

WDOG Unit

UT32M0R500 32-bit Arm™ Cortex® M0+ Microcontroller

Application Note

Cobham.com/HiRel

December 6, 2018

The most important thing we build is trust

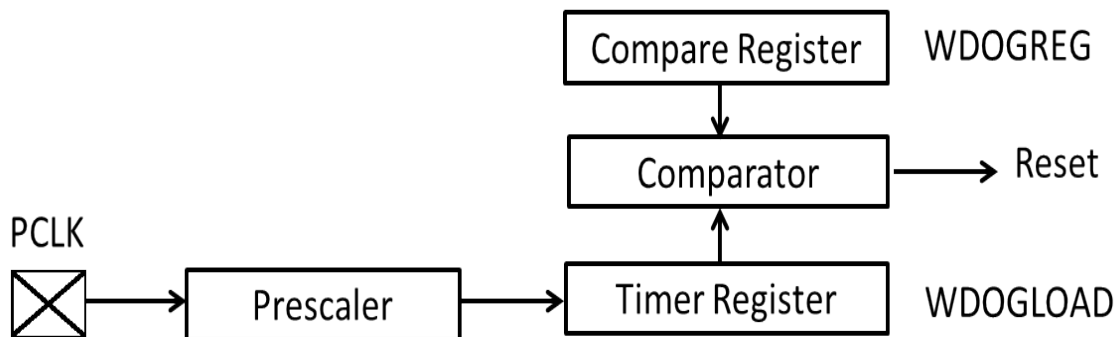
Table 1: Cross Reference of Applicable Products

PRODUCT NAME	MANUFACTURER PART NUMBER	SMD #	DEVICE TYPE	INTERNAL PIC NUMBER
Arm Cortex M0+	UT32M0R500	5962-17212	WDOG Unit	QS30

1.0 Overview

Watch Dog Timer (WDOG) provides a way to recover from software malfunctions by applying a reset to the system. The reset is a warm reset—the microcontroller starts execution at address 0 in Thread Mode (normal mode).

Figure 1 shows the basic diagram of a WDOG timer.

**Figure 1: WDOG timer basic diagram**

The watch dog begins counting down from the WDOGLOAD register value, and will be reloaded by writing to the interrupt clear register (WDOGINTCLR). If the timer counts down to 0 twice, than the WDOGRES signal will be asserted, causing a reset to the system. Optionally, in Window mode, if the counter value is higher than the Window register value, than the WDOGRES will be asserted causing a reset to the system.

PUBLIC

Figure 2 shows the WDOG time-out and Window mode reset.

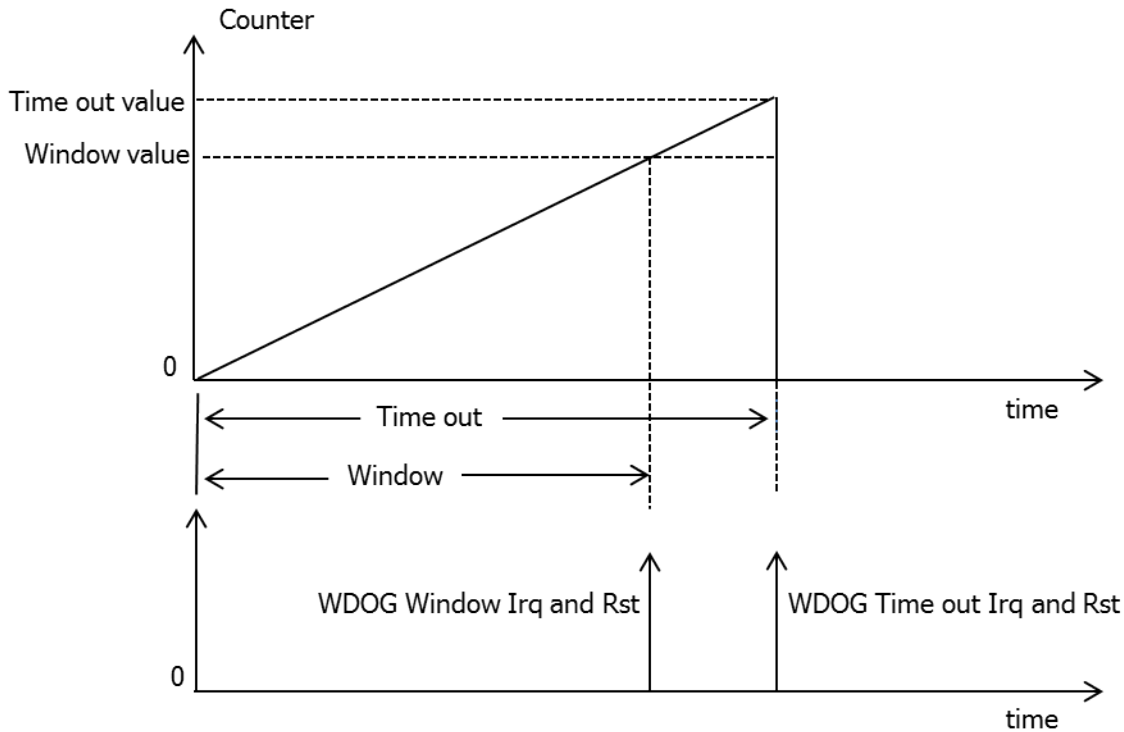


Figure 2: WDOG Time out or Window reset

2.0 [Application Note Layout](#)

This application note (AN) provides a brief description of the WDOG unit’s memory map, configuration and programming.

3.0 [WDOG Unit Hardware](#)

The WDOG Unit is mapped to the memory region from 0x4000_8000 to 0x4000_8FFF. It has 8 registers. For more information on each register, refer to **Chapter 11** of the UT32R500 Functional Manual.

PUBLIC

3.1 WDOG Load Register

The WDOG Load Register (**WDOGLOAD**) contains the reload value to count down from. When the counter times out twice, the WDOGRES signal will be asserted, resetting the system.

3.2 WDOG Control Register

The Control Register (**CTRLREG**) enables reset by setting reset enable, bit [1], to 1; the Window reset enable (optional), bit[2], to 1; the interrupt enable, bit [0], to 1.

3.3 WDOG Interrupt Clear Register

The WDOG Interrupt Clear Register (**WDOGINTCLR**) accepts the specific value **0x1357c0b1** to clear the watchdog interrupt and reload the timer from the value in WDOGLOAD.

3.3.1 WDOG Window Value Register

The WDOG Window Register (**WDOGWINDOW**) holds the value compared to by the timer, and if the timer's value is greater, than the WDOGRES signal will be asserted, causing a reset to the system.

3.3.2 WDOG Lock Register

The WDOG Lock Register (**WDOGLOCK**) locks write access to load, control, and window value registers by setting WDOGLOCK, bit [0], to 1.

4.0 WDOG Initialization

The watch dog time out period of 5 seconds for the example application is given by:

*Time out = pclk * LoadValue*

*Time out = 20ns * 250000000*

Time out = 5 seconds

Code 1 initializes the WDOG timer for reset and interrupt enable with the time out set to 5 seconds, and for specifics on the API's, refer to Wdog_StdPeriph_Lib at www.cobham.com.

```
WDOG_StructInit (&WDOG_InitStruct);

// PCLK=50Mhz, 20ns
// Period = 20ns * LoadValue; LoadValue = Period/20ns=5s/20ns=500000000
WDOG_InitStruct.LoadValue = 250000000;

// Init the watchdog
WDOG_Init (WDOG, &WDOG_InitStruct);
WDOG_Cmd (WDOG, ENABLE);

// WDOG_IntConfig (WATCHDOG_TypeDef *WDOGx, EnableState WindowResetEnable,
// EnableState ResetEnable, EnableState InterruptEnable)
// Enable reset and interrupt, Window reset disabled for now
WDOG_IntConfig (WDOG, DISABLE, ENABLE, ENABLE);
```

Code 1: WDOG Timer Initialization

PUBLIC

5.0 WDOG Unit Programming

Section 3.0 presented some of the basic configurations for the WDOG core unit. The following section show programming examples by making use of Cobham API's for the UT32RM0R500.

Putting it all together, Code 2 shows the main subroutine for “kicking the dog” and an endless loop for testing the WDOG timer reset functionality.

```
int main (void){
    // Initialization and settings from previous sections go here.
    for(;;){
        if(SysTickExpired)
        {
            SysTickExpired=0;
            // Infinite loop to test the watch dog (WDOG) reset.
            while(1);
        }
        // Kick the dog!
        WDOG_PetTheDog (WDOG);
    }
}

void SysTick_Handler(void){
    SysTickExpired=1;
    GPIO_WriteOutputDataBit (GPIO2, PIN_13, SET);// LED on
    __ASM volatile ("nop");
    GPIO_WriteOutputDataBit (GPIO2, PIN_13, RESET);// LED off
}
```

Code 2: Sample program for testing WDOG reset

Figure 3 shows an Oscilloscope plot for the process running and the WDOG timer, timing out after 5 seconds and resetting the system.

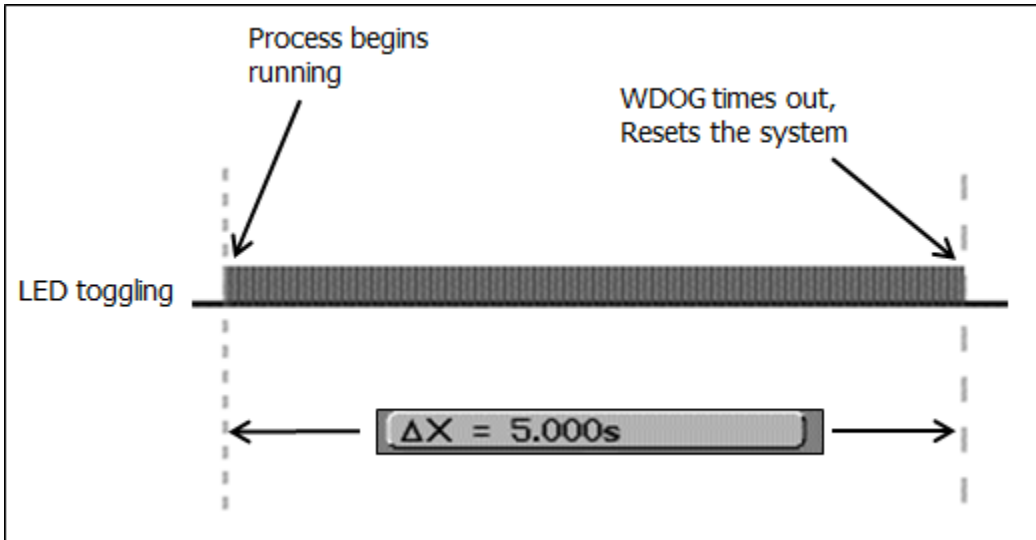


Figure 3: WDOG 5 secs time out reset

6.0 Summary and Conclusion

For unexpected and unknown software behaviors, the Watch Dog Timer (WDOG) provides a way to recover from them by applying a reset to the system and rebooting it to its normal state.

For more information about our UT32M0R500 microcontroller and other products, please visit our website: www.cobham.com/HiRel or email us at info-ams@cobham.com.

REVISION HISTORY

Date	Rev. #	Author	Change Description
12/06/18	1.0.0	JA	Initial Release



Cobham Semiconductor Solutions

The following United States (U.S.) Department of Commerce statement shall be applicable if these commodities, technology, or software are exported from the U.S.: These commodities, technology, or software were exported from the United States in accordance with the Export Administration Regulations. Diversion contrary to U.S. law is prohibited.

Cobham Semiconductor Solutions
4350 Centennial Blvd
Colorado Springs, CO 80907



E: info-ams@aeroflex.com
T: 800 645 8862

Aeroflex Colorado Springs Inc., DBA Cobham Semiconductor Solutions, reserves the right to make changes to any products and services described herein at any time without notice. Consult Aeroflex or an authorized sales representative to verify that the information in this data sheet is current before using this product. Aeroflex does not assume any responsibility or liability arising out of the application or use of any product or service described herein, except as expressly agreed to in writing by Aeroflex; nor does the purchase, lease, or use of a product or service from Aeroflex convey a license under any patent rights, copyrights, trademark rights, or any other of the intellectual rights of Aeroflex or of third parties.

PUBLIC