# XT32H05x

# XT32 microcontroller WDT-W

# Application notes

Rev 0.0.0

**Original Release Date: 28-Sep-2023**

**Revised :**

# Revision History

| Release | Date | Author | Summary of Change |
|---|---|---|---|
| V0.0.0 | 28/09/2023 | Shirling Liu | Initial |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

# Contents

# List of Figures

# List of Tables

# 1 Introduce

This application note serves as a comprehensive guide for software developers, offering essential information on WDT-W (Windows watchdog). It covers fundamental concepts and provides guidelines to ensure proper utilization of WDT-W in software development projects. Whether you're a beginner or an experienced developer, this document will equip you with the necessary knowledge and best practices to effectively configure and utilize WDT-W in your applications.

## 1.1 Required peripherals

This application involves modules as table 1.

Table 1. Modules in example

| Sub-module | Peripheral use | Note |
|---|---|---|
| W-WDT | 32-bit down counter watchdog timer | |
| | | |

## 1.2 Compatible devices

This example is compatible with the devices in Table 2.

Table 2. Device list

| Product | EVB |
|---|---|
| XT32H050 | XB002823 |
| | |

## 2 Design description

### 2.1 Feature overview

WDT-W has a 32-bit down counter watchdog timer. WDT-W counts down from the initial value (TORR register), when the counter reaches zero, it will occur an interrupt and wraps to the selected timeout value (TORR register) and continues to count down 0, user can restart the watchdog counter (HAL_WDT_Refresh) to feed the dog when IRQ generated, otherwise WDT-W counter counts to 0 again, system will be reset.

### 2.2 Design steps

1. Set WDT-W source clock and clock divider.
2. Configure WDT-W parameters: reset pulse length, wraps Range period, interrupt mode.

3. Register IRQ callback function and enable Interrupt.
4. Software feed dog to reload WDT-W counter in IRQ callback, otherwise it will reset system.

## 2.3 Design considerations

## 2.4 Software flowchart

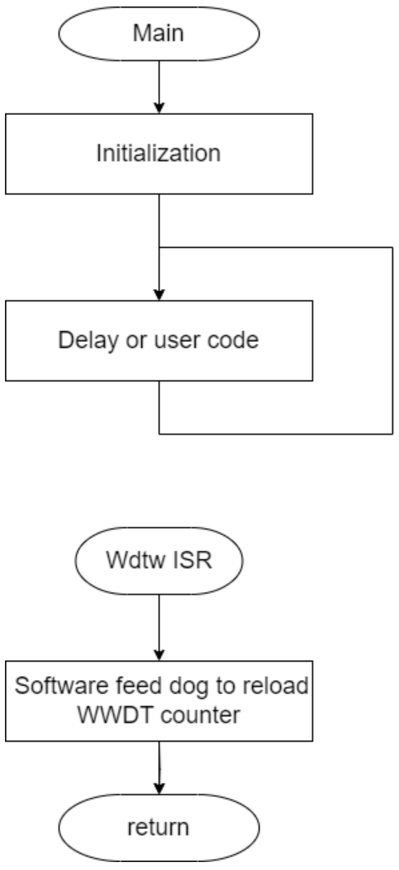

Figure 1. Application flow

## 2.5 Reference code

Configure Peripheral WDT-W using HAL_WDT_Init.

```
    HAL_WDT_RegisterCallback(&hwdtw, XT_Wdtw_TimeoutCB);
    hwdtw.Instance    = WDTW;
    hwdtw.Init.Rpl    = WDT_RPL_2;/* reset pulse length 2 pclk cycles */
    hwdtw.Init.Rmod   = WDT_RMOD_INTRST;    /* interrupt then reset */
    hwdtw.Init.Range = WDT_RANGE_PERIOD_7;
    hwdtw.Init.InitRange = WDT_RANGE_PERIOD_7;
    if (HAL_WDT_Init(&hwdtw) != HAL_OK)
    {
      /* Error_Handler */
    }
    XT_EVB_Led_Off(LED1);
```

XT_Wdtw_TimeoutCB restart the watchdog counter to feed dog.

```
static void XT_Wdtw_TimeoutCB(WDT_HandleTypeDef *hwdt)
{
    HAL_WDT_Refresh(&hwdtw);
    XT_EVB_Led_Toggle(LED1);
}
```

## 2.6 Additional resources

- XT32H0xxB--reference manual