函数入口参数列表用结构体 整合。 修改函数参数，只需要修改结构体成员变量

APB2 GPIO,ADC,TIM,SPI

APB1 DAC,USB,I2C,UART,SPI,TIM

时钟配置

1. SystemInit

1） 复位 RCC 时钟配置为默认复位值（默认开始了 HIS）

2） 外部存储器配置

3） 中断向量表地址配置

2.Stm32\_Clock\_Init(u32 PLL）

1）配置 PLL 相关参数确定 SYSCLK 值

2）配置了 AHB,APB1 和 APB2 的分频系数，也就是确定了 HCLK，PCLK1 和 PCLK2 的时钟值。

3.外设时钟使能(或禁止)

GPIOx\_CLK\_ENABLE........................

端口复用（以GPIO和串口为例，引脚默认为GPIO）

1）先打开对应的 IO 时钟和复用功能外设时钟（使能 GPIOA 和 USART1 时 钟）

2）在 GIPOx\_MODER 寄存器中将所需 IO（对于串口 1 是 PA9,PA10）配置 为复用AF功能。

3）还需要对 IO 口的其他参数，例如上拉/下拉以及输出速度等进行配置

GPIOA 的 7 个寄存器都是 32 位的，所以每个寄存器占有 4

个地址，一共占用 28 个地址，地址偏移范围为（000h~01Bh）

GPIOA指向GPIOA寄存器的基地址（可由上层模块（AHP2）基地址+偏移量算出），结构体GPIOA将这一块区域整合起来，因为结构体存储成员地址是连续的，所以结构体顺序要求和寄存器地址一致

调用函数

1. 从头文件里，看函数参数def，定义参数并初始化（参考库文件）assert\_param判断参数是否有效
2. 调用

每个IO口7个寄存器控制（模式，数据，置位复位，锁存）

0X0 表示模拟输入模式（ADC 用）、0X3 表示推挽输出模式（做输出口用， 50M 速率）、0X8 表示上/下拉输入模式（做输入口用）、0XB 表示复用输出（使用 IO 口的第二 功能，50M 速率）。

Led.c .h里面外设define重定义，给pin重命名

1.io口初始化 开启外设GPIO时钟（配置外设，使能时钟，初始化GPIO pin

typedef struct

{

uint32\_t Pin; 32位，可装地址

uint32\_t Mode;

uint32\_t Pull;

uint32\_t Speed;

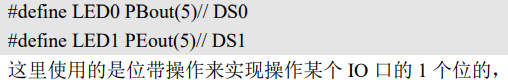
}GPIO\_InitTypeDef

void HAL\_GPIO\_Init(GPIO\_TypeDef \*GPIOx, GPIO\_InitTypeDef \*GPIO\_Init);

0000第3位到第0位

1.位带操作：我们往 Address0 地址写入 1，那么就可以达到往寄存器的第 0 位 Bit0 赋值1 的目的.

2.等同于对结构体中定义变量赋值





端口复用：使能GPIO和复用外设两个时钟，同时把GPIO口设置成复用模式（端口默认可以当做一个额外外设的端口）

端口重映射：即外设默认端口映射到其他pin上去，除了使能上面两个时钟外，使能AIFO时钟，调用重映射函数（更换端口复用默认的外设端口，改为第二套默认端口）



GPIO.h中获取可重映射的外设端口

初始化操作一样

初始化一个结构体；给结构体变量赋值；结构体变量作为函参传入所需初始化函数

1.GPIO

1） 使能 IO 口时钟。调用函数为 RCC\_APB2PeriphClockCmd()。

2） 初始化 IO 参数。调用函数 GPIO\_Init();

3） 操作 IO。操作 IO 的方法就是上面我们讲解的方法。



2.串口

1) 串口时钟使能，GPIO 时钟使能

2) 串口复位

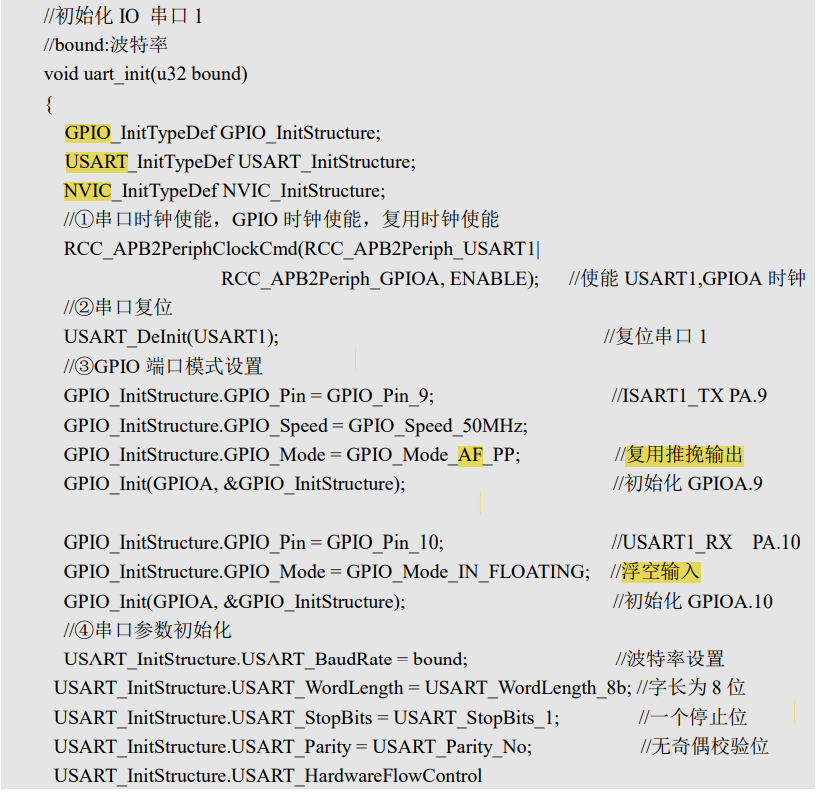
3) GPIO 端口模式设置

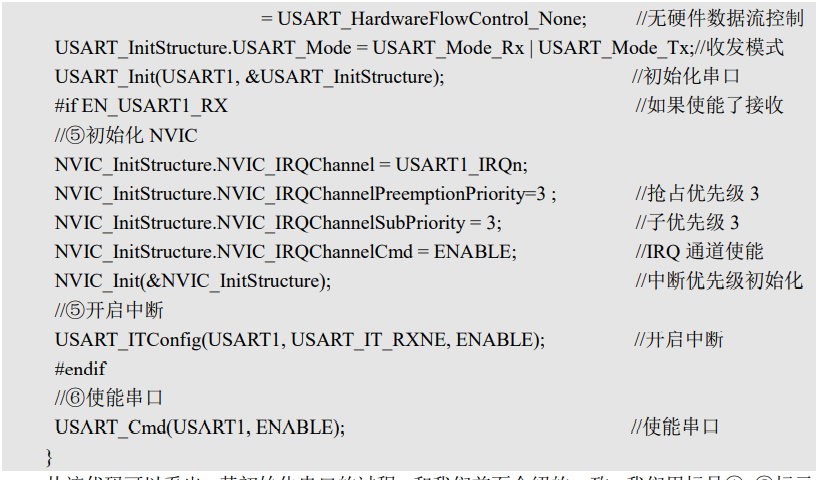
4) 串口参数初始化

5) 开启中断并且初始化 NVIC（如果需要开启中断才需要这个步骤）

6) 使能串口

7) 编写中断处理函数





**3.中断NVIC**

Register:使能，挂起，激活标志，优先级(分组，确定抢占和响应优先级位数void NVIC\_PriorityGroupConfig(uint32\_t NVIC\_PriorityGroup);) （主函数）

void NVIC\_Init(NVIC\_InitTypeDef\* NVIC\_InitStruct)

**4.IO 口外部中断 （以按键为例）**

1）初始化 IO 口为输入。

2）开启 AFIO 时钟 （pin口连接中短线，也需要打开）

3）设置 IO 口与中断线的**映射**关系。

GPIO\_EXTILineConfig(GPIO\_PortSourceGPIOE,GPIO\_PinSource2);

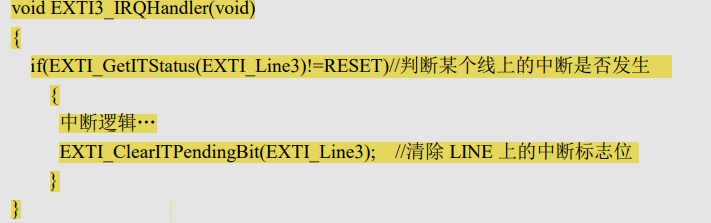
4）初始化线上中断，设置触发条件等。

5）配置中断分组（NVIC），并使能中断。

6）编写中断服务函数。

Tip：一个pin首先默认连接gpio，默认复用一个外设端口。**要更换默认外设端口，或者搭载一根中断线**，则需开启AFIO时钟。并且调用映射配置函数。

以线 0 为例：它对应了 GPIOA.0、GPIOB.0、GPIOC.0、GPIOD.0、 GPIOE.0、GPIOF.0、GPIOG.0。而中断线每次只能连接到 1 个 IO 口上，这样就需要通过配置来 决定对应的中断线配置到哪个 GPIO 上了GPIO\_EXTILineConfig(uint8\_t GPIO\_PortSource, uint8\_t GPIO\_PinSource)



**时钟**

SYSCLK（最大72MHZ，可在系统时钟初始化函数中设置，修改define）

经AHB分频

提供给APB1 (tim2-7) , APB2(gpio,adc，TIM1.8),

HCLK（总线，内核，内存。DMA）

-systick，fclk

这里总结一下 SystemInit()函数中设置的系统时钟大小：

SYSCLK（系统时钟） =72MHz AHB

总线时钟(使用 SYSCLK) =72MHz APB1

总线时钟(PCLK1) =36MHz APB2

总线时钟(PCLK2) =72MHz

PLL 时钟 =72MHz

**寄存器**

控制CR(使能、计数方式、对齐方式、分频)

DMA/中断使能寄存器DIER(更新中断允许位)

预分频PSC【15:0】

时钟来源SMCR

CNT

自动重装载ARR

状态标记SR

捕获比较模式CCMR1/2 高低八位，四个通道

CCER

CCR存储发生捕获时，cnt数据

功能

16位CNT

16位预分频器0 - 2^16-1

4个独立通道CH1-4(输入捕获、 输出比较 、 pwm 、 单脉冲)

外部信号控制定时器互联

中断（cnt计满更新、 触发…….）

**定时器中断**

1. TIM3 时钟使能。
2. 初始化定时器参数,设置自动重装值，分频系数，计数方式等。

3）设置 TIM3\_DIER 允许更新中断，并设置中断类型（区别gpio中断线）

4）TIM3 中断优先级设置。

5）**允许 TIM3 工作，也就是使能 TIM3。**

6）编写中断服务函数。（判断中断是否符合预定类型，清除中断标志位）

定时器pwm输出

定时器输出到pin，首先就要开启定时器和gpio时钟。因为pin端口默认复用不是定时器，所以开启AFIO时钟，并且调用重映射函数，将定时器输出映射到pin管脚。Gpio端口输出模式为复用推挽输出

**Tip**：gpio中断和定时器输出都要开启AFIO

3）初始化gpio。初始化 TIM3,设置 TIM3 的 ARR 和 PSC。

4）初始化channel2pwm模式，设置 TIM3\_CH2 的 PWM 模式，使能 TIM3 的 CH2 输出。

5）使能 TIM3。

6）修改 TIM3\_CCR2 来控制占空比。



定时器输入捕获

1. 开启TIM,GPIO时钟
2. 初始化TIM
3. 设置TIM输入比较参数，开启输入捕获
4. 使能捕获，更新中断DIER
5. 设置中断分组，调用中断服务函数
6. 使能定时器

