M051BN Series Errata Sheet

Errata Sheet for 32-bit NuMicro® Family

Rev. 1.01 — Apr. 16, 2013

Document Information

Abstract	This errata sheet describes the functional problems known at the release date of this document.
Apply to	M051BN Series.

© 2013 Nuvoton Technology Corp.

Table of Contents

© 2013 Nuvoton Technology Corp.

1 Overview

Functional Problem	Description
ADC External Trigger	When ADC external trigger function is enabled, it may cause an additional ADC start trigger.
ADC VALID Bit	The VALID bit of ADC data register will not be set to 1 if program uninterruptedly reads data register.
ADST Re-trigger	After ADC finishes a conversion and clears the ADST bit, software must wait one ADC clock cycle before writing ADST bit to 1 again.
ADC Continuous/Single-Cycle Scan Mode	If the ADST bit is cleared to 0 in Continuous or Single-Cycle Scan mode while ADC is doing conversion, the conversion result of the smallest enabled ADC channel may be incorrect.
ADC DMOF Limitation of Digital Compare Function	When Differential Input mode is enabled, the ADC digital compare function does not support the conversion result in signed format.
ADC DMOF Limitation of Burst Mode	When Differential Input mode is enabled, ADC Burst mode does not support the conversion result in signed format.
BOD Low Power Mode Enable	The Brown-out low power control bit will not be reset by BOD reset.
BOD Reload	When system recovers from BOD reset state, BOD settings in user configuration will be reloaded.
BOD Reset	If WDT time-out reset occurs in BOD reset state, system will exit BOD reset state and start booting for 960 us.
GPIO Wake-up Trigger Level	GPIO wake-up function failed if selected pin is already in active state before system enters Power-down mode.
GPIO Wake-up Transient	GPIO may be woken up by I/O transient no matter it is configured as rising edge or falling edge wake-up.
Power-down Enable Bit Cleared	The Power-down enable bit (PWR_DOWN_EN) will be cleared if a GPIO interrupt occurs before WFI is executed.
PWM Capture Timer	When PWM capture input channel is enabled and PWM capture

© 2013 Nuvoton Technology Corp.

	input channel has a transition before PWM timer starts to count, PWM timer will be halted.
RS-485 Module Clock Setting	When UART module clock frequency is higher than system clock, UART may receive wrong data in RS-485 mode.
Timer Module Reset	Timer 1 register cannot be accessed when Timer 0 is doing module reset. Timer 0 register cannot be accessed when Timer 1 is doing module reset. Timer 3 register cannot be accessed when Timer 2 is doing module reset. Timer 2 register cannot be accessed when Timer 3 is doing module reset.
UART Error Flags	If one of the error flags (BIF/FEF/PEF) is set, writing '1' cannot clear the flag.
UART Buffer Error Flag	When one of the error flags (BIF/FEF/PEF) is set, the BUF_ERR_IF flag will be set at the same time.
WDT Wake-up Flag	When software wants to write 1 to clear WTWKF flag, the internal WDT wake-up signal will always be kept one WDT module clock period and then transit to 0 even if WTWKF flag is read as 1 by CPU.

© 2013 Nuvoton Technology Corp.

2 Functional Problems

2.1 ADC External Trigger

Description:

If the external trigger function is enabled, user can trigger the A/D conversion by STADC pin. The trigger conditions such as low-level trigger, high-level trigger, falling-edge trigger and rising-edge trigger are determined by TRGCOND[1:0] in ADCR register.

Problem:

The ADC controller will generate an additional conversion with the programming sequence listed below.

1. Force the STADC pin at 0V.
2. Enable ADC clock (APBCLK[28]).
3. Complete the following operations within 4 system clock cycles.
   - Enable STADC pin external trigger with falling-edge condition.
   - Enable ADC analog circuit (ADCR[0]).

Workaround:

After ADC module clock is enabled, it is necessary to delay at least 4 system clocks before the external trigger function or ADC analog circuit is enabled.

2.2 ADC VALID Bit

Description:

When an ADC conversion is completed, the result is saved to the data register of the corresponding channel, and VALID bit of both data and status register are set to 1. Reading data register will clear the VALID bit.

Problem:

The VALID bit of ADC data register is set by hardware when an ADC data conversion is ready and is cleared by CPU reading ADC data register. However, if hardware sets VALID bit and

© 2013 Nuvoton Technology Corp.

software reads the VALID bit at the same time, the VALID bit will not be set, which causes the VALID bit probably always 0. For example, if user enables ADC channel 0 to convert the input signal and poll channel 0 data register to check whether the data is valid, it may never get data valid because VALID bit is suppressed by clear read.

Workaround:

User should read the ADF bit or the VALID bit of ADC status register instead of reading the VALID bit of ADC data register to check if a conversion is finished or not.

2.3 ADST Re-trigger

Description:

In Single mode and Single Cycle Scan mode, the ADST bit will be automatically cleared to 0 once a conversion is completed. In Continuous Scan mode, the conversion is stopped if program clears the ADST bit to 0. Whenever ADST is 0, program can set ADST to 1 to re-trigger ADC conversion.

Problem:

When the ADST bit is cleared to 0, if ADST bit is immediately set to 1, ADC can't work correctly.

Workaround:

When the ADST bit is cleared to 0, it is necessary to delay at least one ADC clock cycle before writing the ADST bit to 1 again. For example, if ADC clock rate is 10 MHz, the delay loop of the following example must be greater than 0.1us (1/10 MHz).

```
ADC->ADCHER = 0xFF;              // Enable all ADC channels
ADC->ADCR = 9;                   // Single Cycle Scan mode
ADC->ADCR_BITS.ADST = 1;         // Trigger ADC
while(ADC->ADSR_BITS.ADF==0);    // Wait ADC finishes the conversion
ADC->ADSR = 0x1;                 // Clear the ADF bit
for(u32DelayCount=0; u32DelayCount<0x1000; u32DelayCount++);    // Delay loop
ADC->ADCR BITS.ADST = 1;         // Re-trigger ADC
```

© 2013 Nuvoton Technology Corp.

2.4 ADC Continuous/Single-Cycle Scan Mode

Description:

If the ADST bit is cleared to 0 to stop Continuous or Single-Cycle Scan mode, ADC controller will finish the current conversion and write the result to data register of the corresponding ADC channel.

Problem:

If the ADST bit is arbitrarily cleared to 0 in Continuous or Single-Cycle Scan mode, the conversion result of the smallest enabled ADC channel may be incorrect. The following example shows a case that all ADC channels are enabled in Continuous Scan mode. Then, the ADST bit is cleared to stop ADC conversion and causes the result of ADC channel 0 to be the result of any enabled channel.

```
ADC->ADCHER = 0xFF;             // Enable all ADC channels
ADC->ADCR = 0xD;                // Continuous scan mode
ADC->ADCR_BITS.ADST = 1;        // Trigger ADC
…
ADC->ADCR BITS.ADST = 0;        // Stop ADC when ADC converting
…
// The ADDR[0] may be the result of any  ADC channel.
u32AdcData = ADC->ADDR[0];
```

Workaround:

If more than one ADC channels are enabled in Continuous or Single-Cycle Scan mode, program should read the valid conversion result before clearing the ADST bit. As shown in the following example, program needs to read available channel data before stopping ADC.

```
ADC->ADCHER = 0xFF;             // Enable all ADC channels
ADC->ADCR = 0xD;                // Continuous scan mode
ADC->ADCR_BITS.ADST = 1;        // Trigger ADC
While(ADC->ADSR_BITS.ADF==0); // Wait ADC finishes the conversion
for(u32DataCount=0; u32DataCount<8; u32DataCount++)
{
    u32AdcData = ADC->ADDR[u32DataCount]; // Read the valid data
    …
}
ADC->ADCR BITS.ADST = 0;        // Stop ADC
```

© 2013 Nuvoton Technology Corp.

2.5 ADC DMOF Limitation of Digital Compare Function

Description:

When Differential Input mode is selected, the conversion result can be unsigned format or signed format (2's complement). If the DMOF bit is set to 1, the signed format is selected; otherwise, the unsigned format is selected. In ADC digital compare function, ADC controller will compare the conversion result with CMPD[11:0] setting. If the DMOF bit is cleared to 0, ADC controller compares CMPD[11:0] with conversion result in unsigned format, i.e., the CMPD[11:0] should be an unsigned value. If the DMOF bit is set to 1, ADC controller will compares CMPD[11:0] with conversion result in signed format, i.e., the CMPD[11:0] should be a signed value.

Problem:

When Differential Input mode is enabled, the digital compare function does not support signed format, i.e., ADC controller always treats the CMPD[11:0] as an unsigned value.

	ADC Result Format	CMPD[11:0] Format
DMOF = 0	Unsigned	Unsigned
DMOF = 1	Signed	Unsigned

The above table shows the data format of ADC Result and CMPD[11:0] in Differential Input mode. When DMOF = 0, both ADC Result and CMPD[11:0] are unsigned values. Therefore, it is ok to compare their value. When DMOF = 1, the ADC result format is signed but CMPD[11:0] format is unsigned. Therefore, the compare result will be wrong.

Workaround:

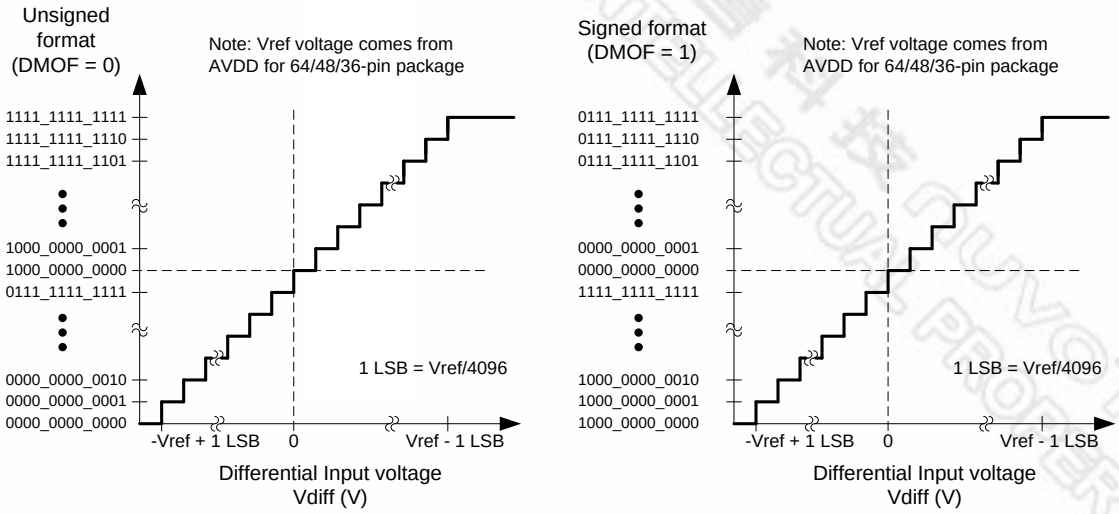
When Differential Input mode is enabled and the DMOF bit is set to 1, the value of CMPD[11:0] needs to be set in unsigned format. The formula to transfer signed value to unsigned value is:

Unsigned output code = (Vin+Vref)*4096/(2*Vref)

Where Vin is the input voltage; Vref is reference voltage of ADC.

© 2013 Nuvoton Technology Corp.

For example, if the DMOF bit is set to 1 and user wants to monitor 1.5V conversion result in 3V application, user can set CMPD[11:0] to (1.5v+3v)*4096/(2*3v)=0xC00 instead of 0x400. Refer to the following diagram for the mapping between signed format and unsigned format. For example, number 0 in unsigned format is mapping to 0x800, number 1 in unsigned format is mapping to 0x801, and so on. The following figure shows the mapping between signed/unsigned format.



2.6 ADC DMOF Limitation of Burst Mode

Description:

If Differential Input mode is enabled, the conversion result can be expressed in unsigned format or signed format (2's complement). If DMOF is set to 1, the conversion result is expressed in signed format; otherwise, the conversion result is expressed in unsigned format.

Problem:

When enabling Differential Input mode, ADC Burst mode does not support the conversion result with signed format. The conversion result can only be expressed in unsigned format.

Workaround:

When both Differential Input mode and Burst mode are enabled, the DMOF bit must be cleared to 0, i.e., user must use unsigned format in this case.

© 2013 Nuvoton Technology Corp.

2.7 BOD Low Power Mode Enable

Description:

BOD (Brown-out Detection) in Low Power mode will be reset to Normal mode when reset occurred. In BOD Normal mode, the BOD response time is faster than that in BOD Low Power mode. The maximum response time of BOD detection is 100ms in Low Power mode. When BOD Low Power mode disabled, the maximum response time of BOD detection is 1/10 KHz (100us).

Problem:

When BOD works in Low Power mode, it will not be reset to Normal mode when BOD reset.

Workaround:

If the BOD reset flag, RSTS_BOD(RSTSRC[4]), is set after BOD reset, program can disable low power mode function to speed up the response time of BOD detection.

2.8 BOD Reload

Description:

When system reboots from BOD reset state, BOD settings will not be reset.

Problem:

When system recovers from BOD reset state, BOD settings in user configuration (Brown-out enable, Brown-out reset enable and Brown-out detector threshold voltage) will be reloaded. This causes the settings of BOD control registers to be reloaded as BOD settings in user configuration.

Workaround:

Do not modify BOD_OUT(BODCR[6]), BOD_VL(BODCR[2:1]) and BOD_EN(BODCR[0]) to keep consistent with BOD settings in user configuration. If it is necessary to change BOD settings, user should modify BOD settings in user configuration and reset system.

© 2013 Nuvoton Technology Corp.

2.9 BOD Reset

Description:

When BOD detection and BOD reset are enabled, the system will be held in BOD reset state until the supply voltage comes back to BOD detection level.

Problem:

When BOD is detected and BOD reset is enabled, the system will be held in BOD reset state. However, in BOD reset state, WDT (watchdog timer) still works if it is enabled. If WDT time-out reset occurs in BOD reset state, the system will exit BOD reset state and start booting for 960 us. After 960 us from booting, the system will go back to BOD reset state.

Workaround:

Before entering BOD reset state, user can disable watchdog timer to prevent WDT from generating time-out event. As shown in the following example, user enables BOD Interrupt mode and IRQ0 for BOD interrupt in main function. User can disable watchdog timer in BOD interrupt handler and then set BOD Reset mode. After BOD Reset mode is set, system will enter BOD reset state immediately.

```
Void main(void)
{
    SYS->BODCR &= ~SYS_BODCR_BOD_RSTEN_Msk; // Set BOD interrupt mode - BODCR[3]
    NVIC_EnableIRQ(BOD_IRQn);   //Enable IRQ0 for BOD interrupt
}

void BOD_IRQHandler(void)
{
    WDT->WTCR = WDT->WTCR & ~WDT_WTCR_WTE_Msk;  //Disable watchdog timer - WTCR[7]
    SYS->BODCR |= SYS_BODCR_BOD_RSTEN_Msk;  //Set BOD reset mode - BODCR[3]
    /* System will enter to BOD reset state */
    ...
}
```

© 2013 Nuvoton Technology Corp.

2.10 GPIO Wake-up Trigger Level

Description:

All GPIO interrupts can be used to wake up CPU. The types of GPIO interrupt could be triggered include rising-edge, falling-edge, high-level or low-level.

Problem:

System cannot be woken up by GPIO edge-trigger interrupt while the corresponding GPIO pin status is already at active edge-trigger level before entering Power-down mode.

For example, If a GPIO pin status is at low level before entering Power-down mode, system cannot be woken up by this GPIO pin when it is set as low edge-trigger interrupt.

Similarly, if the status of a GPIO pin is at high level before entering Power-down mode, system cannot be woken up by this GPIO pin when it is set as high edge-trigger interrupt.

Workaround:

Program should make sure the input I/O status used to wake-up system is stable and does not match with the interrupt active state before system enters Power-down mode.

For example, if the interrupt is set as rising-edge triggered, user should make sure the I/O status of selected pin is at low level before entering Power-down mode; if interrupt is set as falling-edge triggered, user should make sure the I/O status of selected pin is at high level before entering Power-down mode.

2.11 GPIO Wake-up Transient

Description:

System cannot be woken up by GPIO edge-trigger interrupt while I/O pin status is matched with the interrupt active state before system enters Power-down mode. For example, if an I/O state is low before entering Power-down and the I/O is set to be falling edge triggered; the I/O should not be able to wake system up.

Problem:

The transient of some I/O pins will induce internal glitch, which causes system to be woken up by edge-trigger interrupt no matter the I/O pin status is matched with interrupt active state or not before system enters Power-down mode.

© 2013 Nuvoton Technology Corp.

Workaround:

To successfully wake system up, user needs to set the I/O state before entering Power-down mode. In other words, if user wants to wake system up by rising-edge trigger, I/O state must be set to low when Power-down. If user wants to wake system up by falling-edge trigger, user should make sure the I/O state is at high in Power-down mode.

If user does not want system to be woken up by GPIO, the GPIO interrupt should be disabled.

2.12 Power-down Enable Bit

Description:

System will enter Power-down mode if WFI instruction is executed with PWR_DOWN_EN = 1 (PWRCON[7]) and SLEEPDEEP = 1 (SCR[2]). The PWR_DOWN_EN bit will be automatically cleared after system wakes up.

Problem:

The Power-down enable bit - PWR_DOWN_EN (PWRCON[7]) may be cleared if any GPIO interrupt or specific wake-up source interrupt occurred before WFI is executed. This causes system fails to enter Power-down mode.

Workaround:

The Power-down enable bit - PWR_DOWN_EN(PWRCON[7]) needs to be enabled again in GPIO interrupt handler and specific wake-up interrupt handler to make sure this bit is enabled. For example, if system has WDT time-out interrupt and GPIO PA.0 interrupt wake-up sources, program should enable PWR_DOWN_EN bit in both WDT_IRQn and GPAB_IRQn handler.

```
void GPAB_IRQHandler(void)
{
    GPIOA->ISRC |= GPIOA->ISRC; // Clear interrupt source flag
    SYSCLK->PWRCON |= SYSCLK_PWRCON_PWR_DOWN_EN_Msk; // Enable PWR_DOWN_EN bit
}

void WDT_IRQHandler(void)
{
    // Clear WDT time-out interrupt and wake-up flag
    WDT->WTCR |= (WDT_WTCR_WTIF_Msk | WDT_WTCR_WTWKF_Msk);
    SYSCLK->PWRCON |= SYSCLK_PWRCON_PWR_DOWN_EN_Msk; // Enable PWR_DOWN_EN bit
```

© 2013 Nuvoton Technology Corp.

```
}
```

2.13 PWM Timer Capture

Description:

When PWM timer is enabled in capture mode and PWM capture input channel has a transition (rising/falling), the PWM counter value will be latched into CRLRn/CFLRn registers.

Problem:

When PWM timer is enabled in Capture mode and PWM counter is started, PWM timer will be halted if input signal transient before PWM counter really starts. (PWM counter may delay one PWM counter clock source when counter enabled.)

Workaround:

Before the capture input channel - CAPENR is enabled, user should enable PWM timer – CHxEN(PCR) first and make sure PWM timer starts counting by checking PWM data register - PDRx is not equal to 0. As shown in the following example, user enables PWM Timer2 first and then waits until PDR2 is not equal to 0 to make sure PWM Timer2 starts to count. After Timer2 begins to count, user enables capture2 input path – CAPNER[2].

```
void main(void)
{
    ...
    /* Set the PWMA channel 2 for capture function */
    _PWM_SET_TIMER_PRESCALE(PWMA,PWM_CH2, 1); //Set Timer2 prescaler
    _PWM_SET_TIMER_CLOCK_DIV(PWMA,PWM_CH2,PWM_CSR_DIV1);//Set Timer2 clock divider
    _PWM_SET_TIMER_AUTO_RELOAD_MODE(PWMA,PWM_CH2); //Set Timer2 auto-reload mode
    _PWM_ENABLE_CAP_FUNC(PWMA, PWM_CH2); //Enable PWM2 capture function
    PWMA->CNR2 = 0xFFFF; //Set Timer2 loaded value
    _PWM_ENABLE_CAP_FALLING_INT(PWMA, PWM_CH2); //Enable falling edge interrupt
    NVIC_EnableIRQ((IRQn_Type)(PWMA_IRQn)); //Enable PWMA NVIC interrupt

    _PWM_ENABLE_TIMER(PWMA, PWM_CH2); //Enable PWM Timer2 – CH2EN(PCR[16])
    while(PWMA->PDR2==0); //Wait until PWM Timer2 start to count – PDR2
    _PWM_ENABLE_CAP_IN(PWMA, PWM_CH2); //Enable capture2 input path – CAPENR[2]
    ...
```

© 2013 Nuvoton Technology Corp.

```
}
```

2.14 RS485 Module Clock Setting

Description:

The module clock frequency of UART could be higher or lower than system clock frequency, even UART is working in RS-485 mode.

Problem:

When UART module clock frequency is higher than system clock, the received data will be incorrect in RS-485 mode.

Workaround:

Set UART source clock frequency divider (UART_N) and make sure UART module clock frequency (RS-485 operation clock frequency) is less than or equal to half of system clock. For example, if system clock frequency is 50 MHz, the module clock frequency of UART must be lower or equal to 25 MHz.

2.15 Timer Module Reset

Description:

The individual Timer channel can be reset to default settings (chip power-on settings) by setting the corresponding bit of IPRSTC2 [5:2].

Problem:

Timer 0, 1 are running on APB1 bus and Timer 2, 3 are running on APB2 bus. If the specified Timer is at reset state, the other Timer register of the same bus cannot be accessed.

For example, If Timer 1 is at reset state, Timer 0 register cannot be accessed until Timer 1 returns to normal state. If Timer 0 is at reset state, Timer 1 register cannot be accessed until Timer 0 returns to normal state. If Timer 3 is at reset state, the Timer 2 register cannot be accessed until Timer 3 returns to normal state. If Timer 2 is at reset state, Timer 3 register cannot be accessed until Timer 2 returns to normal state.

© 2013 Nuvoton Technology Corp.

Workaround:

Program should avoid setting IPRSTC2 register to reset Timer while Timer is running. It is recommended to set the CRST bit in TCSR register to reset timer function if necessary.

2.16 UART Error Flags

Description:

When Break Interrupt Flag, Parity Error Flag or Frame Error Flag (BIF/PEF/FEF) is set, user needs to write 1 to clear the relative error flag.

Problem:

"Writing 1" is unable to clear Break Interrupt Flag, Parity Error Flag and Frame Error Flag (BIF/PEF/FEF).

Workaround:

When one of the error flags (BIF/PEF/FEF) is set, user needs to flush FIFO (TFR/RFR) to clear the relative error flag.

2.17 UART Buffer Error Flag

Description:

When TX or RX FIFO overflows (TX_OVER_IF=1 or RX_OVER_IF=1) and the Buffer Error interrupt function is enabled, the Buffer Error interrupt will be generated (BUF_ERR_IF=1).

Problem:

When Break Interrupt Flag, Parity Error Flag or Frame Error Flag (BIF/PEF/FEF) is set, the BUF_ERR_IF flag is set to 1 at the same time.

Workaround:

If Buffer Error flag (BUF_ERR_IF) is set, check if TX or RX FIFO overflows or not. If TX or RX FIFO does not overflow, user needs to flush FIFO to clear the Error flag (BIF/PEF/FEF).

© 2013 Nuvoton Technology Corp.

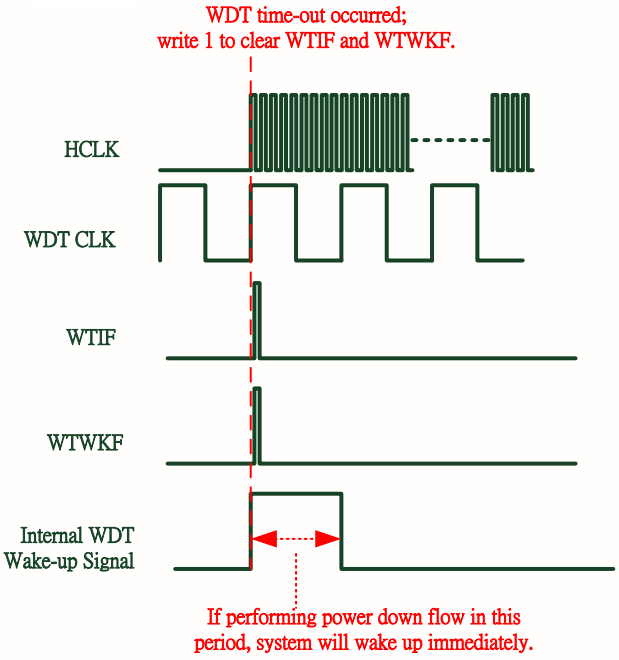
2.18 WDT Wake-up Flag

Description:

When system is woken up by WDT time-out interrupt, the WTWKF (WDT wake-up flag) bit will be set to 1. Program can write 1 to clear this bit and then perform Power-down flow to enter Power-down mode again

Problem:

In general case, system can enter Power-down mode successfully while WTWKF is 0. When system is woken up by WDT time-out interrupt the WTWKF will be set to 1 and the internal WDT wake-up signal is also kept to 1. When program writes 1 to clear WTWKF bit, the WTWKF bit is cleared to 0 immediately but the internal WDT wake-up signal will be kept one WDT module clock period before it is changed to 0. If user performs Power-down flow while internal WDT wake-up signal keeps 1, system will wake up immediately. The detailed timing diagram is shown below.



Workaround:

Once WDT time-out wake-up occurs, program has to delay at least one WDT module clock to enter Power-down mode again. For example, if the current WDT source clock is 10 kHz, the delay time should be more than 100 us to enter next Power-down mode successfully.

© 2013 Nuvoton Technology Corp.

3 Revision History

Revision	Date	Description
1.00	Mar. 29, 2013	Initially issued.
1.01	Apr. 16, 2013	Added PWM Capture problem. Modified some terminologies.

© 2013 Nuvoton Technology Corp.

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.

© 2013 Nuvoton Technology Corp.