M253 Series Errata Sheet

Errata Sheet for 32-bit NuMicro® Family

Document Information

Abstract	This errata sheet describes the functional problem known at the release date of this document.
Apply to	M253 Series.

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro microcontroller and microprocessor based system design. Nuvoton assumes no responsibility for errors or omissions.
All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.
www.nuvoton.com

Table of Contents

1. Overview

Functional Problem	Description
CAN FD register access error	CPU reads an incorrect value from CAN FD register while the message RAM is accessed by CAN controller. The incorrect value is 0, but the correct value should not be 0.
CAN FD sends an error message while receiving a new message	If a sent message and a received message happens at the same time, the sent message data will be overwritten with the received message data during the CAN FD controller reading data from the CAN FD SRAM.

2. Functional Problems

2.1 CAN FD register access error

Description:

CPU reads an incorrect value from CANFD register while the message RAM under is accessed by CAN controller. The incorrect value is 0, but the correct value should not be 0.

Problem:

When the CAN FD controller receives a message from CAN bus, the CAN FD controller will read or write CAN FD SRAM during this read/write cycle. CPU will always read "0" data from CAN FD register.

Workaround:

The BSP adds a function to wrap reading the CANFD register. If the read value is 0, the function will continue to read 48 counts.

Note: This function supports M253_Series_BSP_CMSIS_V3.00.005 and later versions.

```
#define CANFD_READ_REG_TIMEOUT    48                /* CANFD read register time-out count
*/

uint32_t CANFD_ReadReg(__I uint32_t *pu32RegAddr)
{
    uint32_t u32ReadReg;
    uint32_t u32TimeOutCnt = CANFD_READ_REG_TIMEOUT;
    u32ReadReg = 0UL;

    do
    {
        u32ReadReg = inpw(pu32RegAddr);

        if (--u32TimeOutCnt == 0UL)
        {
            break;
        }
    } while (u32ReadReg == 0UL);

    return u32ReadReg;
}
```

2.2 CAN FD sends an error message while receiving a new message

Description:

If a sent message and a received message happens at the same time, the sent message data will be overwritten with the received message data during the CAN FD controller reading data from the CAN FD SRAM.

Problem:

The CAN FD controller will access CAN SRAM after Tx message is triggered or Rx message data is received. If the Tx message and Rx message happens at the same time, the Tx message data will be overwritten by Rx message data during CAN FD controller reading data from CAN FD SRAM. As a result, the CAN FD controller will follow incorrect data to transmit.

Workaround:

The BSP driver has been revised to monitor bus communication status while CPU requests to send a new message. The driver will write data to a message buffer while bus is in idle state to avoid the occurrence of CANFD internal message RAM buffer having read-while-write condition.

Note: This function supports M253_Series_BSP_CMSIS_V3.00.005 and later versions.

```
/* CAN FD communication state.*/
typedef enum
{
    eCANFD_SYNC         = 0,
    eCANFD_IDLE         = 1,
    eCANFD_RECEIVER     = 2,
    eCANFD_TRANSMITTER  = 3
} E_CANFD_COMMUNICATION_STATE;

/* Get Monitors the Module’s CAN Communication State Flag */
#define CANFD_GET_COMMUNICATION_STATE(canfd)    (((canfd)->PSR  & CANFD_PSR_ACT_Msk) >>
CANFD_PSR_ACT_Pos)
```

The Tx trigger timing should be controlled when the CAN bus is in idle state.

```
while(1)
{
__disable_irq();
if(CANFD_GET_COMMUNICATION_STATE (CANFD0) == eCANFD_IDLE)
{
    CANFD_TransmitTxMsg(CANFD0, 0, psTxMsg);
}
__enable_irq();
}
```

Note1 : To avoid long interrupt delay, the CANFD interrupt priority should be set as the highest and CANFD_TransmitTxMsg() is executed during disable interrupt (__disable_irq()).

Note2 : This function supports M253_Series_BSP_CMSIS_V3.00.005 and later versions.

3. Revision History

Date	Revision	Description
2022.08.27	1.00	Initial version.

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.