Microcomputer Technical Information

CP(K), O

QB-V850ESSX2 In-Circuit Emulator for V850ES/SG1, V850ES/SG2,V850ES/SJ2, V850ES/SG3, V850ES/SJ3, V850ES/JG2, V850ES/JJ2 Usage Restrictions		Document No.	ZBG-CD-06-0026	1/1
		Date issued	April 5, 2006	
		Issued by	Development Tool Group Multipurpose Microcomputer Systems Division 4th Systems Operations Unit NEC Electronics Corporation	
Related documents	QB-V850ESSX2 User's Manual: U17091EJ2	Notification classification	√	Usage restriction
				Upgrade
				Document modification
				Other notification

1. Affected product

Product	Control Code[Note]	Remark
QB-V850ESSX2-xxx-yyy	A, B, C, D, E	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 item

Special notes on emulating V850ES/SG3 and V850ES/SJ3 have been added. See the attachment for details.

3. List of restrictions

See the attachment for details.

4. Revision history

QB-V850ESSX2, In-Circuit Emulator for V850ES/SG1, V850ES/SG2,V850ES/SJ2, V850ES/SG3, V850ES/SJ3, V850ES/JG2, V850ES/JJ2 - Usage Restrictions

Document Number	Issued on	Description
ZBG-CD-04-0012	June 8, 2004	Newly created.
ZBG-CD-05-0102	October 28, 2005	Correction of description in No. 11 Addition of support for optional functions (No. 22) Addition of bugs (No. 19 to No. 21, No. 23 and No. 24) Addition of **4. Supported Devices**
ZBG-CD-06-0026	April 5, 2006	Addition of **5. Special Notes on Emulating V850ES/SG3 and V850ES/SJ3**

Notes on Using QB-V850ESSX2

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.

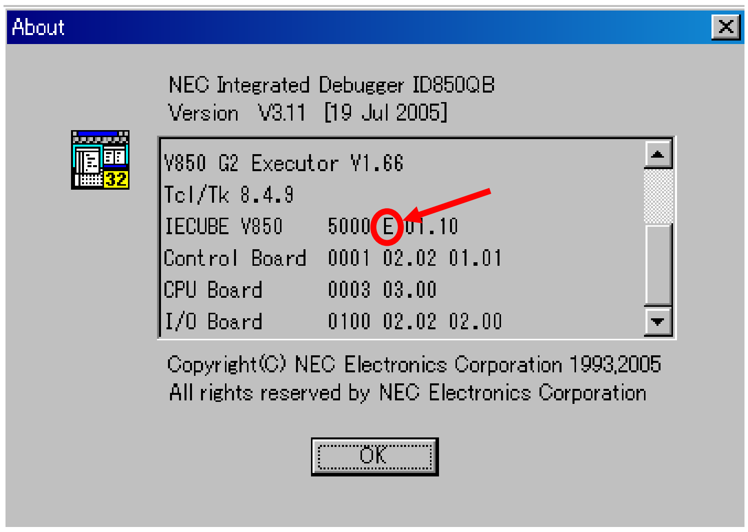
- User's manual of target device
- Restrictions notification document for target device

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	–
D	–
E	–

Note The "control code" is the second digit from the left in the 10-digit serial number (if the product has not been upgraded).
If the product has been upgraded, the control code can be checked by selecting [About] from the [Help] menu while the debugger ID850QB is running.
"X" in "IECUBE V850 **** X **.**" is the control code.

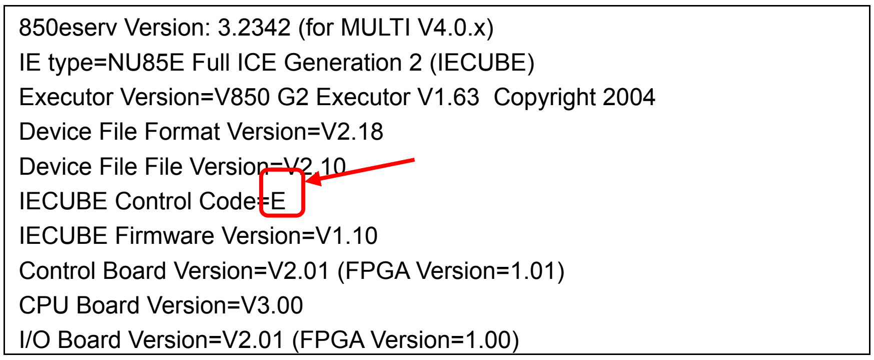


NEC Integrated Debugger ID850QB
Version V3.11 [19 Jul 2005]

```
V850 G2 Executor V1.66
Tcl/Tk 8.4.9
IECUBE V850     5000 E 01.10
Control Board   0001 02.02 01.01
CPU Board       0003 03.00
I/O Board       0100 02.02 02.00
```

Copyright(C) NEC Electronics Corporation 1993,2005
All rights reserved by NEC Electronics Corporation

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=E
IECUBE Firmware Version=V1.10
Control Board Version=V2.01 (FPGA Version=1.01)
CPU Board Version=V3.00
I/O Board Version=V2.01 (FPGA Version=1.00)
```

MULTI is a registered trademark of Green Hills Software, Inc in the United States.

2. Product History

No.	Bugs and Changes/Additions to Specifications		Control Code				
			A	B	C	D	E
1	Bug in watchdog timer during break		Permanent restriction				
2	Bug in 16-bit timer M during break		Permanent restriction				
3	Bug in accessing UAnRX register during break		Permanent restriction				
4	Bug in accessing DR register during break		Permanent restriction				
5	Bug in accessing CBnRX register during		Permanent restriction				
6	Bug in accessing C0RGPT register during break		Permanent restriction				
7	Bug in accessing C0TGPT register during break		Permanent restriction				
8	Bug in accessing C0GNCTRL register during break		Permanent restriction				
9	Restriction on operating frequency		×	×	×	√	√
10	Bug related to DMA transfer forcible termination		×	×	×	√	√
11	Bug in program execution and DMA transfer in internal RAM	(1) Bit manipulation instruction	×	×	×	√	√
		(2) Access for misaligned address	Permanent restriction				
12	Emulator hangs up upon internal reset		×	×	×	√	√
13	Emulator hangs up while downloading data or setting software break		Avoidable by debugger upgrade				
14	Restriction on CLKOUT pin status in standby mode^Note		×	√	√	√	√
15	Restriction on internal ROM misfetch		×	√	√	√	√
16	Data loss occurs when external RAM is connected		Avoidable by debugger upgrade				
17	Illegal break occurs during program execution in internal RAM (1)		Permanent restriction				
18	Restriction on reset input during a break		×	×	×	√	√
19	Bug that occurs upon entering and releasing STOP mode when RESET pin is masked		Permanent restriction				
20	Bugs in A/D conversion function during a break		Permanent restriction				
21	Changes in clock specifications		–	–	–	–	√
22	Support of optional functions		–	–	–	–	√
23	Illegal break occurs during program execution in internal RAM (2)		Permanent restriction				
24	Address is not retained during external bus access		Avoidable by device file upgrade				

×: Applicable, √: Not applicable or already corrected, –: Specification not supported

Note This restriction can be avoided in control code D or later if it is used in combination with the following debugger.

- ID850QB V2.81 or later
- MULTI with EXEC V1.57 or later

3. Details of Bugs and Added Specifications

No. 1 Bug in watchdog timer during break

[Description]

When both the following conditions (a) and (b) are satisfied and a break occurs, the watchdog timer does not stop and a reset or a non-maskable interrupt occurs. If a reset occurs, the debugger hangs up.

(a) The main clock or subclock is selected as the source clock for the watchdog timer

(b) The internal oscillator is stopped (RSTOP flag =1)

[Workaround]

Implement (a) or (b) below.

(a) Use the internal oscillation clock as the source clock.

(b) Do not stop internal oscillator.

Please regard this item as a permanent restriction.

No. 2 Bug in 16-bit timer M during break

[Description]

When a break occurs while the following both (a) and (b) are satisfied, timer M does not stop even if the Peripheral Break function has been set to “Break”.

(a) INTWT, internal oscillation clock ($f_R/8$), or subclock is selected as the source clock of timer M.

(b) The main clock is stopped by setting the MCK flag.

[Workaround]

Implement (a) or (b) below to stop timer M during a break using the Peripheral Break function.

(a) Use the main clock (fxx, fxx/2, fxx/4, fxx/64, or fxx/512) as the source clock.

(b) Do not stop main clock oscillation.

Please regard this item as a permanent restriction.

No. 3 Bug in accessing UAnRX register during break

[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 or DMM function. However, 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. 4 Bug in accessing DR register during break

[Description]

An overrun error occurs under the following conditions (a) and (b).

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

(b) If a DMA transfer from the IEBus data register (DR) is performed during a break[Note], an overrun error occurs when IEBus reception is performed next time.

Note Including breaks by the RAM monitor function or DMM function. However, 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) Set a hardware break when setting a break immediately after reading the DR register.

(b) There is no workaround.

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

No. 5 Bug in accessing CBnRX register during break

[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 or DMM function. However, 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. 6 Bug in accessing C0RGPT register during break

[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 CAN0 module receive history list register (C0RGPT).

(b) If a DMA transfer from the CAN0 module receive history list register (C0RGPT) is performed during a breakNote.

Note Including breaks by the RAM monitor function or DMM function. However, 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 C0RGPT register.

(b) There is no workaround.

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

No. 7 Bug in accessing C0TGPT register during break

[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 CAN0 module transmit history list register (C0TGPT).

(b) If a DMA transfer from the CAN0 module transmit history list register (C0TGPT) is performed during a breakNote.

Note Including breaks by the RAM monitor function or DMM function. However, 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 C0TGPT register.

(b) There is no workaround.

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

No. 8 Bug in accessing C0GNCTRL register during break

[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 CAN0 module control register (C0GMCTRL) is set.

(2) The I/O registerNote is accessed.

(3) The GOM bit of the CAN0 module control register (C0GMCTRL) is cleared.

Note I/O register access except for clearing the GOM bit of the C0GMCTRL 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. 9 Restriction on operating frequency

[Description]

The microcontrollers with “H” in their part numbers cannot be emulated. Consequently, the maximum operating frequency is 20 MHz.

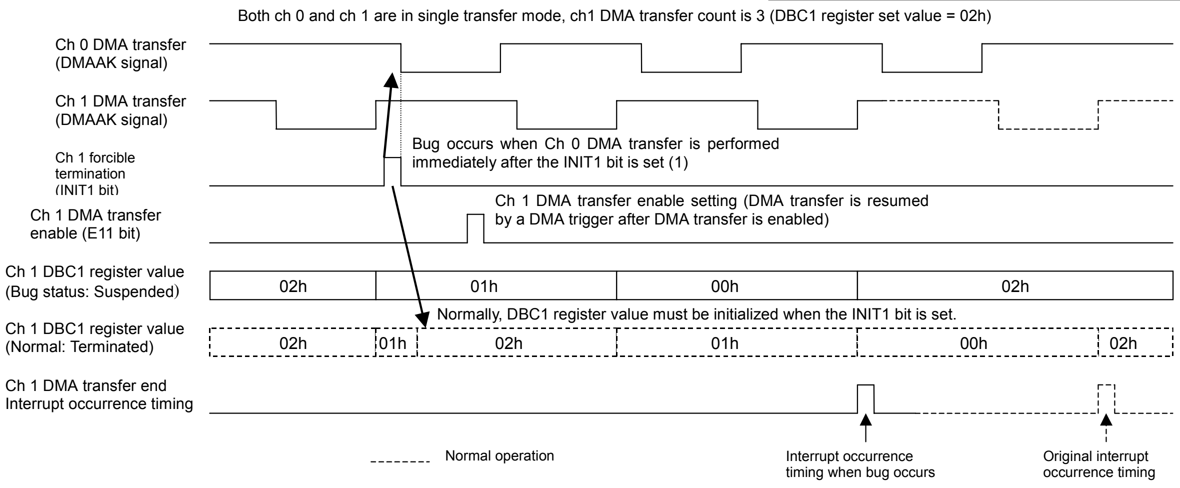
[Workaround]

There is no workaround. Use the emulator at a frequency of 20 MHz or lower.
This restriction has been corrected in control code D or later. Consequently, the maximum operating frequency is 32 MHz.

No. 10 Bug related to DMA transfer forcible termination

[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 (n = 0 to 3). 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.


Both ch 0 and ch 1 are in single transfer mode, ch1 DMA transfer count is 3 (DBC1 register set value = 02h)
Ch 0 DMA transfer (DMAAK signal)
Ch 1 DMA transfer (DMAAK signal)
Ch 1 forcible termination (INIT1 bit)
Bug occurs when Ch 0 DMA transfer is performed immediately after the INIT1 bit is set (1)
Ch 1 DMA transfer enable (E11 bit)
Ch 1 DMA transfer enable setting (DMA transfer is resumed by a DMA trigger after DMA transfer is enabled)
Ch 1 DBC1 register value (Bug status: Suspended)
02h
01h
00h
02h
Normally, DBC1 register value must be initialized when the INIT1 bit is set.
Ch 1 DBC1 register value (Normal: Terminated)
02h
01h
02h
01h
00h
02h
Ch 1 DMA transfer end Interrupt occurrence timing
Normal operation
Interrupt occurrence timing when bug occurs
Original interrupt occurrence timing

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 (n = 0 to 3):

- 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 one 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$^{\text{Note}}$.
By executing twice$^{\text{Note}}$, 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).

Remarks **1.** 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. n = 0 to 3

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

(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 D or later.

No. 11 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 D 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. 12 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]

Workarounds for watchdog timer 2:

- Stop watchdog timer 2 after reset.
- Mask the RESET pin using the pin mask function of the debugger.

Workaround for low-voltage detector (LVI):

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

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

No. 13 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. 14 Restriction on CLKOUT pin status in standby mode

[Description]

Normally a low level is output in standby mode (software STOP or IDLE), but the operating clock is output as is if the mode is shifted from CLKOUT output mode to standby mode. This restriction is not applicable to the operation in HALT mode.

[Workaround]

If outputting CLKOUT in standby mode may cause problems, be sure to set the CLKOUT pin so that it outputs a low level before the mode is shifted from CLKOUT output mode to standby mode using the following procedure.

(1) Clear bit 1 of the PMCCM register to 0 (switches to I/O port)

(2) Clear bit 1 of the PMCM register to 0 (sets output port)

(3) Clear bit 1 of the PCM register to 0 (low-level output)

When this procedure is implemented, set bit 1 of the PMCCM register to 1 immediately after standby release (switching to CLKOUT output mode) to output CLKOUT.

This restriction has been corrected in control code B or later. However, this restriction can be avoided in control code D or later if the emulator is used in combination with the following debugger.

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

No. 15 Restriction on internal ROM misfetch

[Description]

A misfetch may occur during program execution in the internal ROM (an unexpected instruction is executed).

[Workaround]

There is no workaround.

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

No. 16 Data loss occurs when external RAM is connected

[Description]

A write cycle to the external bus is generated when downloading data to the internal ROM is executed or a software break is set to the internal ROM. Therefore, if RAM is connected to the target system, data in the RAM may be lost.

[Workaround]

There is no workaround if the bug occurs as a result of downloading data to the internal ROM. However, it does not cause any problem if the internal RAM values are initialized by program execution after download (all the values in the RAM are overwritten), because the lost data is overwritten by normal values.

If the bug occurs as a result of setting a software break to the internal ROM, 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 ID850QB: Use V2.81 or later.
- When using MULTI: Use in combination with EXEC V1.57 or later.

No. 17 Illegal break occurs during program execution in internal RAM (1)

[Description]

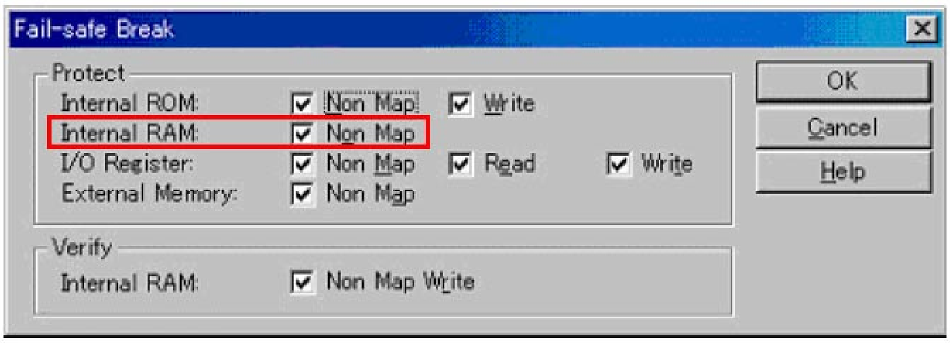
An illegal break may occur when a peripheral I/O register is accessed during program execution in the internal RAM.

[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”.



Fail-safe Break

Protect
- Internal ROM: ☑ Non Map ☑ Write
- Internal RAM: ☑ Non Map
- I/O Register: ☑ Non Map ☑ Read ☑ Write
- External Memory: ☑ Non Map

Verify
- Internal RAM: ☑ Non Map Write

- 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. 18 Restriction on reset input during a break

[Description]

The QB-V850ESSX2 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 D or later.

No. 19 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 reset signal for watchdog timer 2 is held for the period “$2^7 \times$ watchdog timer input clock” after a reset of watchdog timer 2 occurs.

[Workaround]

Implement either of the following workarounds.

- To prevent a reset of watchdog timer 2 from occurring, stop watchdog timer 2 by using software before the reset occurs.
- 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. 20 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               } 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[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, A/D conversion may be performed twice with the values before the writing.) 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.)

(3) If a break occurs during A/D conversion in peripheral break mode, the A/D conversion result immediately after re-execution is invalid. Moreover, 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, then the result of the subsequent one A/D conversion operation is invalid.

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. 21 Changes in clock specifications

[Description]

The oscillator can be connected to the emulator in products with control code E or later.

In products with control code E or later, a 4 MHz oscillator is mounted at shipment and a 5 MHz oscillator is supplied with the emulator.

For details on the clock settings, refer to the QB-V850ESSX2 User's Manual (U17091E) (2nd edition and later).

No. 22 Support of optional functions

[Description]

The following optional functions can be added for a fee to products with control code E or later.

- Memory emulation function
- Coverage measurement function
- TimeMachine^{TM} function

For details, refer to the QB-V850ESSX2 User's Manual (U17091E) (2nd edition and later).

For an application to order the optional functions, consult an NEC Electronics sales representative or distributor.

No. 23 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.

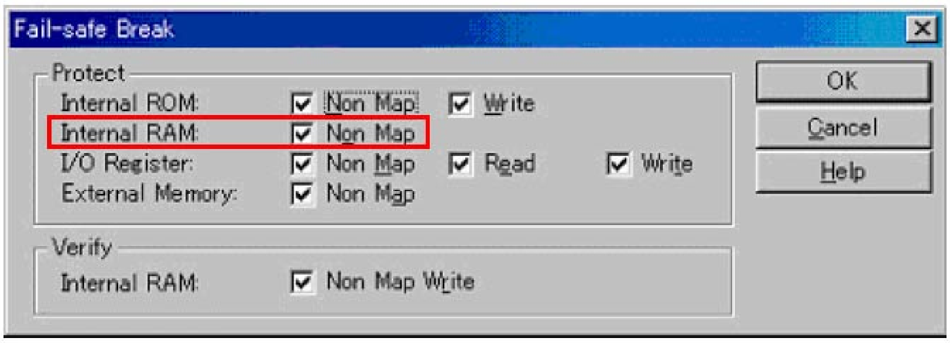
- A program is executed in the internal RAM area.
- Data access for the internal RAM area is performed twice in succession.
- An execution branches to the internal ROM area using a JR or JARL instruction immediately after the above successive data access, or one NOP instruction after the above successive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


Fail-safe Break
Protect
Internal ROM: ✓ Non Map ✓ Write
Internal RAM: ✓ Non Map
I/O Register: ✓ Non Map ✓ Read ✓ Write
External Memory: ✓ Non Map
Verify
Internal RAM: ✓ Non Map Write
OK
Cancel
Help

  - 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.

TimeMachine is a trademark of Green Hills Software, Inc.

No. 24 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/SG2: Use DF703283 V2.11 or later.
- When using V850ES/SJ2: Use DF703288 V2.11 or later.

4. Supported Devices

The QB-V850ESSX2 can be used for emulation of the following devices.

- V850ES/SG1
- V850ES/SG2
- V850ES/SJ2
- V850ES/SG3
- V850ES/SJ3
- V850ES/JG2
- V850ES/JJ2

5. Special Notes on Emulating V850ES/SG3 and V850ES/SJ3

The QB-V850ESSX2 uses the V850ES/SJ2 as the emulation chip.
When performing emulation with the V850ES/SG3 or V850ES/SJ3, therefore, note the following differences in specifications.

(1) Rate of sampling time during conversion by A/D converter is in progress
(2) Generation factor of low-voltage detection interrupt (INTLVI)
(3) Output frequency of internal oscillator
(4) Output resistance of D/A converter

No.	Differences	Emulator	V850ES/SG3, V850ES/SJ3
1	Rate of sampling time during conversion by A/D converter is in progress	4/26 clocks	8/26 clocks
2	Generation factor of low-voltage detection interrupt (INTLVI)	When the power supply voltage drops to lower than the detection voltage	When the power supply voltage drops/rises to lower/higher than the detection voltage
3	Output frequency of internal oscillator	200 kHz (TYP.)	220 kHz (TYP.)
4	Output resistance of D/A converter	3.5 kΩ	6.42 kΩ

6. Cautions

6.1 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-V850ESSX2, the extension probe can be used at the maximum operating frequency because the maximum operating frequency of the target device is 32 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.