# 例程十一　窗口看门狗实验

## 介绍

窗口看门狗用于监测由于外部干扰或不可预知的逻辑条件所产生的软件错误，这样的软件错误通常会导致应用程序不按照预期的方式运行。除非程序在递减计数器的T6位变为0之前刷新递减计数器，看门狗电路将在一个预置的时间间隔后产生系统复位；如果在7位的递减计数器数值达到窗口寄存器数值之前刷新递减计数器，同样会产生系统复位。这就意味着只能在一个有限的时间窗口内刷新递减计数器。

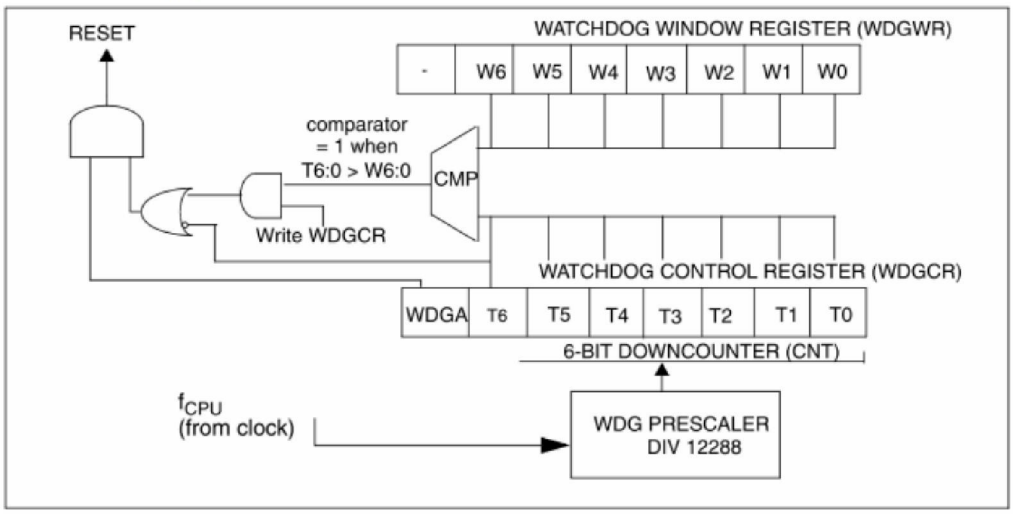
## WWDG主要功能

- 可编程的自由运行递减计数器
- 有条件的复位
  - 如果开启了看门狗，当递减计数器的数值小于 0x40 时产生复位
  - 如果开启了看门狗，当在指定的时间窗口之外重加载递减计数器的数值(见*图 27*) 时产生复位
- 硬件或软件启动看门狗(由选择字节指定)
- 可在HALT指令时产生复位(由选择字节配置)

## WWDG功能说明

如果开启了看门狗(设置了WDGA=1)，当7位的递减计数器(T[6:0]位)从0x40变为0x3F时(即T6变为0)，看门狗产生一个复位信号并把复位引脚拉低。如果软件刷新计数器时，计数器的数值大于窗口寄存器中的数值，同样会产生复位。

图25　窗口看门狗框图




在正常的操作期间，应用程序必须定期地写入WDGCR寄存器，以避免产生复位；这个写的动作必须在计数器的数值小于窗口寄存器的数值时进行。写入WDGCR寄存器的数值必须是介于0xFF和0xC0之间(见*图26*)：

- 开启看门狗：

如果(通过选择字节)选择了软件看门狗，在系统复位后看门狗处于关闭状态。设置WDGCR寄存器中的WDGA位将开启看门狗，随后在下次复位之前将不能关闭看门狗。

如果(通过选择字节)选择了硬件看门狗，看门狗将始终开启，而WDGA位将不起作用。

- 控制递减计数器：

递减计数器是自由运行计数器：即使未开启看门狗，它依然不断地递减计数。当开启看门狗时，必须设置T6位以避免立刻产生复位。

T[5:0]位中包含了看门狗产生复位前允许的时间延迟(见*图26*)；因为写入WDGCR寄存器时，预分频器的状态是不可知的(见*图27*)，所以这个时间延迟介于一个最小和最大数值之间。

窗口寄存器(WDGWR)的数值是指定窗口的高限：为防止复位，必须在递减计数器的数值小于窗口寄存器的数值并大于0x3F时刷新递减计数器。*图27*描述了窗口看门狗操作过程。

*T6位可以用于产生一个软件复位(即设置WDGA位同时清除T6位)*

- 在停止时产生看门狗复位

如果开启了看门狗，并且选择了停止时产生看门狗复位的选项，则执行HALT指令将产生复位。

## 在停止模式下使用WWDG

如果在选择字节中使能了停止模式下的看门狗(HALT指令不产生看门狗复位)，建议在执行HALT指令前先刷新看门狗计数器，以避免在唤醒微控制器后立刻进入不希望的看门狗复位。

## 如何设置看门狗的超时

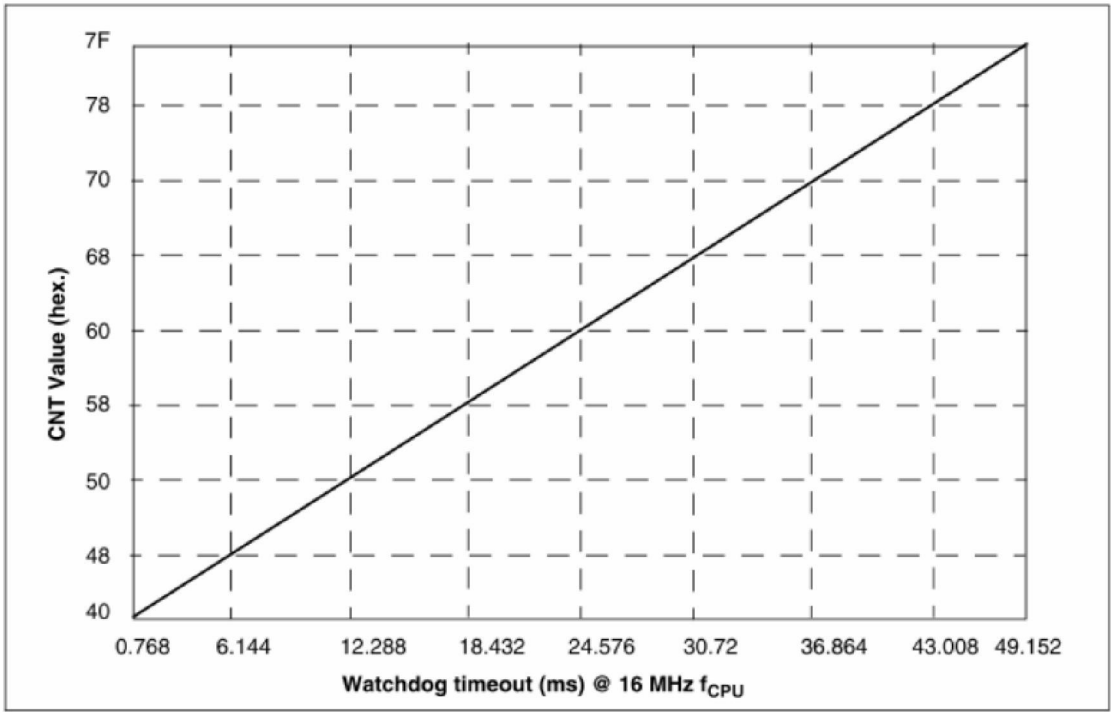
下图显示了看门狗计数器(CNT)中的6位数值，与以毫秒为单位的超时时间的线性关系，这个表可以在不考虑时序变化时作为一个快速的粗略计算参考，如果需要更精确的计算，请使用*图27*的公式。

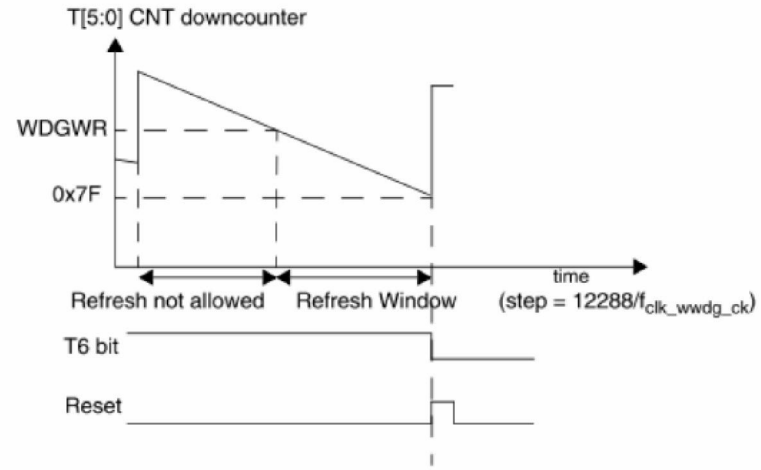
---

警告：每次写入WDGCR寄存器时，首先要置T6位为'1'，以避免立刻产生看门狗复位。

---

图26　大约的超时时间






**Example:**

| Counter code | 0h | 7Fh |
|---|---|---|
| fckc_wwdg_ck | 1 step | 64 steps |
| 16 MHz | 0.768 ms | 49.152 ms |
| 16 MHz/8 | 6.144 ms | 393 ms |

## 15.6 WWDG低功耗模式

表28 WWDG在低功耗模式下的影响

| 模式 | 说明 | |
|---|---|---|
| 等待(Wait) | 看门狗不受影响：递减计数器照常工作 | |
| 停机(Halt) | 选择字节中WWDG_HALT | |
| | 0 | 不产生看门狗复位。微控制器进入停止模式。递减计数器递减一次后停止计数，在微控制器收到一个外部中断或复位之前，它不会再产生看门狗复位。<br>如果收到了一个中断(参考中断映像表，查看停止模式下可以产生哪些中断)，在经过稳定延迟后看门狗将恢复计数。如果系统被复位，除非在选择字节中选择了硬件看门狗，否则看门狗将被关闭。请参考下面15.8的建议。 |
| | 1 | 产生一个复位而不是进入停止模式。 |
| 活跃停机(Active Halt) | X | 不产生复位，微控制器进入Active Halt模式。看门狗计数器停止计数，不再递减。当微控制器收到一个振荡器中断或外部中断，看门狗立刻恢复计数。当微控制器被复位，在经过稳定延迟后看门狗将恢复计数。 |

## 15.7 硬件看门狗选项

如果在选择字节中选择了硬件看门狗选项，则看门狗始终开启，同时WDGCR寄存器中的WDGA将不起作用。请参考数据手册中有关选择字节的说明。

## 在停止模式下使用WWDG

如果开启了看门狗，则建议在停止模式下做如下操作。

在执行HALT指令前先刷新看门狗计数器，以避免在唤醒微控制器后立刻进入不希望的看门狗复位。

## 15.10 WWDG寄存器

### 15.10.1 控制寄存器(WWDG_CR)

地址偏移值：0x00

复位值：0x7F

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| WDGA | T6 | T5 | T4 | T3 | T2 | T1 | T0 |
| rw | rw | rw | rw | rw | rw | rw | rw |

| | |
|---|---|
| 位7 | **WDGA**：开启位(1)<br>该位由软件设置，只能由硬件在复位后清除。当WDGA=1时，看门狗可以产生复位。<br>0：关闭看门狗<br>1：开启看门狗 |
| 位6:0 | **T[6:0]**：7位计数器(MSB至LSB)<br>这些位包含看门狗计数器的数值，每过(大约)12288个fckc_wwdg_ck周期递减一次。当它的内容从0x40变为0x3F(T6被清除)时，则产生一个复位。 |

(1) 如果在选择字节中使能了硬件看门狗功能，则此位不起作用。

### 15.10.2 窗口寄存器(WWDG_WR)

地址偏移值：0x01

复位值：0x7F

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 保留 | W6 | W5 | W4 | W3 | W2 | W1 | W0 |
| rw | rw | rw | rw | rw | rw | rw | rw |

| | |
|---|---|
| 位7 | 保留 |
| 位6:0 | **W[6:0]**：7位计数器(MSB至LSB)<br>这些位包含了窗口的数值，这是需要与递减计数器比较的数值。 |

看完窗口看门狗的具体介绍，发现窗口看门狗也挺强大的吧。下面我从主函数看起，一步一步来分析下

```
int main(void)
{
    LED_Init();
    SetLedON();
    Test_WWDGReset();
    WWDG_Configuration();

    while (1)
    {
      Refresh_WWDG_Window();
      /*Refresh_WWDG_Window()这个函数是俗称喂狗
      如果注释掉这个函数的话，系统每个43.008ms就会复位
      这个函数功能很强大的，也可以实现一个定时器，定时
      间隔要在 WWDG_Configuration()函数里面配置*/
      LED_Display();

     }
}
```

相信大家看到这主函数，都能明白的一些了吧，下再看看一个子函数，了解它的功能和配置。

```
#define CounterInit 0x7f
#define window      0x77
void WWDG_Configuration(void)
{
    /* WWDG Configuration */
    /* Watchdog Window= 0x7F step to 0x3F step
    = (0x7F-0x3F) * 1 step
    = 64 * 1 step
    = 64 * (12288/2Mhz)
    = 393,216ms
    */
    /* Allowed Window = (0x7F-window) * 1 step
    = (0x7F-0x77) * 1 step
    = 7 * 1 step
    = 7 * (12288/2Mhz)
    = 43.008ms
    */
    /* So the non allowed window starts from 0.0ms to 43.008ms
    and the alowed window starts from 43.008ms to 393,216ms */
    WWDG_Init(CounterInit,window);
}
```

大家看得明白吗？下面我分析下吧，大家刚才注意到我在主函数里面没有设置系统时钟，在前面的例程中我也介绍过了，STM8 是如果你没设置系统时钟的话，复位后就是 16M 的 8 分频，也就是系统时钟为 2M.这个函数是配置一个窗口,在 0~43.008ms 内不能喂狗,否则就会产生部位,那只能在 43.008ms~393.216ms 喂狗。有了这个配置函数，可以在下面 Refresh_WWDG_Window()这个函数里面实现定时作用，同时也要喂狗。在我的例程中，我在这个函数里面没实现什么功能，只是在喂狗而已，大家用这个函数是时候可以添加自己的功能代码，实际上就是一

个定时器来的。大家来看看函数的原型

```
void Refresh_WWDG_Window(void)
{
  u8 CounterValue;
  CounterValue = (u8)(WWDG_GetCounter() & 0x7F);

  if(CounterValue < window)
  {
    WWDG_SetCounter(CounterInit);
    /*在此处放你的代码*/
  }
}
```

把程序下载到风驰电子 STM8 开发板上，如果把 Refresh_WWDG_Window()注释掉，则会不断地复位，4 个 LED 不断在闪，如果不注释的话则正常运行，4 个 LED 流水灯流起来。

风驰电子祝您学习愉快～～～!!!!!