Timer B (pulse period/pulse width measurement mode)

1. The timer Bi interrupt request bit goes to “1” when an effective edge of a measurement pulse is input or timer Bi is overflowed. The factor of interrupt request can be determined by use of the timer Bi overflow flag within the interrupt routine.

2. If the timer overflow occurs simultaneously with the input of a measurement pulse, and if the interrupt factor cannot be determined from the timer Bi overflow flag, connect the timers and count the number of overflows.

3. After reset, the timer Bi overflow flag is indeterminate. After setting the count start flag to “1”, if a value is written to the timer Bi mode register after the count timing of count source, then the timer Bi overflow flag becomes “0”.

4. Use the timer Bi interrupt request bit to detect only overflows. Use the timer Bi overflow flag only to determine the interrupt factor within the interrupt routine.

5. When the first effective edge is input after a count is started, an indeterminate value is transferred to the reload register. At this time, timer Bi interrupt request is not generated.

6. The value of the counter is indeterminate at the beginning of a count. Therefore, the timer Bi overflow flag may go to “1” and timer Bi interrupt request may be generated during the interval between a count start and an effective edge input.

7. If changing the measurement mode select bits are set after a count is started, the timer Bi interrupt request bit goes to “1”. Note that the timer Bi interrupt request bit does not change if the same value as before is written to the measurement mode select bits.

8. If the input signal to the TBiIN pin is affected by noise, precise measurement may not be performed in some cases. It is recommended to see that measurements fall within a specific range by use of software.

9. For pulse width measurement, pulse widths are successively measured. Use software to check whether the measurement result is an “H” level width or an “L” level width.

1.5 Precautions for Serial I/O (in clock-synchronous serial I/O)

1.5.1 Transmission/reception

With an external clock selected, and choosing the $\overline{RTS}$ function, the output level of the $\overline{RTSi}$ pin goes to "L" when the data-receivable status becomes ready, which informs the transmission side that the reception has become ready. The output level of the $\overline{RTSi}$ pin goes to "H" when reception starts. So if the $\overline{RTSi}$ pin is connected to the $\overline{CTSi}$ pin on the transmission side, the circuit can transmission and reception data with consistent timing. With the internal clock, the $\overline{RTS}$ function has no effect. Figure 1.5.1 shows an example of wiring.

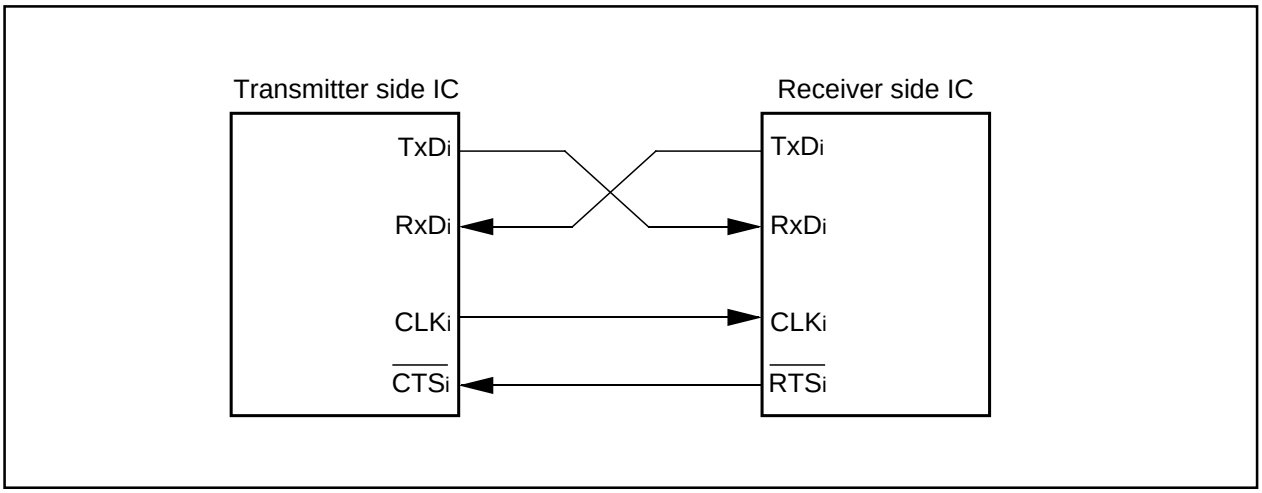


Figure 1.5.1. Example of wiring

1.5.2 Transmission

With an external clock selected, perform the following set-up procedure with the CLKi pin input level = “H” if the CLK polarity select bit = “0” or with the CLKi pin input level = “L” if the CLK polarity select bit = “1”:

(1) Set the transmit enable bit (to “1”)
(2) Write transmission data to the UARTi transmit buffer register
(3) “L” level input to the $\overline{\text{CTSi}}$ pin (when the $\overline{\text{CTS}}$ function is selected)

1.5.3 Reception

1. In operating the clock-synchronous serial I/O, operating a transmitter generates a shift clock. Fix settings for transmission even when using the device only for reception. Dummy data is output to the outside from the TxDi pin (transmission pin) when receiving data.

2. With the internal clock selected, setting the transmit enable bit to “1” (transmission-enabled status) and setting dummy data in the UARTi transmission buffer register generates a shift clock. With the external clock selected, a shift clock is generated when the transmit enable bit is set to “1”, dummy data is set in the UARTi transmit buffer register, and the external clock is input to the CLKi pin.

3. In receiving data in succession, an overrun error occurs when the next reception data is made ready in the UARTi receive register with the receive complete flag set to “1” (before the content of the UARTi receive buffer register is read), and overrun error flag is set to “1”. In this instance, the next data is written to the UARTi receive buffer register, so handle with this problem by writing programs on transmission side and reception side so that the previous data is transmitted again. If an overrun error occurs, the UARTi receive interrupt request bit does not change.

4. To receive data in succession, set dummy data in the lower-order byte of the UARTi transmit buffer register every time reception is made.

5. With an external clock selected, perform the following set-up procedure with the CLKi pin input level = “H” if the CLK polarity select bit = “0” or with the CLKi pin input level = “L” if the CLK polarity select bit = “1”:
   (1) Set receive enable bit (to “1”)
   (2) Set transmit enable bit (to “1”)
   (3) Write dummy data to the UARTi transmit buffer register

6. Output from the $\overline{\text{RTS}}$ pin goes to “L” level as soon as the receive enable bit is set to “1”. This is not related to the content of the transmit buffer empty flag or the content of the transmit enable bit. Output from the $\overline{\text{RTS}}$ pin goes to “H” level when reception starts, and goes to “L” level when reception is completed. This is not related to the content of the transmit buffer empty flag or the content of the receive complete flag.

1.6 Precautions for A-D Converter

1. Write to each bit (except bit 6) of A-D control register 0, to each bit of A-D control register 1, and to bit 0 of A-D control register 2 when A-D conversion is stopped (before a trigger occurs). In particular, when the Vref connection bit is changed from 0 to 1, start A-D conversion after an elapse of 1 µs or longer.

2. To reduce conversion error due to noise, connect a voltage to the AVcc pin and to the Vref pin from an independent source. It is recommended to connect a capacitor between the AVss pin and the AVcc pin, between the AVss pin and the Vref pin, and between the AVss pin and the analog input pin (ANi). Figure 1.6.1 shows the an example of connecting the capacitors to these pins.

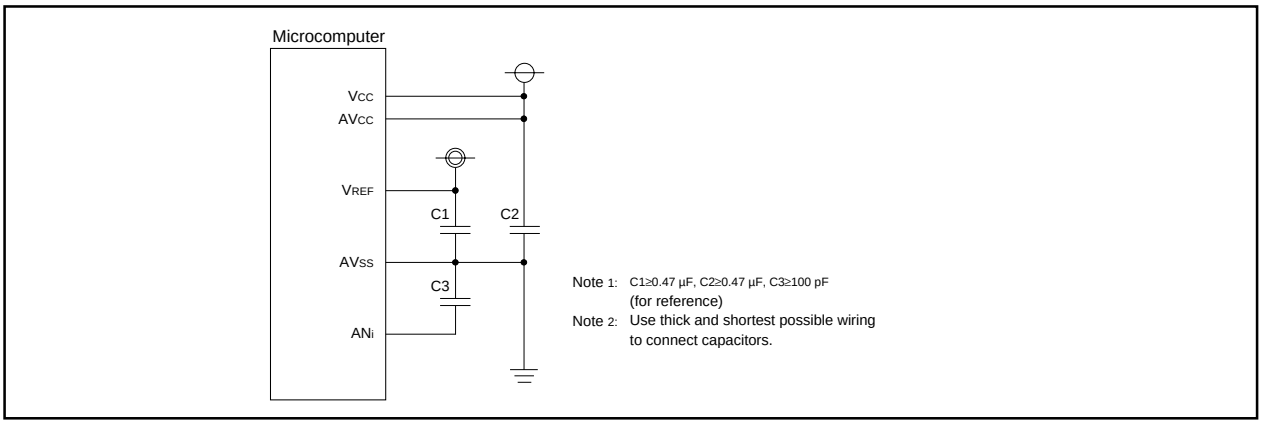


Figure 1.6.1. Use of capacitors to reduce noise

3. Set the direction register of the following ports to input: the port corresponding to a pin to be used as an analog input pin and external trigger input pin (P97).

4. In using a key-input interrupt, none of the 4 pins (AN4 through AN7) can be used as an A-D conversion port (if the A-D input voltage goes to “L” level, a key-input interrupt occurs).

5. Divide the fAD if f(XIN) exceeds 10MHz, and make AD operation clock frequency (ϕAD) equal to or lower than 10MHz. And divide the fAD if VCC is less than 3.0V, and make AD operation clock frequency (ϕAD) equal to or lower than fAD/2. A case without sample & hold function turn AD operation clock frequency (ϕAD) into 250 kHz or more. A case with sample & hold function turn AD operation clock frequency (ϕAD) into 1MHz or more. (M16C/62N)

Divide the fAD and make AD operation clock frequency (ϕAD) equal to or lower than fAD/2. A case without sample & hold function turn AD operation clock frequency (ϕAD) into 250 kHz or more. (M3062GF8NFP/GP)

6. Rewrite to analog input pin select bits after changing A-D operation mode.

7. When using the one-shot or single sweep mode
Confirm that A-D conversion is complete before reading the A-D register. (Note: When A-D conversion interrupt request bit is set, it shows that A-D conversion is completed.)

8. When using the repeat mode or repeat sweep mode 0 or 1
Use the undivided main clock as the internal CPU clock.

1.7 Precautions for Power Control

1. The processor does not switch to stop mode when the $\overline{NMI}$ pin is at "L" level.

2. When returning from stop mode by hardware reset, $\overline{RESET}$ pin must be set to "L" level until main clock oscillation is stabilized.

3. When switching to either wait mode or stop mode, instructions occupying four bytes either from the WAIT instruction or from the instruction that sets the all clock stop control bit to "1" within the instruction queue are prefetched and then the program stops. So put at least four NOPs in succession either to the WAIT instruction or to the instruction that sets the all clock stop control bit to "1".

4. Before the count source for BCLK can be changed from X_{IN} to X_{CIN} or vice versa, the clock to which the count source is going to be switched must be oscillating stably. Allow a wait time in software for the oscillation to stabilize before switching over the clock.

5. Suggestions to reduce power consumption

(a) Ports

The processor retains the state of each programmable I/O port even when it goes to wait mode or to stop mode. A current flows in active I/O ports. A pass current flows in input ports that float. When entering wait mode or stop mode, set non-used ports to input and stabilize the potential.

(b) A-D converter

A current always flows in the V_{REF} pin. When entering wait mode or stop mode, set the Vref connection bit to "0" so that no current flows into the V_{REF} pin.

(c) D-A converter

The processor retains the D-A state even when entering wait mode or stop mode. Disable the output from the D-A converter then work on the programmable I/O ports.

Set D-A register to "00_{16}".

(d) Stopping peripheral functions

In wait mode, stop non-used peripheral functions using the WAIT peripheral function clock stop bit. However, peripheral function clock f_{C32} does not stop so that the peripherals using f_{C32} do not contribute to the power saving. When the MCU running in low-speed or low power dissipation mode, do not enter WAIT mode with this bit set to "1".

(e) Switching the oscillation-driving capacity

Set the driving capacity to "LOW" when oscillation is stable.

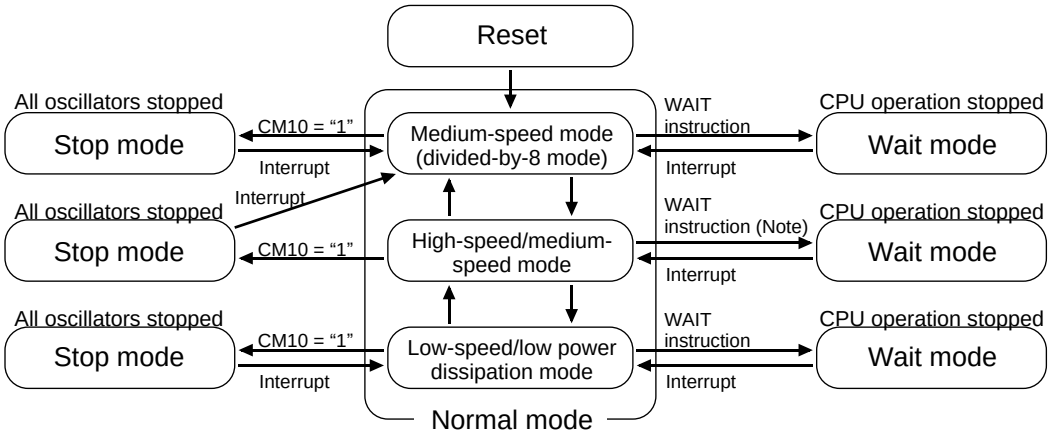
(f) External clock

When using an external clock input for the CPU clock, set the main clock stop bit to "1". Setting the main clock stop bit to "1" causes the X_{OUT} pin not to operate and the power consumption goes down (when using an external clock input, the clock signal is input regardless of the content of the main clock stop bit).

6. Precautions for stop mode and wait mode (M3062GF8NFP/GP)

When shifting to the wait mode in operating with the main clock, set up the bit 6 (CM06) of the system clock control register 0, and the bits 6 and 7 (CM16,CM17) of the system clock control register 1, and execute WAIT instruction after selecting the main clock operation divided by 8 or 16.

Transition of stop mode, wait mode



Note: When shifting to the wait mode in operating with the main clock, set up the bit 6 (CM06) of the system clock control register 0, and the bits 6 and 7 (CM16,CM17) of the system clock control register 1, and execute WAIT instruction after selecting the main clock operation divided by 8 or 16.

Figure 1.7.1. Transition of stop mode, wait mode

1.8 Precautions for External Bus

1. The external ROM version can operate only in the microprocessor mode, so be sure to perform the following:
   • Connect the CNVSS pin to VCC.

2. When the CNVSS pin is reset at “H” level, the contents of internal ROM cannot be read out.

1.9 Electric characteristic differences between mask ROM and flash memory version MCUs

There are differences in electric characteristics, operation margin, noise immunity, and noise radiation between mask ROM and flash memory version MCUs due to the difference in the manufacturing processes.

When manufacturing an application system with the flash memory version and then switching to use of the mask ROM version, please perform sufficient evaluations for the commercial samples of the mask ROM version.

1.10 Precautions for Flash Memory Version

1.10.1 Precautions for Read Array Command (FF16)

The read array mode is entered by writing the command code "FF16" in the first bus cycle. When an even address to be read is input in one of the bus cycles that follow, the content of the specified address is read out at the data bus (D0–D15), 16 bits at a time.

The read array mode is retained intact until another command is written.

However, please begin to read data in the following procedures when a user uses read array command after program command.

(1) Set FF16, FF16, FF16, FF16 to arbitrary continuing four address beforehand
(2) Input the top address which FF16 was set at (in read array mode)
(3) Input the top address till FFFF16 agrees with the value that begins to have been read
(4) Input top address +2
(5) Input top address +2 till FFFF16 agrees with the value that begins to have been read
(6) Input an arbitrary address

1.10.2 Precautions for Program Command (4016)

Program operation starts when the command code "4016" is written in the first bus cycle. Then, if the address and data to program are written in the 2nd bus cycle, program operation (data programming and verification) will start. Make an address in the first bus cycle same as an address to program by the second bus cycle.

Make an address in the first bus cycle same as an address to program by the second bus cycle of program command.

1.10.3 Precautions for Lock Bit Program Command (7716/D016)

By writing the command code "7716" in the first bus cycle and the confirmation command code "D016" in the second bus cycle that follows to the block address of a flash memory block, the system sets the lock bit for the specified block to 0 (locked). Make an address in the first bus cycle same as an address to block by the second bus cycle.

1.10.4 Precautions for stop mode

After returning from stop mode, an unexpected operation may occur (for example, undefined instruction interrupt, BRK instruction interrupt, etc.).

Execute a JMP.B instruction after an instruction to write data to the all clock stop control bit. Disable DMA transfer. A program example is described as follows:

```
        MOV.B:S   #21H, CM1  ; writing to the all clock stop control bit to "1"(stop mode)
        JMP.B     L1
L1 :
        NOP
        NOP
        NOP
        NOP
```

1.10.5 Precautions on CPU Rewrite Mode

Described below are the precautions to be observed when rewriting the flash memory in CPU rewrite mode.

1.10.5.1 Operation speed

During CPU rewrite mode, set the BCLK as shown below using the main clock divide ratio select bit (bit 6 at address 0006_{16} and bits 6 and 7 at address 0007_{16}):

6.25 MHz or less when wait bit (bit 7 at address 0005_{16}) = 0 (without internal access wait state)

10.0 MHz or less when wait bit (bit 7 at address 0005_{16}) = 1 (with internal access wait state)

1.10.5.2 Instructions inhibited against use

The instructions listed below cannot be used during CPU rewrite mode because they refer to the internal data of the flash memory:

UND instruction, INTO instruction, JMPS instruction, JSRS instruction, and BRK instruction

1.10.5.3 Interrupts inhibited against use

The address match interrupt cannot be used during CPU rewrite mode because they refer to the internal data of the flash memory. If interrupts have their vector in the variable vector table, they can be used by transferring the vector into the RAM area. The $\overline{\text{NMI}}$ and watchdog timer interrupts can be used to automatically initialize the flash identification register and flash memory control register 0 to “0”, then return to normal operation. However, these two interrupts' jump addresses are located in the fixed vector table and there must exsist a routine to be executed. Since the rewrite operation is halted when an $\overline{\text{NMI}}$ or watchdog timer interrupts occurs, you must reset the CPU rewrite mode select bit to “1” and the perform the erase/program operation again.

1.10.5.4 Access disable

Write to CPU rewrite mode select bit and user ROM area select bit only when executing out of an area other than the internal flash memory.

1.10.5.5 How to access

For CPU rewrite mode select bit and lock bit disable select bit to be set to “1”, the user needs to write a “0” and then a “1” to it in succession. When it is not this procedure, it is not enacted in “1”. This is necessary to ensure that no interrupt or DMA transfer will be executed during the interval. Write to CPU rewrite mode select bit and user ROM area select bit only when executing out of an area other than the internal flash memory.Also only when $\overline{\text{NMI}}$ pin is “H” level.

1.10.5.6 Writing in the user ROM area

If power is lost while rewriting blocks that contain the flash rewrite program with the CPU rewrite mode, those blocks may not be correctly rewritten and it is possible that the flash memory can no longer be rewritten after that. Therefore, it is recommended to use the standard serial I/O mode or parallel I/O mode to rewrite these blocks.

1.10.5.7 Using the lock bit

To use the CPU rewrite mode, use a boot program that can set and cancel the lock command.

1.10.5.8 Internal reserved area expansion bit (Bit 3 at address 0005_{16})

To use the products which RAM size is over 15 Kbytes or flash memory size is over 192 Kbytes, change into the CPU rewrite mode after setting the internal reserved area expansion bit (bit 3 at address 0005_{16}) to “1”. Even if the CPU rewrite mode select bit (bit 1 at address $03B7_{16}$) is set to “1”, the internal reserved area expansion bit (bit 3 at address 0005_{16}) is not set to “1” automatically.

GRADE A

MSC TECHNICAL NEWS

No.M16C-09-9705

Note on using the A-D converter of the M16C/60 series MCU

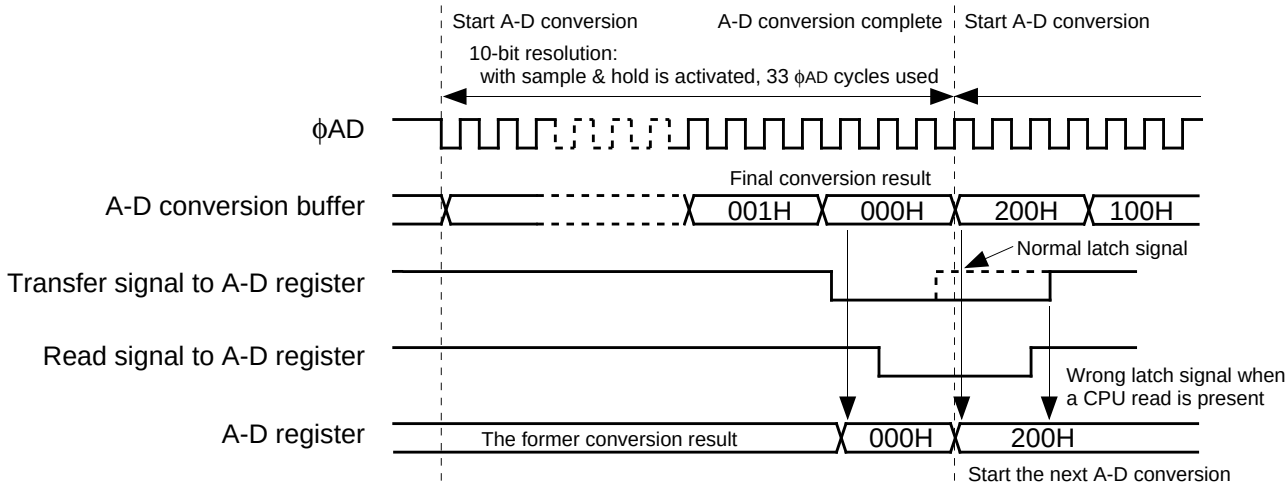
1. Related devices

M16C/60 series

2. Symptoms

After A-D conversion is complete, if the CPU reads the A-D register at the same time as the A-D conversion result is being saved to A-D register, wrong A-D conversion value is saved into the A-D register. This happens when the internal CPU clock is selected from divided main clock or sub-clock.

(When connected an A-D input port and GND)



Normally, A-D conversion value is saved at the rising edge (dashed rising edge) of the latch signal. However, when the CPU is doing a read to A-D register at this time, the A-D register latch signal is delayed, and wrong value is stored at A-D conversion register.

3. Precaution

(1) When using the one-shot or single sweep mode

Confirm that A-D conversion is complete before reading the A-D register.
(Note: When A-D conversion interrupt request bit is set, it shows that A-D conversion is completed.)

(2) When using the repeat mode or repeat sweep mode 0 or 1

Use the undivided main clock as the internal CPU clock.

GRADE A

MESC TECHNICAL NEWS

No.M16C-11-9710

Note on dedicated input pin of the M16C/60 series MCU

1. Related devices

M16C/60 series

2. Note on dedicated input pin

When different power supplied to the system as shown in figure 1, and input voltage of unused dedicated input pin is larger than voltage of VCC pin, do not connect dedicated input pin and power supply directly. Connect to VCC via resistor (approximately 1kohm) as shown in figure 2. This note is also applicable when VINPUT exceeds VCC during power-up.

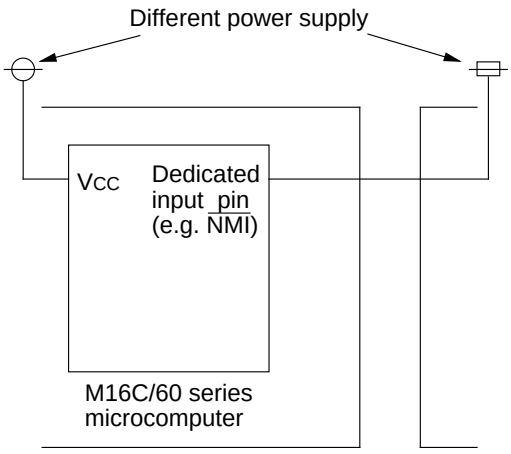


Figure 1. Circuit diagram

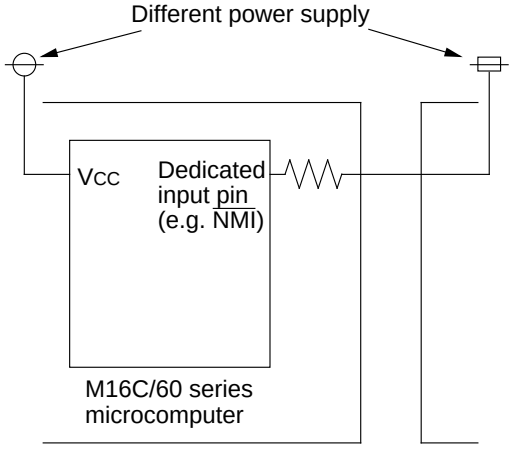


Figure 2. Improved circuit diagram

* The resistor is not necessary when VCC pin voltage is same or larger than dedicated input pin voltage.

3. Cause

When dedicated input pin voltage is larger than VCC pin voltage, latch up occurs.

GRADE A

MESC TECHNICAL NEWS No.M16C-13-9802

Supplemental Description of DMAC for the M16C/60, M16C/61 and M16C/62 Group MCUs

1. Related devices

M16C/60, M16C/61 and M16C/62 groups

2. DMA enable bit

The DMA enable bit is bit 3 of both DMA0 and DMA1 control registers.
When the DMA enable bit is set to "1" the DMAC is in an active state and the following occurs:

a. The value of whichever of the source or destination pointer that is set up as the forward pointer is reloaded into the forward address pointer.
b. The value in the transfer counter reload register is reloaded into the transfer counter.

Therefore, the DMAC will start from the initial conditions once again if the DMA enable bit is set to "1" while in the active state.

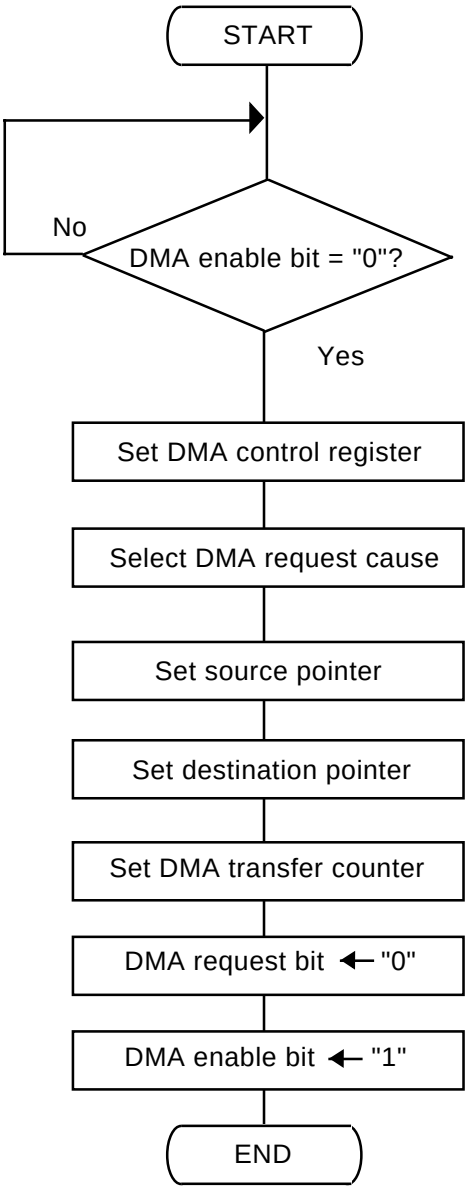
3. DMA request bit

The DMA request bit is bit 2 of both DMA0 and DMA1 control registers.
Regardless of the DMAC status (enable bit set or clear), the request bit is set to "1" when a request signal for a DMA transfer occurs, based on the DMA request factor. The bit is cleared to "0" when data transfer begins. Further, the user can clear ("0") the DMA request bit but not set it.

It is possible that the DMA request bit may become "1" due to the DMA request cause select bits being changed. Therefore the DMA request bit should be cleared ("0") after changing the DMA request cause select bits.

If DMAC is in the active state (enable bit set) when the request bit becomes "1", the data transfer begins immediately. That in turn immediately causes the DMA request bit to be cleared ("0"). Therefore, to best judge the state of the DMAC, the DMA enable bit should be read instead of the request bit.

4. Initialization of DMA-related registers



GRADE A

MESC TECHNICAL NEWS No.M16C-14-9805

Precautions Regarding Writing to M16C/60, M16C/61, M16C/62 and M16C/63 Group MCUs Interrupt Control Registers

1. Related devices

M16C/60, M16C/61, M16C/62 and M16C/63 groups

With the M16C/60 series MCU, setting the interrupt priority level and clearing the interrupt request bit in the interrupt control registers should be done with interrupt disabled.
Executing these operations while interrupt is enabled may result in unintended CPU operations.

2. Symptom

Changing the Interrupt priority LeVeL select bit (ILVL) and clearing the Interrupt Request bit (IR) in the Interrupt Control Registers (ICRs) while the Interrupt enable FLAG (I-FLAG) is "1" may result in unintended operations, such as BRK and other interrupts being generated.

3. Considerations for writing new program

It is recommended that the interrupts must be disabled by clearing the I-FLAG, before setting ILVL or clearing the IR bit in the ICRs.
In order to avoid the influence of the CPU pipeline, a certain number of instructions (eg. NOP) should be inserted between writing to the ICRs and setting the I-FLAG.

The number of instructions (NOPs) required is shown in TABLE.

4. Conditions to be checked for program already written

Please confirm that at least one condition is met for both actions listed below. If any one of the conditions is met, the symptom will not occur.

(1) When changing ILVL

- I-FLAG is "0". (Interrupt disabled) (*Note)
- The processor interrupt priority level (IPL) in the flag register is "7".
- The ILVL changes from a lower level than IPL to a higher level.
- The ILVL before and after the change is lower than IPL.
- The ILVL before and after the change is higher than IPL.
- It is obvious that the corresponding interrupt will not occur while changing the ILVL.

(2) When clearing the IR

- I-FLAG is "0". (Interrupt disabled) (*Note)
- The IPL in the flag register is "7".
- The ILVL during the operation is "0".
- The ILVL is lower than IPL.
- It is obvious that the corresponding interrupt will not occur while clearing the IR.

Note: In order to avoid the influence of the CPU pipeline, a certain number of instructions (eg. NOP) should be inserted between writing to ICRs and setting the I-FLAG.

The number of instructions required is showed in the TABLE.

	When not using HOLD function	When using HOLD function
Example 1	Two NOP instructions required	Four NOP instructions required
Example 2	No NOP instruction required (because there is dummy read)	
Example 3	No NOP instruction required	

5. Program examples

The program examples are described as follow:

(1) For assembler

Example 1:

```
INT_SWITCH1:
  FCLR   I           ; Disable interrupts.
  AND.B  #00h, 0055h ; Clear TA0IC int. priority level and int. request bit.
  NOP                ; Four NOP instructions are required when using HOLD function.
  NOP
  FSET   I           ; Enable interrupts.
```

Example 2:

```
INT_SWITCH2:
  FCLR   I           ; Disable interrupts.
  AND.B  #00h, 0055h ; Clear TA0IC int. priority level and int. request bit.
  MOV.W  MEM, R0     ; Dummy read.
  FSET   I           ; Enable interrupts.
```

Example 3:

```
INT_SWITCH3:
  PUSHC  FLG         ; Push Flag register onto stack
  FCLR   I           ; Disable interrupts.
  AND.B  #00h, 0055h ; Clear TA0IC int. priority level and int. request bit.
  POPC   FLG         ; Enable interrupts.
```

The reason why two NOP instructions (four when using the HOLD function) or dummy read are inserted before FSET I in Examples 1 and 2 is to prevent the interrupt enable flag I from being set before the interrupt control register is rewritten due to effects of the instruction queue.

(2) For C language

```
#pragma   ASM
 INT_SWITCH:
  FCLR   I
#pragma   ENDASM
  TA0IC  & =00 ;     /* Clear TA0IC int. priority level and int. request bit. */
#pragma   ASM
  NOP                ; Four NOP instructions are required when using HOLD function.
  NOP
  FSET   I
#pragma   ENDASM
```

GRADE A

MESC TECHNICAL NEWS No.M16C-19-9903

MESC TECHNICAL NEWS
'No.M16C-16-9902' replace

MESC TECHNICAL NEWS 'No.M16C-16-9902' has an error, so we will correct.
Please replace old Technical News 'No.M16C-16-9902' to corrected Technical News 'M16C/60, M16C/61, M16C/62 Group Precautions for Setting Pull-up Resistors'.

[Attached]

Corrected Technical News 'No.M16C-19-9903'
'M16C/60, M16C/61, M16C/62 Group Precautions for Setting Pull-up Resistors' ..... 1 page

GRADE A

MESC TECHNICAL NEWS No.M16C-19-9903

M16C/60, M16C/61, M16C/62 Group
Precautions for Setting Pull-up Resistors

1. Related devices

M16C/60 group, M16C/61 group, M16C/62 group

2. Precautions

Ports P0 to P10 can be set to apply a pull-up resistor by using the pull-up control registers.

(1) M16C/60 Group, M16C/61 Group

In Memory expansion mode or Microprocessor mode, the settings of pull-up control registers for ports P0 to P5 are invalid.
In Memory expansion mode or Microprocessor mode, ports P0, P1, $P3_1$ to $P3_7$ and P4 can be used as input ports, but internal pull-up resistors can not be connected.

(2) M16C/62 Group

In Memory expansion mode or Microprocessor mode, the settings of pull-up control registers for P0 to P3, $P4_0$ to $P4_3$ and P5 are invalid.
In Memory expansion mode or Microprocessor mode, ports P0, P1, $P3_1$ to $P3_7$ and $P4_0$ to $P4_3$ can be used as input ports, but internal pull-up resistors can not be connected.
(In Memory expansion mode or Microprocessor mode, $P4_4$ to $P4_7$ can be used as general input ports, and pull-up control register can be used to connect the internal pull-up resistors.)

GRADE A

MESC TECHNICAL NEWS No.M16C-25-9905

M16C/60 , M16C/20 Series

Precautions for Wait and Stop modes

1. Related devices

M16C/60 Series , M16C/20 Series

2. Precautions

The M16C has both WAIT and STOP modes. These modes can be used to reduce power consumption when the CPU is not required to perform any work. To return to normal operating mode after issuing a WAIT instruction or setting the all clock stop control bit, perform a hardware reset or use an interrupt. The interrupts for canceling the WAIT and STOP modes must be enabled before entering either mode. The priority level of the interrupts not used for these modes should be set to 0 before switching into the WAIT or STOP modes. Also, if only hardware reset or $\overline{\text{NMI}}$ interrupts are used for canceling the WAIT or STOP modes, all interrupt priority level should be set to 0 before switching into the WAIT or STOP mode.

3. Examples

3.1 Use the following algorithm to enter the WAIT or STOP modes when an interrupt is used to cancel either mode.

• Hardware reset, $\overline{\text{NMI}}$ interrupt, and $\overline{\text{INT0}}$ interrupt is used to cancel either mode

Set the interrupt enable flag (I flag) to “0”	; Disable interrupt
Change the interrupt priority level to 1 or higher	; Enable $\overline{\text{INT0}}$ interrupt (In case of processor interrupt priority level=0)
Change all other interrupt priority levels to 0	; Disable all other interrupts
Insert 4 NOP instructions	; Prevention of irregular interrupts issue. See TECHNICAL NEWS No.M16C-14-9805
Set the interrupt enable flag (I flag) to “1”	; Enable interrupt
WAIT instruction or all clock stop control bit set	; Set the bit 0 of protect register to “1” before set the all clock stop control bit to “1” .
Insert 4 NOP instructions	; NOP instructions are required because M16C instruction queue reads forward 4 bytes from wait or stop instruction when program is stopped.

3.2 When using only hardware reset or $\overline{\text{NMI}}$ interrupt to cancel the STOP or WAIT modes, use the following algorithm to enter the STOP or WAIT modes.

Set the interrupt enable flag (I flag) to “0”	; Disable interrupt
Change all interrupt priority levels to 0	; Disable maskable interrupt
WAIT instruction or all clock stop control bit set	; Set the bit 0 of protect register to “1” before set the all clock stop control bit to “1” .
Insert 4 NOP instructions	; NOP instructions are required because M16C instruction queue reads forward 4 bytes from wait or stop instruction when program is stopped.

GRADE A

MESC TECHNICAL NEWS No.M16C-26-9905

M16C/61 , M16C/62 Group

Precautions for UART2

1. Related devices

M16C/61 Group, M16C/62 Group

2. Precautions

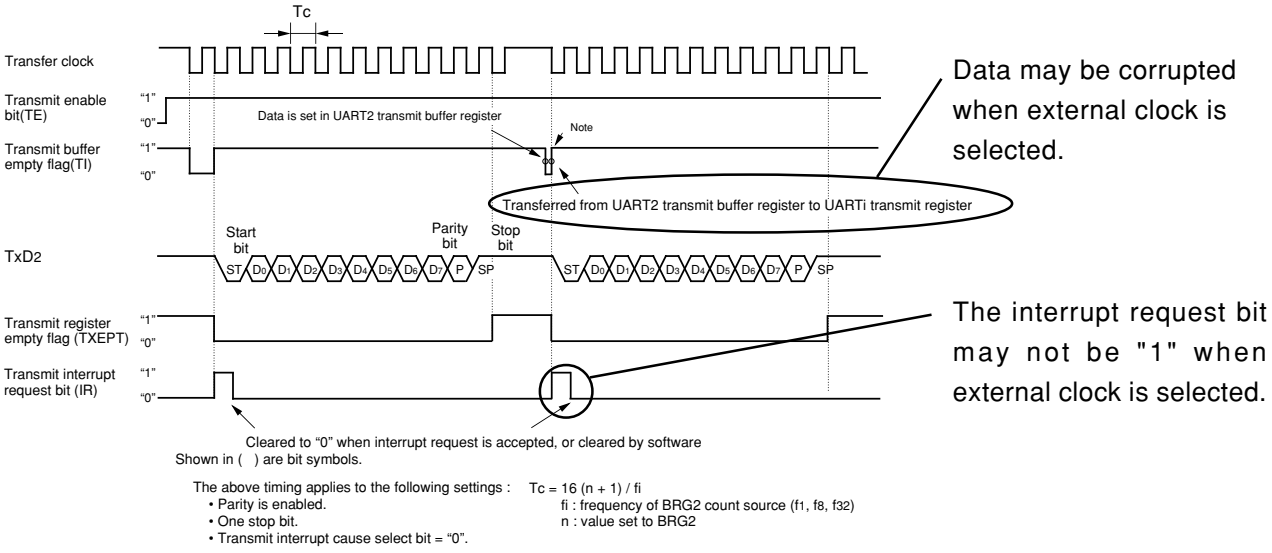
When using UART2 in clock asynchronous serial I/O (UART) mode choose internal clock. If UART2 in clock asynchronous mode is used with external clock, then one of the following may occur;

2.1 The interrupt may not be issued at the end of data transmission when the hardware transfers the data from the transmit buffer register to the transmit register.

2.2 Data may be corrupted when the hardware transfers the data from the transmit buffer register to the transmit register.

This precaution only applies to the UART2 asynchronous serial I/O mode and does not apply to UART0 or UART1. It does not apply to any UART when used in the synchronous clocked serial I/O mode.

Example of transmit with UART mode, Transfer data 8 bits long.



Note: The transmit is started with overflow timing of BRG after having written in a value at the transmit buffer in the above timing.

GRADE A

MESC TECHNICAL NEWS No.M16C-32-9908

M16C/60 Series

Precautions for Address Match Interrupt

1. Related devices

M16C/60 Series

2. Precautions

When using the address match interrupt, please observe the following usage conditions.

(1) Address match interrupt for internal address.
(2) Address match interrupt for external address and 16-bit bus.

When external address and 8-bit bus, you can not use the address match interrupt for external address.

GRADE A

MESC TECHNICAL NEWS No.M16C-49-0004

M16C/80 Series, M16C/60 Series

Cautions for Using Memory Expansion Mode or Microprocessor Mode

1. Affected devices

- M16C/80 Series
- M16C/60 Series

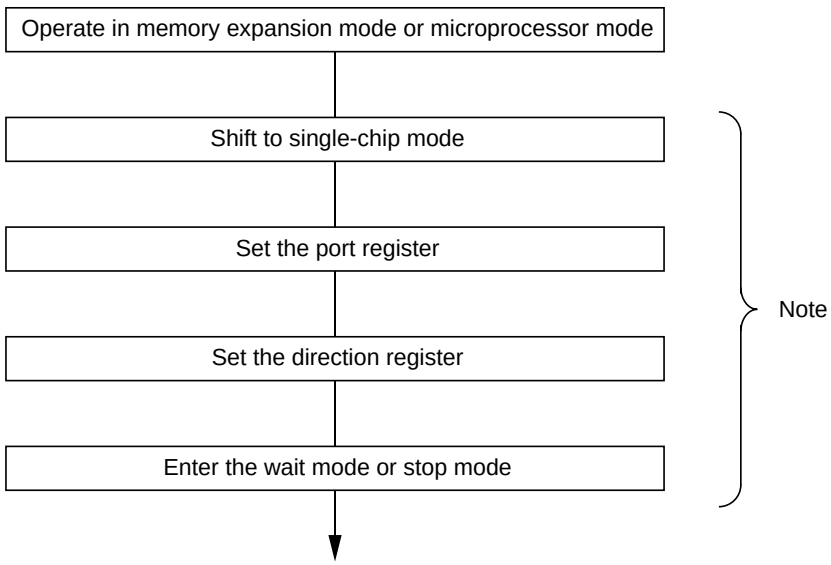
2. Cautions

When the MCU enters wait mode while operating in memory expansion mode or microprocessor mode, a pin functioning as part of the address or data bus retains it's state on the bus before wait mode is entered. Shift to single-chip mode and output an arbitrary value in order to reduce current consumption. By shifting to single-chip mode, a pin which was functioning as part of the bus becomes a general-purpose port and can output an arbitrary value. Set the port registers and direction registers after shifting to single-chip mode (this implies that any control pins ($\overline{CS}$,$\overline{WR}$,$\overline{RD}$,etc..) being used for access of an external device be changed as well).

If the port registers and direction registers are set while in memory expansion mode or microprocessor mode, the operation will be ignored.

This is similar when entering stop mode.

Setting procedure is following.



Note: This program does not work in external area. Transfer a program to internal RAM and work on internal RAM.

GRADE **A**

MESC TECHNICAL NEWS No.M16C-55-0006

M16C Family

Cautions Using Data Registers that Include Write Only Bits

1. Affected devices

• M16C Family

2. Cautions

The registers shown in the table on the following page contain bits that will result in unknown data when read.
If performing a read-modify-write sequence of instructions to a register with write only bits, please reset the write only bits to their previous values before writing back to the register.
If your software accesses these registers frequently, please use a temporary RAM area to change the value, and then transfer it to the register.

Figure 1 shows an example of a register structure. If you execute a 'Read Modify Write' instruction like BSET, BCLR, AND or OR, the values of bits 5-7 may change. (Please see Figure 2)

'Table 10' show instruction table for Read Modify Write.

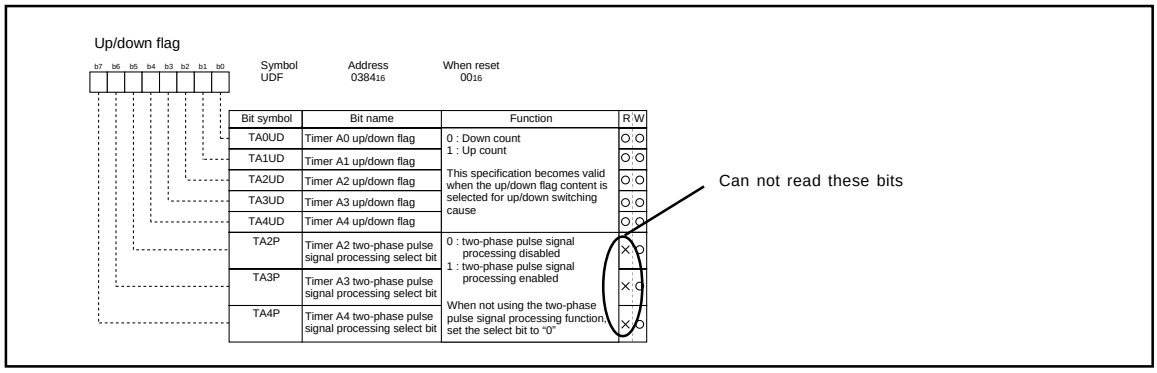


Figure 1. Example of a register structure

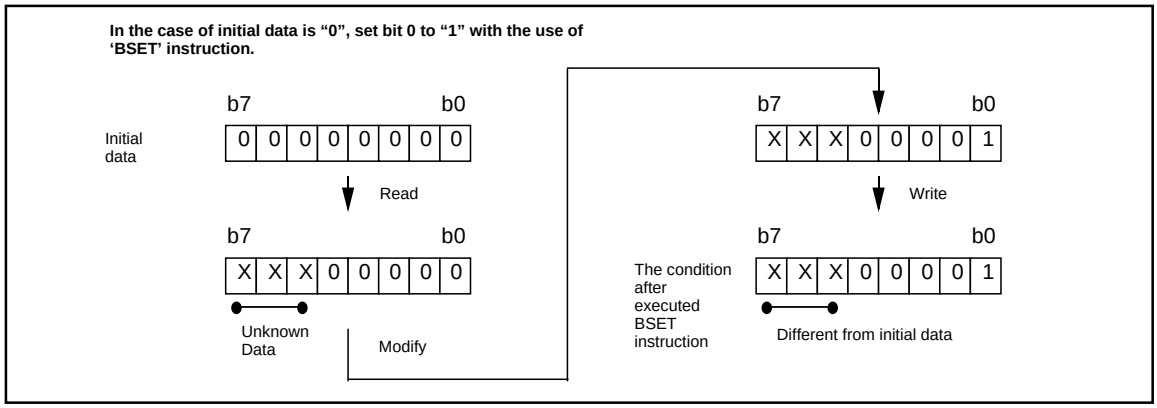


Figure 2. Example of a register change

Table 1. Affected register (M16C/80 group)

Register name	Symbol	Address
UART4 bit rate generator	U4BRG	$02F9_{16}$
UART4 transmit buffer register	U4TB	$02FB_{16}$, $02FA_{16}$
Dead time timer	DTT	$030C_{16}$
Timer B2 interrupt occurrences frequency set counter	ICTB2	$030D_{16}$
UART3 bit rate generator	U3BRG	0329_{16}
UART3 transmit buffer register	U3TB	$032B_{16}$, $032A_{16}$
UART2 bit rate generator	U2BRG	0339_{16}
UART2 transmit buffer register	U2TB	$033B_{16}$, $033A_{16}$
Up/down flag	UDF	0344_{16}
Timer A0 register (Note)	TA0	0347_{16} , 0346_{16}
Timer A1 register (Note)	TA1	0349_{16} , 0348_{16}
Timer A2 register (Note)	TA2	$034B_{16}$, $034A_{16}$
Timer A3 register (Note)	TA3	$034D_{16}$, $034C_{16}$
Timer A4 register (Note)	TA4	$034F_{16}$, $034E_{16}$
UART0 bit rate generator	U0BRG	0361_{16}
UART0 transmit buffer register	U0TB	0363_{16} , 0362_{16}
UART1 bit rate generator	U1BRG	0369_{16}
UART1 transmit buffer register	U1TB	$036B_{16}$, $036A_{16}$

Note: It is affected only in one-shot timer mode and pulse width modulation mode.

Table 2. Affected register (M16C/61 group)

Register name	Symbol	Address
UART2 bit rate generator	U2BRG	0379_{16}
UART2 transmit buffer register	U2TB	$037B_{16}$, $037A_{16}$
Up/down flag	UDF	0384_{16}
Timer A0 register (Note)	TA0	0387_{16} , 0386_{16}
Timer A1 register (Note)	TA1	0389_{16} , 0388_{16}
Timer A2 register (Note)	TA2	$038B_{16}$, $038A_{16}$
Timer A3 register (Note)	TA3	$038D_{16}$, $038C_{16}$
Timer A4 register (Note)	TA4	$038F_{16}$, $038E_{16}$
UART0 bit rate generator	U0BRG	$03A1_{16}$
UART0 transmit buffer register	U0TB	$03A3_{16}$, $03A2_{16}$
UART1 bit rate generator	U1BRG	$03A9_{16}$
UART1 transmit buffer register	U1TB	$03AB_{16}$, $03AA_{16}$

Note: It is affected only in one-shot timer mode and pulse width modulation mode.

Table 3. Affected register (M16C/62 group)

Register name	Symbol	Address
Dead time timer	DTT	$034C_{16}$
Timer B2 interrupt occurrences frequency set counter	ICTB2	$034D_{16}$
SI/O3 bit rate generator	S3BRG	0363_{16}
SI/O4 bit rate generator	S4BRG	0367_{16}
UART2 bit rate generator	U2BRG	0379_{16}
UART2 transmit buffer register	U2TB	$037B_{16}$, $037A_{16}$
Up/down flag	UDF	0384_{16}
Timer A0 register (Note)	TA0	0387_{16} , 0386_{16}
Timer A1 register (Note)	TA1	0389_{16} , 0388_{16}
Timer A2 register (Note)	TA2	$038B_{16}$, $038A_{16}$
Timer A3 register (Note)	TA3	$038D_{16}$, $038C_{16}$
Timer A4 register (Note)	TA4	$038F_{16}$, $038E_{16}$
UART0 bit rate generator	U0BRG	$03A1_{16}$
UART0 transmit buffer register	U0TB	$03A3_{16}$, $03A2_{16}$
UART1 bit rate generator	U1BRG	$03A9_{16}$
UART1 transmit buffer register	U1TB	$03AB_{16}$, $03AA_{16}$

Note: It is affected only in one-shot timer mode and pulse width modulation mode.

Table 4. Affected register (M16C/6H group)

Register name	Symbol	Address
Display RAM address control register	CA	0203_{16} , 0202_{16}
Font RAM address control register	FA	0207_{16} , 0206_{16}
SYRAM address control register	YA	$020B_{16}$, $020A_{16}$
Slice RAM address control register	SA	$020F_{16}$, $020E_{16}$
VBIRAM address control register	EA	0213_{16} , 0212_{16}
Address control register for expansion register	DA	0217_{16} , 0216_{16}
SI/O3 bit rate generator	S3BRG	0363_{16}
SI/O4 bit rate generator	S4BRG	0367_{16}
UART2 bit rate generator	U2BRG	0379_{16}
UART2 transmit buffer register	U2TB	$037B_{16}$, $037A_{16}$
Up/down flag	UDF	0384_{16}
Timer A0 register (Note)	TA0	0387_{16} , 0386_{16}
Timer A1 register (Note)	TA1	0389_{16} , 0388_{16}
Timer A2 register (Note)	TA2	$038B_{16}$, $038A_{16}$
Timer A3 register (Note)	TA3	$038D_{16}$, $038C_{16}$
Timer A4 register (Note)	TA4	$038F_{16}$, $038E_{16}$
UART0 bit rate generator	U0BRG	$03A1_{16}$
UART0 transmit buffer register	U0TB	$03A3_{16}$, $03A2_{16}$
UART1 bit rate generator	U1BRG	$03A9_{16}$
UART1 transmit buffer register	U1TB	$03AB_{16}$, $03AA_{16}$

Note: It is affected only in one-shot timer mode and pulse width modulation mode.

Table 5. Affected register (M16C/6K group)

Register name	Symbol	Address
SI/O3 bit rate generator	S3BRG	0363_{16}
SI/O4 bit rate generator	S4BRG	0367_{16}
UART2 bit rate generator	U2BRG	0379_{16}
UART2 transmit buffer register	U2TB	$037B_{16}$, $037A_{16}$
Up/down flag	UDF	0384_{16}
Timer A0 register (Note)	TA0	0387_{16} , 0386_{16}
Timer A1 register (Note)	TA1	0389_{16} , 0388_{16}
Timer A2 register (Note)	TA2	$038B_{16}$, $038A_{16}$
Timer A3 register (Note)	TA3	$038D_{16}$, $038C_{16}$
Timer A4 register (Note)	TA4	$038F_{16}$, $038E_{16}$
UART0 bit rate generator	U0BRG	$03A1_{16}$
UART0 transmit buffer register	U0TB	$03A3_{16}$, $03A2_{16}$
UART1 bit rate generator	U1BRG	$03A9_{16}$
UART1 transmit buffer register	U1TB	$03AB_{16}$, $03AA_{16}$
Comparator control register	CMPCON	$03DE_{16}$

Note: It is affected only in one-shot timer mode and pulse width modulation mode.

Table 6. Affected register (M16C/6N group)

Register name	Symbol	Address
Dead time timer	DTT	$01CC_{16}$
Timer B2 interrupt occurrences frequency set counter	ICTB2	$01CD_{16}$
SI/O3 bit rate generator	S3BRG	$01E3_{16}$
UART2 bit rate generator	U2BRG	$01F9_{16}$
UART2 transmit buffer register	U2TB	$01FB_{16}$, $01FA_{16}$
Up/down flag	UDF	0384_{16}
Timer A0 register (Note)	TA0	0387_{16} , 0386_{16}
Timer A1 register (Note)	TA1	0389_{16} , 0388_{16}
Timer A2 register (Note)	TA2	$038B_{16}$, $038A_{16}$
Timer A3 register (Note)	TA3	$038D_{16}$, $038C_{16}$
Timer A4 register (Note)	TA4	$038F_{16}$, $038E_{16}$
UART0 bit rate generator	U0BRG	$03A1_{16}$
UART0 transmit buffer register	U0TB	$03A3_{16}$, $03A2_{16}$
UART1 bit rate generator	U1BRG	$03A9_{16}$
UART1 transmit buffer register	U1TB	$03AB_{16}$, $03AA_{16}$

Note: It is affected only in one-shot timer mode and pulse width modulation mode.

Table 7. Affected register (M16C/6V group)

Register name	Symbol	Address
Processor mode register 1	PM1	0005_{16}
OSD reserved register 3	OR3	$027B_{16}$
UART2 bit rate generator	U2BRG	0379_{16}
UART2 transmit buffer register	U2TB	$037B_{16}$, $037A_{16}$
Up/down flag	UDF	0384_{16}
Timer A0 register (Note 1)	TA0	0387_{16} , 0386_{16}
Timer A1 register (Note 1)	TA1	0389_{16} , 0388_{16}
Timer A2 register (Note 2)	TA2	$038B_{16}$, $038A_{16}$
Timer A3 register (Note 2)	TA3	$038D_{16}$, $038C_{16}$
Timer A4 register (Note 1)	TA4	$038F_{16}$, $038E_{16}$
UART0 bit rate generator	U0BRG	$03A1_{16}$
UART0 transmit buffer register	U0TB	$03A3_{16}$, $03A2_{16}$
Port P6 register	P6	$03EC_{16}$
Port P8 register	P8	$03F0_{16}$
Port P9 register	P9	$03F1_{16}$

Note 1: It is affected only in one-shot timer mode.
Note 2: It is affected only in one-shot timer mode and pulse width modulation mode.

Table 8. Affected register (M30201 group)

Register name	Symbol	Address
Up/down flag	UDF	0384_{16}
Timer A0 register (Note)	TA0	0387_{16} , 0386_{16}
Timer X0 register (Note)	TX0	0389_{16} , 0388_{16}
Timer X1 register (Note)	TX1	$038B_{16}$, $038A_{16}$
Timer X2 register (Note)	TX2	$038D_{16}$, $038C_{16}$
UART0 bit rate generator	U0BRG	$03A1_{16}$
UART0 transmit buffer register	U0TB	$03A3_{16}$, $03A2_{16}$
UART1 bit rate generator	U1BRG	$03A9_{16}$
UART1 transmit buffer register	U1TB	$03AB_{16}$, $03AA_{16}$

Note: It is affected only in one-shot timer mode and pulse width modulation mode.

Table 9. Affected register (M30218 group)

Register name	Symbol	Address
Up/down flag	UDF	0384_{16}
Timer A0 register (Note)	TA0	0387_{16} , 0386_{16}
Timer A1 register (Note)	TA1	0389_{16} , 0388_{16}
Timer A2 register (Note)	TA2	$038B_{16}$, $038A_{16}$
Timer A3 register (Note)	TA3	$038D_{16}$, $038C_{16}$
Timer A4 register (Note)	TA4	$038F_{16}$, $038E_{16}$
UART0 bit rate generator	U0BRG	$03A1_{16}$
UART0 transmit buffer register	U0TB	$03A3_{16}$, $03A2_{16}$
UART1 bit rate generator	U1BRG	$03A9_{16}$
UART1 transmit buffer register	U1TB	$03AB_{16}$, $03AA_{16}$

Note: It is affected only in one-shot timer mode and pulse width modulation mode.

Table 10. Instruction table for Read Modify Write

Function	Mnemonic
Bit manipulation	BCLR, BNOT, BSET, BTSTC, BTSTS
Shift	ROLC, RORC, ROT, SHA, SHL
Arithmetic	ABS, ADC, ADCF, ADD, DEC, EXTS, INC, MUL, MULU, NEG,
	SBB, SUB
Logical	AND, NOT, OR, XOR
Jump	ADJNZ, SBJNZ

3. C language programming

Figure 3 shows an example using C programming

(1)

```
#pragma     ADDRESS   UDF   0384h
char   near   UDF;

void   func(void)
{
            ⋮
       UDF  =      0x01   ;
            ⋮
}
```

‘Image of extract’

```
    ⋮
MOV.B       #01h, UDF
    ⋮
```

(2)

```
struct bit_def {
                    char   b0:1;
                    char   b1:1;
                    char   b2:1;
                    char   b3:1;
                    char   b4:1;
                    char   b5:1;
                    char   b6:1;
                    char   b7:1;
                };

union byte_def{
                    struct bit_def bit;
                    char    byte;
                };
#pragma  ADDRESS  UDF  0384h
char  near  UDF;

#pragma  BIT  UDF_RAM
union  byte_def UDF_RAM=0 ;  /*Initial data*/

#define  TA0UD_RAM  UDF_RAM.bit.b0
#define  TA1UD_RAM  UDF_RAM.bit.b1
#define  TA2UD_RAM  UDF_RAM.bit.b2
#define  TA3UD_RAM  UDF_RAM.bit.b3
#define  TA4UD_RAM  UDF_RAM.bit.b4
#define  TA2P_RAM  UDF_RAM.bit.b5
#define  TA3P_RAM  UDF_RAM.bit.b6
#define  TA4P_RAM  UDF_RAM.bit.b7

void func(void);

void   func(void)
{
            ⋮
       TA0UD_RAM = 1 ;
       UDF = UDF_RAM.byte ;
            ⋮
}
```

‘Image of extract’

```
    ⋮
BSET        TA0UD_RAM
MOV.B       UDF_RAM, UDF
    ⋮
```

Figure 3. Example for C programming

GRADE A

MAEC TECHNICAL NEWS No.M16C-71-0105

Setting procedure of processor mode bits

Classification

Corrections and supplementary explanation of document

✔ Notes

Knowhow

Others

Products Effected

M16C/80 Series

M16C/60 Series

1. Precautions

Processor mode bits are allocated to bits 1 and 0 of the processor mode register 0. Regardless of the level of the CNVSS pin, changing the processor mode bits selects the mode. Therefore, never change the processor mode bits when changing the contents of other bits. Do not change the processor mode bits simultaneously with other bits when changing the processor mode bits "01_2" or "11_2". Change the processor mode bits after changing the other bits.

Figure 1 shows the processor mode register 0 of M16C/62A group, and figure 2 shows the setting procedure of processor mode bits.

Processor mode register 0 (Note 1)

b7	b6	b5	b4	b3	b2	b1	b0

Symbol: PM0 — Address: 0004_{16} — When reset: 00_{16} (Note 2)

Bit symbol	Bit name	Function	R	W
PM00	Processor mode bit	b1 b0 0 0: Single-chip mode 0 1: Memory expansion mode 1 0: Do not set 1 1: Microprocessor mode	○	○
PM01			○	○
PM02	R/W mode select bit	0 : RD,BHE,WR 1 : RD,WRH,WRL	○	○
PM03	Software reset bit	The device is reset when this bit is set to "1". The value of this bit is "0" when read.	○	○
PM04	Multiplexed bus space select bit	b5 b4 0 0 : Multiplexed bus is not used 0 1 : Allocated to CS2 space 1 0 : Allocated to CS1 space 1 1 : Allocated to entire space (Note4)	○	○
PM05			○	○
PM06	Port $P4_0$ to $P4_3$ function select bit (Note 3)	0 : Address output 1 : Port function (Address is not output)	○	○
PM07	BCLK output disable bit	0 : BCLK is output 1 : BCLK is not output (Pin is left floating)	○	○

Note 1: Set bit 1 of the protect register (address $000A_{16}$) to "1" when writing new values to this register.
Note 2: If the V_{CC} voltage is applied to the CNV_{SS}, the value of this register when reset is 03_{16}. (PM00 and PM01 both are set to "1".)
Note 3: Valid in microprocessor and memory expansion modes.
Note 4: If the entire space is of multiplexed bus in memory expansion mode, choose an 8-bit width.The processor operates using the separate bus after reset is revoked, so the entire space multiplexed bus cannot be chosen in microprocessor mode.
$P3_1$ to $P3_7$ become a port if the entire space multiplexed bus is chosen, so only 256 bytes can be used in each chip select.

Figure 1. Processor mode register 0

When changing into the following state after reset
• Memory expansion mode
• Entire space multiplexed bus

MOV.B #00110001B, PM0

N.G.

MOV.B #00110000B, PM0 ; Setting other bits except the
; processor mode bits
MOV.B #00110001B, PM0

O.K.

Figure 2. Setting procedure

GRADE A

MAEC TECHNICAL NEWS No.M16C-85-0204

M16C Family

Usage Precautions when Clearing Interrupt Request Bit

Classification

Corrections and supplementary explanation of document

✔Notes

Knowhow

Others

Products Effected

M16C/80 Series

M16C/60 Series

M16C/20 Series

Usage Precautions

When clearing an interrupt request bit of the interrupt control register, depending on the instruction used, an interrupt request bit may not get cleared.

Please use an MOV instruction to clear an interrupt request bit by modifying the interrupt control register.

When modifying the interrupt control register of M16C/60 and M16C/20 series microcontrollers (MCU), ensure that you only modify the interrupt control register when interrupt is disabled or a location in code where an interrupt will not generated.

Program examples of clearing interrupt request bit of M16C/60 series MCU:

Example 1: Modifying the interrupt control register

```
FCLR    I              ; Disable interrupts
MOV.B   #00H,0055H     ; Clear Timer A0 interrupt request bit
MOV.W   MEM,R0         ; Dummy read
FSET    I              ; Enable interrupts
```

Example 2: Clearing the interrupt request bit

```
FCLR    I              ; Disable interrupts
MOV.B   0055H,R0L      ; Read Timer A0 interrupt request bit
AND.B   #0F7H,R0L      ; Clear Timer A0 interrupt request bit
MOV.B    R0L,0055H     ; Write to Timer A0 interrupt request bit
MOV.W   MEM,R0         ; Dummy read
FSET    I              ; Enable interrupts
```

The reason why a dummy read is inserted before “FSET I” in Examples 1 and 2, is to prevent the interrupt enable flag I from being set before the interrupt control register is rewritten due to the effects of the instruction queue.

Moreover, please also refer to the interrupt precautions described in the manual.

MITSUBISHI SEMICONDUCTORS
USAGE NOTES REFERENCE BOOK
M16C/62 (M16C/62N, M3062GF8NFP/GP) Group

June. First Edition 2002

Editioned by
Committee of editing of Mitsubishi Semiconductor Usage Notes Reference Book

Published by
Mitsubishi Electric Corp., Semiconductor Marketing Division

M16C/62 (M16C/62N, M3062GF8NFP/GP) Group
Usage Notes Reference Book

Renesas Electronics Corporation
1753, Shimonumabe, Nakahara-ku, Kawasaki-shi, Kanagawa 211-8668 Japan