NEC

Customer Notification

IE-703239-G1-EM1™

In-Circuit-Emulator

Operating Precautions

Target Device

V850ES/FE2
V850ES/FF2
V850ES/FG2
V850ES/FJ2

Document No. TPS-HE-B-2853
Date Published September 2004

© 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

No.	Outline		IE-703239-G1-EM1		
		Rev.	1.00	1.10	1.11
		Control Code[Note]	A	B	C
1	Operation at EVDD >= 4.0 V (Technical limitation)		✗	✓	✓
2	Operation at BVDD <= 5.5 V (Technical limitation)		✗	✓	✓
3	RC-Oscillation of the Subclock not supported (Technical limitation)		✗	✓	✓
4	Emulation of POC by voltage change not supported (Direction of use)		✗	✗	✗
5	Self-programming not supported (Direction of use)		✗	✗	✗
6	Oscillation stabilzation time after reset cannot be emulated (Direction of use)		✗	✗	✗
7	DMA control registers DCHC0 to DCHC3 cannot be displayed (Direction of use)		✗	✗	✗
8	On-chip debug mode release not supported (Direction of use)		✗	✗	✗
9	Internal reset factor (POC, LVI, WDT) cannot be masked (Direction of use)		✗	✗	✗
10	Port mode control registers (Technical limitation)		✗	✓	✓
11	TMP0 pin function (Technical limitation)		✗	✓	✓
12	Sub clock, code execution between flash macros (Technical limitation)		✗	✗	✓
13	ADC accuracy (Technical limitation)		✗	✗	✓
14	aFCAN transmission / reception (Technical limitation)		✗	✗	✓
15	Reset by WDT or external RESET (Technical limitation)		✗	✗	✓
16	WDT during break (Technical limitation)		✗	✗	✓

No.	Outline		IE-703239-G1-EM1		
		Rev.	1.00	1.10	1.11
		Control Code[Note]	A	B	C
17	16-bit timer M during break (Technical limitation)		✗	✗	✓
18	16-bit timer M compare register (Technical limitation)		✗	✗	✓
19	LVI detection voltage level (Technical limitation)		✗	✗	✓
20	Programmable clock mode register PCLM (Technical limitation)		✗	✗	✓
21	Access of UAnRX register during break (Specification change notice)		✗	✗	✗
22	Access of CBnRX register during break (Specification change notice)		✗	✗	✗
23	Access of CnRGPT register during break (Specification change notice)		✗	✗	✗
24	Access of CnTGPT register during break (Specification change notice)		✗	✗	✗
25	Access of CnGNCTRL register during break (Specification change notice)		✗	✗	✗
26	TMQn/TMPn external event counter function (Specification change notice)		✗	✗	✗
27	SLD instruction precaution (Specification change notice)		✗	✗	✗
28	aFCAN: Rx limitation (Technical limitation)		✗	✗	✗

✓ :Not applicable

✗ :Applicable

Note: The Control Code is indicated by the letter appearing at the 2nd position from the left in the serial number of the product. If the product has been upgraded, a label indicating the new version is attached to the product and the x in V-UP LEVEL x on this label indicates the new control code.

(B) Description of Operating Precautions

No. 1	Operation at EVDD >= 4.0 V (Technical limitation)

Details

When the debugger is connected to the emulator, the regulation of EVDD on the IE-703239-G1-EM1 is restricted to EVDD >= 4.0 V for operation frequencies lower than 20 MHz. Therefore Operating frequencies lower than 20 MHz should not be used when EVDD on the target circuit is < 4.0 V.

Workaround

When the debugger is connected to the emulator use EVDD >= 4.0 V or use an operaton frequency of 20 MHz.

No. 2	Operation at BVDD <= 5.5 V (Technical limitation)

Details

A TBVDD voltage higher than 5.5 V will damage the level shifters and may therefore not be applied.

Workaround

Operate the IE-703239-G1-EM1 at BVDD <= 5.5 V only.

No. 3	RC-Oscillation of the Subclock not supported (Technical limitation)

Details

RC-Oscillation of the Subclock is not supported.

Workaround

Use a crystal in the specified range for subclock operation (see user's manual).

No. 4	Emulation of POC by voltage change not supported (Direction of use)

Details

Emulation of POC (power-on clear) by voltage change is not supported.

Workaround

None.

No. 5	Self-programming not supported (Direction of use)

Details

Emulation of self-programming is not possible.

Workaround

None.

No. 6	Oscillation stabilzation time after reset cannot be emulated (Direction of use)

Details

Emulation of oscillation stabilization time after reset is not possible. However, after STOP mode release the oscillation stabilzation time will be emulated.

Workaround

None.

No. 7	DMA control registers DCHC0 to DCHC3 cannot be displayed (Direction of use)

Details

The DMA channel control registers DCHC0 to DCHC3 cannot be displayed in the debugger, since the status is of these registers is changed by reading.

Workaround

None.

No. 8	On-chip debug mode release not supported (Direction of use)

Details

Emulation of on-chip debug mode release at the time of external reset input is not possible.

Workaround

None.

No. 9	Internal reset factor (POC, LVI, WDT) cannot be masked (Direction of use)

Details

A reset which takes place according to an internal factor (POC, LVI, WDT) cannot be masked.

Workaround

None.

No. 10	Port mode control registers (Technical limitation)

Details

All the peripheral I/O registers become undefined if any of the following bits of the port mode control registers is set to 1:
- bits 2 and 3 of the PMC0 register
- bits 0 to 5 of the PMC3 register

Workaround

Do not set any of the following bits of the port mode control registers to 1:
- bits 2 and 3 of the PMC0 register
- bits 0 to 5 of the PMC3 register

No. 11	TMP0 pin function (Technical limitation)

Details

The alternate function of port pin P32 has changed (V850ES/FJ2: pin27; V850ES/FG2: pin27; V850ES/FF2: pin24; V850ES/FE2: pin24).
Before: P32/ASCKA0/TIP00/TOP00
After: P32/ASCKA0/TIP00/TOP00/TOP01
Therefore the setting of PMC3.PMC32 has changed accordingly.
Also the assignment for PFC3 and PFCE3L have changed:

PFCE32	PFC32	P32 pin control mode specification
0	0	ASCKA0 input
0	1	TOP01 output
1	0	TIP00 input
1	1	TOP00 output

Workaround

There is no workaround.

No. 12	Sub clock, code execution between flash macros (Technical limitation)

Details

When the CPU is operating in sub-clock mode and the main clock is stopped, the CPU will incorrectly recognize the first 4 bytes as NOP instructions when it transitions to another flash macro. However in the following cases this behavior does not occur:
- When returning from another macro to the macro in which the main clock was stopped.
- On transition to a macro where the above behavior already occured once.

Workaround

There is no workaround.

No. 13	ADC accuracy (Technical limitation)

Details

The ADC has an offset of approx. 30 mV.

Workaround

There is no workaround. The offset has been changed to the specified value at control code ’C’.

No. 14	aFCAN transmission / reception (Technical limitation)

Details

The aFCAN macro will under certain timing conditions accompanied by a particular configuration of the message buffers and a specific operational usage not operate as expected. Different behaviors have to be considered. The description of the particular unexpected behavior is organized by the buffer Type: TX and RX, the frame format (extended or standard identifier), and the buffer number.

TX-Buffer Behavior

This section describes all unexpected behaviors linked to the configuration of transmit buffers. Configurations not listed are of no concern and can be used without restrictions.

- Buffer #0 or Buffer #0 and #1 are configured as TX-buffer

When using message buffer #0 as a TX-buffer, the message requested for transmission of this buffer may not be sent at the next possible timing. This behavior is caused when the internal scan for new transmission requests reached buffer #0 and at the same time the transmission of a previously sent message ends. Instead of sending the message object from buffer #0, the aFCAN attempts to send the contents of message buffer #1. In case the TRQ of buffer #1 is set, the message from buffer #1 is sent followed by the transmission of the message in buffer #0. This resembles an inner priority inversion. In case the TRQ of buffer #1 is not set or buffer #1 is not a TX-buffer, the contents of buffer #0 are sent whenever any of the other TRQ-bits in the aFCAN are set or cleared, or when a receive operation is started; i.e. when the next bus activity occurs. This behavior is valid for both frame types, extended or standard identifier format.

- Buffer #1 - #31 are configured as TX-buffer

There are two configurations that lead to the same, unexpected behavior. In the first configuration Buffer #1 through #31 are set up as normal TX-buffers, and in the second configuration buffer #0 through #7 are operated in ABT-mode (automatic block transmission) and the remaining buffers (#8 - 31) are configured as normal TX-buffers. The unexpected behavior occurs as well if only a subset of buffers are configured as normal TX-buffers. When using any message buffer #n in the range of buffer #1 through #31 with extended identifier, an inner priority inversion can be encountered. In that case the message from buffer #n+1 is sent in advance of the message in buffer #n even though the priority of the identifier in buffer #n is higher. This behavior is seen when the internal transmit search algorithms of the aFCAN processing buffer #n meets the start of a transmission, the end of a frame on the bus (RX or TX) that needs to end with an error frame, or the event of transmission request by the CPU or by the ABTmode. In case the ABT-mode is active, the unexpected behavior does only apply for messages in buffer #8 through #31. This behavior applies only if extended identifiers are in use. Applications exclusively using standard Identifiers do not suffer any limitation of this kind.

No. 14	aFCAN transmission / reception (contd.) (Technical limitation)

RX-Buffer Behavior

There are several configurations for the receive buffers that can cause different unexpected behaviors.

- Buffer #0 is configured as RX-buffer with EXT ID handling

There are two kinds of unexpected behavior in this configuration. The first one requires that a newly received message would normally be received in buffer #0. When in this case a transmission request (TRQ) by the host processor is submitted at the time where buffer #0 is scanned by the internal RX-Search algorithm for newly received messages, the storage of the message for buffer #0 will not occur. The message will be stored nowhere.
In a second scenario, the newly received message would under normal conditions not be stored in buffer #0. When in this case a TRQ for buffer #i by the host processor is submitted, which again happens at the same time where the acceptance filtering for the newly received message is active, the message can be stored in buffer #0. The unexpected storage of the message in buffer #0 does only occur when the identifier bits ID15-0 of the received message coincidentally match the values in MCONF/MDLC of the TX-buffer #i. The behavior applies only for extended format frames. Standard format frames can be used without suffering this limitation.

- Buffer #1 - #31 are configured as RX-buffer with EXT ID handling

When a newly received message does not match the acceptance filter criteria for buffer #n (n = 1...31), the following timing can lead to an unexpected behavior. When additionally to that condition a transmission request is set for buffer #i at the time where buffer #n is scanned by the internal RX-Search algorithm, the storage of the message can be performed in buffer #n+1 although this buffer does not match with the acceptance filter criteria for the received message. The unexpected storage of the message in buffer #n+1 does only occur when the identifier bits ID15-0 of the received message coincidentally match the values in MCONF/MDLC of the Txbuffer #i. This unexpected behavior is also possible if only a subset of buffers in the range of #1 through #31 are configured as RX-buffers. The behavior applies only for extended format frames. Standard format frames can be used without suffering this limitation.

Workaround

The application needs to abstain from the usage of message buffer #0 or configure buffer #0 as a receive buffer with standard identifier handling.
The ABT-mode is not affected from this limitation. Thus buffer #0 can be configured as a transmit buffer in that mode.
Do not use extended identifiers for transmit and receive objects. If the application needs to process extended identifiers, the host processor must not issue a transmission request anytime the CAN-bus is busy. For this case the application can monitor the bus status and issue a transmission request right after the bus idle state was detected. Then the TRQ will be submitted in an uncritical point of time. This is even valid when the TRQ is submitted during the first 19 bits of the extended identifier.

No. 15	Reset by WDT or external RESET (Technical limitation)

Details

Access to the I/O registers becomes illegal when a reset by the watchdog timer or an external reset occurs. Read or Wrire acces may not be executed correctly.

Workaround

The expected status can be restored by by executing a reset in the debugger.

No. 16	Watchdog timer during break (Technical limitation)

Details

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

Conditions that need to be fulfilled so that the above behaviour occurs:
(a) The main clock or subclock is selected as the clock source of the watchdog timer and
(b) The ring oscillator is stopped (RSTOP flag = 1).

Workaround

As a workaround to prevent the above behaviour do not stop the ring oscillator clock.

No. 17	Timer M during break (Technical limitation)

Details

When a break occurs while the following conditions (a) and (b) are both fulfilled, timer M does not stop even if the peripheral break function has been set to 'break'.

(a) INTWT, Ring oscillator clock ($f_R/8$) or subclock is selected as the clock source for timer M.
(b) The main clock is stopped by setting the MCK flag.

(Note: The peripheral break function is not supported by the debugger ID850 V2.51.)

Workaround

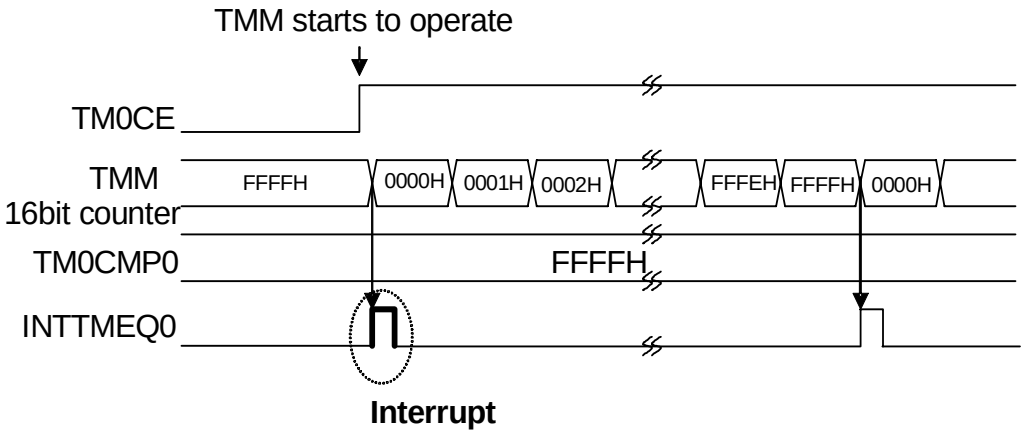
Implement one of the below workarounds to stop timer M during a break using the peripheral break function:

(a) Use the main clock (f_{XX}, $f_{XX}/2$, $f_{XX}/4$, $f_{XX}/64$, $f_{XX}/512$) as the source clock for timer M.
(b) Do not stop the main clock oscillation.

No. 18	Timer M compare interrupt (Technical limitation)

Details

An unexpected interrupt occurs after activation of timer M when the compare register TM0CMP0 contains the value 0xFFFF.



Workaround

Do not set TM0CMP0 to 0xFFFF.

No. 19	LVI detection voltage level (Technical limitation)

Details

The low voltage indicator detection level is higher than specified.
The specified value is 4.2V +/- 0.2V or 4.4V +/- 0.2V.
The actual value is 5.1V

Workaround

There is no workaround. The detection level voltage has been set to the specified value for control code 'C' version.

No. 20	Programmable clock mode register PCLM (Technical limitation)

Details

The divider of the processor clock output is not connected as specified.

Specified options:

PCK1	PCK0	PCL output select
0	0	fxx / 2
0	1	fxx / 4
1	0	fxx / 8
1	1	fxx / 16

Actual options:

PCK1	PCK0	PCL output select
0	0	fxx
0	1	fxx / 2
1	0	fxx / 4
1	1	fxx / 8

Workaround

There is no workaround. Settings have been changed to the specified values at control code ’C’.

No. 21	Access of UAnRX register during break (Specification change notice)

Details

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

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

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

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

Note: Including breaks by the RAM monitor function or DMM function. However the realtime RAM monitor function does not cause this behaviour since it does not set breaks.

Remark: An overrun error also occurs when the UART receives data multiple times during a break (This complies with the specification of the emulator).

Workaround

(a) Do not display the UAnRX register in the I/O register window.
(b) Set a hardware break when setting a break immediately after reading the UAnRX register
(c) There is no workaround.

No. 22	Access of CBnRX register during break (Specification change notice)

Details

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

(a) If a software break occurs immediately after reading the CSIBn receive register (CBnRX).
(b) If a DMA transfer from the CSIBn receive data register (CBnRX) is performed during a break**NOTE**. As a result the communication stops or the DMA controller stops.

Note: Including breaks by the RAM monitor function or DMM function. However the realtime RAM monitor function does not cause this behaviour since it does not set breaks.

Workaround

(a) Set a hardware break when setting a break immediately after reading the CBnRX register.
(b) There is no workaround.

No. 23	Access of CnRGPT register during break (Specification change notice)

Details

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

(a) If a software break occurs immediately after reading the CANn module receive history list register (CnRGPT)
(b) If a DMA transfer from the CANn module receive history list register (CnRGPT) is performed during a break**NOTE**.

Note: Including breaks by the RAM monitor function or DMM function. However the realtime RAM monitor function does not cause this behaviour since it does not set breaks.

Workaround

(a) Set a hardware break when setting a break immediately after reading the CnRGPT register.
(b) There is no workaround.

No. 24	Access of CnTGPT register during break (Specification change notice)

Details

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

(a) If a software break occurs immediately after reading the CANn module transmit history list register (CnTGPT).
(b) If a DMA transfer from the CANn module transmit history list register (CnTGPT) is performed during a break**NOTE**.

Note: Including breaks by the RAM monitor function or DMM function. However the realtime RAM monitor function does not cause this behaviour since it does not set breaks.

Workaround

(a) Set a hardware break when setting a break immediately after reading the CnTGPT register.
(b) There is no workaround.

No. 25	Access of CnGNCTRL register during a break (Specification change notice)

Details

When a register access is performed in the following sequence, an unexpected forcible shutdown may occur after the sequence is complete.

Sequence :
(1) The EFSD bit of the CANn module control register (CnGNCTRL) is set.
(2) The I/O register**NOTE** is accessed.
(3) The GOM bit of the CANn mode control register (CnGNCTRL) is cleared.

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

The conditions under which a forcible shutdown takes place are shown below:
(a) If a break occurs immediately after the I/O register access in (2) occurs.
(b) If a break by the RAM monitor function or the DMM function occurs immediately after the I/O register access in (2) occurs.
(c) Stepwise execution is performed for the I/O register access in (2).

Workaround

Be sure to set the EFSD bit and clear the GOM bit successively when executing a forcible shutdown. Do not perform a register access in the above sequence when not performing a forcible shutdown.

No. 26	TMQn/TMPn external event counter function (Specification change notice)

Details

When the external event counter mode is used and the compare register of timer TMQ or TMP is set to 0x0000 an interrupt occurs after the overflow of the timer (change from 0xFFFF to 0x0000).

Workaround

As a software workaround it is necessary to avoid the setting of 0x0000 to the timer compare register of the timers TMQ or TMP.

No. 27	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 source 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. 27	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. 27	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. 28	aFCAN: Rx limitation (Technical limitation)

Details

RX Limitation

The aFCAN macro may store an incoming message although this message was interrupted by a bus error frame. Thus, the incomplete reception causes that a message buffer is updated with old or incorrect data or that the message is even stored at an incorrect location.

This unexpected behaviour affords that the bus error occurs in a certain relation to the currently present message on the bus. The critical time window starts at the sample point of the LSB of the DLC-field and lasts for the duration of an internal process in the aFCAN macro (RX-search). This time window usually lasts for a few bit times only. The actual length depends on the clock supply for the AFCAN, the CPU accesses during this period, the baud rate and the number of message buffers of the particular AFCAN macro.
In this time window the RX-search evaluates the received identifier of the current message. When the bus error is detected within this window and when the RX-search has just scanned buffer #n for reception and found it is matching, the message will unexpectedly be treated as a received message. As the time window is limited as described above, only a stuff bit error occurring right in this window can cause this behaviour.
There are two types of unexpected behavior for the RX limitation depending on the presence of pending transmission request (TRQ) for any other message buffer.

1. Behaviour at pending TRQ (TRQi = 1)

When the host processor has already submitted a transmit request (TRQ) for at least one buffer, the unexpected reception of the message will take place into the message buffer found by internal RX-search. This is the correct location to store the message i.e. the acceptance filter criteria are correctly fulfilled. However the data part will be updated with the contents of the shift register of the CAN protocol core. As this register is immediately stopped at detection of the bus error, the data provided to the message buffer can not be interpreted by the host processor.

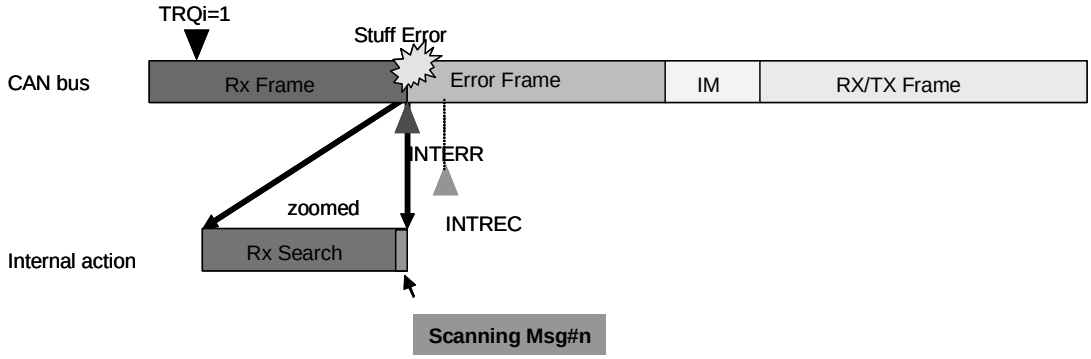


Figure 1: Behavior at pending TRQ

As during a regular reception, the RX-interrupt (if enabled) is generated and the application processes the message object.

No. 28	aFCAN: Rx limitation (Technical limitation)

2. Behavior without pending TRQ (TRQi = 0)
In case the host processor has not submitted a transmit request (TRQ) for any buffer before the detection of the bus error but submits TRQ = 1 after that point in time (see figure below) before the re-transmission of the message interrupted by the stuff bit error started, the unexpected reception of the message will take place.

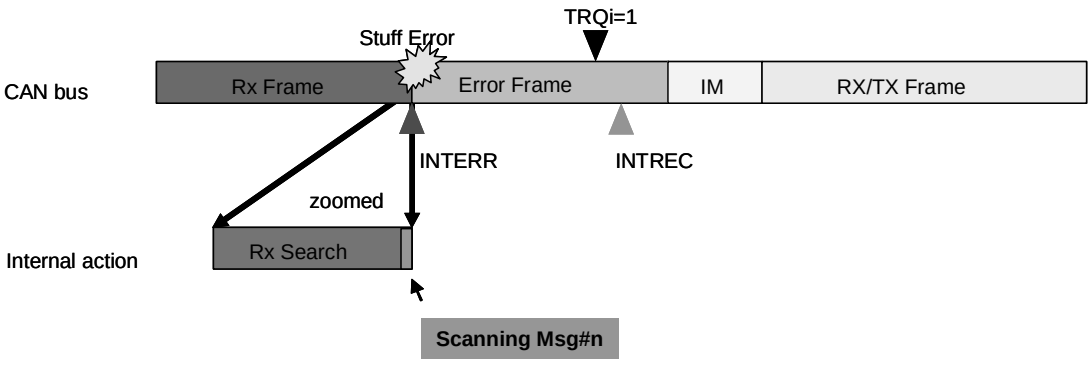


Figure 2: Behavior without pending TRQ

The unexpected storage of the message is issued in the particular message buffer that matches the acceptance filter criteria at the time where the bus error is detected (as described in 1.) or the message buffer #0 is overwritten independently of its configuration.

Impact on application
In typical applications the RX-limitation will lead to transiently incorrect data. In the vast majority of cases the message interrupted by a bus error is repeated by the transmitter right away. Then the application receives correct data shortly after the unexpected reception.
In scenarios where the message buffer #0 is overwritten, the impact for the application depends on the usage of that buffer. If it is configured as a receive buffer, the application receives a message at an unexpected location and will interpret the data to belong to the identifier originally programmed for that buffer. The message buffer #0 needs to be re-configured in order to receive the originally intended message object again.
In case of a transmit message buffer the unexpected storage may falsify a transmit object; i.e. when the unexpected behavior occurs after preparation of the message data but before the actual start of transmission. This scenario is even less likely than the scenario described in 1, which itself has a low probability. However the transmission of a falsified message can lead to repetitive transmission attempts when the original provider of that message (identifier) tries to send its message at the same time. Then the messages most likely will differ in their data part and a bit error is detected. This repetition resumes until one of the nodes enters error passive or bus off state. Then the situation is resolved as all pending TRQ are send with delay or are cancelled (in case of bus off state).

No. 28	aFCAN: Rx limitation (Technical limitation)

Workaround

NEC will update the affected products.
NEC does not recommend a S/W workaround as first choice as it is fairly complex. On the one hand it is based on the control of submitting transmission requests only when the bus is idle. On the other hand a less complex algorithm can be used which does not prevent the unexpected reception but detects it safely and discards the unexpected reception in the CAN S/W driver. Any of these algorithms require that message buffer #0 is not used or that a 'dummy' TRQ in an unused buffer is set. This prevents behaviors as described in 2.

(C) Valid Specification

Item	Date pulished	Document No.	Document Title
1	April 1, 2003	SUD-FT-03-0102	IE-703239-G1-EM1 (Preliminary User's Manual)
2	September 2003	U16480EJ1V0UD00	V850ES/FE2, V850ES/FF2, V850ES/FG2, V850ES/FJ2 Hardware (Preliminary User's Manual)
3	April 2004	U15943EJ3V0UM00	V850ES, 32-Bit Microprocessor Core Architecture (User's Manual)

(D) Revision History

Item	Date pulished	Document No.	Comment
1	April 29, 2003	TPS-HE-B-2850	First release
2	May 14, 2003	TPS-HE-B-2851	Addition of control code B; Addition of customer notifications no. 4 to 9
3	April 26, 2004	TPS-HE-B-2852	Addition of control code C Addition of items 10 to 27
4	September 1, 2004	TPS-HE-B-2853	Addition of item 28