

# STC 库函数

版本：V10

日期：2014-5-28

编写：STC 公司

## 基本说明：

本函数库适用于 STC/IAP15F/L2K61S2 系列 MCU，具体的 MCU 的资源，请参考用户手册中对应的章节。

使用以下的库函数，都必须包含“config.h”文件，里面包含了“STC15Fxxxx.H”头文件。

在自己的工程中，加入库函数文件，并且在使用库函数的 C 文件中，包含对应的头文件。

这个文档仅仅解释各个库函数里的参数定义和取值，具体的应用例子，请参考例程。

相关功能的具体描述，请参考用户手册。

## 函数目录

IO 口初始化函数：GPIO_Inilize	2
定时器初始化函数：Timer_Inilize	3
ADC 初始化函数：ADC_Inilize	4
ADC 电源控制函数：ADC_PowerControl	6
ADC 查询转换函数：Get_ADC10bitResult	6
通用软件延时函数：delay_ms	7
串口初始化函数：USART_Configuration	7
串口 1 写缓冲函数：TX1_write2buff	9
串口 2 写缓冲函数：TX2_write2buff	10
串口 1 写数据块函数：PrintString1	10
串口 2 写数据块函数：PrintString2	10
模拟串口字节发送函数：TxSend	10
模拟串口写数据块函数：PrintString	11
EEPROM 多字节读函数：EEPROM_read_n	11
EEPROM 多字节写函数：EEPROM_write_n	11
EEPROM 扇区擦除函数：EEPROM_SectorErase	12
PCA 初始化函数：PCA_Init	12
PWM 更新占空比函数：UpdatePwm	15
外中断初始化函数：Ext_Inilize	15

## IO 口初始化函数：GPIO\_Initalize

函数名	GPIO_Initalize
函数原形	u8 GPIO_Initalize(u8 GPIO, GPIO_InitTypeDef *GPIOx)
所在文件	GPIO.c
功能描述	对 IO 口初始化
输入参数 1	GPIO: 选择以下之一: GPIO_P0, GPIO_P1, GPIO_P2, GPIO_P3, GPIO_P4, GPIO_P5
输入参数 2	GPIOx: 配置 IO 口的指针, 指定配置的引脚和输入、输出方式, 见下表描述。
返回	U8, 返回 0 表示配置成, 返回非 0 表示配置错误。

GPIO\_InitTypeDef 的定义见于文件“GPIO.H”。

typedef struct

```
{
    u8  Mode;
    u8  Pin;    //要设置的端口
} GPIO_InitTypeDef;
```

Mode: 配置 IO 的模式, 取值见下表:

Mode 取值	功能描述
GPIO_PullUp	准双向口, 内部弱上拉, 可以输出, 也可以当输入, 当输入时, 要先写 1。
GPIO_HighZ	高阻输入, 只能做输入。
GPIO_OUT_OD	开漏输出, 输出 0 时拉低, 输出 1 时高阻, 可以做输入/输出。
GPIO_OUT_PP	推挽输出, 只能做输出, 根据具体电路, 可能要串电阻以限制电流。

Pin: 要配置的某一个或多个 IO, 取值见下表:

Pin 取值	功能描述
GPIO_Pin_0	配置 Px.0。
GPIO_Pin_1	配置 Px.1。
GPIO_Pin_2	配置 Px.2。
GPIO_Pin_3	配置 Px.3。
GPIO_Pin_4	配置 Px.4。
GPIO_Pin_5	配置 Px.5。
GPIO_Pin_6	配置 Px.6。
GPIO_Pin_7	配置 Px.7。
GPIO_Pin_All	配置整个 8 位口。

以上参数可以使用或运算, 比如:

```
GPIO_InitStructure.Pin = GPIO_Pin_0 | GPIO_Pin_1 | GPIO_Pin_7;
```

## 定时器初始化函数: Timer\_Inilize

函数名	Timer_Inilize
函数原形	u8 Timer_Inilize(u8 TIM, TIM_InitTypeDef *TIMx)
所在文件	Timer.c
功能描述	对定时器初始化
输入参数 1	TIM: 选择以下之一: Timer 0, Timer 1, Timer 2。
输入参数 2	TIMx: 配置定时器的指针, 指定配置的功能, 见下表描述。
返回	U8, 返回 0 表示配置成, 返回非 0 表示配置错误。

TIM\_InitTypeDef 的定义见于文件 “timer.H”。

typedef struct

```
{
    u8  TIM_Mode;           //工作模式
    u8  TIM_Polity;         //优先级设置
    u8  TIM_Interrupt;     //中断允许
    u8  TIM_ClkSource;     //时钟源
    u8  TIM_ClkOut;        //可编程时钟输出
    u16 TIM_Value;         //装载初值
    u8  TIM_Run;           //是否运行
} TIM_InitTypeDef;
```

TIM\_Mode: 定时器的工作模式:

TIM_Mode 取值	功能描述
TIM_16BitAutoReload	配置成 16 位自动重装模式。
TIM_16Bit	配置成 16 位 (手工重装) 模式。
TIM_8BitAutoReload	配置成 8 位自动重装模式。
TIM_16BitAutoReloadNoMask	配置成 16 位自动重装模式, 中断自动打开, 并且不能屏蔽 (禁止)。

TIM\_Polity: 中断的优先级:

TIM_Polity 取值	功能描述
PolityHigh	中断设置为高优先级。
PolityLow	中断设置为低优先级 (默认)。

**TIM\_Interrupt:** 中断允许或禁止:

TIM_Interrupt 取值	功能描述
ENABLE	允许中断。
DISABLE	禁止中断（默认）。

**TIM\_ClkSource:** 定时器的时钟源选择:

TIM_ClkSource 取值	功能描述
TIM_CLOCK_1T	定时器的时钟使用 系统时钟 1T 模式。
TIM_CLOCK_12T	定时器的时钟使用 系统时钟 12T 模式。
TIM_CLOCK_Ext	定时器的时钟使用外部输入。

**TIM\_ClkOut:** 定时器溢出时取反对应 IO 输出高速时钟:

TIM_ClkOut 取值	功能描述
ENABLE	允许定时器溢出时取反对应 IO 输出高速时钟。
DISABLE	禁止定时器溢出时取反对应 IO 输出高速时钟。

**TIM\_Value:** 一个 16 位的初值。

**TIM\_Run:** 初始化后是否运行定时器:

TIM_Run 取值	功能描述
ENABLE	初始化后运行定时器。
DISABLE	初始化后停止定时器。

## ADC 初始化函数: ADC\_Inilize

函数名	ADC_Inilize
函数原形	void ADC_Inilize(ADC_InitTypeDef *ADCx)
所在文件	Adc.c
功能描述	对 ADC 初始化
输入参数 1	ADCxx: 配置 ADC 的指针, 见下面描述。
返回	无

ADC\_InitTypeDef 的定义见于文件“ADC.H”。

```
typedef struct
```

```
{
    u8  ADC_Px;           //设置要做 ADC 的 IO,ADC_P10 ~ ADC_P17,ADC_P1_All
    u8  ADC_Speed;        //ADC 速度          ADC_90T,ADC_180T,ADC_360T,ADC_540T
    u8  ADC_Power;        //ADC 功率允许/关闭  ENABLE,DISABLE
    u8  ADC_AdjResult;    //ADC 结果调整,  ADC_RES_H2L8,ADC_RES_H8L2
    u8  ADC_Polity;       //优先级设置      PolityHigh,PolityLow
    u8  ADC_Interrupt;    //中断允许      ENABLE,DISABLE
} ADC_InitTypeDef;
```

ADC\_Px: 设置要做 ADC 的 IO:

ADC_Px 取值	功能描述
ADC_P10	设置 P1.0 为 ADC 输入口。
ADC_P11	设置 P1.1 为 ADC 输入口。
ADC_P12	设置 P1.2 为 ADC 输入口。
ADC_P13	设置 P1.3 为 ADC 输入口。
ADC_P14	设置 P1.4 为 ADC 输入口。
ADC_P15	设置 P1.5 为 ADC 输入口。
ADC_P16	设置 P1.6 为 ADC 输入口。
ADC_P17	设置 P1.7 为 ADC 输入口。
ADC_P1_All	配置整个 8 位 P1 口为 ADC 输入。

以上参数可以使用或运算，比如：

```
ADC_InitStructure.ADC_Px = ADC_P10 | ADC_P11 | ADC_P12;
```

ADC\_Speed: 设置 ADC 的速度:

ADC_Speed 取值	功能描述
ADC_90T	设置 ADC 时钟为 90 个主时钟周期。
ADC_180T	设置 ADC 时钟为 180 个主时钟周期。
ADC_360T	设置 ADC 时钟为 360 个主时钟周期。
ADC_540T	设置 ADC 时钟为 540 个主时钟周期。

ADC\_Power: ADC 电源控制:

ADC_Power 取值	功能描述
ENABLE	初始化后打开 ADC 电源。
DISABLE	初始化后关闭 ADC 电源。

ADC\_AdjResult: ADC 结果调整:

ADC_AdjResult 取值	功能描述
ADC_RES_H2L8	ADC 结果寄存器高字节为结果的高 2 位, 低字节为低 8 位。
ADC_RES_H8L2	ADC 结果寄存器高字节为结果的高 8 位, 低字节为低 2 位。

ADC\_Polity: 中断的优先级:

ADC_Polity 取值	功能描述
PolityHigh	中断设置为高优先级。
PolityLow	中断设置为低优先级 (默认)。

ADC\_Interrupt: 中断允许或禁止:

ADC_Interrupt 取值	功能描述
ENABLE	允许中断。
DISABLE	禁止中断 (默认)。

## ADC 电源控制函数: ADC\_PowerControl

函数名	ADC_PowerControl
函数原形	void ADC_PowerControl(u8 pwr)
所在文件	Adc.c
功能描述	开/关 ADC 电源。
输入参数 1	pwr: 取值 ENABLE 打开 ADC 电源, 取值 DISABLE 关闭 ADC 电源。
返回	无

## ADC 查询转换函数: Get\_ADC10bitResult

函数名	Get_ADC10bitResult
函数原形	u16 Get_ADC10bitResult(u8 channel)
所在文件	Adc.c
功能描述	查询方式进行一次 ADC 转换。
输入参数 1	channel: 要进行转换的 ADC 通道, 取值 0~7 其中一个, 对应 P1.0~P1.7。
返回	10 位 ADC 值。

**通用软件延时函数：delay\_ms**

函数名	delay_ms
函数原形	void delay_ms(unsigned char ms)
所在文件	Delay.c
功能描述	延时程序。包含 config.h，延时时间会根据主频自动适应。
输入参数 1	ms：延时的 ms 数，1~255。
返回	无

**串口初始化函数：USART\_Configuration**

函数名	USART_Configuration
函数原形	u8 USART_Configuration(u8 UARTx, COMx_InitDefine *COMx)
所在文件	Usart.c
功能描述	对串口初始化
输入参数 1	UARTx：要初始化的串口，取值以下之一：USART1，USART2。
输入参数 2	COMx：配置串口的参数指针。
返回	U8，返回 0 表示配置成，返回非 0 表示配置错误。

COMx\_InitDefine 的定义见于文件“USART.H”。

typedef struct

```
{
    u8  UART_Mode;
    u8  UART_BRT_Use;
    u32 UART_BaudRate;
    u8  Morecommunicate;
    u8  UART_RxEnable;
    u8  BaudRateDouble;
    u8  UART_Interrupt;
    u8  UART_Polity;
    u8  UART_P_SW;
    u8  UART_RXD_TXD_Short;
```

```
} COMx_InitDefine;
```

UART\_Mode: 设置 USART 的工作模式:

UART_Mode 取值	功能描述
UART_ShiftRight	串口工作于同步输出方式, 仅仅用于 USART1。
UART_8bit_BRTx	串口工作于 8 位数据, 可变波特率。
UART_9bit	串口工作于 9 位数据, 固定波特率。
UART_9bit_BRTx	串口工作于 9 位数据, 可变波特率。

UART\_BRT\_Use: 使用的波特率发生器:

UART_BRT_Use 取值	功能描述
BRT_Timer1	使用 Timer1 做波特率发生器, 仅仅用于 USART1。
BRT_Timer2	使用 Timer2 做波特率发生器。

UART\_BaudRate: 使用的波特率, 比如:

COMx\_InitStructure.UART\_BaudRate = 115200ul; //UL 表示是 unsigned long。

Morecommunicate: 多机通讯允许:

Morecommunicate 取值	功能描述
ENABLE	允许多机通讯。
DISABLE	禁止多机通讯 (默认)。

UART\_RxEnable: 接收允许:

UART_RxEnable 取值	功能描述
ENABLE	允许接收。
DISABLE	禁止接收。

BaudRateDouble: 波特率加倍 (仅仅用于 USART1):

BaudRateDouble 取值	功能描述
ENABLE	允许波特率加倍。
DISABLE	禁止波特率加倍。

UART\_Interrupt: 中断允许或禁止:

UART_Interrupt 取值	功能描述
ENABLE	允许中断。
DISABLE	禁止中断 (默认)。



UART\_Polity: 中断的优先级:

UART_Polity 取值	功能描述
PolityHigh	中断设置为高优先级。
PolityLow	中断设置为低优先级（默认）。

UART\_P\_SW: 切换 IO: 对于串口 1 的取值:

UART_P_SW 取值	功能描述
UART1_SW_P30_P31	把串口 1 切换到 P3.0、P3.1。
UART1_SW_P36_P37	把串口 1 切换到 P3.6、P3.7。
UART1_SW_P16_P17	把串口 1 切换到 P1.6、P1.7。(必须使用内部时钟)。

UART\_P\_SW: 切换 IO: 对于串口 2 的取值:

UART_P_SW 取值	功能描述
UART2_SW_P10_P11	把串口 2 切换到 P1.0、P1.1。
UART2_SW_P46_P47	把串口 2 切换到 P4.6、P4.7。

UART\_RXD\_TXD\_Short: 内部 TXD 与 RXD 同相缓冲输出做中继: 对于串口 1 的取值:

UART_RXD_TXD_Short 取值	功能描述
ENABLE	允许内部 TXD 与 RXD 同相缓冲输出做中继。
DISABLE	禁止内部 TXD 与 RXD 同相缓冲输出做中继。

## 串口 1 写缓冲函数: TX1\_write2buff

函数名	TX1_write2buff
函数原形	void TX1_write2buff(u8 dat)
所在文件	Usart.c
功能描述	写入串口 1 发送缓冲, 指针+1
输入参数 1	dat: 要发送的一字节数据。
返回	无。

**串口 2 写缓冲函数：TX2\_write2buff**

函数名	TX2_write2buff
函数原形	void TX2_write2buff(u8 dat)
所在文件	Usart.c
功能描述	写入串口 2 发送缓冲，指针+1
输入参数 1	dat: 要发送的一字节数据。
返回	无。

**串口 1 写数据块函数：PrintString1**

函数名	PrintString1
函数原形	void PrintString1(u8 *puts)
所在文件	Usart.c
功能描述	把一个字符串写入串口 1 发送缓冲，遇到 0 结束。
输入参数 1	puts: 要发送的字符串指针。
返回	无。

**串口 2 写数据块函数：PrintString2**

函数名	PrintString2
函数原形	void PrintString2(u8 *puts)
所在文件	Usart.c
功能描述	把一个字符串写入串口 2 发送缓冲，遇到 0 结束。
输入参数 1	puts: 要发送的字符串指针。
返回	无。

**模拟串口字节发送函数：TxSend**

函数名	TxSend
函数原形	void TxSend(u8 dat)
所在文件	Soft_uart.c
功能描述	模拟串口发送，可以定义任意一个 IO 做串口发送，固定为 9600,8,n,1，修改 config 里的时钟频率时，程序会自动适应这个频率，始终保持 9600 的波特率，一般用于测试用途。 当发送时，为了避免受中断的影响，会关掉总中断。
输入参数 1	dat: 要发送的字节。

返回	无。
----	----

## 模拟串口写数据块函数：PrintString

函数名	PrintString
函数原形	void PrintString(u8 *puts)
所在文件	Soft_uart.c
功能描述	模拟串口发送一个字符串，遇到 0 结束。用于测试用途。 调用了 TxSend 函数
输入参数 1	puts: 要发送的字符串指针。
返回	无。

## EEPROM 多字节读函数：EEPROM\_read\_n

函数名	EEPROM_read_n
函数原形	void EEPROM_read_n(u16 EE_address,u8 *DataAddress,u16 number)
所在文件	EEPROM.c
功能描述	从 EEPROM 读出多个字节。
输入参数 1	EE_address: 要读出的 EEPROM 的 16 位首地址。
输入参数 2	DataAddress: 读出数据存放的指针。
输入参数 3	Number: 要读出的字节数，取值 1~65535（根据实际情况确定最大值）。
返回	无。

## EEPROM 多字节写函数：EEPROM\_write\_n

函数名	EEPROM_write_n
函数原形	void EEPROM_write_n(u16 EE_address,u8 *DataAddress,u16 number)
所在文件	EEPROM.c
功能描述	把多个字节写入 EEPROM。
输入参数 1	EE_address: 要写入的 EEPROM 的 16 位首地址。
输入参数 2	DataAddress: 源数据存放的指针。
输入参数 3	Number: 要写入的字节数，取值 1~65535（根据实际情况确定最大值）。
返回	无。

**EEPROM 扇区擦除函数函数：EEPROM\_SectorErase**

函数名	EEPROM_SectorErase
函数原形	void EEPROM_SectorErase(u16 EE_address)
所在文件	EEPROM.c
功能描述	擦除 EEPROM 一个扇区。
输入参数 1	EE_address: 要擦除的扇区内的任意一个 16 位地址。
返回	无。

**PCA 初始化函数：PCA\_Init**

函数名	PCA_Init
函数原形	void PCA_Init(u8 PCA_id, PCA_InitTypeDef *PCAx)
所在文件	PCA.c
功能描述	初始化 PCA。
输入参数 1	PCA_id: 要初始化的 PCA 通道，取以下其一：PCA0, PCA1, PCA2, PCA_Counter。
输入参数 2	PCAx: 初始化参数的结构指针。详情看下面的描述。
返回	无。

PCA\_id: 选择要初始化的 PCA 通道：

PCA_id 取值	功能描述
PCA0	初始化 PCA 0 通道。
PCA1	初始化 PCA 1 通道。
PCA2	初始化 PCA 2 通道。
PCA_Counter	初始化 PCA 公用计数器，这个最好放在最后初始化。

PCA\_InitTypeDef 的定义见于文件“PCA.H”。

```
typedef struct
```

```
{
```

```
    u8  PCA_IoUse;
```

```

u8 PCA_Clock;
u8 PCA_Mode;
u8 PCA_PWM_Wide;
u8 PCA_Interrupt_Mode;
u8 PCA_Polity;
u16 PCA_Value;
} PCA_InitTypeDef;

```

PCA\_IoUse: 选择 PCA 使用的 IO: 初始化 PCA\_Counter 时的取值, 初始化 PCA0 ~ PCA2 时忽略:

PCA_IoUse 取值	功能描述
PCA_P12_P11_P10_P37	把 PCA 切换到 P1.2、P1.1、P1.0、P3.7。
PCA_P34_P35_P36_P37	把 PCA 切换到 P3.4、P3.5、P3.6、P3.7。
PCA_P24_P25_P26_P27	把 PCA 切换到 P2.4、P2.5、P2.6、P2.7。

PCA\_Clock: 选择 PCA 使用的时钟: 初始化 PCA\_Counter 时的取值, 初始化 PCA0 ~ PCA2 时忽略:

PCA_Clock 取值	功能描述
PCA_Clock_1T	PCA 使用系统 1T 做时钟。
PCA_Clock_2T	PCA 使用系统 2T 做时钟。
PCA_Clock_4T	PCA 使用系统 4T 做时钟。
PCA_Clock_6T	PCA 使用系统 6T 做时钟。
PCA_Clock_8T	PCA 使用系统 8T 做时钟。
PCA_Clock_12T	PCA 使用系统 12T 做时钟。
PCA_Clock_Timer0_OF	PCA 使用 Timer0 溢出率做时钟。Time0 要另外初始化, 速度快时不要开 Timer0 中断。
PCA_Clock_ECI	PCA 使用外部 ECI 引脚做时钟。

PCA\_Polity: 中断的优先级: 初始化 PCA\_Counter 时的取值, 初始化 PCA0 ~ PCA2 时忽略:

PCA_Polity 取值	功能描述
PolityHigh	PCA 中断设置为高优先级。
PolityLow	PCA 中断设置为低优先级 (默认)。

PCA\_Interrupt\_Mode: 中断允许或禁止: 初始化 PCA\_Counter 时的取值, 初始化 PCA0 ~ PCA2 时忽略:

PCA_Interrupt_Mode 取值	功能描述
ENABLE	允许公用 PCA 定时器中断。
DISABLE	禁止公用 PCA 定时器中断 (默认)。

PCA\_Interrupt\_Mode: 中断允许或禁止: 初始化 PCA0 ~ PCA2 时的取值, 初始化 PCA\_Counter 时忽略:

PCA_Interrupt_Mode 取值	功能描述
ENABLE	允许 PCA 通道中断。
DISABLE	禁止 PCA 通道中断 (默认)。
PCA_Rise_Active	PCA 通道上升沿中断。
PCA_Fall_Active	PCA 通道下降沿中断。

注意: 上面的参数可以做如下组合:

PCA\_InitStructure.PCA\_Interrupt\_Mode = PCA\_Fall\_Active | ENABLE; //下降沿中断, 允许中断。

PCA\_InitStructure.PCA\_Interrupt\_Mode = PCA\_Rise\_Active | ENABLE; //上升沿中断, 允许中断。

PCA\_InitStructure.PCA\_Interrupt\_Mode = PCA\_Rise\_Active | PCA\_Fall\_Active | ENABLE; //上升沿、下降沿中断, 允许中断。

如果后面使用了 | DISABLE, 则中断被禁止。

PCA\_Clock: 选择 PCA 使用的时钟: 初始化 PCA\_Counter 时的取值, 初始化 PCA0 ~ PCA2 时忽略:

PCA_Clock 取值	功能描述
PCA_Clock_1T	PCA 使用系统 1T 做时钟。
PCA_Clock_2T	PCA 使用系统 2T 做时钟。
PCA_Clock_4T	PCA 使用系统 4T 做时钟。
PCA_Clock_6T	PCA 使用系统 6T 做时钟。

PCA\_Mode: 设置 PCA 通道的工作模式: 初始化 PCA0 ~ PCA2 时的取值, 初始化 PCA\_Counter 时忽略:

PCA_Mode 取值	功能描述
PCA_Mode_PWM	PCA 通道工作于 PWM 输出模式。
PCA_Mode_Capture	PCA 通道工作于输入捕捉模式。
PCA_Mode_SoftTimer	PCA 通道工作于 16 位软件定时器模式。
PCA_Mode_HighPulseOutput	PCA 通道工作于 16 位软件定时器模式, 并且高速输出脉冲。

PCA\_PWM\_Wide: 设置 PCA 通道工作于 PWM 模式时的 PWM 宽度: 初始化 PCA0 ~ PCA2 工作于 PWM 模式时的取值, 初始化 PCA\_Counter 时或 PCA 通道其它模式时忽略:

PCA_PWM_Wide 取值	功能描述
PCA_PWM_8bit	PCA 通道工作于 PWM 输出模式, PWM 宽度为 8 位。
PCA_PWM_7bit	PCA 通道工作于 PWM 输出模式, PWM 宽度为 7 位。
PCA_PWM_6bit	PCA 通道工作于 PWM 输出模式, PWM 宽度为 6 位。

PCA\_Value: 设置 PCA 通道的初值, 初始化 PCA0 ~ PCA2 时的取值, 初始化 PCA\_Counter 时忽略。

## PWM 更新占空比函数: UpdatePwm

函数名	UpdatePwm
函数原形	void UpdatePwm(u8 PCA_id, u8 pwm_value)
所在文件	PCA.c
功能描述	更新 PWM 的占空比。
输入参数 1	PCA_id: 要更新的 PCA 通道, 取以下其一: PCA0, PCA1, PCA2。
输入参数 2	pwm_value: 新的 PWM 占空比值。这是 PWM 输出低电平的时间。
返回	无。

## 外中断初始化函数: Ext\_Inilize

函数名	Ext_Inilize
函数原形	u8 Ext_Inilize(u8 EXT, EXTI_InitTypeDef *INTx)
所在文件	Exti.c
功能描述	外中断初始化。
输入参数 1	EXT: 要初始化的外中断号, 取如下值之一: EXT_INT0, EXT_INT1, EXT_INT2, EXT_INT3, EXT_INT4。
输入参数 2	INTx: 初始化结构参数的指针。
返回	U8, 返回 0 表示配置成, 返回非 0 表示配置错误。

EXTI\_InitTypeDef 的定义见于文件 “Exti.H”。

typedef struct

```
{
    u8  EXTI_Mode;
    u8  EXTI_Polity;
    u8  EXTI_Interrupt;
} EXTI_InitTypeDef;
```

EXTI\_Mode: 设置外中断的工作模式: 初始化 INT0、INT1 时的取值, 初始化 INT2、INT3、INT4 时忽略, 固定为下降沿中断。

EXTI_Mode 取值	功能描述
EXT_MODE_RiseFall	外中断工作于上升、下降沿中断。

EXT_MODE_Fall	外中断工作于下降沿中断。
---------------	--------------

EXTI\_Polity: 中断的优先级: 初始化 INT0、INT1 时的取值, 初始化 INT2、INT3、INT4 时忽略, 固定为低优先级中断。

EXTI_Polity 取值	功能描述
PolityHigh	外中断设置为高优先级。
PolityLow	外中断设置为低优先级 (默认)。

EXTI\_Interrupt: 中断允许或禁止:

EXTI_Interrupt 取值	功能描述
ENABLE	允许外部中断。
DISABLE	禁止外部中断。