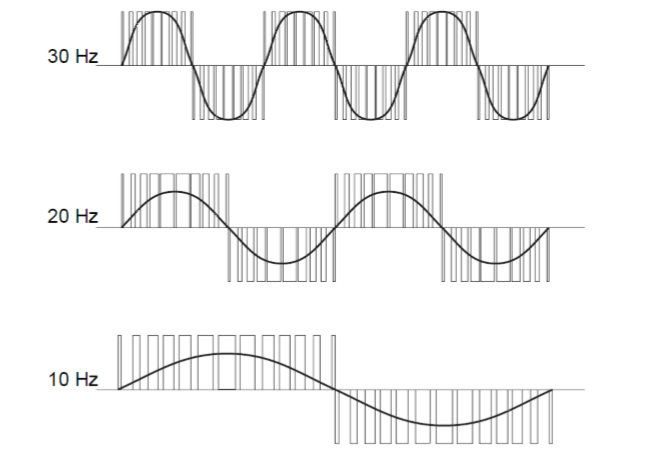
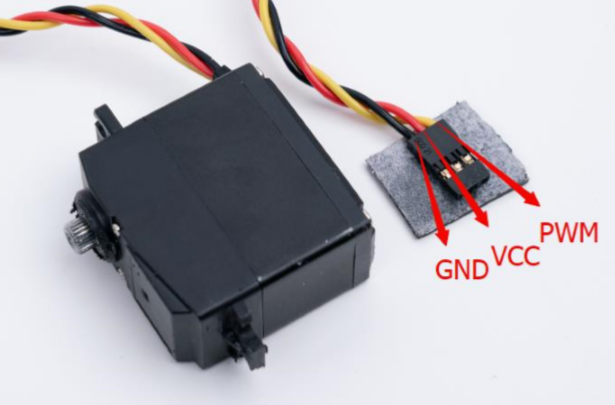
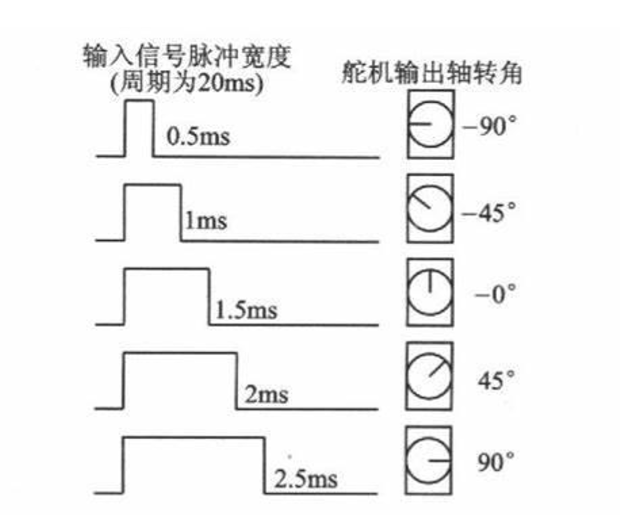
# PWM概念

1. PWM 即脉冲宽度调制是英文“Pulse Width Modulation”的缩写，简称脉宽调试。是利用微处理器的数字输出来对模拟电路进行控制的一种非常有效的技术
2. 
3. 一个周期内高电平的持续时间占总周期的比例成为占空比，通过修改占空比，可以改变输出的等效模拟电压。

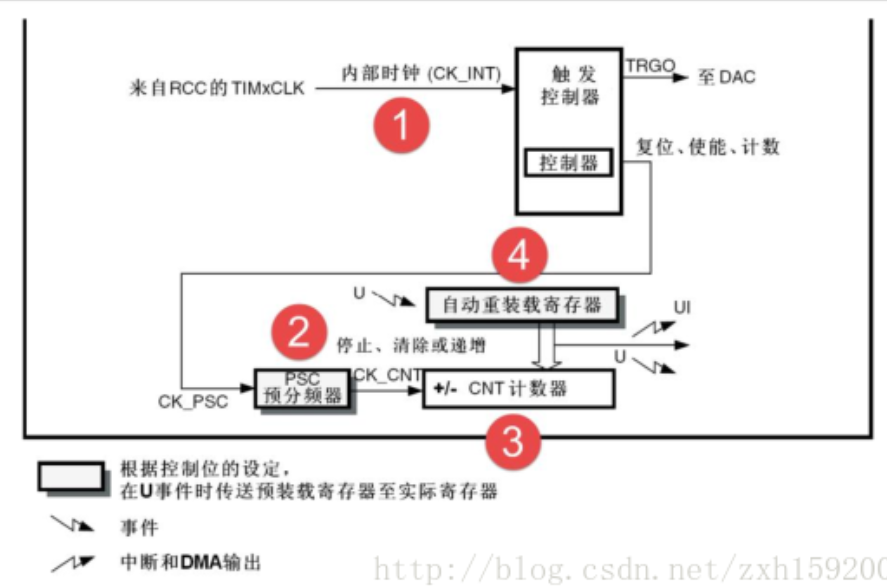
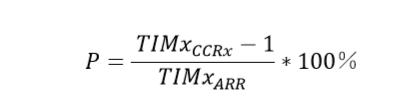
# 舵机概念

1. 舵机是机器人中的常见的执行部件，通常使用特定频率的 PWM 进行控制。舵机的主要组成部位由一个小型的电机和传动机构（齿轮组）构成，多被用于操控飞行器上的舵面，故而得名舵机。由于控制简单，价格便宜，在 RoboMaster 比赛中，用于简单的动 作控制，例如使用舵机控制弹仓盖的开合。下图为市场上常见的舵机。
2. 
3. 舵机使用的 PWM 信号一般为频率 50Hz，高电平时间 0.5ms-2.5ms 的 PWM 信号，不同占 空比的 PWM 信号对应舵机转动的角度，以 180 度舵机为例，对应角度图如下图所示。
4. 

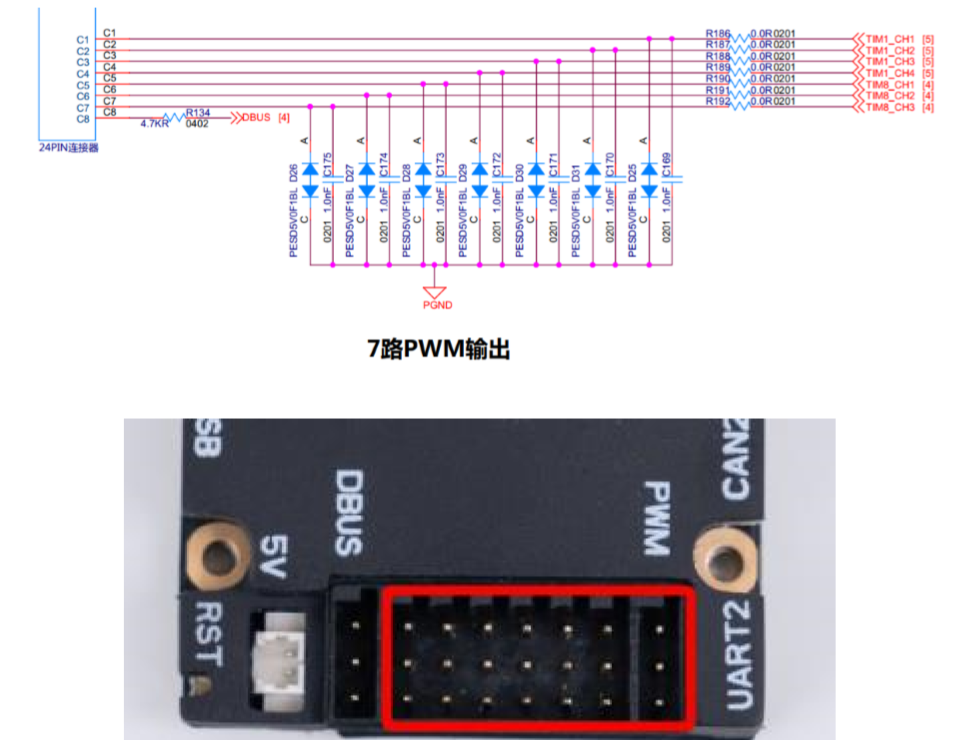
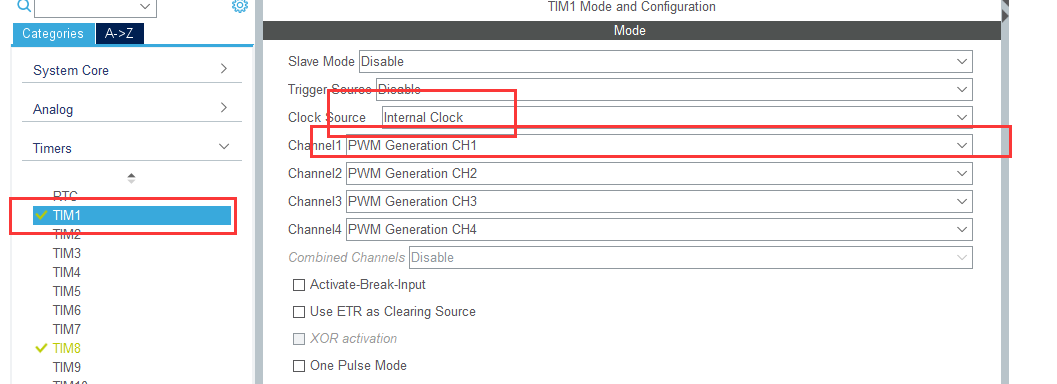
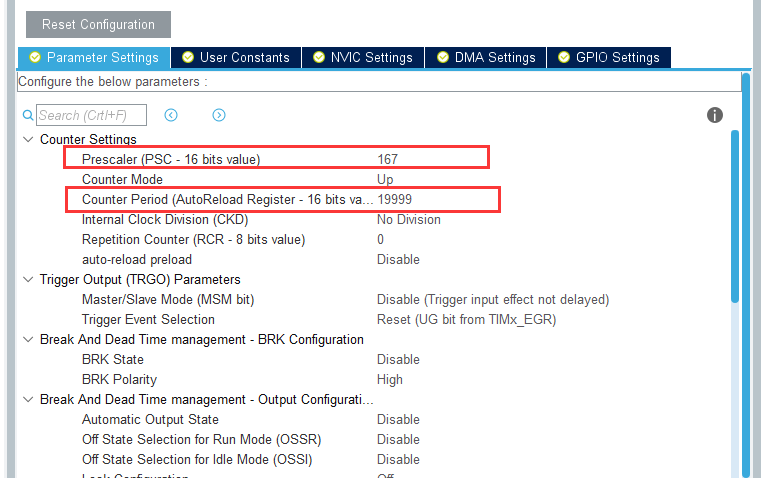
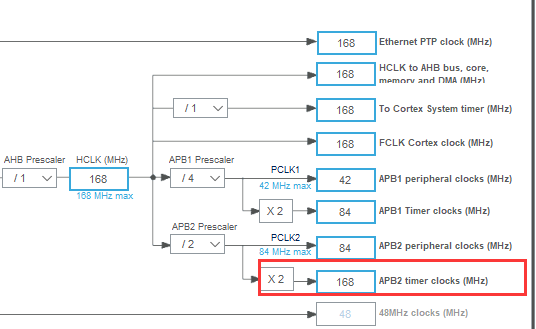
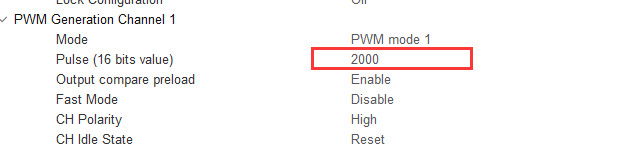
# STM32定时器的概念

1. 定时器的基本功能是计时功能，如同闹铃一般，设定好对应的时间后，会在设定的时刻响起铃声。
2. 在使用定时器时，会涉及到三个非常重要的概念——分频，计数，重载。这三个概念可以结合生活中使用的时钟来理解。
   1. 分频：时钟上不同的指针需要有不同的速度，也就是不同的频率，从而精确的表示时间，比如秒针，分针，时针，这三者相邻的频率之比都是 60:1，即秒针每转过 60 格分针转 动 1 格，分针转动 60 格时针转动 1 格，所以分针对于秒针的分频为 60。
   2. 计数：时钟所对应的值都是与工作时间成正比的，比如秒针转动 10 格，意味着过了 10 秒，同样定时器中的计数也是和计数时间成正比的值，频率越高增长速度越快。
   3. 重载：时、分、秒的刻度都是有上限的，一个表盘最多记 12 小时，60 分钟，60 秒， 如果继续增加的话就会回到 0。同样的在定时器中也需要重载，当定时器中的计数值达 到重载值时，计数值就会被清零。
3. 与定时器有关的三个重要定时器
   1. 预分频寄存器 TIMx\_PSC
      1. 时钟源处的时钟信号经过预分频寄存器，按照预分频寄存器内部的值进行分频。比如时钟源的频率为 16MHz，而预分频寄存器中设置的值为 16：1，那么通过预分频后进入定时器的 时钟频率就下降到了 1MHz。
   2. 计数器寄存器 TIMx\_CNT
   3. 自动重装载寄存器 TIMx\_ARR

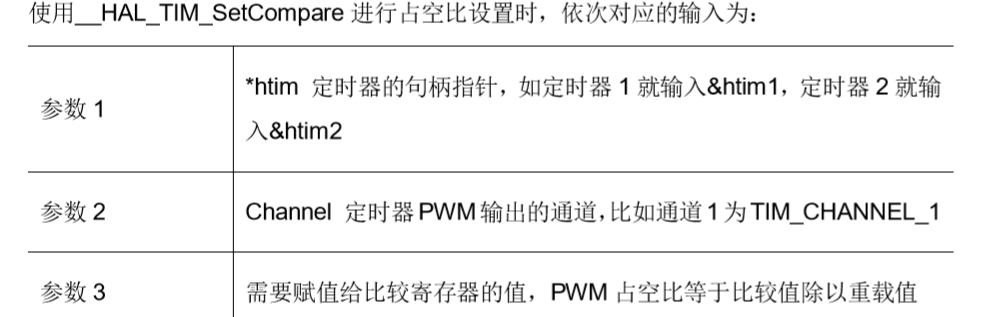
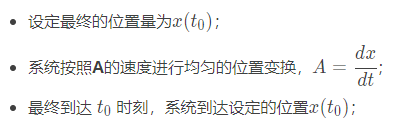
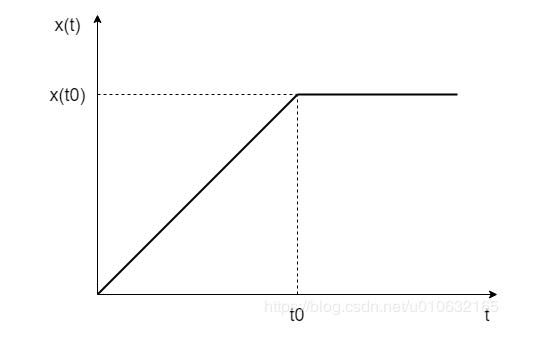
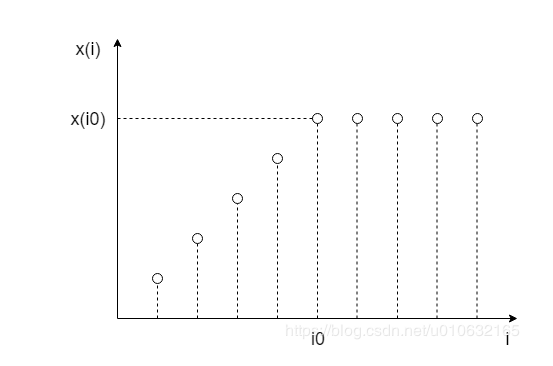
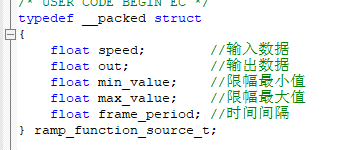
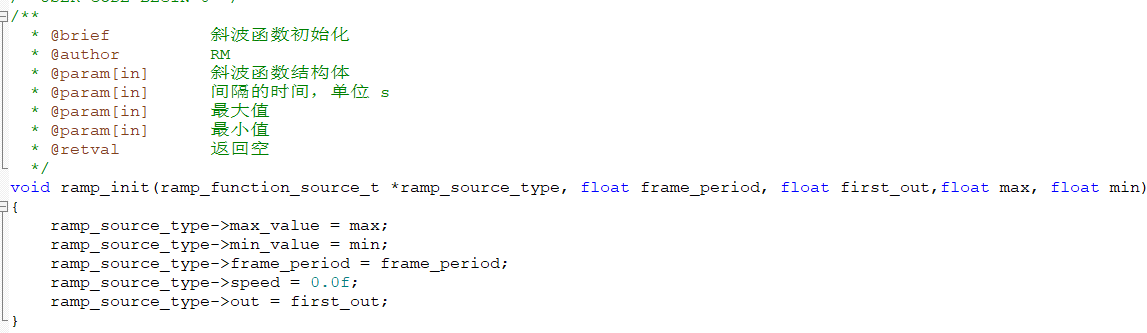
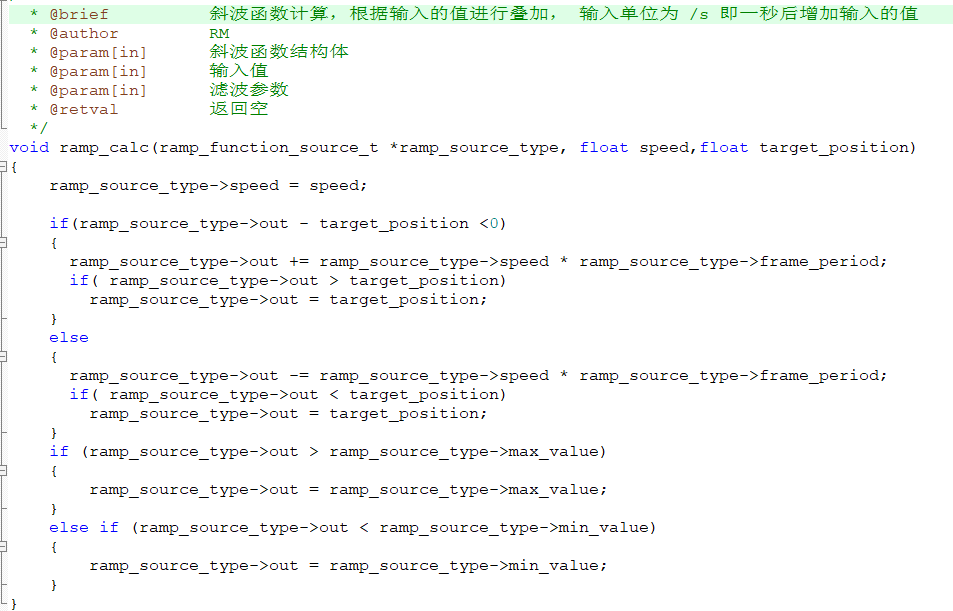
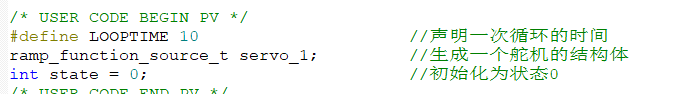
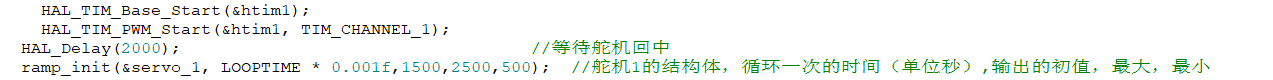
在已经分频后的定时器时钟驱使下，TIMx\_CNT 根据该时钟的频率向上计数，直到TIMx\_CNT 的值增长到与设定的自动重装载寄存器 TIMx\_ARR 相等时， TIMx\_CNT 被清空， 并重新从 0 开始向上计数，TIMx\_CNT 增长到 TIMx\_ARR 中的值后被清空时产生一个定时 中断触发信号。综上定时器触发中断的时间是由设定的 TIMx\_PSC 中的分频比和TIMx\_ARR 中的自动重装载值共同决定的。

1. 
2. 定时器中的比较寄存器（TIMx\_CCRx）
   1. 当定时器以 PWM模式工作时，会自动将 TIMx\_CCRx 的值与 TIMx\_CNT（计数寄存器）中 的值做比较，当 TIMx\_CNT 中的值小于 TIMx\_CCRx 的值时，PWM 输出引脚输出高电平，大于时则输出低电平。因此知道了 PWM 信号的周期和占空比可以通过设置比较寄存器 TIMx\_CCRx 和定时器重载寄存器 TIMx\_ARR 来控制。PWM的占空比可以通过下图公式计 算
      1. 
   2. 一个定时器工作在 PWM输出模式下时，有 4 个通道可以进行 PWM信号的输出，每一个定 时器都有对应标号的比较寄存器，比如 5 号定时器的 1 号通道对应的比较寄存器为TIM5\_CCR1。

# CUBEMX的配置

1. 首先查阅开发板手册，找到PWM输出引脚对应的引脚编号和对应的定时器
2. 
3. 比如我们现在使用TIM1 1通道的C1，即图中最右边的PWM引脚
4. 
5. 找到TIM1，选择内部时钟源，通道一设置为输出PWM
6. ‘
7. 在参数设置中将分配器设置为167，因为TIM1是挂载在APB2高速总线上的，速度为168MHZ，又因为分频器是从0开始计数的，填入167相当于将168MHZ的168分之一为1MHZ
8. 
9. 将重装载寄存器设置为19999，同理也是从0开始计数，相当于累加到20000的时候完成一个周期，因为信号源是上面的分频器，所以以1Mhz的速度不断累积，加到20000的时候时间正好为20ms，相当于50hz，满足驱动舵机的信号要求
10. 
11. 下面设置1通道的初始化比较寄存器，已经知道舵机在2ms的时候，相当于一个周期的十分之一，为45°角度，所以填入计数寄存器20000的十分之一，为2000。
    1. 由此可以扩展得到此程序可以控制舵机最小的旋转角度为：
       1. 舵机的行程-90~90°，对应的脉宽0.5ms~2.5ms，对应的比较寄存器值500~2500。舵机的可调范围为180°，对应比较寄存器2000的范围，于比较寄存器每增加1，对应舵机增加180/2000，为0.09°
       2. 也可以通过将预分频器减小，比如设置为83，那么进入计算器的脉冲频率就为2Mhz，要想还是输出50hz的PWM，那么重装载寄存器就可设置为原来的两倍40000，对应的比较寄存器范围1000~5000，那么舵机的最小移动量就为原来的一半，为0.045°，实现了更精细的控制。但是注意有的舵机并不能相应这么精细的控制。

# KEIL代码部分

1. HAL库关于定时器的函数
   1. HAL\_TIM\_Base\_Start(&htim1);开启定时器
   2. HAL\_TIM\_PWM\_Start(&htim1, TIM\_CHANNEL\_1);开启对应通道PWM
   3. \_\_HAL\_TIM\_SetCompare(&htim1, TIM\_CHANNEL\_1, 500);设置对应通道比较寄存器的值，可以实现动态改变舵机的角度
   4. 
2. 实验1：让舵机1在两个角度直接来回切换
   1. 在初始化函数中开启定时器和对应通道的PMW（实测可以指开启PWM，不需要开启定时器）
   2. 在循环函数中将对应比较寄存器设置为一个值，然后等待一会让舵机到达指定角度，然后设置为另一个角度，如此循环往复
3. 实验2：控制舵机角度的同时控制舵机的速度
   1. 斜坡函数
      1. 斜波函数就相当于控制系统中均速变换的位置信号，在三环控制的位置中，相当于这样一个过程;
      2. 
      3. 
      4. 
      5. 在每一次循环中获取递增的值，直到输出值达到了目标值，则一直保持目标值输出
   2. C语言实现
      1. 
      2. 定义结构体，结构体好处是可以很方便的实现多个舵机的控制
      3. 
      4. 初始化斜坡函数，将必要的参数保存在结构体
      5. 
      6. 斜波计算函数，输入目标的速度和位置，函数经过判断后，将当前的位置量累加上这一间隔需要增加的位置量。从而实现控制舵机旋速度
   3. 流程
      1. 
      2. 定义一次循环的时间单位毫秒，声明一个舵机实例，定义初始状态为0
      3. 
      4. 在初始PWM的时候等待舵机到达默认的位置，然后初始化斜坡函数，将舵机1的结构体，循环时间（单位s），舵机的初始位置，最大最小值带入
      5. 
      6. 在循环中使用switch语句判断状态，计算舵机在当前状态下下一个循环时刻输出的位置，判断是否到达，然后进入下一个状态。之后使用HAL函数将输出的值使用PWM发送到舵机，之后延迟一定的时间
      7. 