第八天笔记

1. STM32定时器概述
2. 定时器的概述

定时器的基本功能，但是STM32的定时器除了具有定时功能之外，也具有定时器中断功能，还具有输入捕获（检测外部信号）以及输出比较功能（输出不同的脉冲），可以利用STM32定时器输出某种频率的脉冲信号来控制产品（控制灯的亮度、控制直流电机的转速、控制舵机的角度.......）

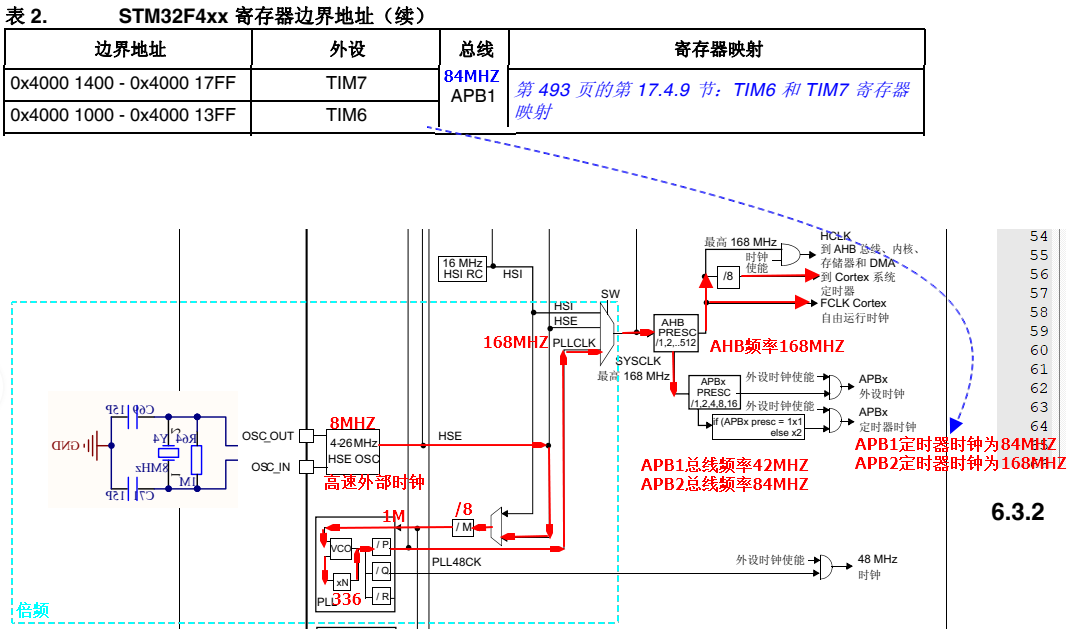
1. 定时器的种类

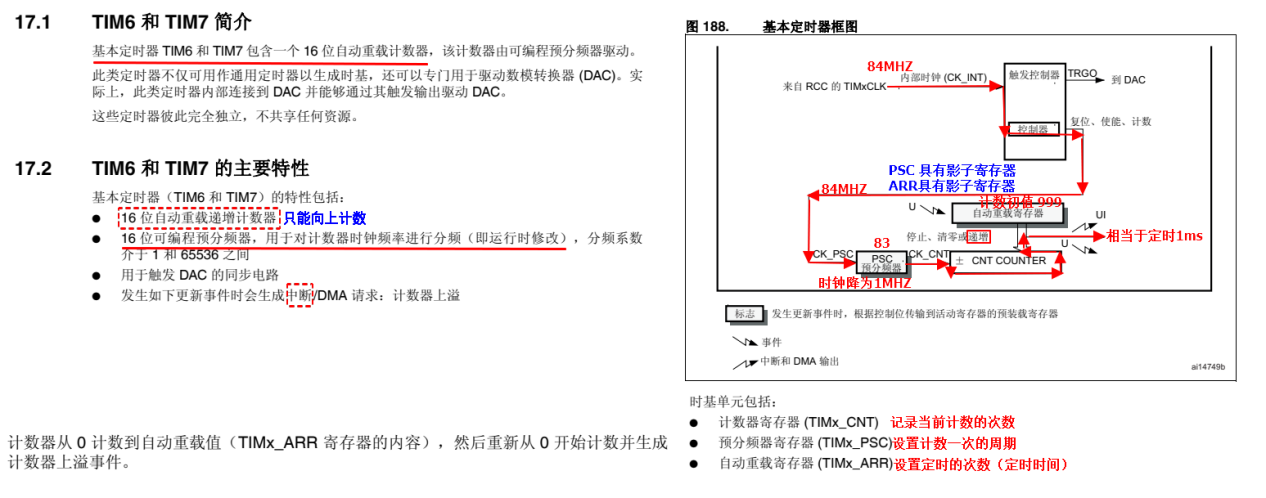
STM32由于资源丰富，所以提供了多种定时器，一共提供14个定时器（不包含系统定时器、不包含看门狗定时器），分为高级定时器（TIM1和TIM8）、通用定时器（TIM2~TIM5 + TIM9~TIM14，一共10个）、基本定时器（TIM6和TIM7）。

1. 定时器的使用

* 基本定时器

STM32提供TIM6和TIM7作为基本定时器，属于STM32内部资源，都是16bit定时器，并没有和GPIO引脚连接，TIM6和TIM7都挂载在APB1总线下，而APB1定时器时钟频率为84MHZ。





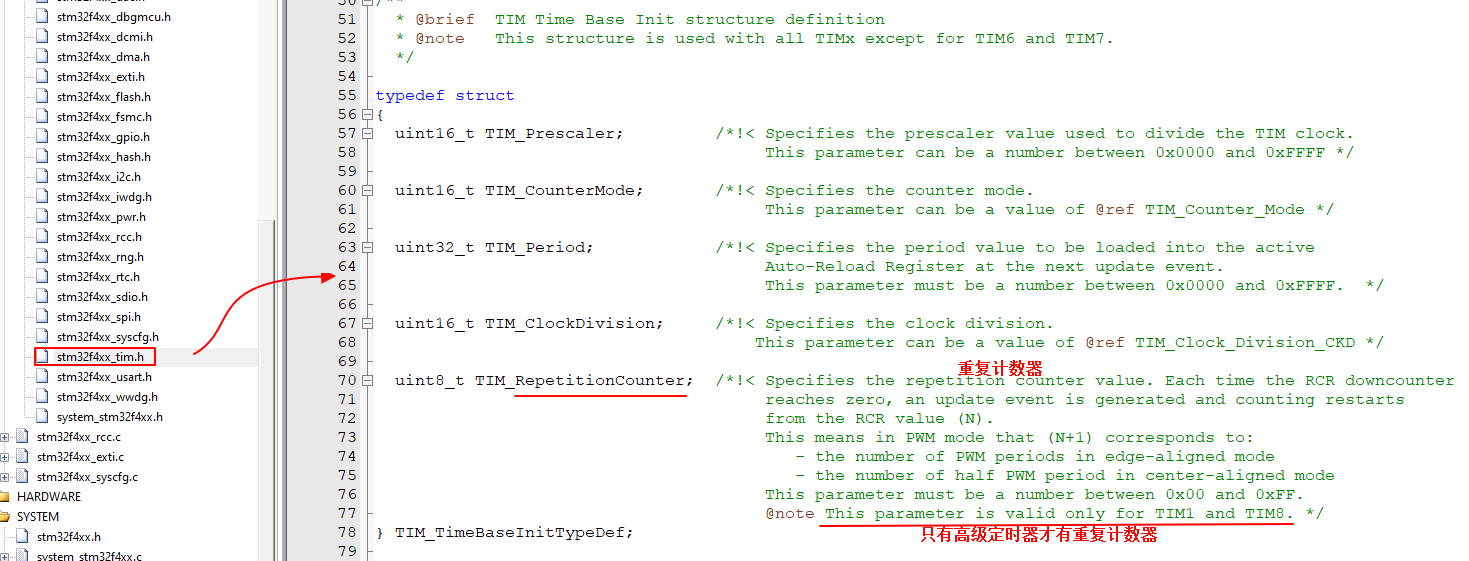
所以想要对基本定时器的定时功能进行设置，只需要把TIM\_CNT、TIM\_PSC、TIM\_ARR寄存器设置正确即可。

举个例子：TIM6的时钟频率为84MHZ，如果想要降低频率，公式：1MHZ = 84MHZ /(83+1)，频率就降为1MHZ，也就意味着TIM\_CNT寄存器累计计数1次的时间是1us，如果打算利用TIM6定时1ms，也就是说TIM\_ARR寄存器的值应该设置为1000-1，只要TIM\_CNT == TIM\_ARR，就说明时间到了，如果时间到了则TIM\_CNT清0，然后重新开始计数。

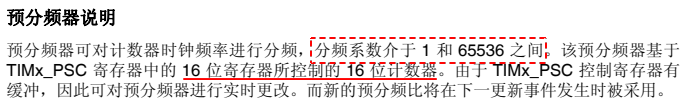
对时钟进行分频的目的就是为了延长定时时间，但是不是必须分频，由用户指定是否分频。

思考：那应该如何去配置基本定时器的相关参数？？ 可以通过结构体进行配置

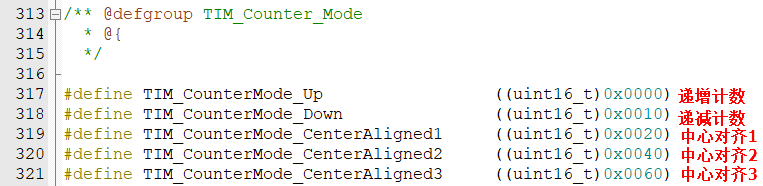
可以查阅stm32f4xx\_tim.h头文件中包含了定时器的参数配置结构体，如下图所示

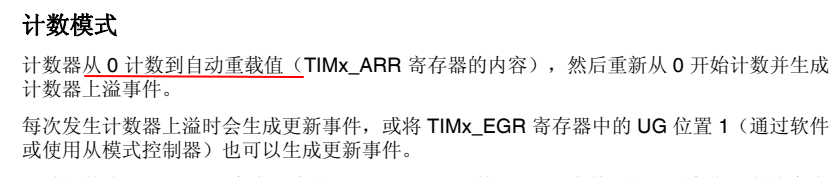


* 设置定时器的分频系数 TIM\_Prescaler 范围 0x0000 and 0xFFFF



* 设置定时器的计数模式 TIM\_CounterMode 基本定时器只能选择向上计数





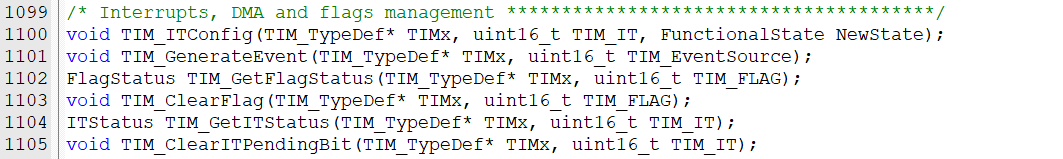
* 设置定时器的计数周期 TIM\_Period 指的是重载寄存器中的值 0x0000 and 0xFFFF
* 设置定时器的时钟分频因子 TIM\_ClockDivision 在输入捕获的采样使用 不需要设置
* 设置定时器的重复计数器 TIM\_RepetitionCounter 只有高级定时器需要设置

如何利用基本定时器实现定时中断的功能？流程可以参考stm32f4xx\_tim.c中的代码注释

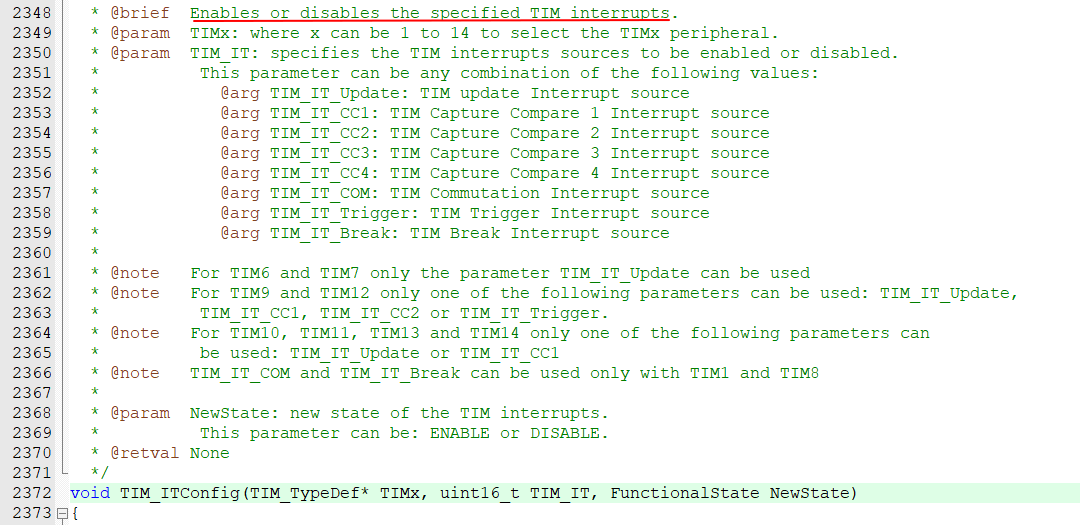


![1_43BUT0RRX](K_YAFWSL0V](data:image/jpeg;base64,)

1. 打开基本定时器的时钟 TIM6和TIM7 在APB1总线
2. 设置定时器的相关参数（预分频、计数模式、计数周期） + 初始化定时器
3. 设置定时器的中断源
4. 设置NVIC的相关参数（中断通道、中断优先级） + 初始化NVIC
5. 打开定时器
6. 编写定时器中断服务函数 中断服务函数名字去汇编文件复制（检测状态+清除状态）



如果打算设置定时器的中断源，可以调用一个TIM\_ITConfig函数 来设置中断的触发条件



函数原型

void TIM\_ITConfig(TIM\_TypeDef\* TIMx, uint16\_t TIM\_IT, FunctionalState NewState)

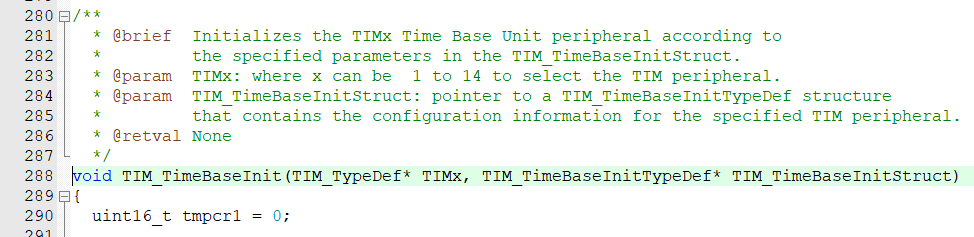
函数参数

参数一：TIMx 打算产生中断的定时器号 如TIM6

参数二：TIM\_IT 指定定时器的中断源 一般为TIM\_IT\_Update 表示更新时中断

参数三：NewState 打开或关闭中断源 ENABLE or DISABLE

初始化定时器，可以调用TIM\_TimeBaseInit函数实现



函数原型

voidTIM\_TimeBaseInit(TIM\_TypeDef\* TIMx, TIM\_TimeBaseInitTypeDef\* TIM\_TimeBaseInitStruct)

函数参数

参数一：TIMx 打算配置参数的定时器号 如TIM6

参数二：TIM\_TimeBaseInitStruct 指的是配置的参数信息 记得取地址 &

**练习：根据基本定时器的理解，编写代码实现利用TIM6每隔200ms让LED0的电平翻转一次，利用TIM7每隔400ms让LED1的电平翻转一次，并对比观察效果是否一致。（30分钟）**



* 通用定时器

STM32一共提供10个通用定时器（TIM2~TIM5、TIM9~TIM14），TIM2和TIM5是32bit定时器，其他的定时器都是16bit定时器。

TIM2~TIM5的计数方式有多种可以选择，分别为递增计数、递减计数、递增/递减计数。

**递增计数**：计数器从 0 计数到自动重载值（TIMx\_ARR 寄存器的内容），然后重新

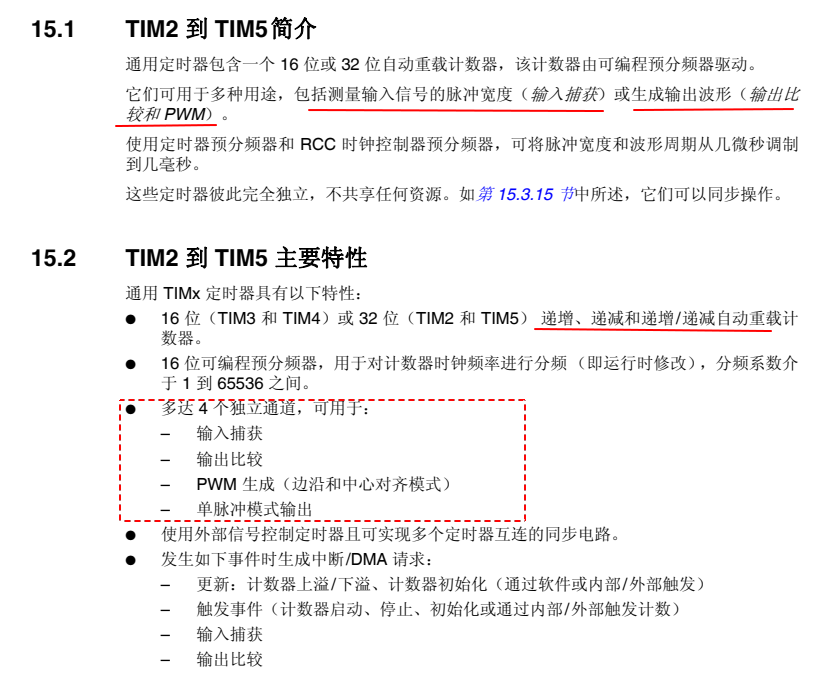
从 0 开始计数并生成计数器上溢事件。

**递减计数**：计数器从自动重载值（TIMx\_ARR 寄存器的内容）开始递减计数到 0，

然后重新从自动重载值开始计数并生成计数器下溢事件。

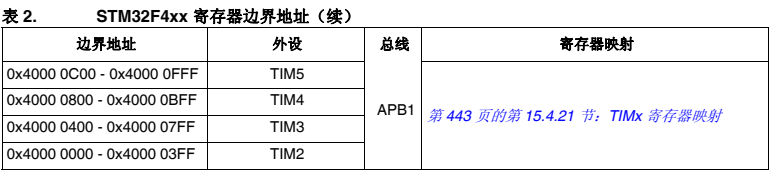
**中心对齐**：计数器从 0 开始计数到自动重载值（TIMx\_ARR 寄存器的内容）— 1，

生成计数器上溢事件；然后从自动重载值开始向下计数到 1 并生成计数器下溢 事件。之后从 0 开始重新计数。

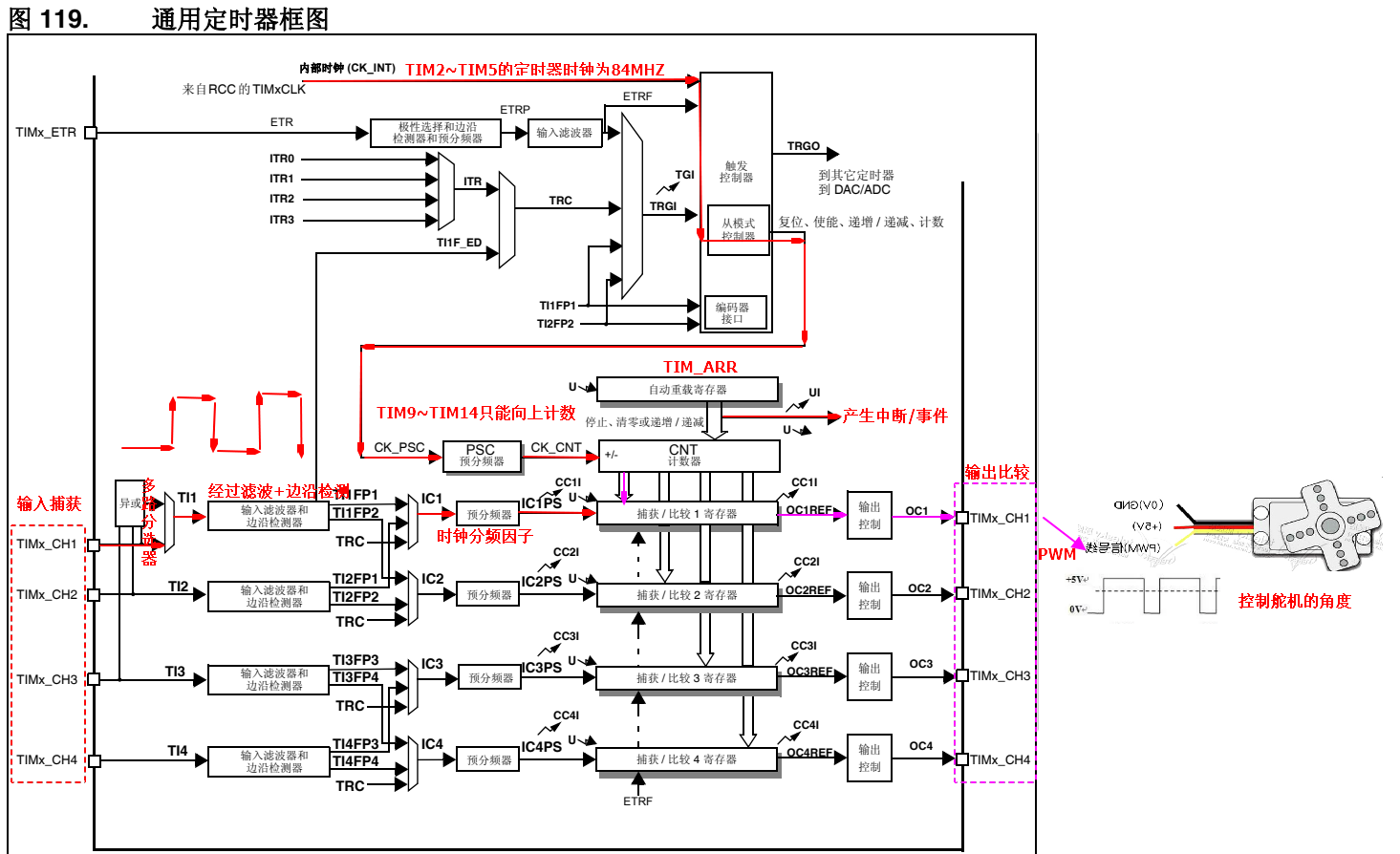


相比于基本定时器而言，通用定时器增加了**输入捕获**功能以及**输出比较**功能，并且通用定时器具有独立通道，就可以把GPIO引脚连接到某个通道，利用通道进行输入信号的检测以及脉冲信号的输出。

对于通用定时器TIM2~TIM5，都挂载在APB1定时总线下，定时器的频率为84MHZ，如下图



可以知道，通用定时器具有多种功能，分别为定时功能、输入捕获、输出比较，如下图所示



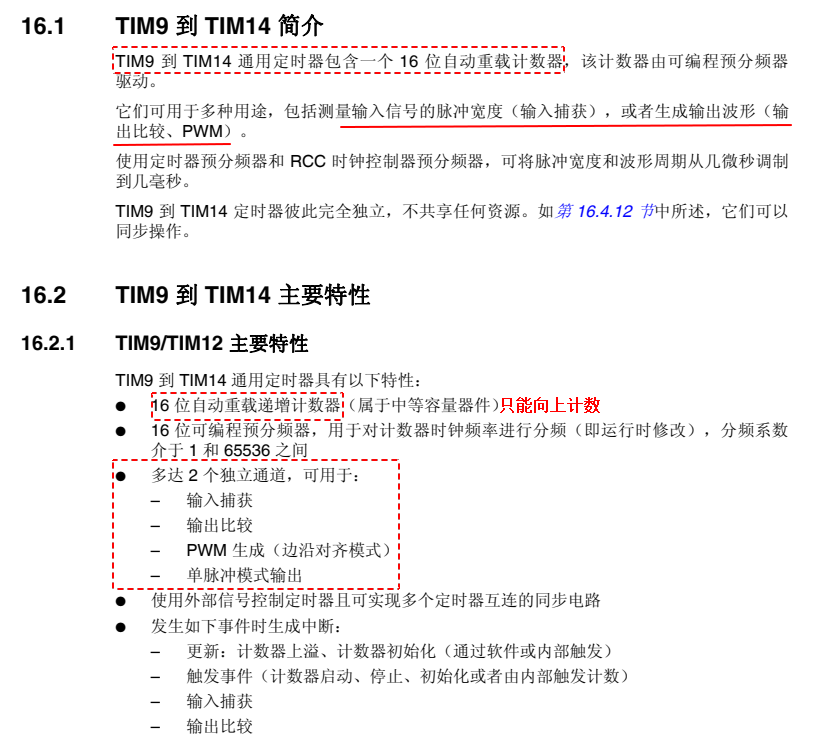
**定时功能**：基本上使用流程和基本定时器一致，只不过通用定时器的计数方式更灵活。

输入捕获：可以把定时器的某个通道连接到GPIO引脚上，然后从外部输入脉冲信号，经过 通道的滤波以及边沿检测之后，可以记录某个电平信号的脉冲宽度以及周期。

输出比较：可以把定时器的某个通道连接到GPIO引脚上，主动从引脚输出一个固定的脉冲， 原理很简单，其实就是计数器（TIM\_CNT）如果超过比较寄存器中的值，就可以 输出一个电平信号（高电平或者低电平）。

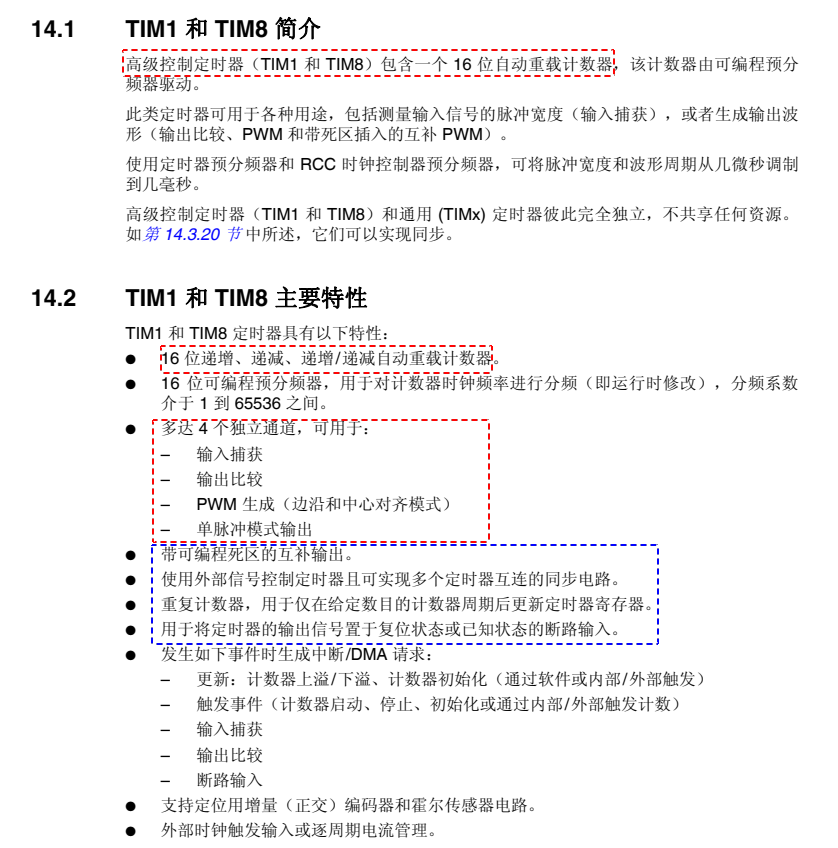
**练习：利用通用定时器的定时功能实现周期性的中断，要求利用4个通用定时器来控制4盏LED的电平翻转，要求LED0每隔200ms电平翻转，LED1每隔400ms电平翻转，LED2每隔500ms电平翻转，LED3每隔1s电平翻转。**

对于TIM9~TIM14而言，也可以进行定时功能，同样也具有输入捕获以及输出比较功能，但是只能采用向上计数的方式，并且相比于TIM2~TIM5，只有2个独立通道。



* 高级定时器

STM32中提供了TIM1和TIM8高级定时器，相比于其他定时器而言，高级定时器增加了带可编程死区控制的互补输出、重复计数器、断路输入功能（一般用于控制工业中的高精度电机），但是如果作为定时功能来说，和其他的定时器没啥区别。



高级定时器TIM1和TIM8都挂载在APB2总线下，所以TIM1和TIM8的定时时钟频率为168MHZ。

预习：中文参考手册的第26章 USART 串口

准备：蓝牙模块 + 超声波模块 + 温湿度模块 （记得明天带过来）

作业：利用STM32CubeMX实现定时器的定时功能，如利用图形界面实现LED灯的控制。