NEC

Customer Notification

V850E, V850ES Architectures

32-bit Single-Chip Microcontrollers

Operating Precautions

Document No. U20081EE1V1IF00
Date Published November 2009

© NEC Electronics (Europe) GmbH

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 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)

(A) Table of Operating Precautions 5

(B) Description of Operating Precautions. 6

(C) Valid Specification 16

(D) Revision History 17

(A) Table of Operating Precautions

		Core	
No.	**Outline**	V850E1	V850ES
1	SLD instruction precaution (Specification change notice)	✗	✗
2	MUL/MULU instruction precaution (Specification change notice)	✓	✗

✓ : Not applicable
✗ : Applicable

(B) Description of Operating Precautions

No. 1	SLD instruction precaution (Specification change notice)

Details

If a conflict occurs between the decode operation of the instruction (<2> in the examples mentioned below) immediately before the **sld** instruction (<3> in the examples) following a special instruction (<1> in the examples) and an interrupt request before execution of the special instruction is complete, the execution result of the special instruction may not be stored in a register as expected.
This situation may only occur when the same register is used as the destination register of the special instruction and the **sld** instruction, and when the register value is referenced by the instruction followed by the **sld** instruction.

Conditions under which the conflict occurs:

The situation may occur when all the following conditions (1) to (3) are satisfied.

(1) Either condition (I) or (II) is satisfied

Condition (I):
The same register is used as the destination register of a special instruction (see below) and the subsequent **sld** instruction and as the source register (reg1) of an instruction shown below followed by the **sld** instruction (See Example 1).

mov **reg1**,reg2	not **reg1**,reg2	satsubr **reg1**,reg2	satsub **reg1**,reg2
satadd **reg1**,reg2	or **reg1**,reg2	xor **reg1**,reg2	and **reg1**,reg2
tst **reg1**,reg2	subr **reg1**,reg2	sub **reg1**,reg2	add **reg1**,reg2
cmp **reg1**,reg2	mulh **reg1**,reg2		

Condition (II):
The same register is used as the destination register of a special instruction (see below) and the subsequent **sld** instruction and as the destination register (reg2) of an instruction shown below followed by the **sld** instruction (See Examples 2 and 3).

not reg1,**reg2**	satsubr reg1,**reg2**	satsub reg1,**reg2**	satadd reg1,**reg2**
satadd imm5,**reg2**	or reg1,**reg2**	xor reg1,**reg2**	and reg1,**reg2**
tst reg1,**reg2**	subr reg1,**reg2**	sub reg1,**reg2**	add reg1,**reg2**
add imm5,**reg2**	cmp reg1,**reg2**	cmp imm5,**reg2**	shr imm5,**reg2**
sar imm5,**reg2**	shl imm5,**reg2**		

Special instruction:

- **ld** instruction: ld.b, ld.h, ld.w, ld.bu, ld.hu
- **sld** instruction: sld.b, sld.h, sld.w, sld.bu, sld.hu
- Multiply instruction: mul, mulh, mulhi, mulu

(2) When the execution result of the special instruction (see above) has not been stored in the destination register before execution of the instruction (instruction of condition (I) or (II)) immediately before the **sld** instruction starts in the CPU pipeline.

No. 1	SLD instruction precaution (Specification change notice)

(cont.)

(3) When the decode operation of the instruction (instruction of condition (I) or (II)) immediately before the **sld** instruction and interrupt request servicing conflict.

Examples of instruction sequences that may cause the conflict:

Example 1:

```
<1>  ld.w   [r11], r10
                ⋮
<2>  mov    r10, r28
<3>  sld.w  0x28, r10
```

This situation occurs when the decode operation of the **mov** instruction (<2>) immediately before the **sld** instruction (<3>) and interrupt request servicing conflict before the execution of the special instruction **ld** (<1>) is complete.

Example 2:

```
<1>  ld.w   [r11], r10
                ⋮
<2>  cmp    imm5, r10
<3>  sld.w  0x28, r10
<4>  bz label
```

This situation occurs when the decode operation of **cmp** (<2>) immediately before the **sld** instruction (<3>) and interrupt request servicing conflict before execution of the special instruction **ld** (<1>) is complete. As a result, the compare result of the **cmp** instruction becomes undefined, which may cause an unexpected operation of the branch instruction **bz** (<4>).

Example 3:

```
<1>  ld.w   [r11], r10
                ⋮
<2>  add    imm5, r10
<3>  sld.w  0x28, r10
<4>  setf   c, r16
```

This situation occurs when the decode operation of the **add** instruction (<2>) immediately before the **sld** instruction (<3>) and interrupt request servicing conflict before execution of the special instruction **ld** (<1>) is complete. As a result, the result of the **add** instruction and the depending status flags become undefined, which may cause an unexpected operation of the **setf** instruction (<4>).

Workaround

(1) Do not use the **sld** instruction (e. g. by avoiding code optimization that makes use of **sld**).

(2) If a code sequence as described above is used (a **sld** instruction following an instruction that can be executed in parallel), insert a **nop** instruction before the **sld** instruction.

(3) If a code sequence as described above is used (a **sld** instruction following an instruction that can be executed in parallel), exchange the order of the previous two instructions as long as the program algorithm is not disturbed:

No. 1	SLD instruction precaution (Specification change notice)

(cont.)

Example:
1. (before implementing workaround)
ld.w [r11], r10
...
add r11, r12
mov r10, r28
sld.w 0x28, r10

2. (after implementing workaround)
ld.w [r11], r10
...
mov r10, r28
add r11, r12
sld.w 0x28, r10

(4) When assembler code is used:
Avoid the crictical code sequences as described above.

No. 2	MUL/MULU instruction precaution (Specification change notice)

Details:

The subsequent behaviour occurs if a specific instruction sequence (sequence 1 or 2 below) is executed.

- The result of executing a multiplication instruction is not stored in the relevant general-purpose register.
- As a result of executing a `ld` instruction for a mis-aligned address, the data at an unexpected address is read and stored in the relevant general-purpose register.

Sequence 1:

In the following instruction sequence, the RAM is read by one of the instructions in (2) at the same time as the RAM is accessed by a DMA transfer:

(1) `ld` or `sld`: A load instruction for the internal ROM
(2) `ld` or `sld`: A load instruction for the internal RAM
(3) `mul` or `mulu`: An instruction that multiplies word data and whose result is truncated to 32 bits**Note 1**
(4) ... **Note 2**
(5) `ld` or `sld`: A load instruction for a mis-aligned address in the internal ROM or RAM

Notes: 1. For a mul or mulu instruction, the operation described here occurs if `r0` is specified for the third operand (`reg3`), or the same register is specified for the second operand (`reg2`) and third operand (`reg3`), as shown below:

- `mul reg1, reg2, reg3` (`reg3` = `r0` or `reg2` = `reg3`)
- `mul imm9, reg2, reg3` (`reg3` = `r0` or `reg2` = `reg3`)
- `mulu reg1, reg2, reg3` (`reg3` = `r0` or `reg2` = `reg3`)
- `mulu imm9, reg2, reg3` (`reg3` = `r0` or `reg2` = `reg3`)

2. This behaviour applies if it takes 2 clock cycles or less between the instructions in (3) and (5).

This behaviour does not apply in any of the following cases:

- DMA is not used to transfer data to or from the internal RAM.
- The data read by the load instruction in (1) is used to specify the target address of the load instruction in (2).
- The data read by the load instruction in (2) is referenced by the multiplication instruction in (3).
- The data obtained by the multiplication instruction in (3) is used to specify the target address of the load instruction in (5).
- The data obtained by the multiplication instruction in (3) is referenced by an instruction executed between the instructions in (3) and (5).
- At least one of the following instructions is executed between the instructions in (3) and (5):
  - A multiplication instruction (`mul, mulh, mulhi, mulu`)
  - A bit manipulation instruction (`clr1, not1, set1, tst1`)
  - A special instruction (`callt, dispose, switch`)
- The instruction in (5) is a load instruction that accesses the memory in bytes (`ld.b, ld.bu, sld.b` or `sld.bu`).
- The instructions in (1) to (5) are located in an external memory or the internal RAM.

No. 2	MUL/MULU instruction precaution (Specification change notice)

Sequence 2:

In the following instruction sequence, access by the instruction in (1) ends at the same time as the instruction in (3) accesses the internal RAM:

(1) `ld` or `sld`: A load instruction for an external memory, the CAN controller, a USB peripheral I/O register or the data flash area
(2) ... **Note 1**
(3) `ld` or `sld`: A load instruction for the internal RAM
(4) `mul` or `mulu`: An instruction that multiplies word data and whose result is truncated to 32 bits**Note 2**
(5) ... **Note 3**
(6) `ld` or `sld`: A load instruction for a mis-aligned address in the internal ROM or RAM

Notes: 1. This behaviour applies if a different instruction is executed between the load instructions in (1) and (3), and the access by the instruction in (1) ends at the same time as the instruction in (3) accesses the internal RAM.

2. For a mul or mulu instruction, the operation described here occurs if `r0` is specified for the third operand (`reg3`), or the same register is specified for the second operand (`reg2`) and third operand (`reg3`), as shown below:

- `mul reg1, reg2, reg3` (`reg3` = `r0` or `reg2` = `reg3`)
- `mul imm9, reg2, reg3` (`reg3` = `r0` or `reg2` = `reg3`)
- `mulu reg1, reg2, reg3` (`reg3` = `r0` or `reg2` = `reg3`)
- `mulu imm9, reg2, reg3` (`reg3` = `r0` or `reg2` = `reg3`)

3. This behaviour applies if it takes 2 clock cycles or less between the instructions in (4) and (6).

This behaviour does not apply in any of the following cases:

- None of the following functions is used: external bus, CAN controller, USB peripheral I/O register, data flash area.
- The data read by the load instruction in (1) is used to specify the target address of the load instruction in (3).
- The data read by the load instruction in (1) is referenced by an instruction executed between the instructions in (1) and (3).
- The data read by the load instruction in (3) is referenced by the multiplication instruction in (4).
- The data obtained by the multiplication instruction in (4) is used to specify the target address of the load instruction in (6).
- The data obtained by the multiplication instruction in (4) is referenced by an instruction executed between the instructions in (4) and (6).
- At least one of the following instructions is executed between the instructions in (4) and (6):
  - A multiplication instruction (`mul, mulh, mulhi, mulu`)
  - A bit manipulation instruction (`clr1, not1, set1, tst1`)
  - A special instruction (`callt, dispose, switch`)
- The instruction in (6) is a load instruction that accesses the memory in bytes (`ld.b, ld.bu, sld.b` or `sld.bu`).
- The instructions in (1) to (6) are located in an external memory or the internal RAM.

No. 2	MUL/MULU instruction precaution (Specification change notice)

Workaround:

The microcontroller will not be modified. Instead, compilers will be modified to automatically prevent instruction sequences to which this behaviour applies from being generated. Please note that this workaround does not work for instructions in assembly code (nevertheless the CA850 outputs a message for such instructions).

Development environment required to work around this behaviour:

Use one of the following compilers:

- NEC Electronics compiler CA850: V3.42 or later.
- GHS compiler CC850: V516 with Patch CPDW9XNT-CDR-V85X-V516-PATCH02 or later.
- IAR compiler ICCV850: V3.71 or later.

Note that the outputs from the above versions might differ from the outputs from the older versions.

Compiler updates are available through NEC-EE TOOLWEB and can be downloaded from following pages:

GHS compiler:
Go to www.eu.necel.com/update and select the order code CPDW9X/NT-CDR-V850
or select the following link:
http://www.eu.necel.com/update?id=26

IAR compiler:
The updated version is expected to be available by begin of december 2009.
Go to www.eu.necel.com/update and select the order code EWV850-FULL-EE
or select the following link:
http://www.eu.necel.com/update?id=18

If you are using a compiler other than one shown here, contact your compiler vendor.

Action for already developed systems:

In order to check whether the behaviour applies for an already developed system proceed as follows:

First judgment: Check the product name and usage conditions
Second judgment: Check the compiling conditions
Third judgment: Check the software using the check tool

Please see the following sections for details on the judgement steps.

No. 2	MUL/MULU instruction precaution (Specification change notice)

First judgment: Check the product name and usage conditions

Checking the product name:

The behaviour only applies to V850ES core devices and emulators for these devices (not to V850, V850E1, and V850E2 core devices or the emulators for these devices).

V850ES core devices are products whose third digit from the right in the part number is 2, 3, 6, or 7 (position of letter x in the below example):

µPD70(F)3**x**yz

The described behaviour does not apply for the following products:
V850ES/KE1 (µPD703207, µPD70F3207H)
V850ES/ST2 (µPD703220)
V850ES/ST3 (µPD703221)
V850ES/KE1+ (µPD703302, µPD70F3302)
V850ES/IK1 (µPD703327, µPD703329, µPD70F3329)
V850ES/HE2 (µPD70F3700, µPD70F3701)
V850ES/HF2 (µPD70F3702, µPD70F3703, µPD70F3704)
V850ES/IE2 (µPD70F3713, µPD70F3714)
V850ES/KE2 (µPD70F3726)

If one of the above devices is used the behaviour is not applicable. In this case further checking is not necessary.

If a product with a part number starting with µPD76 is used, contact an NEC Electronics sales representative or distributor for whether the behaviour applies.

Checking the product usage conditions

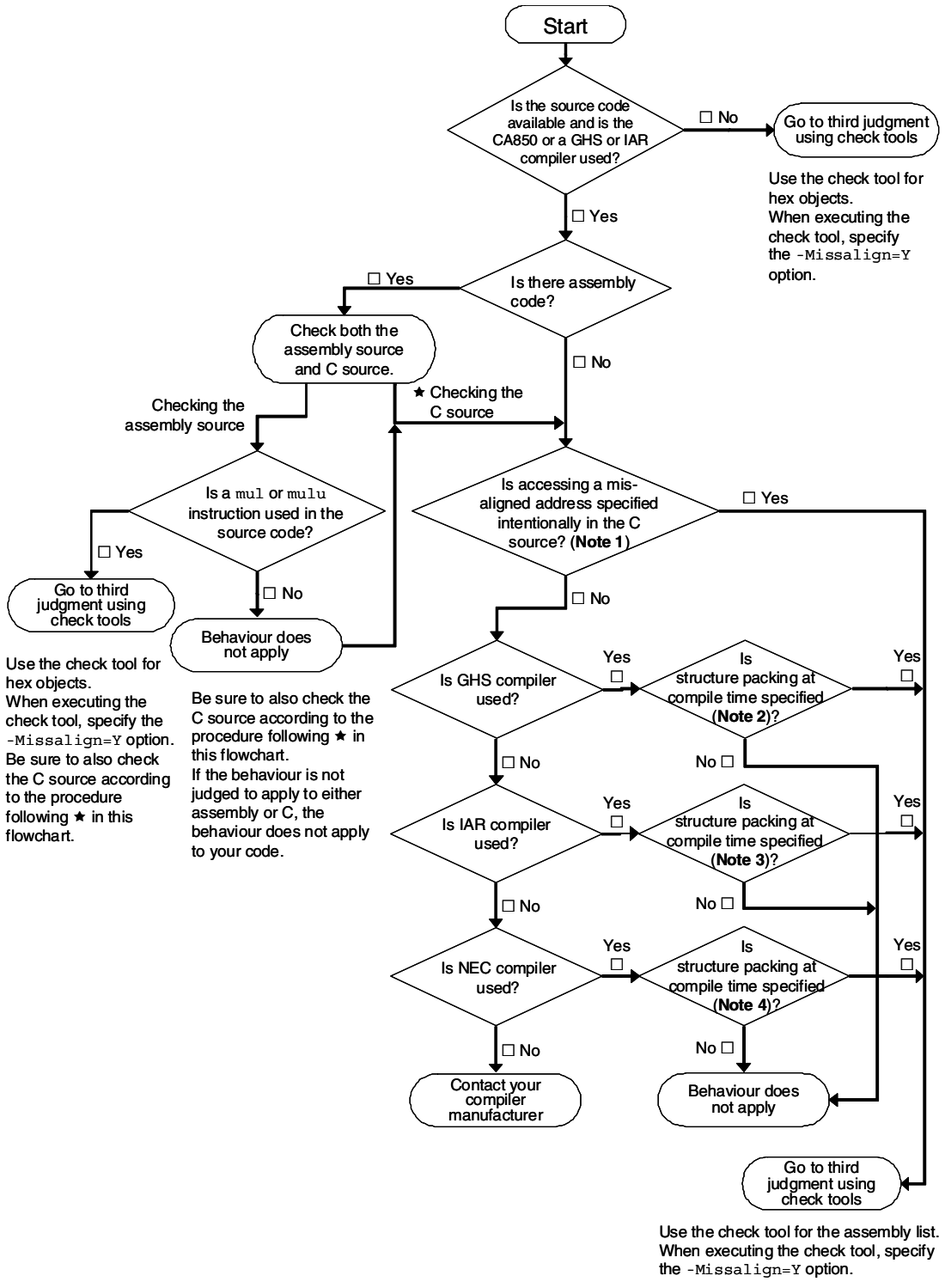
Select “Yes” or “No” for whether each feature from (1) to (6) below is used. If the product does not incorporate a feature, select “Not relevant” for the feature. If there is no item for which “Yes” is selected, the behaviour does not apply. In this case further checking is not necessary. Otherwise proceed with the second judgement.

(1) Data is transferred to or from the internal RAM using DMA.
( ) Yes ( ) No ( ) Not relevant
(2) An on-chip CAN controller is used.
( ) Yes ( ) No ( ) Not relevant
(3) An on-chip USB controller is used.
( ) Yes ( ) No ( ) Not relevant
(4) An internal data flash is used.
( ) Yes ( ) No ( ) Not relevant
(5) An on-chip Ethernet controller is used.
( ) Yes ( ) No ( ) Not relevant
(6) An external bus interface is used.
( ) Yes ( ) No ( ) Not relevant

No. 2	MUL/MULU instruction precaution (Specification change notice)

Second judgment: Check the compiling conditions

Use the following flowchart to judge whether the behaviour applies. If the behaviour might apply, a third judgment using check tools is required.



No. 2	MUL/MULU instruction precaution (Specification change notice)

Notes: 1. A mis-aligned address might be accessed if a pointer to a `char` is cast to a pointer to an `int` to reference the `int`, as shown below.

```
func()
{
  char  *ptr_char="abcdef" ;
  int    data,  *ptr_int ;

    ptr_int= (int *) (ptr_char+2);
   data = *ptr_int;
}
```

2. When using the GHS compiler CC850:
The behaviour might apply if either of the following conditions is satisfied:
1. `#pragma pack(1)` or `#pragma pack(2)` is specified in the source code and `-misalign_pack` is specified as a compiler option.
2. `-pack=1` or `-pack=2` and `-misalign_pack` are specified as compiler options.

3. When using the IAR compiler ICCV850:
The behaviour might apply if the following condition is satisfied:
1. `#pragma pack(1)` or `#pragma pack(2)` is specified in the source code and compiler option `--allow_misaligned_data_access` is set.

4. When using the NEC Electronics compiler CA850:
The behaviour might apply if either of the following conditions is satisfied:
1. `#pragma pack(1)` or `#pragma pack(2)` is specified in the source code.
2. Compiler option `-Xpack=1` or `-Xpack=2` is set.

Third judgment: Check the software using the check tool

If the occurance of the described behaviour cannot be excluded by the above checks, continue judgement using a check tool prepared by NEC Electronics.

The check tool is available through NEC-EE TOOLWEB and can be downloaded from the following pages. Alternatively please contact your NEC Electronics sales representative:

GHS compiler:
Go to www.eu.necel.com/update and select the order code CPDW9X/NT-CDR-V850
or select the following link:
http://www.eu.necel.com/update?id=26

IAR compiler:
Go to www.eu.necel.com/update and select the order code EWV850-FULL-EE
or select the following link:
http://www.eu.necel.com/update?id=18

No.	
No. 2	MUL/MULU instruction precaution (Specification change notice)

Application of this behaviour to NEC Electronics embedded software products:

The NEC compiler CA850 does not generate code containing the mul or mulu instruction in which reg2 = reg3. Nevertheless, code containing mul or mulu in which reg3 = reg0 could be generated.

This behaviour does not apply to the following NEC Electronics real-time OSs and middleware products:

- Real-time OSs: RX850, RX850 Pro, RX850V4
- Middleware: GOFAST

For NEC Electronics products other than the above, contact an NEC Electronics sales representative or distributor. For third-party products, contact the vendor of the product.

(C) Valid Specification

Item	Date pulished	Document No.	Document Title
1	February 2004	U14559EJ3V1UM00	V850E1 Architecture (User's Manual)
2	April 2004	U15943EJ3V0UM00	V850ES Architecture Manual (User's Manual)

(D) Revision History

Item	Date pulished	Document No.	Comment
1	Feb 17, 2004	TPS-HE-B-2170	1st release.
2	November 2009	U20081EE1V0IF00	Added item 2.