《手把手教你学STM32》

主讲人　：正点原子团队
硬件平台：正点原子STM32开发板
版权所有：广州市星翼电子科技有限公司
淘宝店铺：http://eboard.taobao.com
技术论坛：www.openedv.com 开源电子网
公众平台：“正点原子”
官方网站：www.alientek.com
联系电话：13922348612

ALIENTEK

《手把手教你学STM32》

■ PWM输出实验

适用平台

- ~~✓ STM32F1xx 开发板（正点原子）~~
- ✓ STM32F4xx 开发板（正点原子）

✓ PWM输出实验

■ 参考资料：

- ● 探索者STM32F4开发板：

  《STM32F4开发指南-库函数版本》-第14章 PWM实验

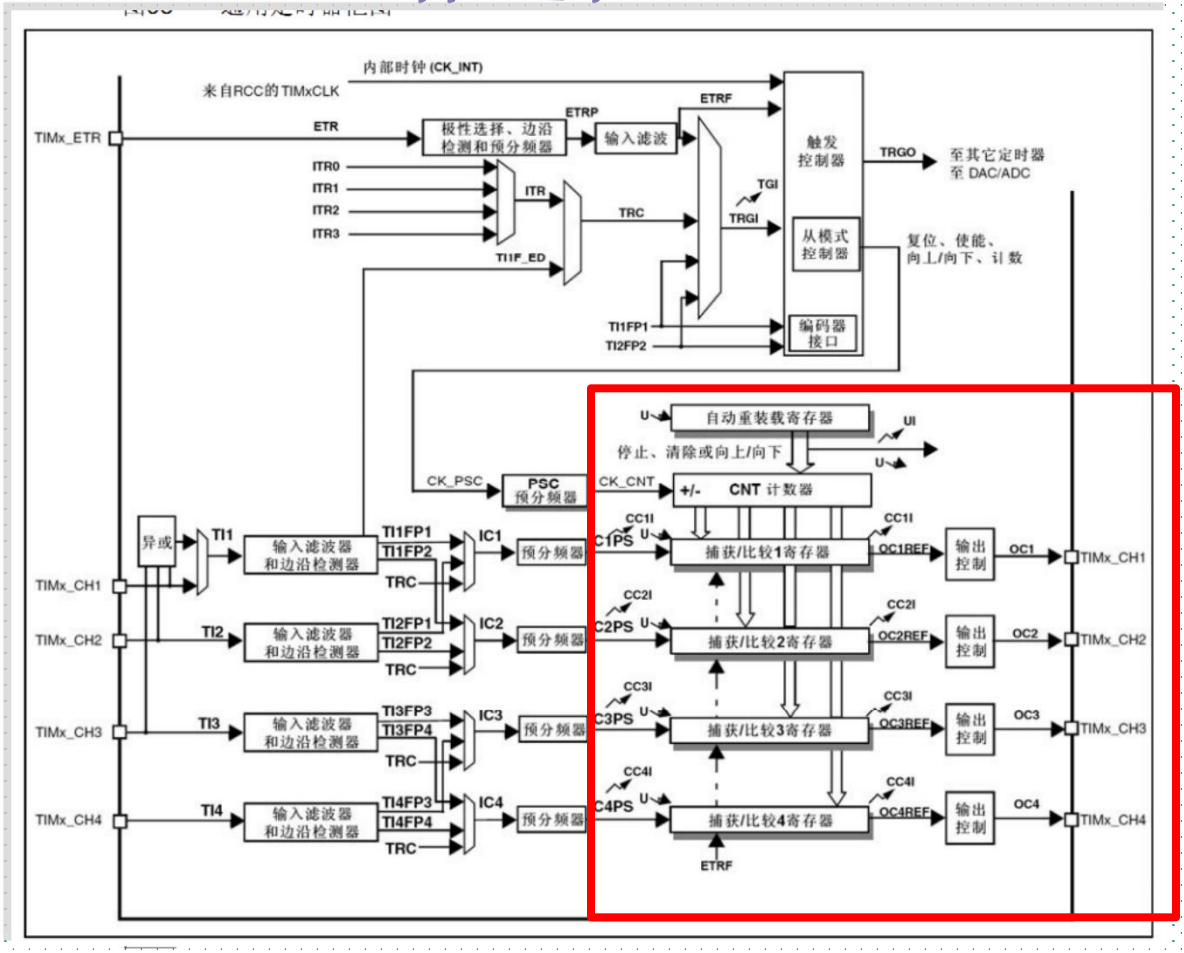
- □ STM32F4xx官方资料：

  《STM32F4xx中文参考手册》-第16章 通用定时器

目录

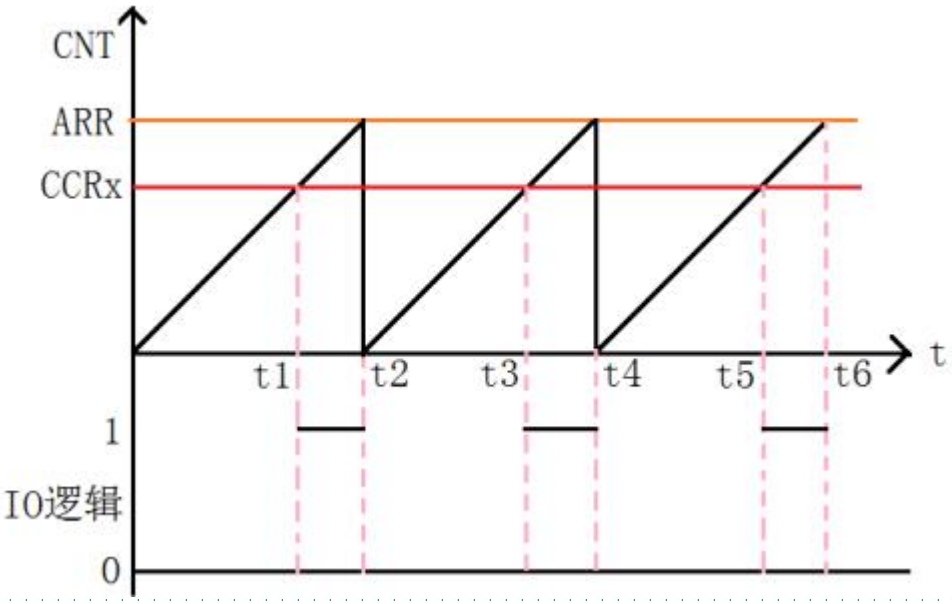
✓ 通用定时器PWM概述

◆STM32 PWM工作过程



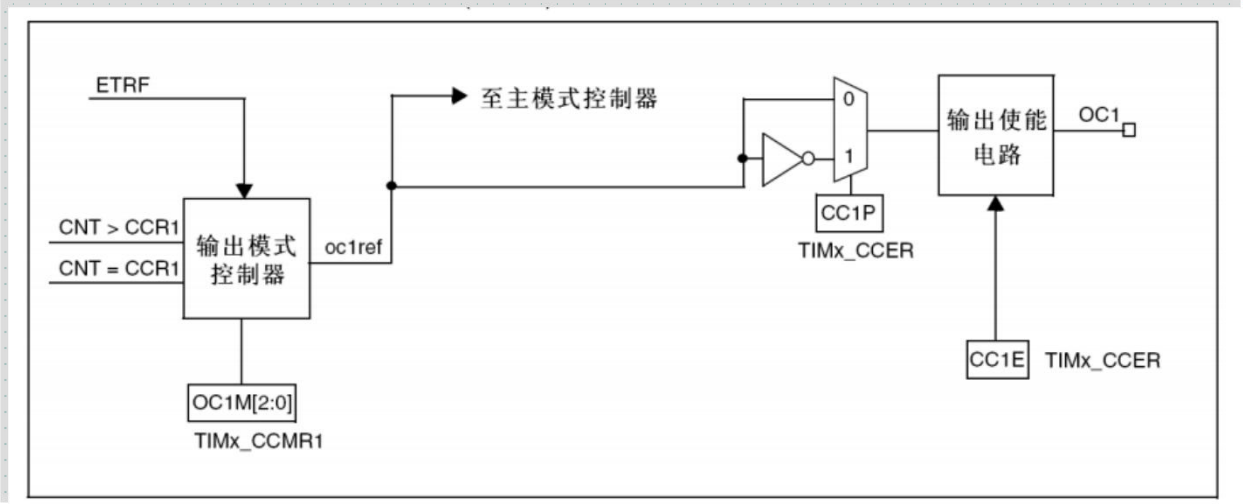
✓ 通用定时器PWM概述

◆STM32 PWM工作过程



✓ 通用定时器PWM概述

◆STM32 PWM工作过程（通道1为例）



CCR1:捕获比较(值)寄存器（x=1,2,3,4):设置比较值。

CCMR1: OC1M[2:0]位：

对于PWM方式下，用于设置PWM模式1【110】或者PWM模式2【111】

CCER:CC1P位：输入/捕获1输出极性。0：高电平有效，1：低电平有效。

CCER:CC1E位：输入/捕获1输出使能。0：关闭，1：打开。

✓ 通用定时器PWM概述

◆ PWM模式1 & PWM模式2

寄存器TIMx_CCMR1的OC1M[2:0]位来分析：

位6:4	**OC1M[2:0]**：输出比较1模式 (Output compare 1 enable) 该3位定义了输出参考信号OC1REF的动作，而OC1REF决定了OC1的值。OC1REF是高电平有效，而OC1的有效电平取决于CC1P位。 000：冻结。输出比较寄存器TIMx_CCR1与计数器TIMx_CNT间的比较对OC1REF不起作用； 001：匹配时设置通道1为有效电平。当计数器TIMx_CNT的值与捕获/比较寄存器1(TIMx_CCR1)相同时，强制OC1REF为高。 010：匹配时设置通道1为无效电平。当计数器TIMx_CNT的值与捕获/比较寄存器1(TIMx_CCR1)相同时，强制OC1REF为低。 011：翻转。当TIMx_CCR1=TIMx_CNT时，翻转OC1REF的电平。 100：强制为无效电平。强制OC1REF为低。 101：强制为有效电平。强制OC1REF为高。 110：PWM模式1－ 在向上计数时，一旦TIMx_CNT<TIMx_CCR1时通道1为有效电平，否则为无效电平；在向下计数时，一旦TIMx_CNT>TIMx_CCR1时通道1为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 111：PWM模式2－ 在向上计数时，一旦TIMx_CNT<TIMx_CCR1时通道1为无效电平，否则为有效电平；在向下计数时，一旦TIMx_CNT>TIMx_CCR1时通道1为有效电平，否则为无效电平。 注1：一旦LOCK级别设为3(TIMx_BDTR寄存器中的LOCK位)并且CC1S='00'(该通道配置成输出)则该位不能被修改。 注2：在PWM模式1或PWM模式2中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到PWM模式时，OC1REF电平才改变。

✓ 通用定时器PWM概述

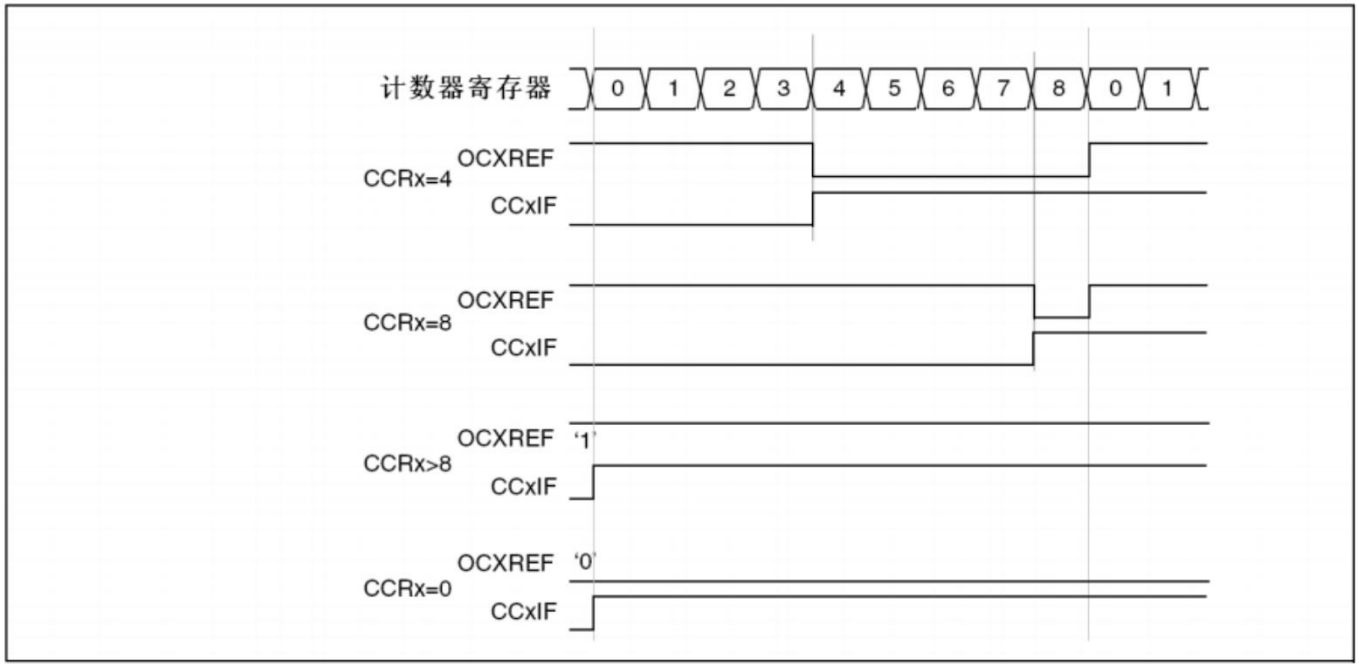
◆ STM32 PWM工作过程

向上计数配置

当TIMx_CR1寄存器中的DIR位为低的时候执行向上计数。参看14.3.2节。

下面是一个PWM模式1的例子。当TIMx_CNT<TIMx_CCRx时PWM信号参考OCxREF为高，否则为低。如果TIMx_CCRx中的比较值大于自动重装载值(TIMx_ARR)，则OCxREF保持为'1'。如果比较值为0，则OCxREF保持为'0'。 下图为TIMx_ARR=8时边沿对齐的PWM波形实例。

图128　边沿对齐的PWM波形(ARR=8)



✓ 通用定时器PWM概述

◆ STM32 PWM

PWM 模式

脉冲宽度调制模式可以产生一个由TIMx_ARR寄存器确定频率、由TIMx_CCRx寄存器确定占空比的信号。

在TIMx_CCMRx寄存器中的OCxM位写入'110'(PWM模式1)或'111'(PWM模式2)，能够独立地设置每个OCx输出通道产生一路PWM。必须设置TIMx_CCMRx寄存器OCxPE位以使能相应的预装载寄存器，最后还要设置TIMx_CR1寄存器的ARPE位，(在向上计数或中心对称模式中)使能自动重装载的预装载寄存器。

void TIM_OCxPreloadConfig(TIM_TypeDef TIMx, uint16_t TIM_OCPreload);*

void TIM_ARRPreloadConfig(TIM_TypeDef TIMx, FunctionalState NewState);*

✓ 通用定时器PWM概述

◆ 自动重载的预装载寄存器

图106　计数器时序图，当ARPE=1时的更新事件(预装入了TIMx_ARR)

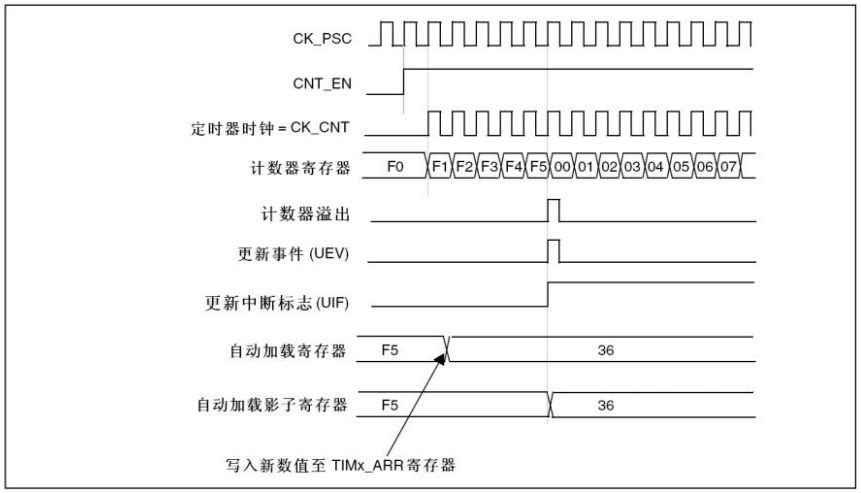
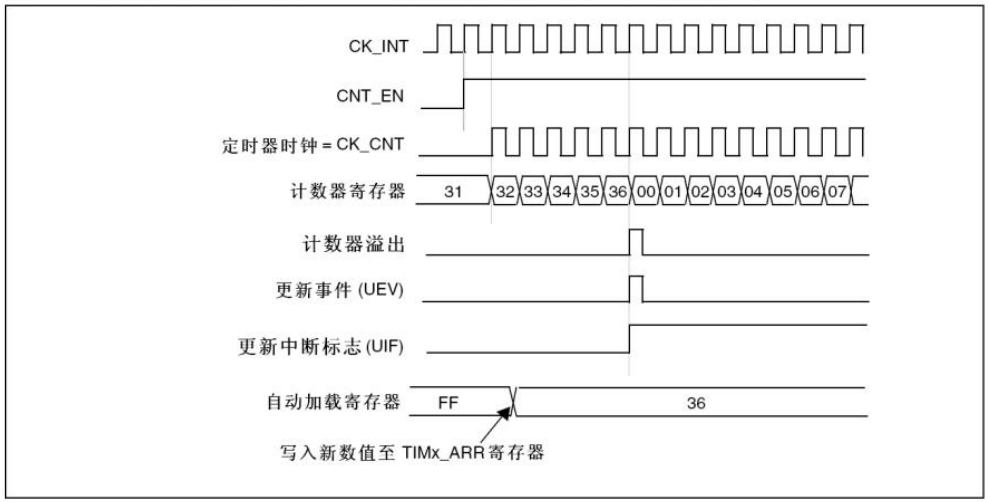


图105　计数器时序图，当ARPE=0时的更新事件(TIMx_ARR没有预装入)



void TIM_ARRPreloadConfig(TIM_TypeDef TIMx, FunctionalState NewState);*

简单的说，ARPE=1,ARR立即生效。。。APRE=0,ARR下个比较周期生效。

✓ 通用定时器PWM概述

◆STM32 定时器14输出通道引脚

PF9	I/O	FT	(4)	TIM14_CH1 / FSMC_CD/ EVENTOUT
PA7	I/O	FT	(4)	SPI1_MOSI/ TIM8_CH1N / TIM14_CH1/TIM3_CH2/ ETH_MII_RX_DV / TIM1_CH1N / RMII_CRS_DV/ EVENTOUT

这里需要纠正，通用定时器**9-14**，有的有**2**个通道，有的只有一个。

◆Datasheet中表格会有详细说明

✓ PWM输出库函数概述

◆ PWM输出库函数

void TIM_OCxInit(TIM_TypeDef TIMx, TIM_OCInitTypeDef* TIM_OCInitStruct);*

```
typedef struct
{
  uint16_t TIM_OCMode;  //PWM模式1或者模式2
  uint16_t TIM_OutputState; //输出使能 OR失能
  uint16_t TIM_OutputNState;
  uint16_t TIM_Pulse; //比较值，写CCRx
  uint16_t TIM_OCPolarity; //比较输出极性
  uint16_t TIM_OCNPolarity;
  uint16_t TIM_OCIdleState;
  uint16_t TIM_OCNIdleState;
} TIM_OCInitTypeDef;
```

```
TIM_OCInitStructure.TIM_OCMode = TIM_OCMode_PWM2; //PWM模式2
TIM_OCInitStructure.TIM_OutputState = TIM_OutputState_Enable; //比较输出使能
TIM_OCInitStructure. TIM_Pulse=100;
TIM_OCInitStructure.TIM_OCPolarity = TIM_OCPolarity_High; //输出极性:TIM输出比较极性高
TIM_OC2Init(TIM3, &TIM_OCInitStructure);  //根据T指定的参数初始化外设TIM3 OC2
```

✓ PWM输出库函数概述

◆ 设置比较值函数：

void TIM_SetCompareX(TIM_TypeDef TIMx, uint16_t Comparex);*

◆ 使能输出比较预装载：

void TIM_OCxPreloadConfig(TIM_TypeDef TIMx, uint16_t TIM_OCPreload);*

◆ 使能自动重装载的预装载寄存器允许位：

void TIM_ARRPreloadConfig(TIM_TypeDef TIMx, FunctionalState NewState);*

✓ 手把手写PWM输出实验

◆ 要求：

使用定时器14的PWM功能，输出占空比可变的PWM波，用来驱动LED灯，从而达到LED【PF9]亮度由暗变亮，又从亮变暗，如此循环。

✓ 手把手写PWM输出实验

◆PWM输出配置步骤：

① 使能定时器14和相关IO口时钟。

使能定时器14时钟：RCC_APB1PeriphClockCmd();

使能GPIOF时钟：RCC_AHB1PeriphClockCmd ();

② 初始化IO口为复用功能输出。函数：GPIO_Init();

GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF; //复用功能

③ GPIOF9复用映射到定时器14

GPIO_PinAFConfig(GPIOF,GPIO_PinSource9,GPIO_AF_TIM14);

④ 初始化定时器：ARR,PSC等：*TIM_TimeBaseInit();*

⑤ 初始化输出比较参数:*TIM_OC1Init();*

⑥ *使能预装载寄存器：TIM_OC1PreloadConfig(TIM14, TIM_OCPreload_Enable);*

⑦ 使能自动重装载的预装载寄存器允许位*TIM_ARRPreloadConfig(TIM14,ENABLE);*

⑧ 使能定时器。

⑨ 不断改变比较值CCRx，达到不同的占空比效果:*TIM_SetCompare1();*

✓ 3. 手把手写代码

手把手写PWM输出实验

GO!!

谢谢您对“正点原子”团队的支持

硬件平台：正点原子STM32开发板
版权所有：广州市星翼电子科技有限公司
淘宝店铺：http://eboard.taobao.com
技术论坛：www.openedv.com