32-Bit Microcontroller V850/SV1 Usage Restrictions		Document No.	SBG-T-2477-1-E	1/2
		Date issued	September 11, 2001	
		Issued by	Microcomputer Engineering Dept. Solution Engineering Div. NEC Electron Devices NEC Corporation	
Related documents	User's manual -Hardware: U14462EJ User's manual -Architecture: U10243EJ Data sheet (Flash memory): U14622EJ Data sheet (Mask ROM): U13953EJ	Notification classification	√	Usage restriction
				Upgrade
				Document modification
				Other notification

CP(K), O

1. Affected products

V850/SV1 flash memory version

μPD70F3038/F3038Y	(on-chip flash memory/RAM: 384KB/16KB)
μPD70F3040/F3040Y	(on-chip flash memory/RAM: 256KB/16KB)

V850/SV1 mask ROM version

μPD703038/3038Y	(on-chip mask ROM/RAM: 384KB/16KB)
μPD703040/3040Y	(on-chip mask ROM/RAM: 256KB/16KB)
μPD703039/3039Y	(on-chip mask ROM/RAM: 256KB/8KB)
μPD703041/3041Y	(on-chip mask ROM/RAM: 192KB/8KB)

2. Details of restrictions

This notification concerns the following restrictions. For details, see attachment 1.

- No.10 Restriction on A/D converter
- No.11 Caution on using port 9
- No.12 Restriction on DMA priority when using an external serial clock

3. Workaround

The following workaround is available for each restriction. For details, see attachment 1.

- No.10 Read the ADCR value during A/D conversion (the ADM0.CE bit = 1).
- No.11 Be sure to set the BIC bit of the SYC register to 0 to use port 9 as an I/O port. After system reset, the BIC bit is reset to 0.
- No.12 If the interrupt source is generated in synchronization with the external clock, do not simultaneously set multiple DMA activation triggers for that interrupt source.

4. Modification schedule

Please regard these as permanent usage restrictions and cautions.

5. List of restrictions

Notes on using the V850/SV1, including the restriction history and detailed information, will be described on the following pages.

Notes on Using V850/SV1

1. Product Version

μPD70F3040/F3040Y:	Rank K, E, P
μPD70F3038/F3038Y:	Rank K
μPD703038/3038Y/703039/3039Y/3040/3040Y/3040/3041Y:	Rank K

* The rank is indicated by the letter appearing as the 5th digit from the left in the lot number marked on each product.

2. Revision History

μPD70F3040, 70F3040Y

No.	Restrictions	Rank		
		K	E	P
1	Restriction on register hardware set/reset operation when conflicts with DMA	Δ	Δ	Δ
2	Restriction on 8-bit timers (TM2 to TM5) in cascade mode	Δ	Δ	Δ
3	Restriction on variable-length CSI	Δ	Δ	Δ
4	Bug in reset start operation of the flash memory product	√	×	√
5	Bug related to I^2C bus function (Y model only)	Δ	Δ	Δ
6	Restriction on one-shot pulse output mode of 16-bit timers (TM0, TM1)	Δ	Δ	Δ
7	Restriction when DMA conflict occurs	Δ	Δ	Δ
8	Caution on interrupt servicing after EI instruction	Δ	Δ	Δ
9	Bug in power save function when an external memory is used	Δ	Δ	Δ
10	Restriction on A/D converter	Δ	Δ	Δ
11	Caution on using port 9	Δ	Δ	Δ
12	Restriction on DMA priority when using an external serial clock	Δ	Δ	Δ

√: Restriction does not apply, Δ: Restriction will also apply in future, ×: Restriction applies

μPD70F3038, 70F3038Y

No.	Restrictions	Rank
		K
1	Restriction on register hardware set/reset operation when conflicts with DMA	Δ
2	Restriction on 8-bit timers (TM2 to TM5) in cascade mode	Δ
3	Restriction on variable-length CSI	Δ
4	Bug in reset start operation of the flash memory product	√
5	Bug related to I^2C bus function (Y model only)	Δ
6	Restriction on one-shot pulse output mode of 16-bit timers (TM0, TM1)	Δ
7	Restriction when DMA conflict occurs	Δ
8	Caution on interrupt servicing after EI instruction	Δ
9	Bug in power save function when an external memory is used	Δ
10	Restriction on A/D converter	Δ
11	Caution on using port 9	Δ
12	Restriction on DMA priority when using an external serial clock	Δ

√: Restriction does not apply, Δ: Restriction will also apply in future, ×: Restriction applies μPD703038, 703038Y, 703039, 703039Y, 703040, 703040Y, 703040, 703041Y

No.	Restrictions	Rank
		K
1	Restriction on register hardware set/reset operation when conflicts with DMA	Δ
2	Restriction on 8-bit timers (TM2 to TM5) in cascade mode	Δ
3	Restriction on variable-length CSI	Δ
4	Bug in reset start operation of the flash memory product	√
5	Bug related to I^2C bus function (Y model only)	Δ
6	Restriction on one-shot pulse output mode of 16-bit timers (TM0, TM1)	Δ
7	Restriction when DMA conflict occurs	Δ
8	Caution on interrupt servicing after EI instruction	Δ
9	Bug in power save function when an external memory is used	Δ
10	Restriction on A/D converter	Δ
11	Caution on using port 9	Δ
12	Restriction on DMA priority when using an external serial clock	Δ

√: Restriction does not apply, Δ: Restriction will also apply in future, ×: Restriction applies

3. Details of Bugs and Additions to Specification

No. 1 Restriction on register hardware set/reset operation when conflicting with DMA

[Description]

If a cycle in which the hardware of a register on which a bit manipulation instruction is execured is set/reset[*1] and conflicts with a DMA cycle, setting/resetting the hardware of the register is canceled.

Example: If the DMA cycle started by a DMA start request (such as INTCSIn) coincides with the DMA transfer count end interrupt (INTDMAn) in a bit manipulation instruction cycle to an interrupt control register (DMAICn)

*1: Interrupt control register (xxIC), in-service priority register (ISPR)

[Workaround]

Do not use the bit manipulation instructions in a DMA cycle. Change the register value in byte units.

[Restriction]

Do not specify the xxIC register or ISPR register as the DIOAn register (DMA peripheral address register).

(Do not access the xxIC and ISPR register by using DMA.)

No. 2 Restriction on 8-bit timers (TM2 to TM5) in cascade mode

[Description]

When the 8-bit timers TM2 to TM5 are used in 16-bit cascade mode, the values of the higher 8 bits of the compare register cannot be set correctly.

[Workaround]

Stop both the higher and lower timers before setting the value of the compare register in 16-bit cascade mode. (Previously, it was only necessary to stop the lower timer.)

No. 3 Restriction on variable-length CSI

[Description]

Do not use the variable-length serial interface (CSI4) with a baud rate higher than the CPU operating clock; otherwise data cannot be transferred correctly.

[Workaround]

Set the baud rate of CSI4 to a value lower than the CPU operating clock.

No. 4 Bug in reset start operation of the flash memory product

[Description]

The microcontroller may become inoperable after the power is turned on or the system is reset.

[Workaround]

It is necessary to secure time for the voltage to rise sufficiently before the first fetch of the program. To do this, please insert ***40 or more*** NOP instructions at the start of the program.

(1) When using NMI

If an NMI is used (handler address: 00000010H), it is recommended to change it to external input maskable interrupt INTP0 (handler address: 00000090H), etc.

(2) When using NMI

If non-maskable interrupt INTWDT (handler address: 00000020H) of the watchdog timer is used, it is recommended to change it to maskable interrupt INTWDTM (handler address: 00000080H) of the watchdog timer.

The maskable interrupt INTWDTM is generated during an overflow by setting the fourth bit (WDTM4) of the watchdog timer mode register (WDTM) to 0.

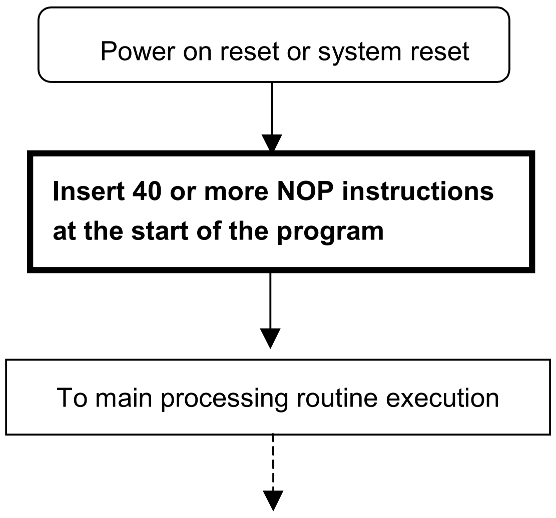

Power on reset or system reset
Insert 40 or more NOP instructions at the start of the program
To main processing routine execution

Figure) NOP Insertion Program Flowchart

Table) Interrupt Vector Table

Type	Interrupt/Exception Source	Handler Address	Number of NOP Instructions That Can Be Inserted
Reset	RESET	00000000H	8
Non-maskable	NMI	00000010H	8
Non-maskable	INTWDT	00000020H	16
Software exception	TRAP0	00000040H	8
	TRAP1	00000050H	8
Exception trap	ILGOP	00000060H	8
Maskable	INTWDTM	00000080H	16
	INTP0	00000090H	8
	:	:	:
	:	:	:

*1 NOP instruction = 2 bytes

No. 5 Bug related to I^2C bus function (Y model only) (refer to attachment 2 for details)

[Description]

The maximum operating frequency of the T model (model that includes the I^2C bus) is 17 MHz (MAX.). When using the product at 17 to 20 MHz, therefore, specification for the high-speed mode (400 kHz) and low-speed mode (100 kHz) cannot be satisfied. For the operation other than the I^2C bus, an operation at 20 kHz can be guaranteed.

No. 6 Restriction on one-shot pulse output mode of 16-bit timers (TM0, TM1)

[Description]

If the software trigger (compare register match) is selected when the 16-bit timers (TM0, TM1) are in one-shot pulse output mode, do not use the pin TI0n0 as a general-purpose port (P30/P32); otherwise, a one-shot pulse may be output at an unintended timing, depending on the level change of the I/O port.

Remark n = 0, 1.

No. 7 Restriction when DMA conflict occurs (refer to attachment 3 for details)

[Description]

If a bit operation on an interrupt register coincides with the acknowledge processing of a vectored interrupt of the register on which the bit operation is being performed, and with the start of DMA, when interrupts are enabled, the interrupt in question is issued twice.

[Workaround]

Issue a DI before, and an EI after performing a bit operation on an interrupt register (xxICn). Alternatively, clear the interrupt flag at the beginning of an interrupt processing routine.

No. 8 Caution on interrupt servicing after EI instruction (refer to attachment 4 for details)

[Description]

An interrupt detection time of about 7 clock cycles is required from the occurrence of an interrupt request until the interrupt can be received. Instructions continue to be executed during this period, and if a DI instruction is executed (disable interrupts), then interrupts are disabled. As a result, all interrupts will be held until the next EI instruction (enable interrupts) is issued.

In addition, this interrupt detection time is also required after executing the EI instruction, which means that a minimum of 7 clock cycles must elapse before interrupts can be received after the EI instruction has been executed. For this reason, if the DI instruction is executed less than 7 clock cycles after the EI instruction is executed, interrupts are held.

[Workaround]

In order for interrupts to be able to be received, be sure to put seven clock-cycles worth of instructions**Note** between the EI instruction and the subsequent DI instruction.

Note: Use instructions other than the following.

IDLE/STOP mode setting instructions
EI instruction
DI instruction
RETI instruction
LDSR instruction (on PSW)
Access to interrupt control registers (xxICn)

No. 9 Bug in power save function when an external memory is used (refer to attachment 5 for details)

[Description]

If the affected products are used under the following conditions, a discrepancy may occur between the address indicated by the program counter (PC) and the address at which the instruction is actually read following the release of a power save mode.

This may result in the CPU ignoring a 4- or 8-byte instruction from between 4 bytes and 16 bytes after an instruction is executed to write to the PSC register, which could in turn result in the execution of an erroneous instruction. Note that this bug only occurs if all of conditions (1) to (3) below are met.

[Conditions]

(1) A power save mode (IDLE or STOP) is set while an instruction is being executed on the external ROM.

(2) The power save mode is released by an interrupt.

(3) The next instruction is executed while interrupts are in a pending state following the release of the power save mode.

Note that interrupts are held pending under any of the following conditions.

<1> The NP flag of the PSW register is 1. (NMI servicing in progress/set by software)

<2> The ID flag of the PSW register is 1. (Interrupt servicing in progress/DI instruction/set by software)

<3> The EI (interrupt enable) state had been set during interrupt servicing to enable multiple interrupt servicing, but was released by an interrupt with the same or lower priority than the interrupt being serviced.

[Workaround]

(1) Do not use a power save mode (IDLE or STOP) while an instruction is being executed on the external ROM.

(2) When using a power save mode (IDLE or STOP) while an instruction is being executed on the external ROM, be sure to set the status that the interrupt can be acknowledged after the power save mode is released.

(3) If it is necessary to use a power save mode (IDLE or STOP) while an instruction is being executed on the external ROM, take the software countermeasures shown below.

<1> Insert 6 NOP instructions 4 bytes after an instruction that writes to the PSC register.

<2> Insert the br $+2 instruction after the NOP instructions to eliminate the PC discrepancy.

No. 10 Restriction on A/D converter

[Description]

If the value of the A/D conversion result register (ADCR) is read after the conversion operation is stopped (the ADM0.CE bit = 0), an undefined value may be read.

[Workaround]

Read the ADCR value during A/D conversion (the ADM0.CE bit = 1).

No. 11 Caution on using port 9

[Description]

When port 9 is used as an output port (the MM register = 00H), if the BIC bit of the SYC register is set to 1, port 9 does not operate as output port correctly.

[Workaround]

Be sure to set the BIC bit of the SYC register to 0 to use port 9 as an I/O port. After system reset, the BIC bit is reset to 0.

No. 12 Restriction on DMA priority when using an external serial clock

[Description]

When CSI is used with an external serial clock, if DMA transfer is performed by inputting the DMA activation trigger interrupt (INTCSI) to multiple DMA channels, the priority order of the DMA channels may be changed.

[Workaround]

If the interrupt source is generated in synchronization with the external clock do not simultaneously set multiple DMA activation triggers for that interrupt source.

4. Other Cautions

None.

No. 5 Bug related to I^2C bus function

(1) Restart operation

[Description]

Address data will be corrupted if the transfer address to the IIC0 register is set immediately after setting the STT bit before restart.

[Workaround]

Write the address data to the IIC0 shift register when the start condition trigger is cleared (STT = 0) and after the start condition has been detected (STD = 1).

(2) Operation while reserving the device for communications (during multi-master operation)

[Description]

If slave mode is selected while reserving the microcontroller for communications, it will not function as a slave.

[Workaround]

Write the STT bit to 0 at the same time as writing the WREL bit to 1. Then, write the STT bit to 1 again after the stop condition is detected.

(3) Start operation after the device is reserved for communications (during multi-master operation)

[Description]

Malfunction results when a start condition (STT = 1) is generated after detecting a stop condition (SPT = 1).

[Workaround]

Check that the TRC bit is 1 (transmit operation) after verifying that the MSTS bit is 1 (master communication status). If the TRC bit is 0 at this time, set LREL and release communication, and then reserve communications (STT = 1) again after the stop condition is detected.

(4) Conflict between STT set timing and start condition detection

[Description]

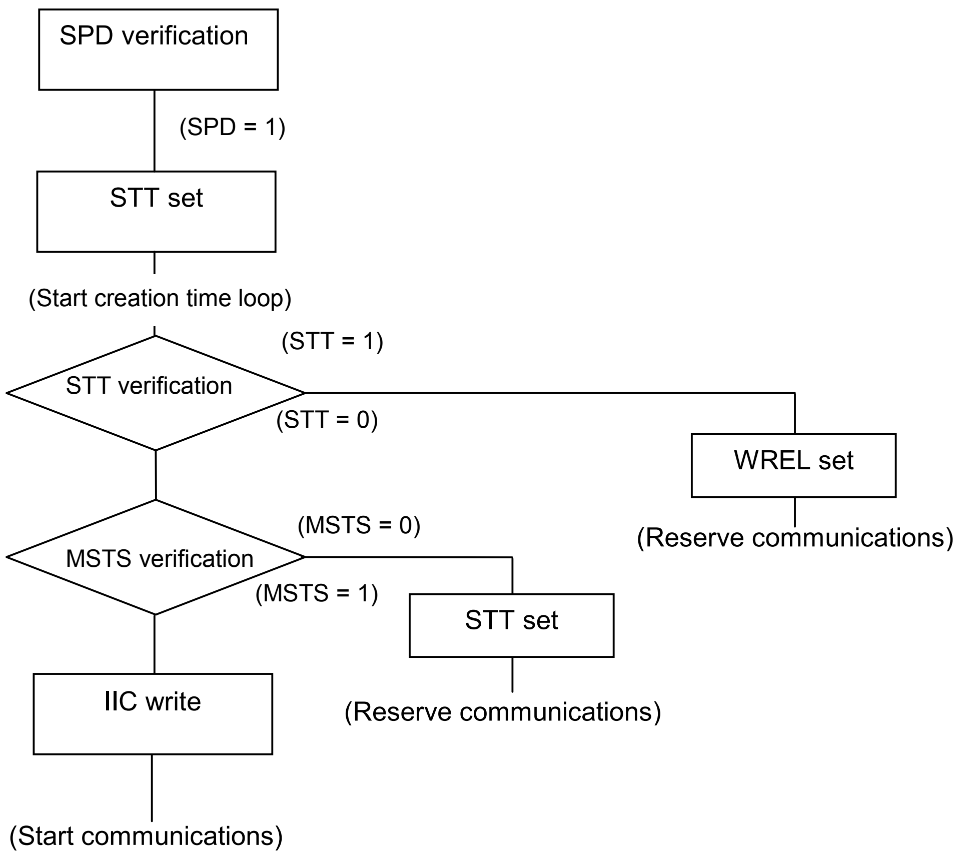
The following bugs may occur as a result of conflict between STT set timing and start condition detection.

- STT bit set operation may be rejected.
- The device may be incorrectly recognized as the master (MAST = 1) and the STT bit cleared.
- A WAIT condition may result from a fall in SCL if STT is set and SCL is in a high-level state (SPD = 1) when the target device has generated a start condition.

[Workaround]

STT set timing (during multi-master operation)

Provide software that performs the operations in the flow chart below.



Ensure that the time from SPD verification to WREL set in the above flow chart is as follows.

In high-speed mode:	17.5 μs or less
In standard mode:	70.0 μs of less

(5) When the STD bit and SPD bit are 1

[Description]

The start condition will occur twice if a start condition is issued when both the STD bit and SPD bit are 1.

[Workaround]

There is no fail-proof software workaround. The probability that this bug occurs can be reduced with the method below.

After confirmation that STD and SPD are both 1 when the SCL level and SDA are high, the bus appears to have been released for an interval greater than the time taken to send 1 byte of data, even though a start condition has been issued. Consequently, the user should make sure that STD and SPD are not both equal to 1 by clearing the STD bit by setting the LREL bit (wait state after releasing communication).

(6) Transfer clock setting

The maximum operating frequency of the T model (model that includes the I^2C bus) is 17 MHz (MAX.). When using the product at 17 to 20 MHz, therefore, specification for the high-speed mode (400 kHz) and low-speed mode (100 kHz) cannot be satisfied. For the operation other than the I^2C bus, an operation at 20 kHz can be guaranteed.

Refer to the data sheet for details.

No. 7 Restriction when DMA conflict occurs

[Description]

If the three conditions below occur at the same time while interrupts are enabled (EI), an interrupt that should only occur once occurs twice.

(1) A bit manipulation instruction (set1, clr1, not1, tst1) is executed on an interrupt request flag (xxIFn) of an interrupt control register (xxICn).
(2) Interrupt processing occurs involving the hardware of the same register as in (1).
(3) DMA is started while executing the above bit operation instruction.

Remark xx: Identifier of each peripheral unit (WDT, P, WTI, TM, CS, SER, SR, ST, AD, DMA, WT)
N: Peripheral unit number (0 to 7)

When the above conditions (1) to (3) coincide and interrupt processing begins, the interrupt request flag that should be reset by hardware is not reset, so that after returning from the interrupt processing routine (RETI instruction), the interrupt processing is executed again. (See figures 1 and 2.)

Example During a bit operation on the interrupt request flag (CSIF0) of the CSIC0 register using the clr1 instruction, the non-masked INTCSI0 interrupt occurs, which then conflicts with the start of DMA. In this case, the INTCSI0 interrupt processing is executed twice.

[Workaround]

(1) Issue the DI instruction before executing a bit operation instruction on the interrupt request flags (xxIFn) of an interrupt control register (xxICn) by software. Then issue an EI instruction afterwards, and do not execute interrupt processing immediately after the bit operation instruction. (See figure 3.)
(2) When interrupt processing begins, interrupts are disabled by hardware (the DI state), so clear the interrupt request flag in every interrupt processing routine before executing the EI instruction. (See figure 4.)

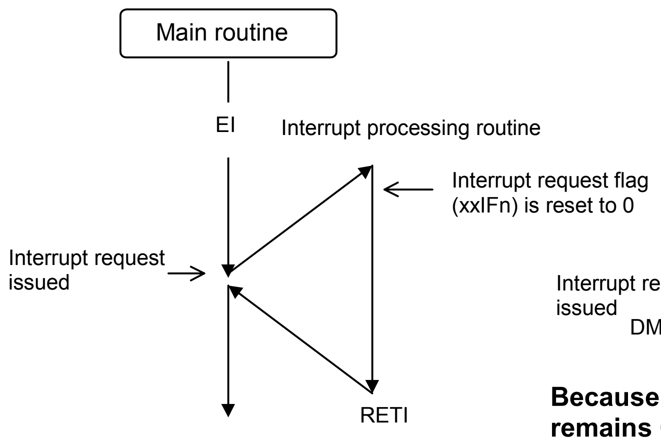

Main routine
EI
Interrupt processing routine
Interrupt request flag (xxIFn) is reset to 0
Interrupt request issued
RETI

Figure 1. Normal Interrupt Processing

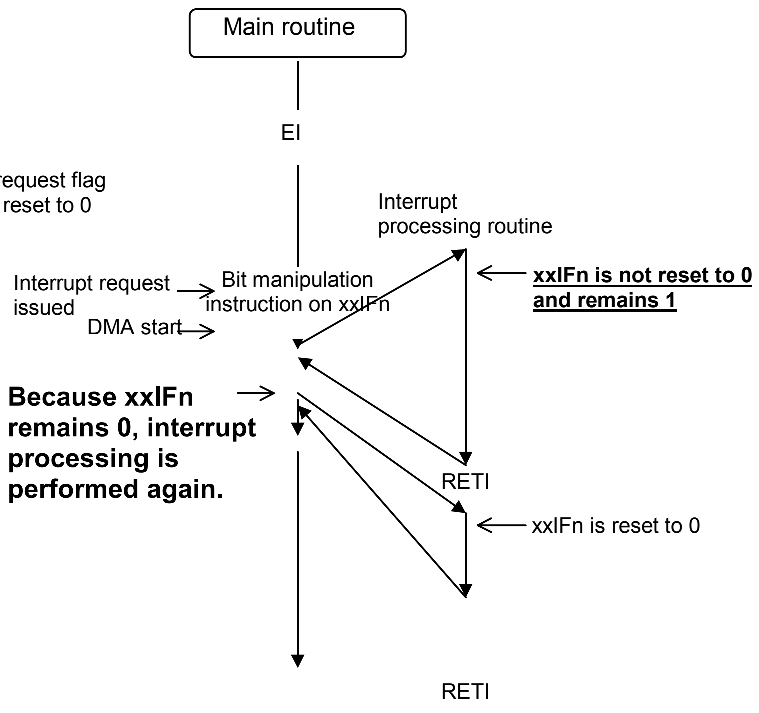

Main routine
EI
Interrupt processing routine
Interrupt request issued
Bit manipulation instruction on xxIFn
DMA start
xxIFn is not reset to 0 and remains 1
Because xxIFn remains 0, interrupt processing is performed again.
RETI
xxIFn is reset to 0
RETI

Figure 2. Interrupt Processing Causing This Bug

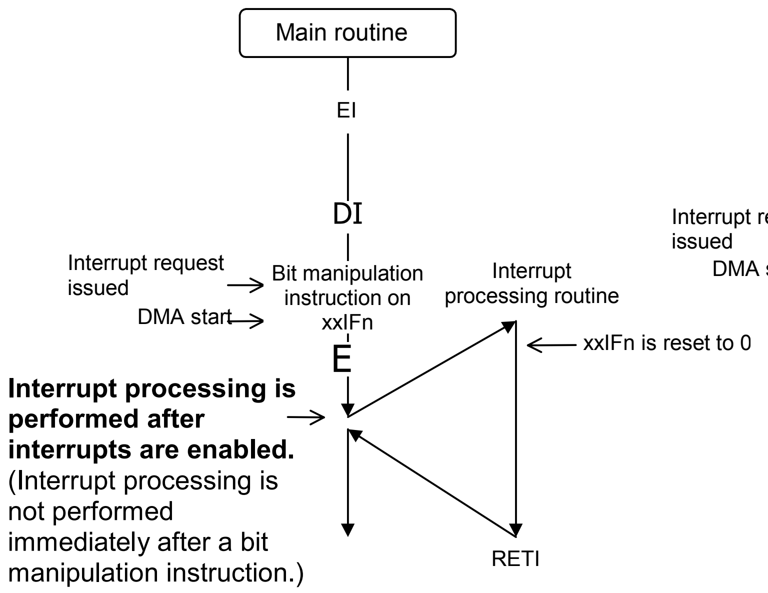

Main routine
EI
DI
Interrupt request issued
Bit manipulation instruction on xxIFn
DMA start
E
Interrupt processing routine
xxIFn is reset to 0
Interrupt processing is performed after interrupts are enabled.
(Interrupt processing is not performed immediately after a bit manipulation instruction.)
RETI

Figure 3. Workaround (1)

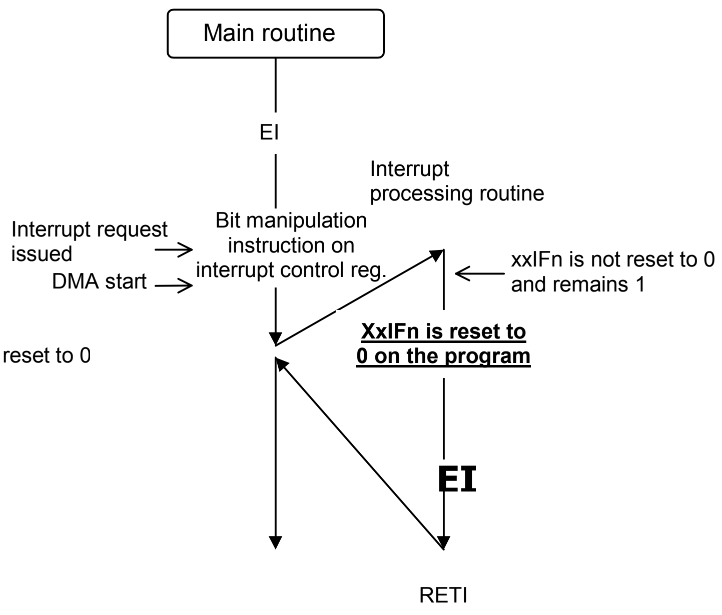

Main routine
EI
Interrupt processing routine
Interrupt request issued
Bit manipulation instruction on interrupt control reg.
DMA start
xxIFn is not reset to 0 and remains 1
XxIFn is reset to 0 on the program
EI
RETI

Figure 4. Workaround (2)

Remark xx: Identifier of each peripheral unit (WDT, P, WTI, TM, CS, SER, SR, ST, AD, DMA, WT)
N: Peripheral unit number (0 to 7)

No. 8 Caution on interrupt servicing after EI instruction

[Description]

An interrupt detection time of about 7 clock cycles is required from the occurrence of an interrupt request until the interrupt can be received. Instructions continue to be executed during this period , and if a DI instruction is executed (disable interrupts), then interrupts are disabled. As a result, all interrupts will be held until the next EI instruction (enable interrupts) is issued.

In addition, this interrupt detection time is also required after executing the EI instruction, which means that a minimum of 7 clock cycles must elapse before interrupts can be received after the EI instruction has been executed. For this reason, if the DI instruction is executed less than 7 clock cycles after the EI instruction is executed, interrupts are held.

Example Case where the EI instruction is ineffective

Program example

```
     DI
      :              ; MK flag = 0 (enable interrupt processing)
      :              ; Interrupt request issued (IF flag = 1)
     EI
     JR LP1
      :              ; Less than 3 clock cycles elapse from EI to DI (3 clock cycles)
      :
      :
LP1:
     DI              ; Interrupt request is not received.
      :
```

[Workaround]

In order for interrupts to be able to be received, be sure to put seven clock cycles worth of instructions[Note] between the EI instruction and the subsequent DI instruction.

Note: Use instructions other than the following:

- IDLE/STOP mode setting instructions
- EI instruction
- DI instruction
- RETI instruction
- LDSR instruction (on PSW)
- Access to interrupt control registers (xxICn)

Example of modified program

```
      DI
       :               ; MK flag = 0 (enable interrupt processing)
       :               ; Interrupt request issued (IF flag = 1)
      EI
      NOP              ; 1 system clock cycle
      NOP              ; 1 system clock cycle
      NOP              ; 1 system clock cycle
      NOP              ; 1 system clock cycle
      JR LP1           ; 3 system clock cycles (branch to LP1 routine)
       :

LP1 :                  ; LP1 routine
DI                     ; Interrupt processing is executed at 8th clock cycle after the EI
                         instruction.
 :
```

No. 9 Bug in power save function when an external memory is used

[Description]

If the affected products are used under the following conditions, a discrepancy may occur between the address indicated by the program counter (PC) and the address at which the instruction is actually read following the release of a power save mode.
This may result in the CPU ignoring a 4- or 8-byte instruction from between 4 bytes and 16 bytes after an instruction is executed to write to the PSC register, which could in turn result in the execution of an erroneous instruction. Note that this bug only occurs if all of conditions (1) to (3) below are met.

[Conditions]

(1) A power save mode (IDLE or STOP) is set while an instruction is being executed on the external ROM.
(2) The power save mode is released by an interrupt.
(3) The next instruction is executed while interrupts are in a pending state following the release of the power save mode.
Note that interrupts are held pending under any of the following conditions.
<1>The NP flag of the PSW register is 1. (NMI servicing in progress/set by software)
<2> The ID flag of the PSW register is 1. (Interrupt servicing in progress/DI instruction/set by software)
<3>The EI (interrupt enable) state had been set during interrupt servicing to enable multiple interrupt servicing, but was released by an interrupt with the same or lower priority than the interrupt being serviced.

[Bug operation]

The operation of the bug is shown below using the power save mode setting example from the user's manual.
(rD: PSC setting value, rX: Value written to PSW, rY: Value written back to PSW, assuming PSW has been set)

```
ldsr  rX,5             ; Sets PSW to the value of rX
st.b  r0,PRCMD[r0]     ; Writes to PRCMD
st.b  rD,PSC[r0]       ; Sets the PSC register        (PSC setting)
ldsr  rY,5             ; Returns the value of PSW     (After 4 bytes)
nop                    ; 2 to 5 NOP instructions      (After 6 bytes)  ┐
nop                    ;                              (After 8 bytes)  │  Bug occurs here
nop                    ;                              (After 10 bytes) │   <1>Discrepancy with
nop                    ;                              (After 12 bytes) ├     PC
nop                    ;                              (After 14 bytes) │   <2>Instructions
(Next instruction)     ;                              (After 16 bytes) ┘     ignored
```

[Workaround]

(1) Do not use a power save mode (IDLE or STOP) while an instruction is being executed on the external ROM.

(2) If it is necessary to use a power save mode (IDLE or STOP) while an instruction is being executed on the external ROM, take the software countermeasures shown below.

<1> Insert 6 NOP instructions 4 bytes after an instruction that writes to the PSC register.

<2> Insert the br $+2 instruction after the NOP instructions to eliminate the PC discrepancy.

[Program example]

(rD: PSC setting value, rX: Value written to PSW, rY: Value written back to PSW, assuming PSW has been set)

```
ldsr  rX,5              ; Sets PSW to the value of rX
st.b  r0,PRCMD[r0]      ; Writes to PRCMD
st.b  rD,PSC[r0]        ; Sets the PSC register
ldsr  rY,5              ; Returns the value of PSW

nop                     ; <1> 6 or more NOP instructions
nop
nop
nop
nop
nop

br  $+2                 ; <2> Eliminates PC discrepancy
```