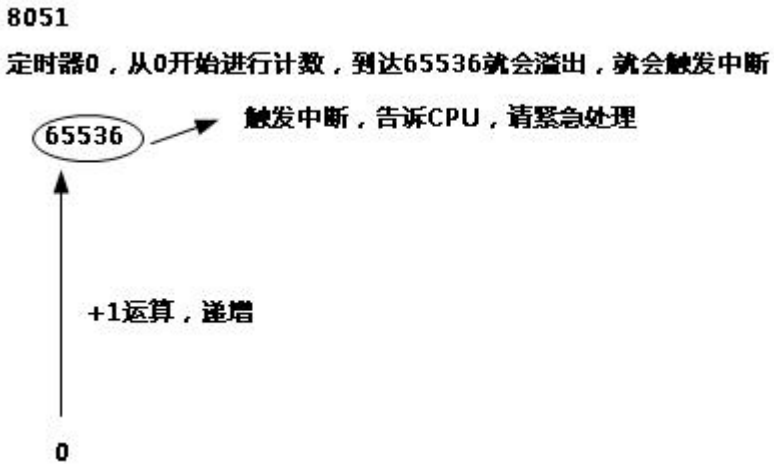
一、系统定时器

SysTick 叫做系统滴答时钟、系统定时器，属于 Cortex-M4 内核中的一个外设(外围设备)，它 24bit 向下递减的计数器。



二、系统定时器的中断使用方法

1.代码的初始化

If you only want to generate a periodic SysTick interrupt, the easiest way is to use a CMSIS-Core function called “SysTick_Config”:

```
uint32_t SysTick_Config(uint32_t ticks);
```

For example, if you have a clock frequency of 30MHz and you want to trigger a SysTick exception of 1KHz, you can use:

如果使用CPU的频率为30MHz，而同时想触发1秒产生1000次中断，初始化如下

```
SysTick_Config(SystemCoreClock / 1000);
```

```
    //初始化系统定时器，1S 内核触发 1000 次中断，说白了定时 1ms
    SysTick_Config(SystemCoreClock/1000);
```

2. 中断服务函数的编写

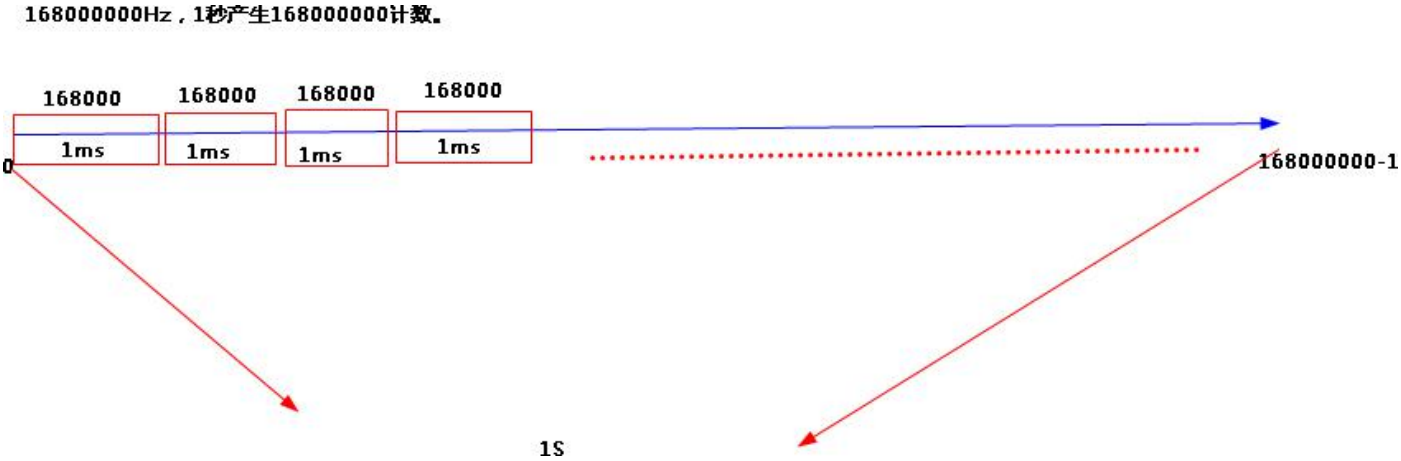
```
void SysTick_Handler(void)
{
    static uint32_t cnt=0;

    cnt++;

    //到达 500ms 的定时
    if(cnt >= 500)
    {
        cnt=0;

        PFout(9)^=1;

    }
}
```

注：如果发现中断服务函数定时不准确，请检查
1）SysTick_Config 函数是否填写参数正确
2）检查 PLL 的配置是否准确

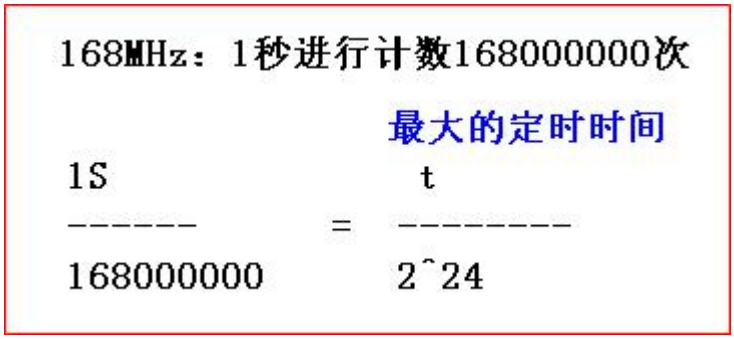
练习 1：
使用系统定时器实现 LED 不同的闪烁时间。
LED0 100ms
LED1 330ms
LED2 1500ms
LED3 2200ms

3. 定时时间的计算

SysTick_Config(SystemCoreClock/**频率**);



思考题，让系统定时器触发 1 秒中断是否可以？如果不可以，最大的定时时间又是什么？
答：不能触发 1 秒中断。



在额定频率情况下，最大定时时间 = 2^24 /168000000 ≈ 99.86ms
在超频的频率（216MHz）下，最大定时时间 = 2^24 /216000000≈77.67ms

测试结果：

```
//初始化系统定时器，1S 内核触发 1000 次中断，说白了定时 1ms，能够成功
//SysTick_Config(SystemCoreClock/1000);

//初始化系统定时器，1S 内核触发 10 次中断，说白了定时 100ms,现象失败
SysTick_Config(SystemCoreClock/10);

//初始化系统定时器，1S 内核触发 11 次中断，说白了定时 90.90ms,能够成功
SysTick_Config(SystemCoreClock/11);
```

三、系统定时器的用途

两个方面：

没有操作系统：只用于延时

有操作系统（ucos2 ucos3 freertos....）:为操作系统提供精准的定时中断（1ms~50ms）

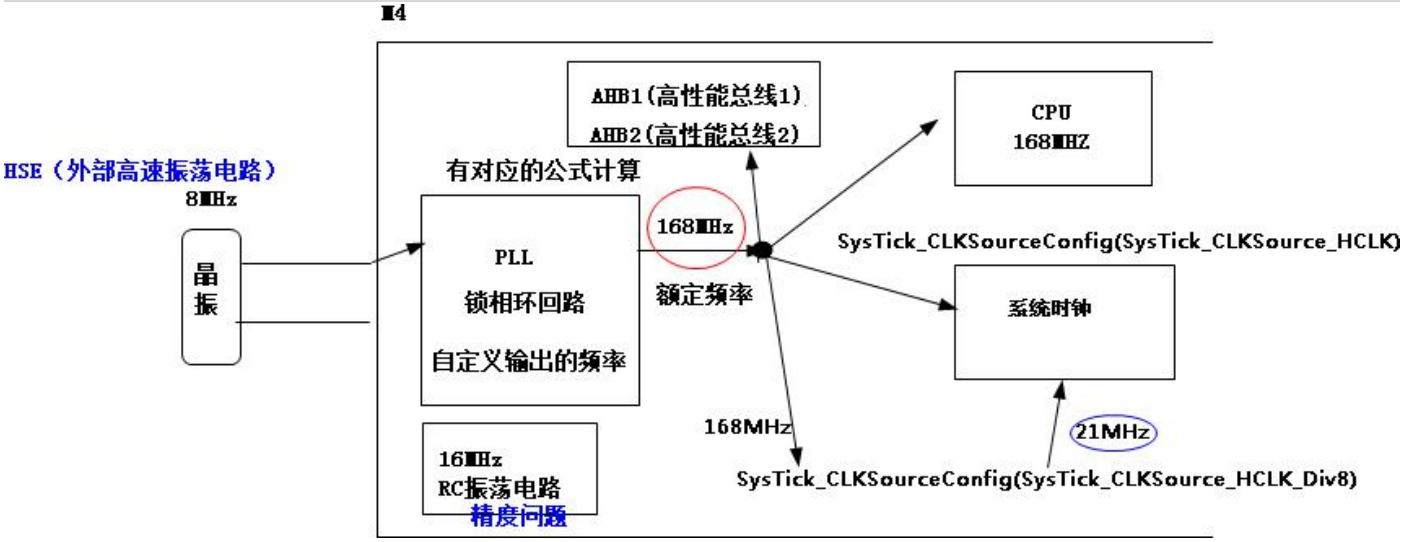
四、使用系统定时器用于延时的用途

If you want to use the SysTick timer in polling mode, you can use the count flag in the SysTick Control and Status Register (SysTick->CTRL) to determine when the timer reaches zero. For example, you can create a timed delay by setting the SysTick timer to a certain value and waiting until it reaches zero:

```
SysTick->CTRL = 0; // Disable SysTick
SysTick->LOAD = 0xFF; // Count from 255 to 0 (256 cycles)
SysTick->VAL = 0; // Clear current value as well as count flag
SysTick->CTRL = 5; // Enable SysTick timer with processor clock
while ((SysTick->CTRL & 0x00010000)==0);// Wait until count flag is set
SysTick->CTRL = 0; // Disable SysTick
```

1. 配置系统定时器的时钟源

```
/**
  * @brief  Configures the SysTick clock source.
  * @param  SysTick_CLKSource: specifies the SysTick clock source.
  *   This parameter can be one of the following values:
  *     @arg SysTick_CLKSource_HCLK_Div8: AHB clock divided by 8 selected as SysTick clock source.
  *     @arg SysTick_CLKSource_HCLK: AHB clock selected as SysTick clock source.
  * @retval None
  */
void SysTick_CLKSourceConfig(uint32_t SysTick_CLKSource)
{
  /* Check the parameters */
  assert_param(IS_SYSTICK_CLK_SOURCE(SysTick_CLKSource));
  if (SysTick_CLKSource == SysTick_CLKSource_HCLK)
  {
    SysTick->CTRL |= SysTick_CLKSource_HCLK;
  }
  else
  {
    SysTick->CTRL &= SysTick_CLKSource_HCLK_Div8;
  }
}
```



2. 系统定时器寄存器

Table 9.7 SYSTICK Control and Status Register (0xE000E010)

Bits	Name	Type	Reset Value	Description
16	COUNTFLAG	R	0	Read as 1 if counter reaches 0 since last time this register is read; clear to 0 automatically when read or when current counter value is cleared
2	CLKSOURCE	R/W	0	0 = External reference clock (STCLK) 1 = Use core clock
1	TICKINT	R/W	0	1 = Enable SYSTICK interrupt generation when SYSTICK timer reaches 0 0 = Do not generate interrupt
0	ENABLE	R/W	0	SYSTICK timer enable

Table 9.8 SYSTICK Reload Value Register (0xE000E014)

Bits	Name	Type	Reset Value	Description
23:0	RELOAD	R/W	0	Reload value when timer reaches 0

Table 9.9 SYSTICK Current Value Register (0xE000E018)

Bits	Name	Type	Reset Value	Description
23:0	CURRENT	R/Wc	0	Read to return current value of the timer. Write to clear counter to 0. Clearing of current value also clears COUNTFLAG in SYSTICK Control and Status Register

2. 当 SysTick 使用 168MHz 系统时钟频率时，代码编写如下：

```
void delay_us(uint32_t nus)
{

    SysTick->CTRL = 0;                        // Disable SysTick
    SysTick->LOAD = (SystemCoreClock/1000000)*nus; // 计数值
    SysTick->VAL = 0;                         // Clear current value as well as count flag
    SysTick->CTRL = 5;                        // Enable SysTick timer with processor clock
    while ((SysTick->CTRL & 0x00010000)==0);// Wait until count flag is set
    SysTick->CTRL = 0;                        // Disable SysTick
}

void delay_ms(uint32_t nms)
{
    SysTick->CTRL = 0;                        // Disable SysTick
    SysTick->LOAD = (SystemCoreClock/1000)*nms; // 计数值
    SysTick->VAL = 0;                         // Clear current value as well as count flag
    SysTick->CTRL = 5;                        // Enable SysTick timer with processor clock
    while ((SysTick->CTRL & 0x00010000)==0);// Wait until count flag is set
    SysTick->CTRL = 0;                        // Disable SysTick
```

```
}
```

最大的延时为 99.86ms

3. 当 SysTick 使用 168MHz 系统时钟频率并进行 8 分频时，代码编写如下：

```
void delay_us(uint32_t nus)
{
    SysTick->CTRL = 0;                        // Disable SysTick
    SysTick->LOAD = (SystemCoreClock/8/1000000)*nus; // 计数值
    SysTick->VAL = 0;                         // Clear current value as well as count flag
    SysTick->CTRL = 1;                        // Enable SysTick timer with processor clock
    while ((SysTick->CTRL & 0x00010000)==0);// Wait until count flag is set
    SysTick->CTRL = 0;                        // Disable SysTick
}


void delay_ms(uint32_t nms)
{
    SysTick->CTRL = 0;                        // Disable SysTick
    SysTick->LOAD = (SystemCoreClock/8/1000)*nms; // 计数值
    SysTick->VAL = 0;                         // Clear current value as well as count flag
    SysTick->CTRL = 1;                        // Enable SysTick timer with processor clock
    while ((SysTick->CTRL & 0x00010000)==0);// Wait until count flag is set
    SysTick->CTRL = 0;                        // Disable SysTick
}
```

思考题，当前最大的延时时间是多少？如何优化代码，支持秒级别或更长时间的延时？

最大的延时时间 = 2^24 / 21000000 ≈ 798.91ms

```
void delay_ms(uint32_t nms)
{
    uint32_t m,n;

    m = nms/500;

    n = nms %500;

    //m 个 500ms 的延时
    while(m--)
    {
        SysTick->CTRL = 0;                        // Disable SysTick
        SysTick->LOAD = (SystemCoreClock/8/1000)*500; // 计数值
        SysTick->VAL = 0;                         // Clear current value as well as count flag
        SysTick->CTRL = 1;                        // Enable SysTick timer with processor clock，当使用
21MHz 的时候，1；当使用 168MHz 的时候，5；
```

```
        while ((SysTick->CTRL & 0x00010000)==0);// Wait until count flag is set
        SysTick->CTRL = 0;                      // Disable SysTick
    }

    //不足 500ms 的延时
    if(n)
    {
        SysTick->CTRL = 0;                      // Disable SysTick
        SysTick->LOAD = (SystemCoreClock/8/1000)*n; // 计数值
        SysTick->VAL = 0;                       // Clear current value as well as count flag
        SysTick->CTRL = 1;                      // Enable SysTick timer with processor clock，当使用
21MHz 的时候，1；当使用 168MHz 的时候，5；
        while ((SysTick->CTRL & 0x00010000)==0);// Wait until count flag is set
        SysTick->CTRL = 0;                      // Disable SysTick

    }
}
```