NEC

Customer Notification

V850™ Series

CPDW9X/NT-CDR-V85X

Operating Precautions

850ESERV2 Target Server

IECUBE
QB-V850MINI-EE
QB-MINI2

Document No. U19303EE1V0IF00
Date Published: April 2009

DISCLAIMER

The related documents in this customer notification may include preliminary versions. However, preliminary versions may not have been marked as such.

The information in this customer notification is current as of its date of publication. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC's data sheets or data books, etc., for the most up-to-date specifications of NEC PRODUCT(S). Not all PRODUCT(S) and/or types are available in every country. Please check with an NEC sales representative for availability and additional information.

No part of this customer notification may be copied or reproduced in any form or by any means without prior written consent of NEC. NEC assumes no responsibility for any errors that may appear in this customer notification. NEC does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from the use of NEC PRODUCT(S) listed in this customer notification or any other liability arising from the use of such PRODUCT(S).

No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC or others. Descriptions of circuits, software and other related information in this customer notification are provided for illustrative purposes of PRODUCT(S) operation and/or application examples only. The incorporation of these circuits, software and information in the design of customer's equipment shall be done under the full responsibility of customer. NEC assumes no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.

While wherever feasible, NEC endeavors to enhance the quality, reliability and safe operation of PRODUCT(S) the customer agree and acknowledge that the possibility of defects and/or erroneous thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects and/or errors in PRODUCT(S) the customer must incorporate sufficient safety measures in their design, such as redundancy, fire-containment and anti-failure features.

The customer agrees to indemnify NEC against and hold NEC harmless from any and all consequences of any and all claims, suits, actions or demands asserted against NEC made by a third party for damages caused by one or more of the items listed in the enclosed table of content of this customer notification for PRODUCT(S) supplied after the date of publication.

Applicable Law:

The law of the Federal Republic of Germany applies to all information provided by NEC to the Customer under this Operating Precaution document without the possibility of recourse to the Conflicts Law or the law of 5th July 1989 relating to the UN Convention on Contracts for the International Sale of Goods (the Vienna CISG agreement).

Düsseldorf is the court of jurisdiction for all legal disputes arising directly or indirectly from this information. NEC is also entitled to make a claim against the Customer at his general court of jurisdiction.

If the supplied goods/information are subject to German, European and/or North American export controls, the Customer shall comply with the relevant export control regulations in the event that the goods are exported and/or re-exported. If deliveries are exported without payment of duty at the request of the Customer, the Customer accepts liability for any subsequent customs administration claims with respect to NEC.

Notes: (1) "**NEC**" as used in this statement means NEC Electronics Corporation and also includes its direct or indirect owned or controlled subsidiaries.

(2) "**PRODUCT(S)**" means 'NEC semiconductor products' *(NEC semiconductor products* means any semiconductor product developed or manufactured by or for NEC) and/or 'TOOLS' *(TOOLS* means 'hardware and/or software development tools' for NEC semiconductor products' developed, manufactured and supplied by 'NEC' and/or 'hardware and/or software development tools' supplied by NEC but developed and/or manufactured by independent 3rd Party vendors worldwide as their own product or on contract from NEC)

Table of Contents

Introduction

This document describes both, limitations and additional features available in the 850eserv2 target server used together with Green Hills Multi debugger and the NEC emulation hardware, as ther is

- IECUBE,
- MINICUBE and
- MINICUBE2

The listed features are sorted in tables and have unique numbers organized like this:

Tool	Number Prefix	Comment
IECUBE	IE<n>	Generic issues of target server and IECUBE
	T<n>	Specific issues of the 850eserv2 trace in the IECUBE
	FS<n>	Issues about Flash Self Programming emulation and data flash programming in IECUBE
Minicube	OC<n>	Issues concerning the Minicube Debugger (**O**n-**C**hip-**D**ebug Interface)
Minicube-2	M<n>	Issue concerning the Minicbube-2 Debugger

1. Operating Precautions for IECUBE

<table>
<tr><th>No</th><th>Description</th></tr>
<tr><td>IC1</td><td>The limitation of the conflict with software break and break
[Limitation] When Multi stopped by break(compulsion break, after hardware break, fail-safe break, etc) near the point of software break, if MULTI restarts after this break, MULTI may execute the illegal instruction.
When the following two conditions are satisfied at the same time, MULTI misrecognizes a stop of execution occurred by a software break even if it occurred by a compulsion break.
In this case, MULTI will set an incorrect value which is 2 bytes smaller than a correct value to the PC at the restart.
(1) A program execution is stopped at a branch destination instruction.
(2) A 2byte instruction is placed on immediately before of a branch destination instruction.

[Example] As the following program, if MULTI broken on 0x1100, in the specification of hard ware, MULTI doesn't know a break reason. MULTI assumes that a program execution has been stopped by a software break point, and it sets 0x10FE to the PC. Therefore, in case of (2), MULTI will stop at wrong address and it will set wrong value to the PC.
(1) Execute the instruction on 0x10FE and break
→ Displays the break PC value to 0x10FE.
(2) Execute the branch instruction on 0x100C and break
→ MULTI assumes that a break reason is same as the case(1). The PC value is set as 0x10fe.
0x1008 st.b r20,PM0
0x100C jr 0x1100 ← branch to 0x1100
:
:
0x10FE bz 0x1514 ← Set software break point
0x1100 mov r1,r2 ← Branch destination from 0x100C
0x1102 nop
[Workaround] Please use the hardware break instead of the software break.</td></tr>
<tr><td>IC2</td><td>Illegal break during a program execution on internal RAM
[Limitation] If you access to the peripheral I/O register during program execution on internal RAM,
unexpected break may occur.
[Workaround] Be sure to disable the guard break function for internal RAM before executing a program.
[Example] flsf ramgrd
(As this command set, the guard break function becomes disabled)
flsf ramgrdv
(As this command set, the guard break function becomes disabled)</td></tr>
<tr><td>IC3</td><td>Reset by watchdog timer
[Limitation] Sometimes the debugger may freeze when watchdog timer reset occurred.
[Workaround] Do not generate the watchdog timer reset like below.
• Mask the reset pin by "pinmask" command
• Stop the watchdog timer.
• Do not use the watch dog timer as the reset mode.
• Do not generate the watch dog timer reset by clearing the watch timer before it becomes full.
This limitation does not happen by upgrade of hardware. (*1)</td></tr>
</table>

No	Description
IC4	Combination with Microsoft Windows [Limitation] Please do not use Microsoft Windows's Standby feature, Pause feature, and Suspend feature after 850eserv2 invocation. Please do not use "User change" feature. [Workaround] None
IC5	Hardware limitation for timer [Limitation] If timer condition is set, the accumulate number of the "tmevent" command becomes 0 when the count value exceed 8589934591 (0x1FFFFFFFF). Timer can not count until 8589934591(0x1FFFFFFFF), MULTI does not display the overflow mark of TotalTime. [Workaround] None
IC6	When a break occurs while target RESET is active [Limitation] When a break occurs while target RESET is active, the debugger freezes depending on the timing. [Workaround] Mask RESET pin by "pinmask" command so that RESET does not become active. This limitation does not happen by upgrade of hardware. (*1)
IC7	Limitation for real-timeRAM [Limitation] When you use real-time RAM function under the following two conditions, real-time RAM memory is correctly displayed. • When you set the address where 256Bytes is consecutive for real-time RAM • When you mis-aline access for 256Bytes Boundary [Workaround] None

*1 The Hardware of the following control code number or later number does not be happened the limitation.

QB-V850ES/FX2	Control code B
QB-V850E/IA4	Control code B
QB-V850ES/KX1H	Control code B
QB-V850ES/RS1	Control code C
QB-V850ES/SX2	Control code D

No	Description
IC8	At the 850eserv2 invocation, before downloading program, please set CPU clock frequency by dclock command. If the dclock command is not set correctly, 850eserv2 cannot access the IECUBE correctly. Please specify either main clock or sub clock by ”dclock” command. Refer to the 850eserv2 Users Manual for details. [Example] CPU clock frequency is 5MHz, sub clock is 32768Hz dclock 5000 32768 swoff 1 2 3 1. Set the main clock value (Input decimal value/ KHz). 2. Set the sub clock value (Input decimal value/ Hz) 3. The switch of whether to use a sub clock swoff : unused sub clock swon : use sub clock
IC9	When setting the option byte, RESET operation is needed. Setting of the option byte becomes effective by RESET operation.(When you set the option byte by the program, please RESET operation after download.)
IC10	Time measurement function and set the pass count by Link event precautions If you measure time and if MULTI internally set the break point, you cannot get the correct time data. If you set passcount by “link” command and if MULTI internally set the break point, the pass count is cleared, so MULTI cannot break on your setting point. The reason is because data is cleared as MULTI stops by internally break point. In the case that the correct data cannot get, please restrain the software break point by MULTI internally by the following way. The workaround and the point that MULTI internally set the software break point. • Start of main (MULTIV3.5 is relevant and MULTIV4.0 is not relevant.) [Workaround] When MULTI is invoked, please set “-noentbp“ option. • Start of .syscall section [Workaround] Before downloading, please set “off” at ‘syscalls’ command by 850eserv2. • Step execution on C source [Workaround] Please set ”FASTSTEP=0“ on MULTI command pane. Regarding -noentbp,FASTSTEP, please refer to the manual of MULTI. Regarding syscalls command, please refer to the manual of 850eserv2.
IC11	Timer measurement function precautions2 If you select the timeout break of section time count (set the ‘ct’ option of “tmevent” command), the debugger is stop at set time count, but it does not stop count until break status. Because the timecount valuecounted with 50MHz(20ns), the error of count is bigger when you select lower CPU clock, because it takes much times until the debugger become break status completely. If you set high speed cpu clock, the difference of the count of setting and the count of timeout break becomes relatively small.
IC12	RealtimeRAM related precautions If you display some the RealtimeRAM by using the “rem” command or the “rmemview” command, the response of a RealtimeRAM display becomes bad by setting too short interval. If you will display some the RealtimeRAM, please set the interval over 1 second (1000 mill second). [Example] Case of setting the interval as 1 second (1000 mill second) rmem t=0x1 rmemview t=0x1000
IC13	The address 0x7A is a system reserved area for V850ES/SG2,V850ES/SJ2 devices. Please set the 0x7A as the value (0) which is written in device users manual.

No	Description
IC14	Fail Safe break related precautions If program fetch executed from unusing area by program, usually NoMap Break is happened by a fail safe function. Exception is 16 Bytes from top of that unusing area, so Non Map Break does not be happened there. [Example] Case of Internal ROM as 256K(Internal ROM until 0x3ffff) Bytes on V850ES/SG2 At the 16Bytes area (0x40000-0x4000F) a fail safe function is not effective. If you use a fail safe function, please do not write program at the very end of the last memory.
IC15	The time measurement related precaution If the filling value of resolution is measured, the result of measurement is not displayed. [Example] When less than 81920nsec is measured by using 4K rates, the measurement result is not displayed. Case of using Run-Break, the result of Pass=1 and Total=0 is displayed.
IC16	Event related precautions (1) The event condition is not flushed after download again. (2) The access event break is delayed (the specified address is passed before execution stops).
IC17	The ROM correction feature cannot be emulated.
IC18	The cache memory cannot be emulated.
IC19	The device for BootSwap function related precautions When using IECUBE, a BootSwap function cannot be used even if by the device supports the BootSwap function.
IC20	When a hardware or software breakpoint has been set for the HALT instruction or an instruction that makes the debugger enter STOP mode, if the program is executed from that instruction, the relevant mode will be entered briefly but immediately released, and execution will resume from the next instruction.
IC21	(1) The ECR register cannot be modified by register window. (2) In case of V850E1 core, DBPC,DBPSW,DIR,BPC0,BPC1,BPAV0,BPAV1,BPAM0,BPAM1 registers cannot be modified by register window. (3) In case of V850ES core, DBPC,DBPSW registers cannot be modified by register window.
IC22	If you set software break point at internal RAM or instruction RAM, please do not rewrite the code to which the break point is set with the user program. If you set the break point to this point, the break function does not work. In case of the system that rewrites the code by the user program, please set the software break point, after you write by the user program.
IC23	If you use I/O registers that need a operation to PRCMD as preparation, the I/O registers are not written correctly when you do the following operation. (1) Step a instruction which write PRCMD. (2) Stop at a instruction which write I/O registers, then execute from it.
IC24	When two instructions are executed simultaneously Example 1. When a breakpoint is not set Address A: MOV r1, r2 Address A+2: XOR r1, r2 Two instructions are stepped from Address A, where one instruction should be stepped. Example 2. When a breakpoint is set at address A+2 Address A: MOV r1, r2 [B] Address A+2: XOR r1, r2 A break does not occur at address A+2 if instructions are “executed continuously” from address A. Example 3. When a hardware break is set at address A+2 before execution of instructions Address A: MOV r1, r2 [B] Address A+2: XOR r1, r2 The instruction at Address A is executed and break occurs if instructions are “executed in steps” from address A

No	Description
IC25	The PC indicates the address after halt if a break occurs in the HALT status.
IC26	Before performing real-time execution, step execution is first performed on an instruction located at the PC. This causes an error in the time measurement result in the timer. In addition, when the program operation is checked using the oscilloscope or logic analyzer, the measured timing may differ between when Go is executed at a certain location and Go is executed one instruction before that location.
IC27	CPU or peripheral I/O cannot be reset as the following case. • During a break (Because the reset pin is masked during a break)
IC28	When you use Exec and 850eserv2, MULTI displays the following warning message and MULTI can not connect. EXEC library Vx.xx is too old. Please use EXEC Vx.xx or later. Please use the newest Exec version.

No	Description
T1	Hardware limitation for trace time tag (1) Limitation for external measurement clock [Limitation] If the external measurement clock is used for trace time tag (set the 'tt=ae' option of "tmode" command), the time tag may be wrong depending on the timing. [Workaround] Please use the CPU clock. If the CPU clock is used for trace time tag, the time tag does not become wrong. Otherwise, if you use the external measurement clock, you can reduce the chances of the getting wrong time tag by using low division rate. (2) Limitation for section trace [Limitation] If the section trace is used (set the 's' or 'e' options of "trace" command), a time tag may notcontinue. [Workaround] None. Please do not use the section trace function.

No	Description
T2	Trace data related precautions (1) Disassemble list is not displayed at the user program running and the trace compensate mode is off. (cp= off at ”tmode” command.) (2) Read access frame and write access frame may be displayed as reverse order. (3) Invalid frame may be displayed for Step command execution. If turn on the trace compensate mode, the frame may not be displayed. (4) The RETI instruction at the address 0x7c is not displayed. (5) The trace stops by hitting software break point, the software break point instruction is displayed even the instruction has not executed. (6) The same two flames may be displayed for several instructions if the tm option mode is bdp (BranchPC + DataAddress + Data + DataAccessPC) at the “tmode” command. Example : prepare, dispose, callt If the trace compensate mode is on, you can reduce the chances of displaying of this frame. Or please use the other mode which is not the combination of B (BranchPC) and P (DataAccessPC). (7) The Status of trace data by using “Frame” mode sfm_S : It means the starting frame of Section Trace or Command Qualify Trace sfm_E : It means the Ending frame of Section Trace or Command Qualify Trace over : It means the time tag is overflow The time tag after this status does not means the time from a trace start, but the time from overflow. (8) The trace data of FullStop and FullBreak Multi set software break point on the start of main function by the default. If you select FullStop and FullBreak (Set m=f(fullstop), m=b(fullbreak) at the “tmode” command), the trace data acquired from break point on the start of main function. Timer measurement function precautions When there is the break point which MULTI set internally between timer measurement section, timer is cleared because MULTI internally breaks. Case of ‘RUN-BREAK’, the debugger displays timer from the internal break point. Case of ‘TIMER EVENT’, the debugger does not display timer because the debugger breaks before end event. The internal break point by setting MULTI is two cases. • The head of Main function • The head of .syscall section If you want to get the expected time measurement result, please do not set the internal break point by using the following method. • The method that MULTI does not set the internal break point on the head of Main function When you start MULTI, please add “-noentbp” option. >multi –noentbp sample.out • The method that MULTI does not set the internal break point on the head of syscall section Please turn off the mode which the break point sets on .syscall section by setting ‘syscalls’ command as off.

No	Description
T3	The workaround and the point that MULTI internally set the software break point. • Start of main(MULTI V3.5 is relevant and MULTI V4.0 is not relevant.) [Workaround] When MULTI connected, please set “-noentbp“ option. • Start of .syscall section [Workaround] Before downloading, please set “off” at sysycalls command by 850eserv2. • Step execution on C source [Workaround] Please set ”FASTSTEP=0“ on MULTI command pane. Regarding -noentbp,FASTSTEP, please refer to the manual of MULTI. Regarding syscalls command, please refer to the manual of 850eserv2.
T4	The trace data of FullBreak If you select FullBreak (Set m=b(fullbreak) at the “tmode” command), as the hardware specification, a programe breaks before the trace buffer becomes full.
T5	Command Qualify Trace If you want to get the Command Qualify Trace, please set the trace compensate mode is on. (cp=on at “tmode” command.) As hardware specification, If there are not the branch instruction in command qualify area, the trace data is uncollectible
T6	Time tag of trace data related precautions (1) The time tag of trace data The time tag of IECUBE causes the error of a maximum of 32 clock(s), in the environment of 50MHz of external measurement clock, in the environment of the value of frequency by setting the “timebase” command of cpu clock. Furthermore, if you set the hardware break point or software break point when a program is executing, the time tag causes the error, by stopping cpu momentary. (2) The time tag by using cpu clock If you get the time tag by using the cpu clock with default setting of the "timebase" command(r=x1), the time tag value may be shown as 0x0 because a time tag is counted up by 4 cpu clocks. If you want to know the time tag more accurate, please use ‘r=xq’ option of the "timebase" command.
T7	Tips to prevent trace data drop Default setting is Real-time mode (tc=r) and perform branch address trace and data access trace in all trace modes (tm=bd).In the case of this setting, trace data drop may happen. To prevent the drop, there are two ways. Please refer “tmode” command in the user’s manual. ● Select trace control mode as the Complete trace mode (tc=f) in the “tmode” command. The mode automatically stops CPU during the trace before the trace drop happen. Therefore the execution speed may become slower than the Real-time mode. ● Change the trace information collection method By restrict ing trace information than the default setting, the trace data drop probability may be reduced. ● To focus on address information tm=b(BranchPC) Perform branch address trace tm=a(BranchPC+DataAccessPC+InstructionPC) Perform PC trace ● To focus on data information tm=d(DataAddress+Data) Perform access data trace tm=dp(DataAddress+Data+DataAccessPC) Perform access data trace and access PC To get more information than the default, please use below two options. tm=bdp(BranchPC+DataAddress+Data+DataAccessPC) Perform branch address trace and access data trace tm=ad(BranchPC+DataAccessPC+InstructionPC+DataAddress+Data) Perform access data trace and PC trace Note: The ‘PC trace’ means branch address, access data address and some instruction address. It does not mean trace all PCs.

2. Special Notes On Flash Memory Self-Programming Emulation

The following lists the devices that support Flash Memory Self-Programming emulation function, as of March 2009

Table1 Flash Memory Self-Programming Emulation Supported Device

Flash Memory Process	Device
Type01	μPD70F3229Y, V850E/RS1, V850E/IA3, V850E/IA4, V850ES/Sx2, V850ES/Fx2, V850ES/Hx2, V850ES/Jx2, V850ES/IK1, V850ES/IE2
Type03	V850ES/Kx1(Only for microcontrollers with on-chip single-power-supply flash memory), V850ES/Kx1+, V850ES/Kx2
Type04	V850ES/Hx3, V850ES/Sx3, V850ES/Fx3, V850ES/Fx3-L, V850ES/Jx3, V850ES/Jx3-L, V850ES/Jx3-H, V850ES/Jx3-U

List of Availability of Emulation for Flash Function

The following table lists whether or not each Flash Function can be emulated, and restrictions when performing Flash Memory Self-Programming emulation function with the MULTI.
(Emulated: Can be emulated, Restriction: Can be emulated with some restrictions, Not emulated: Cannot be emulated)

Table2 List of Availability of Emulation for Flash Function(Type01)

Flash Function	Functional Outline		Availability of Emulation
FlashEnv	Flash environment initialization/end function		Emulated
FlashBlockErase	One block erasure function		Emulated
FlashWordWrite	One word writing function		Restriction(*1)
FlashBlockIVerify	One block internal verify processing function		Emulated
FlashBlockBlankCheck	One block blank check function		Emulated
FlashGetInfo	Flash information aquisition function		
	Option = 2	CPU number and total number of blocks held by CPU	Restriction(*2)
	Option = 3	Security information	Emulated
	Option = 4	Acquisition of boot area swapping information	Restriction(*3)
	Option = 5 +Block Number	Acquisition of last address of block	Emulated
FlashSetInfo	Flash information setting function		Restriction(*4)
FlashStatusCheck	Flash information setting function		Restriction(*3)
FlashBootSwap	Boot area block swapping function		Not emulated
FlashSetUserHandler	User interrupt handler registration function		Emulated
FlashFLMDCheck	FLMD0 pin status check function		Emulated
FlashSetInfoEx	Flash information setting function[1]		Restriction(*3)
FlashNWordRead	Function for reading *n* words[1]		Restriction(*1)

*1 If an address in the guard area is specified as the third argument, a failsafe break occurs at an unexpected address. (Limitations/Precautions No.6)
*2 The device name (four-digit number) of the device file set with 850eserv2 is returned as the CPU number.
*3 When using IECUBE, a BootSwap function cannot be used even if by the device supports the BootSwap function. In FlashGetInfo function, the boot area swapping information is not reflected. In FlashBootSwap function and FlashNWordRead function, the boot area swapping setting is ignored.(Limitations/Precautions No.14)
*4 About the behavior of the FlashStatusCheck after executing the FlashBlockErase and the FlashBlockBlankCheck.Since the changing timing of the return value of the FlashStatusCheck from FE_BUSY to FE_OK is different in the emulation and the real execution, please take care. (Limitations/Precautions No.5)

[1] This function has been added in flash self programming Ver. 5.00 and later.

Table3　List of Availability of Emulation for Flash Function(Type03)

Flash Fucntion	Functional Outline		Availability of Emulation
FlashEnv	Flash environment initialization/end function		Emulated
FlashBlockErase	One block erasure function		Emulated
FlashWordWrite	One word writing function		Restriction(*1)
FlashBlockIVerify	One block internal verify processing function		Emulated
FlashBlockBlankCheck	One block blank check function		Emulated
FlashGetInfo	Flash information aquisition function		
	Option = 2	CPU number and total number of blocks held by CPU	Restriction(*2)
	Option = 3	Security information	Emulated
	Option = 4	Acquisition of boot area swapping information	Restriction(*3)
	Option = 5 + Block Number	Acquisition of last address of block	Emulated
FlashSetInfo	Flash information setting function		Restriction(*3)
FlashBootSwap	Boot area block swapping function		Not emulated
FlashFLMDCheck	FLMD0 pin status check function		Emulated
FlashWordRead	Data reading function		Restriction(*1)
FlashIVerify	Internal verify function (for EEPROM)		Not emulated
FlashBlankCheck	Blank check function (for EEPROM)		Not emulated
EEPROM_Init	EEPROM area initialization function (for EEPROM)		Not emulated
EEPROM_Write	EEPROM write function (for EEPROM)		Not emulated
EEPROM_Read	EEPROM read function (for EEPROM)		Not emulated
EEPROM_Copy	EEPROM copy function (for EEPROM)		Not emulated
EEPROM_VChK	EEPROM valid area check function (for EEPROM)		Not emulated
EEPROM_Erase	EEPROM erase function (for EEPROM)		Not emulated

*1　If an address in the guard area is specified as the third argument, a failsafe break occurs at an unexpected address. (Limitations/Precautions No.6)
*2　The device name (four-digit number) of the device file set with 850eserv2 is returned as the CPU number.
*3　When using IECUBE, a BootSwap function cannot be used even if by the device supports the BootSwap function. In FlashGetInfo function, the boot area swapping information is not reflected. In FlashBootSwap function and FlashNWordRead function, the boot area swapping setting is ignored.(Limitations/Precautions No.14)

Table4　List of Availability of Emulation for Flash Function(Type04)

Flash Function	Functional Outline		Availability of Emulation
FlashInit	Flash library initialization		Emulated
FlashEnv	Flash environment initialization/end function		Emulated
FlashBlockErase	One block erasure function		Emulated
FlashWordWrite	One word writing function		Restriction(*1)
FlashBlockIVerify	One block internal verify processing function		Emulated
FlashBlockBlankCheck	One block blank check function		Emulated
FlashGetInfo	Flash information aquisition function		
	Option = 2	CPU number and total number of blocks held by CPU	Restriction(*2)
	Option = 3	Security information	Emulated
	Option = 4	Acquisition of boot area swapping information	Restriction(*3)
	Option = 5	Reset vector address	Emulated
	Option = 6 +Block Number	Acquisition of last address of block	Emulated
FlashSetInfo	Flash information setting function		Restriction(*3)
FlashStatusCheck	Flash information setting function		Restriction(*4)
FlashBootSwap	Boot area block swapping function		Not emulated
FlashFLMDCheck	FLMD0 pin status check function		Emulated

*1　If an address in the guard area is specified as the third argument, a failsafe break occurs at an unexpected address. (Limitations/Precautions No.6)

*2　The device name (four-digit number) of the device file set with 850eserv2 is returned as the CPU number.

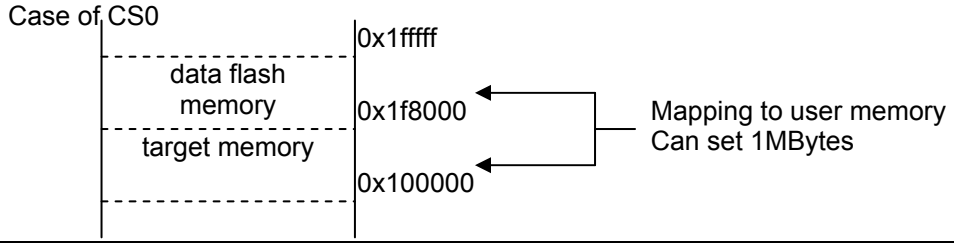
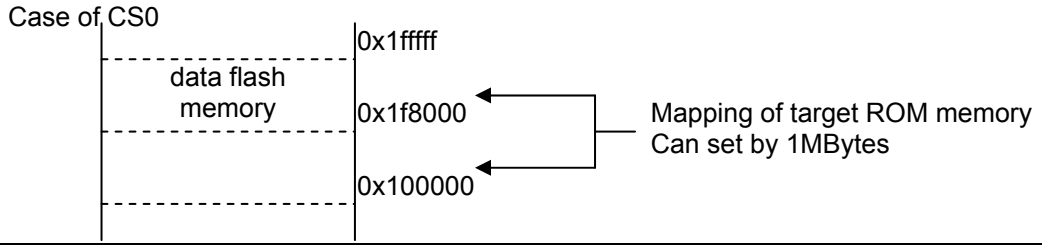
*3　When using IECUBE, a BootSwap function cannot be used even if by the device supports the BootSwap function. In FlashGetInfo function, the boot area swapping information is not reflected. In FlashBootSwap function and FlashNWordRead function, the boot area swapping setting is ignored.(Limitations/Precautions No.14)

*4　About the behavior of the FlashStatusCheck after executing the FlashBlockErase and the FlashBlockBlankCheck.Since the changing timing of the return value of the FlashStatusCheck from FE_BUSY to FE_OK is different in the emulation and the real execution, please take care. (Limitations/Precautions No.5)

No	Description
FS1	In the following cases, the Flash Memory Self-Programing emulation cannot be made effective. (1) When you does not select the default size of internal ROM. [Workaround] Please select the default size of internal ROM (2) When you use more then two Break Before The Execution [Workaround] Please delete a break point of before the execution.
FS2	When you make the Flash Memory Self-Programming emulation effective (setting “flashself” command as on), the following functions are limitation. (1) Can not change the size of internal ROM and the size of internal RAM. (2) If MULTI does the break by event etc before initializing the stack pointer(SP) register by the appropriate value, MULTI cause the illegal break of stack area. Please set the appropriate value to SP register before MULTI does the break. (3) If your IECUBE has the following restriction, the illegal break may happen when you use the Flash Memory Self-Programming emulation. To avoid the illegal break, please uncheck the ‘Non Map’ check box for internal RAM on the Fail safe break dialog. The restriction of the illegal break in case of execution on an internal RAM.
FS3	Flash Memory Self-Programming emulation related precautions(For IECUBE Sx2, Fx2, Kx1+, RS1 and IA4). From 0x0 to 0x3 is reserved for Flash Memory Self-Programming emulation. When you make the Flash Memory Self-Programming emulation effective and use 0x0 as the reset vector, the emulator changes the code of 0x0 as ‘jr 0xfffd6’, and the reset vector becomes 0x4. Therefore, please allocate the startup from 0x4. If you want to use the same code as the emulation on the real chip, the following code is recommended. # RESET handler (if 0x0) .section ".RESET",.text jr __start ← Overwritten by ‘jr 0xfffd6’ jr __start
FS4	The PC is set as an address which was set by “flashreset”, if you specify an address excluding 0x0. If you set 0x0 as the reset vector, it becomes 0x4.
FS5	About the behavior of the FlashStatusCheck after executing the FlashBlockErase and the FlashBlockBlankCheck. Since the changing timing of the return value of the FlashStatusCheck from FE_BUSY to FE_OK is different in the emulation and the real execution, please take care.
FS6	If the specified address by the third argument of the FlashWordWrite and the FlashWordRead is on the Guard Area, it causes a Fail safe break because it is the illegal memory access. Please adjust the argument of them correctly.
FS7	To validate the commands of the FlashSelf emulation (“fsecflag”, “fmacroerr” and “flashreset”), you need to reset the CPU and rerun the program after the setting. If you rerun the program without CPU reset, the setting may be invalid.
FS8	Please allocate the stack area in 84(54H) bytes or more. MULTI use the stack area 84(54H) bytes or more when MULTI does break or does Flash Memory Self-Programming emulation. If you enable the interrupt function, It is necessary to stack 84(54H) byte more as a work area of MULTI. In case of multiple interrupt, an interrupt requires 84(54H) bytes stack area each.
FS9	The data in the internal RAM is corrupted after a CPU reset. Normally, the internal RAM data after reset is not guaranteed in the actual device, but note that the operation may vary.
FS10	If a flash function is not used in accordance with the specifications or an unsupported flash function is called, "1" is returned.
FS11	The execution time of the firmware is different from an actual device.
FS12	In the Flash Memory Self-Programming emulation, the stack pointer(SP) must be set because the Flash Memory Self-Programming emulation uses the stack.
FS13	In the Flash Memory Self-Programming emulation, response of the interrupt takes long time than the actual device.

No	Description
FS14	The device for BootSwap function related precautions. When using IECUBE, a BootSwap function cannot be used even if by the device supports the BootSwap function.
FS15	There are the following limitations in the Flash Memory Self-Programming emulation(Type04). (1) The Flash Memory Self-Programming library converted to ROM cannot do Status Check error emulation. Please do not convert Flash Memory Self-Programming library to ROM. (2) The setting of the work area of internal RAM is needed. In default, it is set as 48 bytes from last address of the internal RAM. If you want to change the work area, please use the "work=address" option of the "fmacroerrsc" command.If the work area cannot allocate at the specified address, it becomes an error. (3) In case of a device which has 1MB Flash, internal ROM from 0xFF200 to the end of ROM is used by the monitor. (4) In case of a device which has more than 1MB Flash, the Flash Memory Self-Programming emulation is not supported. (5) STEP execution is not available in the Flash Memory Self-Programming library. (6) When the necessary instruction for the Flash Memory Self-Programing emulation can not detect in the load module, an error may occur. In such case, please download the load module again. StatusCheck set error 0xc1e: user system err (Instruction code for Flash self emulation doesn’t exist) (7) In the Flash Memory Emulation function, the “flashreset” command doesn't show an error even if you specify the address which is outside the flash memory area.

3. Special Notes On Data Flash Function

No	Description
FS16	When you access to the data flash merory area by way of the library, unexpected break may occur. To avoid this phenomenon, please disable the guard break function for internal RAM (set “ramgrd” option of “flsf” command) and disable the gurad break function for SFR(set “sfrgrd”,”sfrwp”,”sfrrp” option of “flsf” command)
FS17	When you set the user memory from 0xX00000 to 0xXf7fff, the data flash memory area which is from 0xX80000 to 0xXffffff is considered as the user memory too. It is because the Exec allocates the user memory by the unit of 1MBytes. In this case, the fail-safe-break doesn't happen even if a program writes to the data flash memory area. 
FS18	When you set the data flash memory area by “dfmap” command, the area from 0xX00000 to 0xX7ffff becomes accessable even if you don't set anything to the area. It is because the Exec handles the data flash memory area by the unit of 1MBytes. In this case, the fail-safe break doesn't happen even if a program reads the area from 0xX00000 to 0xX7ffff. 
FS19	Cannot set the software break point at data flash memory area. If you set the break point at data flash area, please use the hardware break point.

4. Operating Precautions for MINICUBE

No	Description
OC1	V850ES/KJ1, V850ES/KJ1+(For IE-V850E1-CD-NW only) [Limitation] Using the V850ES/KJ1 and V850ES/KJ1+ device, if you connect with DCK frequency of 20MHz, cannot connect 850eserv2 and it displays the following message. " 0xc70 : device depend err(DCU access error)" [Workaround] Please set the DCK frequency to 10MHz. By setting "-2m" option during 850eserv2 connection, DCK frequency is set as 10MHz.
OC2	PCMCIA interface card(For IE-V850E1-CD-NW only) [Limitation] IE-70000-CD-IF-A (PCMCIA interface card for NEC Electronics emulators) and the IE-V850E1-CD-NW cannot be used at the same time on the same personal computer. Debugger outputs error. [Workaround] When you use the IE-V850E1-CD-NW, please check that IE-70000-CD-IF-A is not inserted to the PCMCIA slot of your computer.
OC3	Host machine [Limitation] The On-Chip Debug Emulator may not respond when after returning from suspend mode depending on the host machine. [Workaround] Do not use suspend mode. When the IE-V850E1-CD-NW does not respond, remove the IE-V850E1-CD-NW from the host machine and re-insert it. When the QB-V850MINI does not respond, disconnect the USB cable and re-connect it.
OC4	Combination with Microsoft Windows [Limitation] Please do not use Microsoft Windows's Standby feature, Pause feature, and Suspend feature after 850eserv2 invocation. Please do not use "User change" feature. [Workaround] None
OC5	Precaution for internal software break point of the MULTI (1) Since **the software breakpoint set in the internal flash memory is implemented by the ROM correction function, the maximum number of the software break varies by device.** Please refer to the device manual for the number. (2) Internally the software break point of the MULTI MULTI internally uses some software break points. As a result, it decreases break points that the user can set. When you set software break point, MULTI might display error message and might not be able to set the break point. The following is internal break point the MULTI use, and workaround. Top of main function(Only the V3.5 of the MULTI sets it. The V4.0 doesn't.) [Workaround] Please use "-noentbp" option, when you invoke the MULTI. Top of .syscall section. [Workaround] Please use "syscalls off" command to the 850eserv2. Step-execution in C source. [Workaround] Please set "FASTSTEP=0" from the MULTI command pane Regarding -noentbp,FASTSTEP, please refer to the manual of MULTI. Regarding sysycalls command, please refer to the manual of 850eserv2.

No	Description
OC6	**To use 850eserv2 for internal flash memory device debugging (invoke 850eserv2 with –cdnw option), please specify DCLOCK command.** Please set correct value to the command, otherwize flash update does not perform correctly or destroy the flash memory. (1) Use the device file other than DF703166 V1.00, DF703134 V1.00 Input the oscillation frequency. (2) Use the device file DF703166 V1.00 • When using clock through mode Input 1/3 of the oscillation frequency • x2 mode Input 2/3 of the oscillation frequency. • x3 mode Input the oscillation frequency. (3) Use the device file DF703134 V1.00 • When using clock through mode Input 1/10 of the oscillation frequency. • When using PLL mode Input the oscillation frequency. These settings forcibly change the values of the peripheral I/O register PCC and PLLCTL temporarily so that the clock speed becomes the maximum. (4) Use the device D703318Y Please use the following proper frequency. • When use clock through mode 4.0V≦VDD≦5.5V，REGC=VDD ---------> Frequency = 2.0-10.0MHz 4.0V≦VDD≦5.5V，REGC= Capacitor --> Frequency = 2.0MHz 2.7V≦VDD＜4.0V，REGC=VDD ---------> Frequency = 2.0MHz • When using PLL mode 4.5V≦VDD≦5.5V，REGC=VDD ---------> Frequency = 2.0-5.0MHz 4.0V≦VDD＜4.5V，REGC=VDD ---------> Frequency = 2.0-4.0MHz 4.0V≦VDD≦5.5V，REGC=Capacitor --> Frequency = 2.0MHz 2.7V≦VDD＜4.0V，REGC=VDD ---------> Frequency = 2.0MHz Refer to the 850eserv2 Users Manual for details. [Example] CPU clock frequency is 5MHz, sub clock is 32768Hz dclock 5000 32768 swoff 1 2 3 1 Set the main clock value (Input decimal value/ KHz). 2 Set the sub clock value (Input decimal value/ Hz) 3 The switch of whether to use a sub clock swoff : Specifies the use of the sub clock. swon : Specifies the unuse of the sub clock.

No	Description
OC7	Precaution for timer measurement function When MULTI are measuring of timer, in the case of being set the break point by MULTI internally, the correct data cannot get. The reason is because data is cleared as MULTI stops by internally break point. In the case that the correct data cannot get, please restrain the software break point by MULTI internally by the following way. The workaround and the point that MULTI internally set the software break point. • Start of main (MULTIV3.5 is relevant and MULTIV4.0 is not relevant.) [Workaround] When MULTI connected, please set "-noentbp" option. • Start of .syscall section [Workaround] Before downloading, please set "off" at 'syscalls' command by 850eserv2. • Step execution on C source [Workaround] Please set"FASTSTEP=0" on MULTI command pane. Regarding noentbp,FASTSTEP, please refer to the manual of MULTI. Regarding sysycalls command, please refer to the manual of 850eserv2.
OC8	The address 0x7A is a system reserved area for V850ES/SG2,V850ES/SJ2 devices. Please set the 0x7A as the value (0) which is written in device users manual.
OC9	**If bit 7 of address 0x79 in the ID code is set to 0, the use of the OCD emulator is disabled**, due to the specification of the security unit mounted on the chip. As a result, the 850eserv2 cannot be activated. **Please use the bit of address 0x79 to 1 always.**
OC10	Note of using V850E2/ME3 (1) When you set the before break point ('e' option by "brs" command), the device can not the parallel execution. Please note that this behavior is different from a using device. (2) When you set the after break point('f' option by "brs" command), the break point is slipped (the specified address is passed before execution stops).
OC11	The ROM correction function cannot be emulated.
OC12	To use the CACHE command, please specify right value to the command. Otherwise, cache memory access by the debugger does not work correctly.
OC13	When a hardware or software breakpoint has been set for the HALT instruction or an instruction that makes the debugger enter STOP mode, if the program is executed from that instruction, the relevant mode will be entered briefly but immediately released, and execution will resume from the next instruction.
OC14	(1) The ECR register cannot modified by register window. (2) The DBPC, the DBPSW and the register for debug function(DIR,BPCn,BPAVn,BPAMn,BPDVn, BPDMn n=0,1,2,3) cannot be modified.
OC15	If you set software break point at internal RAM or instruction RAM, please do not rewrite the code to which the break point is set with the user program. If you set the break point to this point, the break function does not work. In case of the system that rewrites the code by the user program, please set the software break point, after you write by the user program.
OC16	If you use I/O registers that need a operation to PRCMD as preparation, the I/O registers are not written correctly when you do the following operation. (1) Step a instruction which write PRCMD. (2) Stop at a instruction which write I/O registers, then execute from it.
OC17	If connect, error message will be output when the 850eserv2 is activated while the FLMD0 signal is high. Set the FLMD0 signal to low level before activating the 850eserv2. In case of IE-V850E1-CDNW, handle the FLMD0 pin in accordance with the User ' s Manual 3.2.7 <1>. (At this time, the 850eserv2 automatically manipulates the FLMD0 pin level.) In case of QB-V850MINI, handle the FLMD0 pin in accordance with the User ' s Manual 3.4.3 (7) (a). (At this time, the 850eserv2 automatically manipulates the FLMD0 pin level.)
OC18	The access event break is delayed because of the specifications of DCU (the specified address is passed before execution stops).

No	Description
OC19	The internal RAM with the BootSwap device If you reset CPU (‘a’ option is specified or no option is specified at “reset” command) with BootSwap device, the contents of the internal RAM memory are broken. Example : V850ES/SG2(μPD70F3259Y) The head 150Mbytes of the internal RAM contents are broken.
OC20	When two instructions are executed simultaneously Example 1. When a breakpoint is not set Address A: MOV r1, r2 Address A+2: XOR r1, r2 Two instructions are stepped from Address A, where one instruction should be stepped. Example 2. When a breakpoint is set at address A+2 Address A: MOV r1, r2 [B] Address A+2: XOR r1, r2 A break does not occur at address A+2 if instructions are “executed continuously” from address A. Example 3. When a hardware break is set at address A+2 before execution of instructions Address A: MOV r1, r2 [B] Address A+2: XOR r1, r2 The instruction at Address A is executed and break occurs if instructions are “executed in steps” from address A
OC21	The PC indicates the address after halt if a break occurs in the HALT status.
OC22	Before performing real-time execution, step execution is first performed on an instruction located at the PC. This causes an error in the time measurement result in the timer. In addition, when the program operation is checked using the oscillator or analog logic, the measured timing may differ between when Go is executed at a certain location and Go is executed one instruction before that location.
OC23	Since the software breakpoint set in the internal flash memory is implemented by the ROM correction function, it is made invalid temporarily by a target reset or internal reset generated by the watchdog timer. The software break is made valid after a hardware break or forced break occurs.
OC24	If a breakpoint is set in the vicinity of address 0 , error message will be displayed as a result of target reset or internal reset generated by the watchdog timer. Do not set breakpoints in the vicinity of address 0.

No	Description
OC25	(1) Restriction on execution events (when using RCU0, RCU1, or RCU2) If the address of an execution event is set in the vicinity of another execution event, the second event cannot be detected normally. This condition does not apply when the event at the second address is executed again using a branch, etc. The second event cannot be detected under the following conditions. • The interval between the first and second instruction is within 4 bytes (internal ROM, internal RAM) • The first and second instruction are executed consecutively (target) [Execution example] <Program example (internal ROM)> 00FE nop 0100 nop ← 1st instruction 0102 nop ↕ Within 4 bytes 0104 nop 0106 nop ← 2nd instruction 0108 nop <Event example> Event Name :Evt0001 Event Status :Execution Address :0x100 <Event link example> Link Name :Lnk0001 Phase1 :Evt0001 Phase2 :Evt0002 Event Name :Evt0002 Event Status :Execution Address :0x106 <Program example (target)> 00FFFFFE nop 01000000 nop ← 1st instruction 01000002 nop ← 2nd instruction <Event example> Event Name :Evt0001 Event Status :Execution Address :0x100 <Event link example> Link Name :Lnk0001 Phase1 :Evt0001 Phase2 :Evt0002 Event Name :Evt0002 Event Status :Execution Address :0x102 (2) Restriction on access events (when using RCU0, RCU1, or RCU2) If the address of an access event is set in the vicinity of another access event, the second event cannot be detected normally. This condition does not apply when the event at the second address is executed again using a branch, The second event cannot be detected under the following conditions. • The interval between the first and second instruction is within 4 bytes (accessing the internal ROM or internal RAM) • The interval between the first and second instruction is within 28 bytes (target) <Program example (internal ROM)> 0100 mov 0x1000, gp 0106 ld.b 0x10[gp], r6 ← 1st instruction 010a nop ↕ Within 2 bytes 010c ld.b 0x12[gp], r7 ← 2nd instruction 0110 nop <Event example> Event Name :Evt0001 Event Status :R/W Access Size :Byte Address :0x1010 <Event link example> Link Name :Lnk0001 Phase1 :Evt0001 Phase2 :Evt0002 Event Name :Evt0002 Event Status :R/W Access Size :Byte Address :0x1012 <Program example (target)> 0100 mov 0x100000, gp 0106 ld.b 0x10[gp], r6 ← 1st instruction 010a nop : ↕ Within 28 bytes : : 0110 nop 0120 ld.b 0x12[gp], r7 ← 2nd instruction <Event example> Event Name :Evt0001 Event Status :R/W Address :0x1010 <Event link example> Link Name :Lnk0001 Phase1 :Evt0001 Phase2 :Evt0002 Event Name :Evt0002 Event Status :R/W Address :0x1012

<table>
<tr><th>No</th><th>Description</th></tr>
<tr><td>OC26</td><td>
Restriction on event detection during misalign access
(1) Restriction on write access event
No events can be detected.
(2) Restriction on read access event
Events can be detected by setting the read access event as shown below.
<table>
<tr><th colspan="2">Operation of Execution Instruction</th><th colspan="3">Event Condition</th><th rowspan="2">Detection Status</th><th></th></tr>
<tr><th>Size</th><th>address</th><th>address</th><th>Access Size</th><th>Data</th></tr>
<tr><td rowspan="4">Word</td><td>(Multiple of 4)+0</td><td>(Multiple of 4)+0</td><td>Word</td><td>0x44332211</td><td>Detected</td><td></td></tr>
<tr><td>(Multiple of 4)+1</td><td>(Multiple of 4)+1</td><td>Byte</td><td>0x22</td><td>Detected</td><td>(a)</td></tr>
<tr><td>(Multiple of 4)+2</td><td>(Multiple of 4)+2</td><td>Half Word</td><td>0x4433</td><td>Detected</td><td>(b)</td></tr>
<tr><td>(Multiple of 4)+3</td><td>(Multiple of 4)+3</td><td>Byte</td><td>0x44</td><td>Detected</td><td></td></tr>
<tr><td rowspan="4">Half Word</td><td>(Multiple of 4)+0</td><td>(Multiple of 4)+0</td><td>Half Word</td><td>0x2211</td><td>Detected</td><td></td></tr>
<tr><td>(Multiple of 4)+1</td><td>(Multiple of 4)+1</td><td>Byte</td><td>0x22</td><td>Detected</td><td>(c)</td></tr>
<tr><td>(Multiple of 4)+2</td><td>(Multiple of 4)+2</td><td>Half Word</td><td>0x4433</td><td>Detected</td><td></td></tr>
<tr><td>(Multiple of 4)+3</td><td>(Multiple of 4)+3</td><td>Byte</td><td>0x44</td><td>Detected</td><td></td></tr>
</table>
<pre>
[Memory status]
              +0 +1 +2 +3
    3FF8000   11 22 33 44

[Execution example (a)]

    0xFFE nop   <Event example>
    01000 mov 0x3FF8001, gp   Event Name   :Evt0001
    01006 nop   Event Status :R
    01008 ld.w 0x0[gp],   r6   Access Size :Byte
                                        Address   :0x3FF8001
                                        Data   :0x22

[Execution example (b)]

                                        <Event example>
    0xFFE nop   Event Name   :Evt0001
    01000 mov 0x3FF8002, gp   Event Status :R
    01006 nop   Access Size :Half Word
    01008 ld.w 0x0[gp],   r6   Address   :0x3FF8002
                                        Data   :0x4433

[Execution example (c)]

                                        <Event example>
    0xFFE nop   Event Name   :Evt0001
    01000 mov 0x3FF8001, gp   Event Status :R
    01006 nop   Access Size :Byte
    01008 ld.w 0x0[gp],   r6   Address   :0x3FF8001
</pre>
</td></tr>
<tr><td>OC27</td><td>The WAIT mask option by PINMASK command can be used only in products that include RCU1 or RCU2 (under development).</td></tr>
<tr><td>OC28</td><td>Whether an internal reset generated by the watchdog timer can be masked by the RESET mask option by PINMASK command or not varies depending on the device used.</td></tr>
<tr><td>OC29</td><td>
Restrictions on event detection using bit manipulation instruction

When the access size of an event is set to Byte and the event is set at an address other than a multiple of 4, if the address is accessed by a bit manipulation instruction, an incorrect event may be detected or no event may be detected.
When the access size of an event is set to Bit and the event is set at an address other than a multiple of 4, if the address is accessed by a bit manipulation instruction, an incorrect event may be detected or no event may be detected.

</td></tr>
</table>

No	Description
OC30	A pin reset during a break is masked and the CPU or peripheral I/O cannot be reset. In addition, the CPU or peripheral I/O may not be reset sometimes if a pin reset or internal reset occurs when data is overwritten by Memview or MEMORY command during user program execution.
OC31	CPU or peripheral I/O cannot be reset as the following case. • During a break (Because the reset pin is masked during a break)
OC32	When you use Exec and 850eserv2, MULTI displays the following warning message and MULTI can not connect. EXEC library Vx.xx is too old. Please use EXEC Vx.xx or later. Please use the newest Exec version.
OC33	No break occurs in an interrupt service routine acknowledged during self-programming, even if an event breakpoint has been set. (V850E1, D70F3259)
OC34	Do not set a software break in the internal flash memory area. (V850E1, V850ES)
OC35	A forcible break can be accepted even if flash memory self-programming is in progress. However, user cannot debug correctly then. After the forcible break, re-execute immediately or reset the CPU. (V850E1, D70F3259)
OC36	Cannot use disassemble feature or Memview due to the internal flash memory value has not updated correctly in the middle of the execution of the self-programming function. (V850E1, V850ES)
OC37	Do not use the peripheral break function, if you use the flash self programming function in your program. Please set “pb” command OFF. If you set “pb” command ON, the contents of the internal flash memory may destroyed.

5. Operating Precautions for MINICUBE-2

No	Description
M1	Use sub clock with connection UART [Limitation] In case of a condition that the sub clock is working and a program is stopped by user or debugger, the monitor forcibly starts oscillation of main clock even if you have stopped it. [Workaround] None
M2	Reset [Limitation] MULTI will stop a program running by internal reset or target reset (If the reset is unmasked condition) because of the specification of the MINICUBE2. [Workaround] None
M3	Write SFR from MULTI [Limitation] You can't write the SFR which need a special sequence from the MULTI command pane except the PCC and the CKC. [Workaround] None
M4	Restriction on conflict between software break and user break [Limitation] When Multi stopped by break(compulsion break, after hardware break, fail-safe break, etc) near the point of software break, if MULTI restarts after this break, MULTI may execute the illegal instruction. When the following two conditions are satisfied at the same time, MULTI misrecognizes a stop of execution occurred by a software break even if it occurred by a compulsion break. In this case, MULTI will set an incorrect value which is 2 bytes smaller than a correct value to the PC at the restart. (3) A program execution is stopped at a branch destination instruction. (4) A 2byte instruction is placed on immediately before of a branch destination instruction. [Example] As the following program, if MULTI broken on 0x1100, in the specification of hard ware, MULTI doesn't know a break reason. MULTI assumes that a program execution has been stopped by a software break point, and it sets 0x10FE to the PC. Therefore, in case of (2), MULTI will stop at wrong address and it will set wrong value to the PC. (3) Execute the instruction on 0x10FE and break → Displays the break PC value to 0x10FE. (4) Execute the branch instruction on 0x100C and break → MULTI assumes that a break reason is same as the case(1). The PC value is set as 0x10fe. 0x1008 st.b r20,PM0 0x100C jr 0x1100 ← branch to 0x1100 : : 0x10FE bz 0x1514 ← Set software break point 0x1100 mov r1,r2 ← Branch destination from 0x100C 0x1102 nop [Workaround] Please use the hardware break instead of the software break.
M5	Combination with Microsoft Windows [Limitation] Please do not use Microsoft Windows's Standby feature, Pause feature, and Suspend feature after 850eserv2 invocation. Please do not use "User change" feature. [Workaround] None
M6	The internal RAM with the BootSwap device [Limitation] If you reset CPU ("a" option is specified or no option is specified at "reset" command) with BootSwap device, the data of the internal RAM memory will be broken. [Workaround] None.

No	Description
M7	Precaution for the option byte The monitor forcibly stops the watchdog timer on the MINICUBE2. Therefore, please don't set the value of option byte(data of 0x7a) which cannot stop the watchdog timer. Please refer to the device manual for the option byte.
M8	The variable reset vector function is not supported.
M9	Precaution for internal software break point of the MULTI (3) Since **the software breakpoint set in the internal flash memory is implemented by the ROM correction function, the maximum number of the software break varies by device.** Please refer to the device manual for the number. (4) Internally the software break point of the MULTI MULTI internally uses some software break points. As a result, it decreases break points that the user can set. When you set software break point, MULTI might display error message and might not be able to set the break point. The following is internal break point the MULTI use, and workaround. Top of main function(Only the V3.5 of the MULTI sets it. The V4.0 doesn't.) [Workaround] Please use "-noentbp" option, when you invoke the MULTI. Top of .syscall section. [Workaround] Please use "syscalls off" command to the 850eserv2. Step-execution in C source. [Workaround] Please set "FASTSTEP=0" from the MULTI command pane Regarding -noentbp,FASTSTEP, please refer to the manual of MULTI. Regarding sysycalls command, please refer to the manual of 850eserv2.
M10	Precaution for timer measurement function When MULTI are measuring of timer, in the case of being set the break point by MULTI internally, the correct data cannot get. The reason is because data is cleared as MULTI stops by internally break point. In the case that the correct data cannot get, please restrain the software break point by MULTI internally by the following way. The workaround and the point that MULTI internally set the software break point. • Start of main (MULTIV3.5 is relevant and MULTIV4.0 is not relevant.) [Workaround] When MULTI connected, please set "-noentbp" option. • Start of .syscall section [Workaround] Before downloading, please set "off" at 'syscalls' command by 850eserv2. • Step execution on C source [Workaround] Please set"FASTSTEP=0" on MULTI command pane. Regarding noentbp,FASTSTEP, please refer to the manual of MULTI. Regarding sysycalls command, please refer to the manual of 850eserv2.
M11	At the 850eserv2 starting, please set CPU clock frequency by dclock command before downloading a program. If the dclock command is not set correctly, 850eserv2 cannot access the emulator correctly. Please specify either main clock or sub clock by "dclock" command. Refer to the 850eserv2 Users Manual for details. [Example] CPU clock frequency is 5MHz, sub clock is 32768Hz dclock 5000 32768 swoff 1 2 3 4 Set the main clock value (Input decimal value/ KHz). 5 Set the sub clock value (Input decimal value/ Hz) 6 The switch of whether to use a sub clock swoff : Specifies the use of the sub clock. swon : Specifies the unuse of the sub clock.

No	Description
M12	**If bit 7 of address 0x79 in the ID code is set to 0, the use of the N-Wire emulator is disabled**, due to the specification of the security unit mounted on the chip. As a result, the 850eserv2 cannot be activated. **Please use the bit of address 0x79 to 1 always.**
M13	Precaution for using the self-programming function If you debug a self-programming by using MINICUBE2, please do not the following actions. Because MULTI may hang-up. (1) During the self-programming environment, break by software break, hardware break or compulsion break(push the stop button) (2) Step execution to the self-flash environment.
M14	The ROM correction function cannot be emulated.
M15	When a hardware (break after the execution) or software breakpoint has been set for the HALT instruction or an instruction that makes the debugger enter STOP mode, if the program is executed from that instruction, the relevant mode will be entered briefly but immediately released, and execution will resume from the next instruction.
M16	(1) The ECR register cannot modified by register window. (2) The DBPC, the DBPSW and the register for debug function (DIR, BPCn, BPAVn, BPAMn, BPDVn, BPDMn n=0～1) cannot be modified by register window.
M17	If you use I/O registers that need an operation to PRCMD as preparation, the I/O registers are not written correctly when you do the following operation. (1) Step an instruction which write PRCMD. (2) Stop at an instruction which write I/O registers, then execute from it.
M18	The access event break is delayed because of the specifications of DCU (the specified address is passed before execution stops).
M19	When two instructions are executed simultaneously Example 1. When a breakpoint is not set Address A: MOV r1, r2 Address A+2: XOR r1, r2 Two instructions are stepped from Address A, where one instruction should be stepped. Example 2. When a breakpoint is set at address A+2 Address A: MOV r1, r2 [B] Address A+2: XOR r1, r2 A break does not occur at address A+2 if instructions are “executed continuously” from address A. Example 3. When a hardware break is set at address A+2 before execution of instructions Address A: MOV r1, r2 [B] Address A+2: XOR r1, r2 The instruction at Address A is executed and break occurs if instructions are “executed in steps” from address A
M20	The PC indicates the address after halt if a break occurs in the HALT status.
M21	Before performing real-time execution, step execution is first performed on an instruction located at the PC. This causes an error in the time measurement result in the timer. In addition, when the program operation is checked using the oscillator or analog logic, the measured timing may differ between when Go is executed at a certain location and Go is executed one instruction before that location.

No	Description
M22	(3) Restriction on execution events (when using RCU0, RCU1, or RCU2) If the address of an execution event is set in the vicinity of another execution event, the second event cannot be detected normally. This condition does not apply when the event at the second address is executed again using a branch, etc. The second event cannot be detected under the following conditions. • The interval between the first and second instruction is within 4 bytes (internal ROM, internal RAM) • The first and second instruction are executed consecutively (target) [Execution example] <Program example (internal ROM)> 00FE nop 0100 nop ← 1st instruction 0102 nop ↕ Within 4 bytes 0104 nop 0106 nop ← 2nd instruction 0108 nop <Event example> Event Name :Evt0001 Event Status :Execution Address :0x100 <Event link example> Link Name :Lnk0001 Phase1 :Evt0001 Phase2 :Evt0002 Event Name :Evt0002 Event Status :Execution Address :0x106 <Program example (target)> 00FFFFFE nop 01000000 nop ← 1st instruction 01000002 nop ← 2nd instruction <Event example> Event Name :Evt0001 Event Status :Execution Address :0x100 <Event link example> Link Name :Lnk0001 Phase1 :Evt0001 Phase2 :Evt0002 Event Name :Evt0002 Event Status :Execution Address :0x102 (4) Restriction on access events (when using RCU0, RCU1, or RCU2) If the address of an access event is set in the vicinity of another access event, the second event cannot be detected normally. This condition does not apply when the event at the second address is executed again using a branch, The second event cannot be detected under the following conditions. • The interval between the first and second instruction is within 4 bytes (accessing the internal ROM or internal RAM) • The interval between the first and second instruction is within 28 bytes (target) <Program example (internal ROM)> 0100 mov 0x1000, gp 0106 ld.b 0x10[gp], r6 ← 1st instruction 010a nop ↕ Within 2 bytes 010c ld.b 0x12[gp], r7 ← 2nd instruction 0110 nop <Event example> Event Name :Evt0001 Event Status :R/W Access Size :Byte Address :0x1010 <Event link example> Link Name :Lnk0001 Phase1 :Evt0001 Phase2 :Evt0002 Event Name :Evt0002 Event Status :R/W Access Size :Byte Address :0x1012 <Program example (target)> 0100 mov 0x100000, gp 0106 ld.b 0x10[gp], r6 ← 1st instruction 010a nop : ↕ Within 28 bytes 0110 nop 0120 ld.b 0x12[gp], r7 ← 2nd instruction <Event example> Event Name :Evt0001 Event Status :R/W Address :0x1010 <Event link example> Link Name :Lnk0001 Phase1 :Evt0001 Phase2 :Evt0002 Event Name :Evt0002 Event Status :R/W Address :0x1012

<table>
<tr><th>No</th><th>Description</th></tr>
<tr><td>M23</td><td>
Restriction on event detection during misalign access
(3) Restriction on write access event
No events can be detected.
(4) Restriction on read access event
Events can be detected by setting the read access event as shown below.
<table>
<tr><th colspan="2">Operation of Execution Instruction</th><th colspan="3">Event Condition</th><th rowspan="2">Detection Status</th><th></th></tr>
<tr><th>Size</th><th>address</th><th>address</th><th>Access Size</th><th>Data</th></tr>
<tr><td rowspan="4">Word</td><td>(Multiple of 4)+0</td><td>(Multiple of 4)+0</td><td>Word</td><td>0x44332211</td><td>Detected</td><td></td></tr>
<tr><td>(Multiple of 4)+1</td><td>(Multiple of 4)+1</td><td>Byte</td><td>0x22</td><td>Detected</td><td>(a)</td></tr>
<tr><td>(Multiple of 4)+2</td><td>(Multiple of 4)+2</td><td>Half Word</td><td>0x4433</td><td>Detected</td><td>(b)</td></tr>
<tr><td>(Multiple of 4)+3</td><td>(Multiple of 4)+3</td><td>Byte</td><td>0x44</td><td>Detected</td><td></td></tr>
<tr><td rowspan="4">Half Word</td><td>(Multiple of 4)+0</td><td>(Multiple of 4)+0</td><td>Half Word</td><td>0x2211</td><td>Detected</td><td></td></tr>
<tr><td>(Multiple of 4)+1</td><td>(Multiple of 4)+1</td><td>Byte</td><td>0x22</td><td>Detected</td><td>(c)</td></tr>
<tr><td>(Multiple of 4)+2</td><td>(Multiple of 4)+2</td><td>Half Word</td><td>0x4433</td><td>Detected</td><td></td></tr>
<tr><td>(Multiple of 4)+3</td><td>(Multiple of 4)+3</td><td>Byte</td><td>0x44</td><td>Detected</td><td></td></tr>
</table>
<pre>
[Memory status]
              +0 +1 +2 +3
    3FF8000   11 22 33 44
[Execution example (a)]
    0xFFE nop   <Event example>
    01000 mov 0x3FF8001, gp   Event Name   :Evt0001
    01006 nop   Event Status :R
    01008 ld.w 0x0[gp],   r6   Access Size :Byte
                                        Address   :0x3FF8001
                                        Data   :0x22

[Execution example (b)]
                                        <Event example>
    0xFFE nop   Event Name   :Evt0001
    01000 mov 0x3FF8002, gp   Event Status :R
    01006 nop   Access Size :Half Word
    01008 ld.w 0x0[gp],   r6   Address   :0x3FF8002
                                        Data   :0x4433

[Execution example (c)]
                                        <Event example>
    0xFFE nop   Event Name   :Evt0001
    01000 mov 0x3FF8001, gp   Event Status :R
    01006 nop   Access Size :Byte
    01008 ld.w 0x0[gp],   r6   Address   :0x3FF8001
                                        Data   :0x22
</pre>
</td></tr>
<tr><td>M24</td><td>
Restrictions on event detection using bit manipulation instruction
• When the access size of an event is set to Byte and the event is set at an address other than a multiple of 4, if the address is accessed by a bit manipulation instruction, an incorrect event may be detected or no event may be detected.
• When the access size of an event is set to Bit and the event is set at an address other than a multiple of 4, if the address is accessed by a bit manipulation instruction, an incorrect event may be detected or no event may be detected.
</td></tr>
</table>

No	Description
M25	CPU or peripheral I/O cannot be reset as the following case. • During a break (Because the reset pin is masked during a break)
M26	When you connect AZ850 made by NEC Electronics, the hardware trace method cannot be used. Use the software trace method instead.
M27	When you use Exec and 850eserv2, MULTI displays the following warning message and MULTI can not connect. EXEC library Vx.xx is too old. Please use EXEC Vx.xx or later. Please use the newest Exec version.

(A) Valid Specification

Item	Date published	Document No.	Document Title
1	April 2009	sv-v850e2-us-77	V850/850E ICE Server Reference Manual Rev. 7.7

(B) Revision History

Item	Date published	Document No.	Comment
1	19-April-2009	U19303EE1V0IF00	First release.