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.

GRADE A

MAEC TECHNICAL NEWS No.M16C-84-0204

M16C/62N Flash Memory Versions and M3062GF8NFP/GP Usage Precaution on Stop Mode

Classification

Corrections and supplementary explanation of document

✔Notes

Knowhow

Others

Products Effected

M16C/62N Flash memory version

M3062GF8NFP/GP

1. Usage precaution

An undefined operation can occur after returning to normal operation mode from a stop mode because an undefined interrupt is generated or, a BRK instruction occurred, etc, as can be seen on the code below.

```
NOP
NOP
NOP
NOP
BSET 0, CM1          ;Set the all clock stop control bit to “1” (stop mode)
NOP
NOP
NOP
NOP
```

This behavior will not occur on the emulator.

2. Causes

An erroneous value may be read out of flash memory when a STOP mode is entered while reading data out of the flash memory. When this erroneous value is executed, an undefined operation may occur due to an undefined interrupt, a BRK instruction, etc.

For MASK ROM versions, this behavior will not occur. A correct value is always read out even when a STOP mode is encountered while reading out from ROM.

3. Generating conditions

This behavior may occur on M16C/62N Flash memory versions and M3062GF8NFP/GP.

An instruction code to be executed is stored in the instruction queue buffer temporarily. The number of instruction code stored in the instruction queue buffer varies according to the program assignment (even, odd addresses) and the number of memory wait.

(1) In Figure 1-(1), the erroneous instruction read will not be executed because interrupt processing has higher priority. -> This behavior will not occur.

(2) In Figure 1-(2), the erroneous instruction read will be executed because it occurred before interrupt processing. -> This behavior will occur.

4. Countermeasures

Execute the JMP.B instruction after writing to the all clock stop control bit. The instruction code prefetched at the time of shifting to stop mode is latched in the flash memory before entering into stop mode by executing a JMP.B instruction.
Therefore, erroneous value is not prefetched as instruction code.

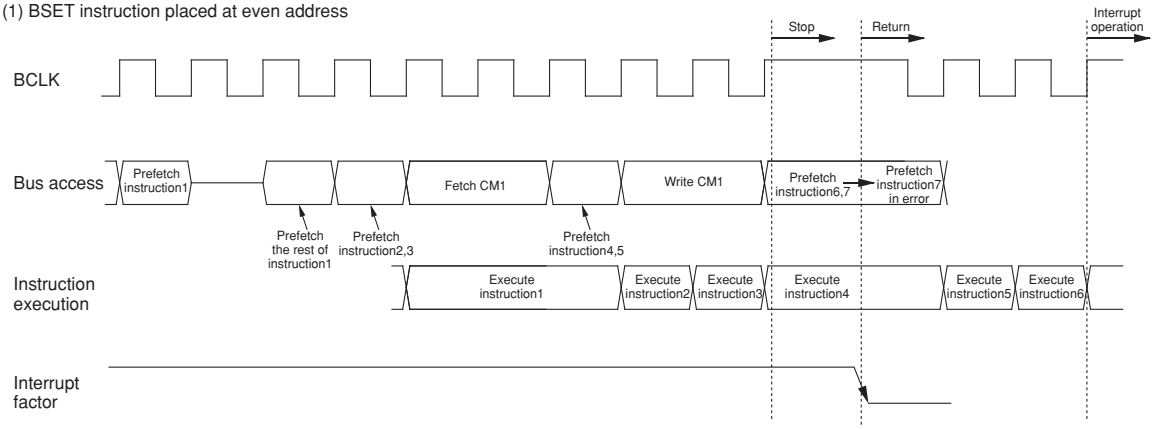
Disable DMA transfer.

Figures 2 and 3 show the countermeasure program examples and operation timing.

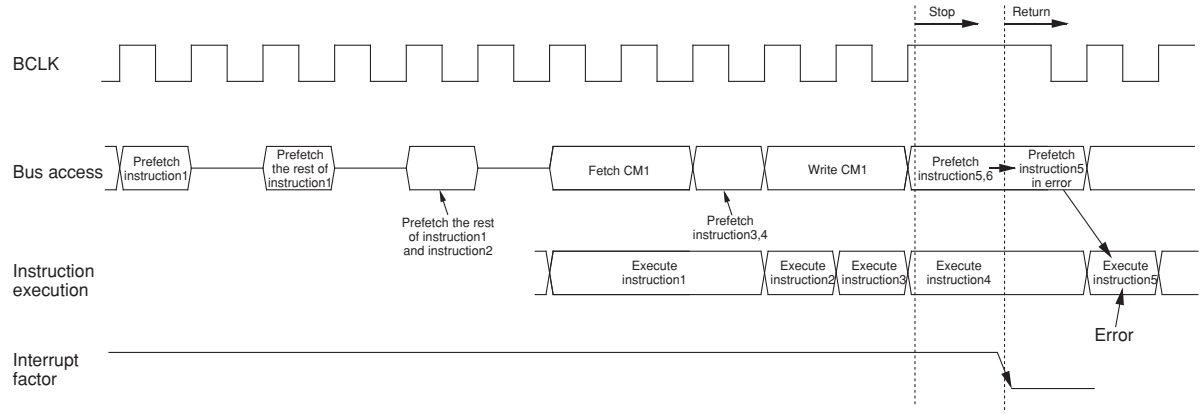
Sample program

```
BSET 0, CM1     ; Instruction 1 (4 bytes)
NOP             ; Instruction 2 (1 byte)
NOP             ; Instruction 3 (1 byte)
NOP             ; Instruction 4 (1 byte)
NOP             ; Instruction 5 (1 byte)
```

(1) BSET instruction placed at even address



(2) BSET instruction placed at odd address



Note 1: This is a brief outline and some parts about return from interrupt were omitted.
Note 2: The BSET instruction accompanies read operation of the corresponding address. This causes to inhibit prefetching and storing access of the instruction code to the instruction queue buffer and generate the said phenomenon easily.

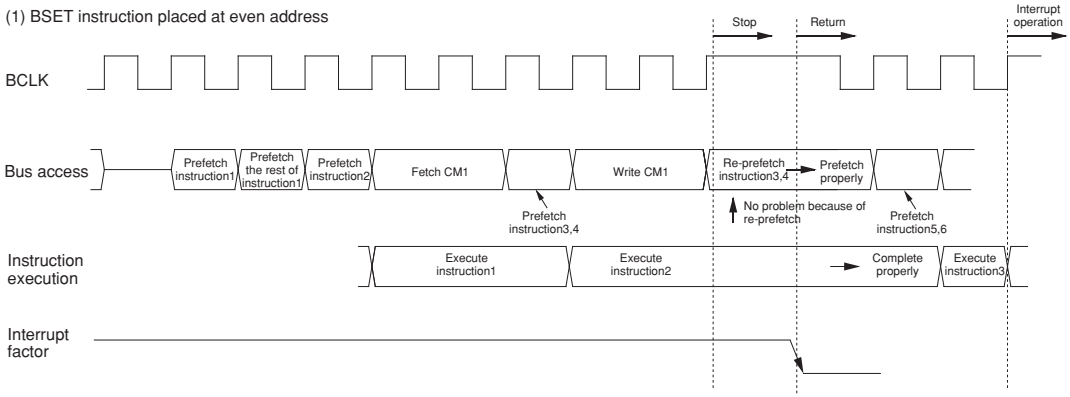
Figure 1. Program example and operation timing of the behavior described in “1. Usage precaution.”

Countermeasure program

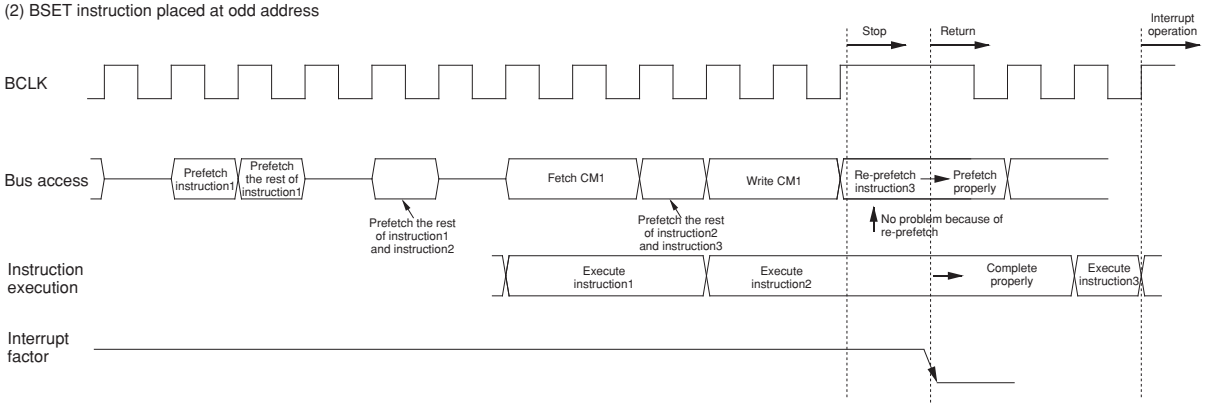
Example 1

```
        BSET  0, CM1    ; Instruction 1 (4 bytes)
        JMP.B     L1    ; Instruction 2 (2 bytes)
L1:
        NOP             ; Instruction 3 (1 byte)
        NOP             ; Instruction 4 (1 byte)
        NOP             ; Instruction 5 (1 byte)
        NOP             ; Instruction 6 (1 byte)
```

(1) BSET instruction placed at even address



(2) BSET instruction placed at odd address



Note: This is a brief outline and some parts about return from interrupt were omitted.

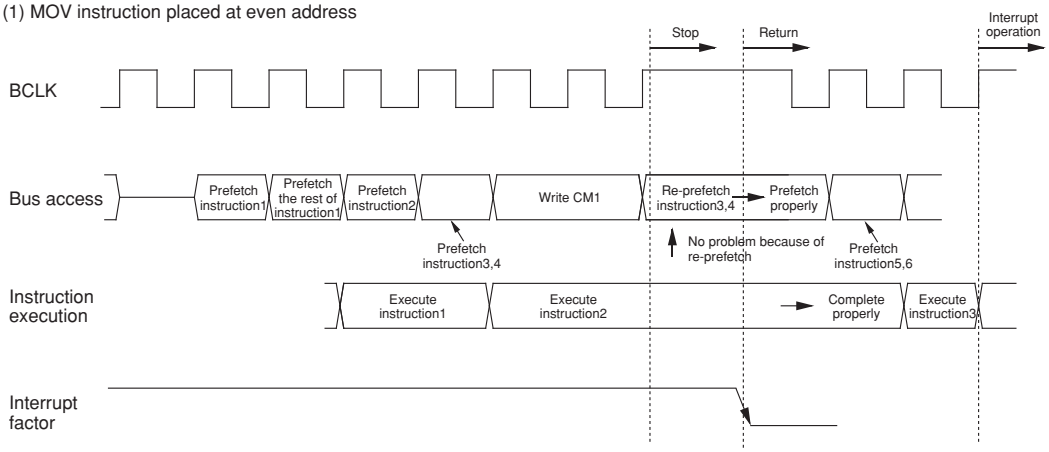
Figure 2. Countermeasure program example and operation timing (Example 1)

Countermeasure program

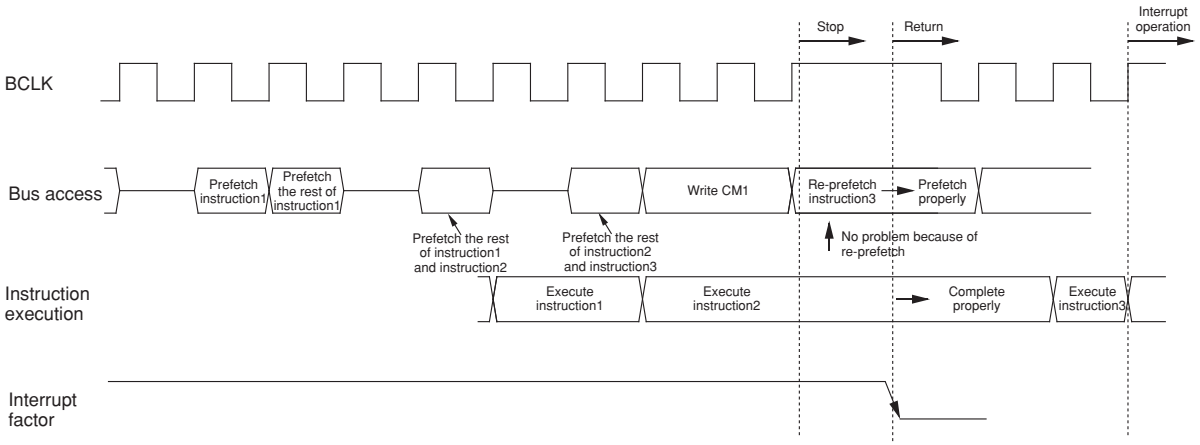
Example 2

```
        MOV.B:S  #21H, CM1  ; Instruction 1 (4 bytes)
        JMP.B    L1         ; Instruction 2 (2 bytes)
L1:
        NOP                 ; Instruction 3 (1 byte)
        NOP                 ; Instruction 4 (1 byte)
        NOP                 ; Instruction 5 (1 byte)
        NOP                 ; Instruction 6 (1 byte)
```

(1) MOV instruction placed at even address



(2) MOV instruction placed at odd address



Note: This is a brief outline and some parts about return from interrupt were omitted.

Figure 3. Countermeasure program example and operation timing (Example 2)