Microcomputer Technical Information

CP(K), O

QB-V850ESFX2 In-Circuit Emulator for V850ES/FE2, V850ES/FF2, V850ES/FG2, V850ES/FJ2, μPD703229Y, μPD70F3229Y Usage Restrictions		Document No.	ZBG-CD-05-0093		1/3
		Date issued	October 25, 2005		
		Issued by	Development Tool Group Multipurpose Microcomputer Systems Division 4th Systems Operations Unit NEC Electronics Corporation		
Related documents	QB-V850ESFX2 User's Manual: ZUD-CD-05-0129	Notification classification	√	Usage restriction	
				Upgrade	
				Document modification	
				Other notification	

1. Target product

Product	Control Code[Note]	Remark
QB-V850ESFX2-xxx-yyy	A, B, C	xxx and yyy are arbitrary codes.

Note For the identification method for the control code, see the Operating Precautions supplied with the product.

2. New restrictions

Restrictions No. 17 and No. 18 have been added. See the attachment for details.

No. 17 Illegal break occurs during program execution in internal RAM (2)

No. 18 Address is not retained during external bus access

The erroneous description on restriction No. 7 has been corrected. After correction, regard one of the restriction conditions as a permanent restriction.

- Corrected item: No. 7 Bug in program execution and DMA transfer in internal RAM
- Before correction:

[Description]

When a DMA transfer for the internal RAM and a bit manipulation instruction (SET1, CLR1, or NOT1) allocated in the internal RAM or a data access instruction for a misaligned address are executed simultaneously, the CPU may deadlock due to conflict between the internal bus operations. At this time, only a reset can be acknowledged. (An NMI or interrupt cannot be acknowledged.)

[Workaround]

Implement any of the following workarounds.

- Do not perform a DMA transfer for the internal RAM when an instruction allocated in the internal RAM is being executed.

- Do not execute an instruction allocated in the internal RAM when a DMA transfer for the internal RAM is being performed.

This bug has been corrected in control code B or later.

- After correction:

[Description]

When a DMA transfer for the internal RAM and an instruction of (1) or (2) described below are executed simultaneously, the CPU may deadlock due to conflict between the internal bus operations. At this time, only a reset can be acknowledged. (An NMI or interrupt cannot be acknowledged.)

(1) A bit manipulation instruction (SET1, CLR1, or NOT1) allocated in the internal RAM

(2) A data access instruction for a misaligned address allocated in the internal RAM

[Workaround]

Implement either of the following workarounds.

(1) For a bit manipulation instruction (SET1, CLR1, or NOT1) allocated in the internal RAM

- Do not perform a DMA transfer for the internal RAM when a bit manipulation instruction allocated in the internal RAM is being executed.
- Do not execute a bit manipulation instruction allocated in the internal RAM when a DMA transfer for the internal RAM is being performed.

This restriction has been corrected in control code B or later.

(2) For a data access instruction for a misaligned address allocated in the internal RAM

- Do not perform a DMA transfer for the internal RAM when a data access instruction for a misaligned address allocated in the internal RAM is being executed.
- Do not execute a data access instruction for a misaligned address allocated in the internal RAM when a DMA transfer for the internal RAM is being performed.

Please regard this item as a permanent restriction.

3. Workarounds

See the attachment for details on workarounds for restrictions added in this edition.

4. Modification schedule

No. 17 This item is not planned for correction. Please regard this item as a permanent restriction.

No. 18 This restriction will be corrected by upgrading the device file according to the following schedule.

DF703229 V2.01 (next version): Mid-November, 2005

DF703239 V2.11 (next version): Mid-November, 2005

* Note that this schedule is subject to change without notice. For the detailed release schedule of the modified products, contact an NEC Electronics sales representative.

5. List of restrictions

See the attachment for details.

6. Addition of supported devices

The devices described below are now supported by the QB-V850ESFX2.

The use of the newly supported devices will be added to the “QB-V850ESFX2 User’s Manual (ZUD-CD-05-0129)”.

- Devices newly supported by QB-V850ESFX2:

  V850ES/HE2, V850ES/HF2, V850ES/HG2, V850ES/HJ2

- QB-V850ESFX2 User’s Manual:

  Revision schedule: September 26, 2005

  Document title: QB-V850ESFX2 User’s Manual

  Document number: ZUD-CD-05-0129

7. Document revision history

QB-V850ESFX2 In-Circuit Emulator for V850ES/FE2, V850ES/FF2, V850ES/FG2, V850ES/FJ2, μPD703229Y, μPD70F3229Y - Usage Restrictions

Document Number	Issued on	Description
ZBG-BG-05-0003	April 22, 2005	1st edition
ZBG-CD-05-0093	October 25, 2005	2nd edition - Addition of restrictions No. 17 and No. 18 - Correction of erroneous description in No. 7 - Addition of supported devices - Deletion of restrictions No. 1 to No. 13 (because these items have been moved to CHAPTER 4 in QB-V850ESFX2 User’s Manual (ZUD-CD-05-0129))

Notes on Using QB-V850ESFX2

This document describes restrictions applicable only to the emulator and restrictions that are planned for correction in the emulator.

Refer to the following documents for the restrictions in the target device.

- User's manual of target device
- Restrictions notification document for target device

Also refer to the user's manual of the emulator for cautions on using the emulator.

1. Product Version

Control Code[Note]	Remark
A	–
B	–
C	–

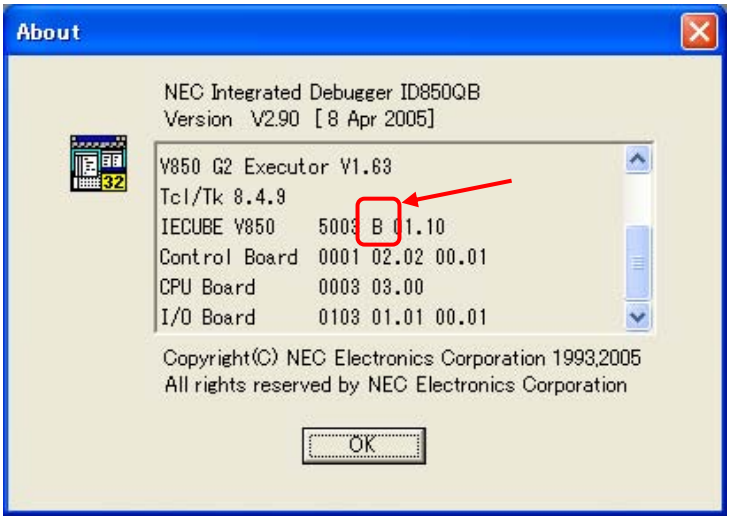
Note The “control code” is the second digit from the left in the 10-digit serial number printed on the sticker attached to the bottom side of IECUBE (if it has not been upgraded).

If the product has been upgraded, the control code can also be checked with the following methods while the debugger is running.

- When using ID850QB

  Click the [Help] menu and then click the About submenu to display the About dialog box.

  “X” in “IECUBE V850 **** X **.**” is the control code.



- When using Green Hills Software (GHS)’s debugger MULTI®

  Execute the version command of 850eserv.

  “X” in “IECUBE Control Code=X” is the control code.

```
850eserv Version: 3.2342 (for MULTI V4.0.x)
IE type=NU85E Full ICE Generation 2 (IECUBE)
Executor Version=V850 G2 Executor V1.63  Copyright 2004
Device File Format Version=V2.18
Device File File Version=V2.10
IECUBE Control Code=B
IECUBE Firmware Version=V1.10
Control Board Version=V2.02 (FPGA Version=0.01)
CPU Board Version=V3.00
I/O Board Version=V1.01 (FPGA Version=0.01)
```

2. Product History

No.	Bugs and Changes/Additions to Specifications		Control Code		
			A	B	C
~~4~~[Note]	~~Caution on flash self programming function~~		~~Permanent restriction~~		
~~2~~[Note]	~~Caution on Non-Map break~~		~~Permanent restriction~~		
~~3~~[Note]	~~Caution on DBTRAP instruction~~		~~Permanent restriction~~		
~~4~~[Note]	~~PSC register access~~		~~Permanent restriction~~		
~~5~~[Note]	~~Caution on DBPC, DBPSW, and ECR registers~~		~~Permanent restriction~~		
~~6~~[Note]	~~Caution on trace display order~~		~~Permanent restriction~~		
~~7~~[Note]	~~Caution on extension probe~~		~~Permanent restriction~~		
~~8~~[Note]	~~Simultaneous execution of two instructions when hardware break is set~~		~~Permanent restriction~~		
~~9~~[Note]	~~Caution on on-chip debug function~~		~~Permanent restriction~~		
~~10~~[Note]	~~Caution on standby mode~~		~~Permanent restriction~~		
~~11~~[Note]	~~Operation during break~~		~~Permanent restriction~~		
~~12~~[Note]	~~Caution on RAM hold voltage flag~~		~~Permanent restriction~~		
~~13~~[Note]	~~Caution on power consumption~~		~~Permanent restriction~~		
1	Bug in accessing UAnRX register during break (n = 0 to 3)		Permanent restriction		
2	Bug in accessing CBnRX register during break (n = 0 to 2)		Permanent restriction		
3	Bug in accessing CnRGPT register during break (n = 0 to 3)		Permanent restriction		
4	Bug in accessing CnTGPT register during break (n = 0 to 3)		Permanent restriction		
5	Bug in accessing CnGNCTRL register during break (n = 0 to 3)		Permanent restriction		
6	Bug related to DMA transfer forcible termination (n = 0 to 3)		×	√	√
7	Bug in program execution and DMA transfer in internal RAM	(1) Bit manipulation instruction	×	√	√
		(2) Access for misaligned address	Permanent restriction		
8	Emulator hangs up upon internal reset		×	√	√
9	Emulator hangs up while downloading data or setting software break		Avoidable by debugger upgrade		
10	Data loss occurs when external RAM is connected		Avoidable by debugger upgrade		
11	Restriction on emulation of POC circuit and clock monitor		Avoidable by debugger upgrade		
12	Restriction on setting flash memory mask option		Avoidable by debugger upgrade		
13	Illegal break occurs during program execution in internal RAM (1)		Permanent restriction		
14	Restriction on reset input during a break		×	√	√
15	Bug that occurs upon entering and releasing STOP mode when RESET pin is masked		Permanent restriction		
16	Bugs in A/D conversion function during a break		Permanent restriction		
17	Illegal break occurs during program execution in internal RAM (2)		Permanent restriction		
18	Address is not retained during external bus access		Avoidable by device file upgrade		

×: Applicable, √: Not applicable or already corrected

Note The description of this restriction has been moved to CHAPTER 4 in the QB-V850ESFX2 In-Circuit Emulator User's Manual (ZUD-CD-05-0129).

3. Details of Bugs and Added Specifications

No. 1 Bug in accessing UAnRX register during break (n = 0 to 3)

[Description]

An overrun error occurs under the following conditions (a) to (c).

(a) If a break occurs after reading the UART receive buffer register (UAnRX) and the UAnRX register is displayed in the I/O register window of the debugger, an overrun error occurs when UART reception is performed next time.

(b) If a software break occurs immediately after reading the UART receive buffer register (UAnRX), an overrun error occurs when UART reception is performed next time regardless of whether or not the UAnRX register is displayed in the I/O register window.

(c) If a DMA transfer from the UART receive buffer register (UAnRX) is performed during a break[Note], an overrun error occurs when UART reception is performed next time.

Note Including breaks by the RAM monitor function, DMM function, step execution, fail-safe break, or breaks due to event change while the program is running. The real-time RAM monitor function does not cause this bug because it does not set breaks.

Remark An overrun error also occurs when UART receives data multiple times during a break. This is an emulator specification.

[Workaround]

(a) Do not display the UAnRX register in the I/O register window.

(b) Set a hardware break when setting a break immediately after reading the UAnRX register.

(c) There is no workaround.

Please regard items (a), (b), and (c) as permanent restrictions.

No. 2 Bug in accessing CBnRX register during break (n = 0 to 2)

[Description]

When the CSIBn receive data register (CBnRX) is read, it usually starts the next reception operation. Under the following conditions (a) and (b), however, the next reception operation is not started even if CBnRX is read.

(a) If a software break occurs immediately after reading the CSIBn receive data register (CBnRX).

(b) If a DMA transfer from the CSIBn receive data register (CBnRX) is performed during a break[Note].

As a result, communication stops or the DMA controller stops.

Note Including breaks by the RAM monitor function, DMM function, step execution, fail-safe break, or breaks due to event change while the program is running. The real-time RAM monitor function does not cause this bug because it does not set breaks.

[Workaround]

(a) Set a hardware break when setting a break immediately after reading the CBnRX register.

(b) There is no workaround.

Please regard items (a) and (b) as permanent restrictions.

No. 3 Bug in accessing CnRGPT register during break (n = 0 to 3)

[Description]

Under the following conditions (a) and (b), the read pointer (RGPT) that should be incremented is not incremented, and the same data as previously read is read.

(a) If a software break occurs immediately after reading the CANn module receive history list register (CnRGPT).

(b) If a DMA transfer from the CANn module receive history list register (CnRGPT) is performed during a break[Note].

Note Including breaks by the RAM monitor function, DMM function, step execution, fail-safe break, or breaks due to event change while the program is running. The real-time RAM monitor function does not cause this bug because it does not set breaks.

[Workaround]

(a) Set a hardware break when setting a break immediately after reading the CnRGPT register.

(b) There is no workaround.

Please regard items (a) and (b) as permanent restrictions.

No. 4 Bug in accessing CnTGPT register during break (n = 0 to 3)

[Description]

Under the following conditions (a) and (b), the read pointer (TGPT) that should be incremented is not incremented, and the same data as previously transmitted is transmitted.

(a) If a software break occurs immediately after reading the CANn module transmit history list register (CnTGPT).

(b) If a DMA transfer from the CANn module transmit history list register (CnTGPT) is performed during a break[Note].

Note Including breaks by the RAM monitor function, DMM function, step execution, fail-safe break, or breaks due to event change while the program is running. The real-time RAM monitor function does not cause this bug because it does not set breaks.

[Workaround]

(a) Set a hardware break when setting a break immediately after reading the CnTGPT register.

(b) There is no workaround.

Please regard items (a) and (b) as permanent restrictions.

No. 5 Bug in accessing CnGNCTRL register during break (n = 0 to 3)

[Description]

When a register access is performed in the following sequence, a forcible shutdown that should not take place normally may occur after the sequence is complete.

[Sequence for bug occurrence]

(1) The EFSD bit of the CANn module control register (CnGMCTRL) is set.

(2) The I/O register[Note] is accessed.

(3) The GOM bit of the CANn module control register (CnGMCTRL) is cleared.

Note I/O register access except for clearing the GOM bit of the CnGMCTRL register

Conditions under which a forcible shutdown takes place are shown below.

(a) If a break occurs immediately after the I/O register access in (2) occurs

(b) If a break by the RAM monitor function or DMM function occurs immediately after the I/O register access in (2) occurs

(c) Stepwise execution is performed for the I/O register access in (2)

[Workaround]

Be sure to set the EFSD bit and clear the GOM bit successively when executing a forcible shutdown.

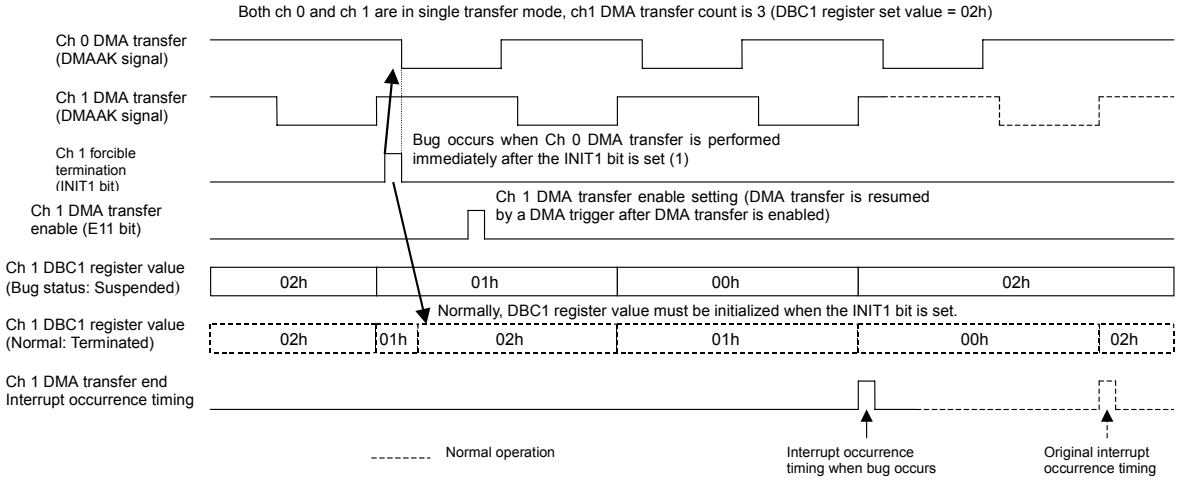
Do not perform register access in the above sequence when not performing a forcible shutdown.

Please regard this item as a permanent restriction.

No. 6 Bug related to DMA transfer forcible termination (n = 0 to 3)

[Description]

When terminating a DMA transfer by setting the INITn bit of the DCHCn register, the transfer may not be terminated, but just suspended, even though the INITn bit is set (1). As a result, when the DMA transfer of a channel that should have been terminated is resumed, the DMA transfer will terminate after an unexpected number of transfers are completed and a DMA transfer completion interrupt may occur. This bug occurs if a DMA transfer is executed immediately after a forcible termination is set (by setting the INITn bit) (see the figure below).



This bug does not depend on the number of transfer channels, transfer type (2-cycle or flyby), transfer target (between memory and memory, memory and I/O; including internal resources), transfer mode (single, single-step, or block), or trigger (external request, interrupt from internal

peripheral I/O, or software), and can occur with any combination of the above elements that can be set under the specifications. In addition, another channel may affect the occurrence of this bug.

The following registers are buffer registers with a 2-stage FIFO configuration of master and slave. If these registers are overwritten during a DMA transfer or in the DMA-suspended status, the value is written to the master register, and reflected in the slave register when the DMA transfer of the overwritten channel is terminated.
The “initialization” in the above figure means that the contents of the master register are reflected in the slave register.

2-stage FIFO configuration registers:

- DMA source address register (DSAnH, DSAnL)
- DMA destination address register (DDAnH, DDAnL)
- DMA transfer count register (DBCn)

[Workaround]

This bug can be avoided by implementing either of the following procedures using the software.

(1) Stop all the transfers from DMA channels temporarily

The following measure is effective if the following condition is satisfied.

- Except for the following workaround processing, the program does not assume that the TCn bit of the DCHCn register is 1. (Since the TCn bit of the DCHCn register is cleared (0) when it is read, execution of the following procedure (b) under <5> clears this bit.)

[Procedure to avoid bug]

<1> Disable interrupts (DI state).
<2> Read the DMA restart register (DRST) and transfer the ENn bit of each channel to a general-purpose register (value A).
<3> Write 00H to the DMA restart register (DRST) twice[Note].
By executing twice[Note], the DMA transfer is definitely stopped before proceeding to <4>.
<4> Set (1) the INITn bit of the DCHCn register of the channel that should be terminated forcibly.
<5> Perform the following operations for value A read in <2>. (Value B)
(a) Clear (0) the bit of the channel that should be terminated forcibly
(b) If the TCn and ENn bits of the channel that is not terminated forcibly are 1 (AND makes 1), clear (0) the bit of the channel.
<6> Write value B in <5> to the DRST register.
<7> Enable interrupts (EI state).

Note Execute three times if the transfer target (transfer source or transfer destination) is the internal RAM.

Remark Be sure to execute <5> to prevent the ENn bit from being set illegally for channels that are terminated normally during the period of <2> and<3>.

(2) Repeat setting the INITn bit until the forcible DMA transfer termination is correctly performed (n = 0 to 3)

[Procedure to avoid bug]

<1> Copy the initial transfer count of the channel that should be terminated forcibly to a general-purpose register.

<2> Set (1) the INITn bit of the DCHCn register of the channel that should be terminated forcibly.

<3> Read the value of the DMA transfer count register (DBCn) of the channel that should be terminated forcibly and compare the value with the one copied in <1>. If the values do not match, repeat <2> and <3>.

Remarks 1. When the DBCn register is read in procedure <3>, the remaining transfer count will be read if the DMA is stopped due to this bug. If the forcible DMA termination is performed correctly, the initial transfer count will be read.

2. Note that it may take some time for forcible termination to take effect if this workaround is implemented in an application in which DMA transfers other than for channels subject to forcible termination are frequently performed.

This bug has been corrected in control code B or later.

No. 7 Bug in program execution and DMA transfer in internal RAM

[Description]

When a DMA transfer for the internal RAM and an instruction of (1) or (2) described below are executed simultaneously, the CPU may deadlock due to conflict between the internal bus operations. At this time, only a reset can be acknowledged. (An NMI or interrupt cannot be acknowledged.)

(1) A bit manipulation instruction (SET1, CLR1, or NOT1) allocated in the internal RAM

(2) A data access instruction for a misaligned address allocated in the internal RAM

[Workaround]

Implement either of the following workarounds.

(1) For a bit manipulation instruction (SET1, CLR1, or NOT1) allocated in the internal RAM

- Do not perform a DMA transfer for the internal RAM when a bit manipulation instruction allocated in the internal RAM is being executed.
- Do not execute a bit manipulation instruction allocated in the internal RAM when a DMA transfer for the internal RAM is being performed.

This restriction has been corrected in control code B or later.

(2) For a data access instruction for a misaligned address allocated in the internal RAM

- Do not perform a DMA transfer for the internal RAM when a data access instruction for a misaligned address allocated in the internal RAM is being executed.
- Do not execute a data access instruction for a misaligned address allocated in the internal RAM when a DMA transfer for the internal RAM is being performed.

Please regard this item as a permanent restriction.

No. 8 Emulator hangs up upon internal reset

[Description]

The emulator may hang up when a reset is generated by watchdog timer 2 or the low-voltage detector (LVI).

[Workaround]

Stop watchdog timer 2 after reset.

Do not emulate the low-voltage detector (LVI).

This restriction has been corrected in control code B or later.

No. 9 Emulator hangs up while downloading data or setting software break

[Description]

The emulator may hang up if the WAIT pin or HLDRQ pin is at the active level while data is being downloaded to the internal ROM area or a software break is set to the internal ROM area.

[Workaround]

If the WAIT and HLDRQ pins are not used, mask the WAIT and HLDRQ pins using the pin mask function of the debugger.

If the WAIT and HLDRQ pins are used, do not make these pins active while data is being downloaded to the internal ROM area or a software break is set to the internal ROM area.

This restriction can be avoided by upgrading the debugger to the following version.

- When using ID850QB: Use V2.81 or later.
- When using MULTI: Use in combination with EXEC V1.57 or later.

No. 10 Data loss occurs when external RAM is connected

[Description]

When there is external RAM in the target system and the bus control pins are active, the data in the CS0 area (0x100000 to 0x1FFFFF) in the external RAM may be overwritten by downloading a program to the internal ROM area or by setting a software break in the internal ROM.

[Workaround]

Initialize the external RAM value by running the downloaded program.

In addition, do not use a software break for the internal ROM space; use a hardware break instead.

This restriction can be avoided by upgrading the debugger to the following version.

- When using ID850: Use V2.81 or later.
- When using MULTI: Use in combination with EXEC V1.57 or later.

No. 11 Restriction on emulation of POC circuit and clock monitor

[Description]

The POC circuit and the clock monitor cannot be emulated.

[Workaround]

There is no workaround.

This restriction will be corrected by upgrading the debugger and device file.

No. 12 Restriction on setting flash memory mask option

[Description]

The value set by the program is reflected in the option data area (0x007A), which is targeted by the flash memory mask option. However, the emulator operates according to the 00H setting.

[Workaround]

There is no workaround.

This restriction can be avoided by upgrading the debugger to the following version.

- When using ID850: Use V2.81 or later.
- When using MULTI: Use in combination with EXEC V1.57 or later.

No. 13 Illegal break occurs during program execution in internal RAM (1)

[Description]

An illegal break may occur when a peripheral I/O register is accessed during program execution in the internal RAM.

[Workaround]

Cancel the fail-safe break setting for the internal RAM in the debugger.

- When using ID850QB

  Click the [Detail] button in the Fail-safe Break field in the Configuration window and clear the check box for “Internal RAM”.

- When using MULTI

  Cancel the fail-safe break for “ramgrd” and “ramgrdv” using the Target flsf command.

Please regard this item as a permanent restriction.

No. 14 Restriction on reset input during a break

[Description]

The QB-V850ESFX2 may hang up if a break occurs when the RESET pin is active (low level).

[Workaround]

Mask the RESET pin using the pin mask function of the debugger.

This restriction has been corrected in control code B or later.

No. 15 Bug that occurs upon entering and releasing STOP mode when RESET pin is masked

[Description]

When the RESET pin is masked using the pin mask function of the debugger and watchdog timer 2 is used in reset mode, the CPU’s operating clock is switched to the internal oscillation clock after STOP mode is released, depending on the timing for entering and releasing STOP mode (one of (1) to (4) in the following table). After the clock is switched to the internal oscillation clock, the CPU continues the operation with the internal oscillation clock until the CPU reset button on the debugger is pressed.

No.	Operating Clock for Watchdog Timer 2	Timing at Which This Bug Occurs
(1)	Main clock	STOP mode is entered during the period from when a reset of watchdog timer 2 occurs until the reset is released[Note]
(2)	Subclock	STOP mode is entered during the period from when a reset of watchdog timer 2 occurs until the reset is released[Note]
(3)	Internal oscillation clock	STOP mode is entered during the period from when a reset of watchdog timer 2 occurs until the reset is released[Note]
(4)		The internal oscillation clock is stopped during the period from when a reset of watchdog timer 2 occurs until the reset is released[Note], and then STOP mode is entered

Note The period in which watchdog timer 2 generates a reset signal while the reset signal of watchdog timer 2 is masked as a result of masking RESET using the pin mask function of the debugger.

[Workaround]

Do not use watchdog timer 2.

To generate a reset of watchdog timer 2, do not mask the RESET pin using the pin mask function of the debugger.

Please regard this item as a permanent restriction.

No. 16 Bugs in A/D conversion function during a break

[Description]

(1) A/D conversion does not start if any one of the following conditions <a> to <c> is satisfied in peripheral break mode (mode in which peripheral functions are stopped during a break). In addition, no interrupt requests are generated upon completion of the A/D conversion.

<a> A break occurs from when an A/D conversion start trigger is generated[Note 1] until the execution of two instructions ends[Note 2].

Example: In software trigger mode

```
* set1 0x7, ADA0M0   ┐
* nop                │ A/D conversion does not start if a break occurs
* nop                │ during this period.
* nop                ┘
* nop                ┐ If a break occurs after this point, A/D conversion starts normally.
                     ┘ (Caution must still be exercised for the bugs described in (2) and (3).)
```

<b> If execution is started using an A/D conversion start instruction in software trigger mode, and a software break or a break before execution is set to the instruction.

Example:

```
B set1 0x7, ADA0M0   ← A/D conversion does not start if an attempt is made to
                       start A/D conversion using the instruction in this line.
```

<c> A break occurs while an A/D conversion operation is stopped, and an attempt is made to start A/D conversion during this break[Note 3].

(2) If a break occurs during A/D conversion in peripheral break mode, the A/D conversion result immediately after re-execution is invalid.

(3) If a break occurs[Note 2] during A/D conversion in peripheral break mode, a write is performed[Note 5] on an A/D-related register[Note 4], and the A/D conversion is re-executed, then conversion is performed once or twice with the values before the writing.

- If the break occurs in normal conversion operation mode: Once or twice
- If a break occurs during A/D conversion in high-speed conversion mode, and the ADA0CE bit is cleared and re-set during the break: Twice
- Other: Once

After this conversion is completed, A/D conversion starts with the values after the writing. Consequently, an invalid A/D conversion result is obtained and it seems as though invalid interrupts occur once or twice for the operation. (Normally, re-conversion is performed immediately after re-execution with values newly set to the A/D-related register.)

Notes 1. Starting conversion by DMA transfer, external trigger, and timer trigger are included in this condition, in addition to starting conversion triggered by instruction execution.

2. Includes the following break sources.

- Step execution
- Fail-safe break
- RAM monitoring (real-time RAM monitoring does not apply)
- DMM
- Change of event while the program is running

Among these sources, RAM monitoring, DMM, and a change of event while the program is running is implemented through an instantaneous break, so the actual break point cannot be specified, and thus the A/D conversion unexpectedly becomes invalid.

3. DMA transfer, external trigger, and timer trigger are included in this condition, in addition to a write access to the ADA0CE bit in the IO register window.

4. A/D-related registers: ADA0M0, ADA0M1, ADA0M2, ADA0S, ADA0PFT, and ADA0PFM

5. Cases such that the write setting is applied in the IO register window, or through DMA transfer.

[Workaround]

Do not set peripheral break mode if you want to avoid this bug entirely, or observe all of the following.

- Do not set breaks between the A/D conversion start trigger and the end of A/D conversion.
- Do not perform step execution of an A/D conversion start instruction in software trigger mode.
- Do not perform write accesses to A/D-related registers during a break.
- Disable the RAM monitoring function.
- Do not use DMM.
- Do not change events while the program is running.

Please regard this item as a permanent restriction.

No. 17 Illegal break occurs during program execution in internal RAM (2)

[Description]

A non-map break occurs if all of the following conditions are satisfied, even if the program itself is correct.

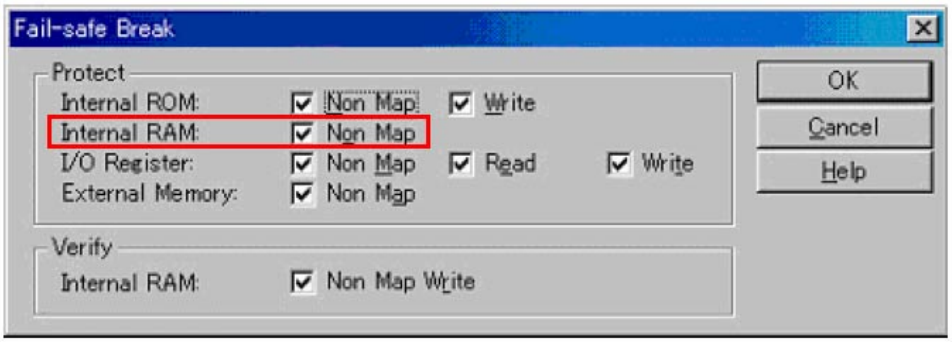
- A program is executed in the internal RAM area.
- Data access for the internal RAM area is performed twice in succession.
- An execution branches to the internal ROM area using a JR or JARL instruction immediately after the above successive data access, or one NOP instruction after the above successive data access.

[Workaround]

Implement either of the following workarounds.

- Cancel the fail-safe break setting for the internal RAM in the debugger.
  - When using ID850QB

    Click the [Detail] button in the Fail-safe Break field in the Configuration window and clear the check in the check box for “Internal RAM”.



  - When using MULTI

    Cancel the fail-safe break for “ramgrd” and “ramgrdv” using the Target flsf command.

- Insert two or more NOP instructions between the successive data access for the internal RAM area and an instruction to branch to the internal ROM area.

Please regard this item as a permanent restriction.

No. 18 Address is not retained during external bus access

[Description]

When the multiplexed bus output mode is selected for the external bus and its data bus size is 8 bits, the address is not retained after the T2 state is entered, but the low level is output.

[Workaround]

There is no workaround.

This restriction can be avoided by upgrading the device file to the following version.

- When using V850ES/FE2, FF2, FG2, or FJ2: Use DF703239 V2.11 or later.
- When using μPD703229Y or μPD70F3229Y: Use DF703229 V2.01 or later.

4. Cautions

4.1 Cautions on Extension Probe

- When using the extension probe, there is a restriction on the maximum operating frequency at which a high-speed signal such as a clock or external bus can be propagated. (See the table below.)
  In the QB-V850ESFX2, the extension probe can be used at the maximum operating frequency because the maximum operating frequency of the target device is 20 MHz.

Use of Clock Signal (CLKOUT, BUSCLK, SDCLK, etc.)	Use of External Bus	Upper Limit of Frequency When Using Extension Probe
Used	Used	32 MHz
	Not used	
Not used	Used	64 MHz
	Not used	80 MHz

- An impedance of approx. 50 Ω is applied to the extension probe.
- The signal level decreases by approx. 0.1 V when it passes through the extension probe. This degrades the precision of analog signal propagation upon A/D conversion, etc.
- A delay of approx. 5 ns (propagation delay) occurs when a signal passes through the extension probe.
  Consequently, it may be necessary to set data waits or address waits when using the external bus.
- Be sure to connect IECUBE and the target to the GND line of the extension probe when using the extension probe; otherwise the level of the propagated signal may degraded.