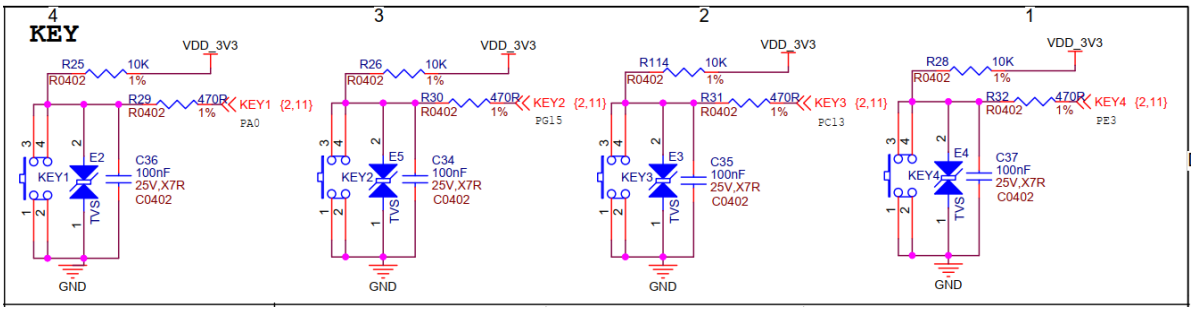
GPIO中断编程

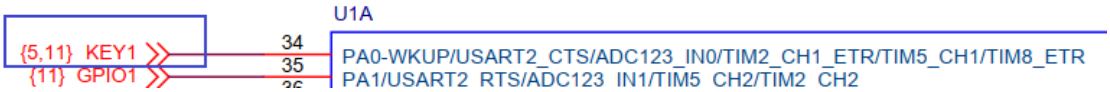
目的：实现KEY1中断，按下、松开按键，串口输出相应信息。

1.1 查看原理图确定引脚

- 100ASK STM32F103按键原理图



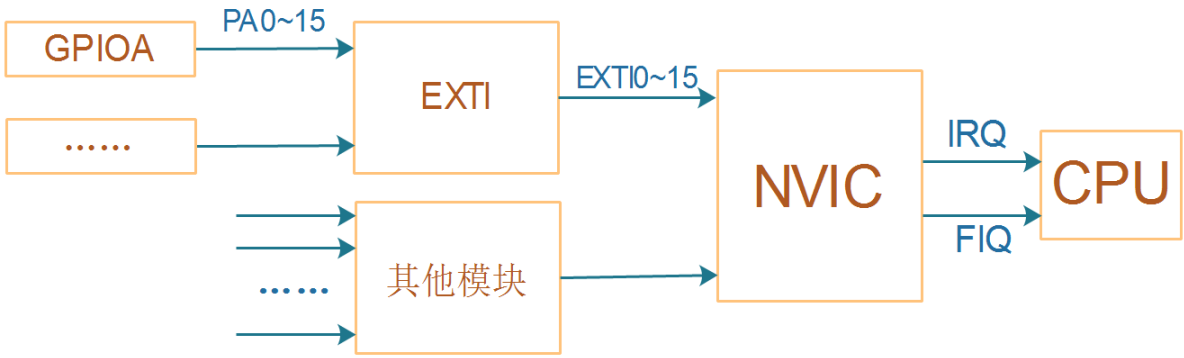
- KEY1用的是PA0引脚



1.2 STM32F103的GPIO中断

参考资料：`STM32F103数据手册.pdf`、`ARM Cortex-M3与Cortex-M4权威指南.pdf`、`PM0056.pdf`

对于GPIO中断，STM32F103又引入了`External interrupt/event controller (EXTI)`。
用来设置GPIO的中断类型，如下图：



EXTI可以给NVIC提供16个中断信号：EXTI0~EXTI15。
那么某个EXTIx，它来自哪些GPIO呢？这需要设置GPIO控制器。

1.2.1 GPIO控制器

STM32F103的GPIO控制器中有AFIO_EXTICR1~AFIO_EXTICR4一共4个寄存器
名为：External interrupt configuration register，外部中断配置寄存器。
用来选择某个外部中断EXTIx的中断源，示例如下：

9.4.3 External interrupt configuration register 1 (AFIO_EXTICR1)

Address offset: 0x08

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTI3[3:0]				EXTI2[3:0]				EXTI1[3:0]				EXTI0[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31:16 Reserved

Bits 15:0 **EXTIx[3:0]:** EXTI x configuration (x= 0 to 3)

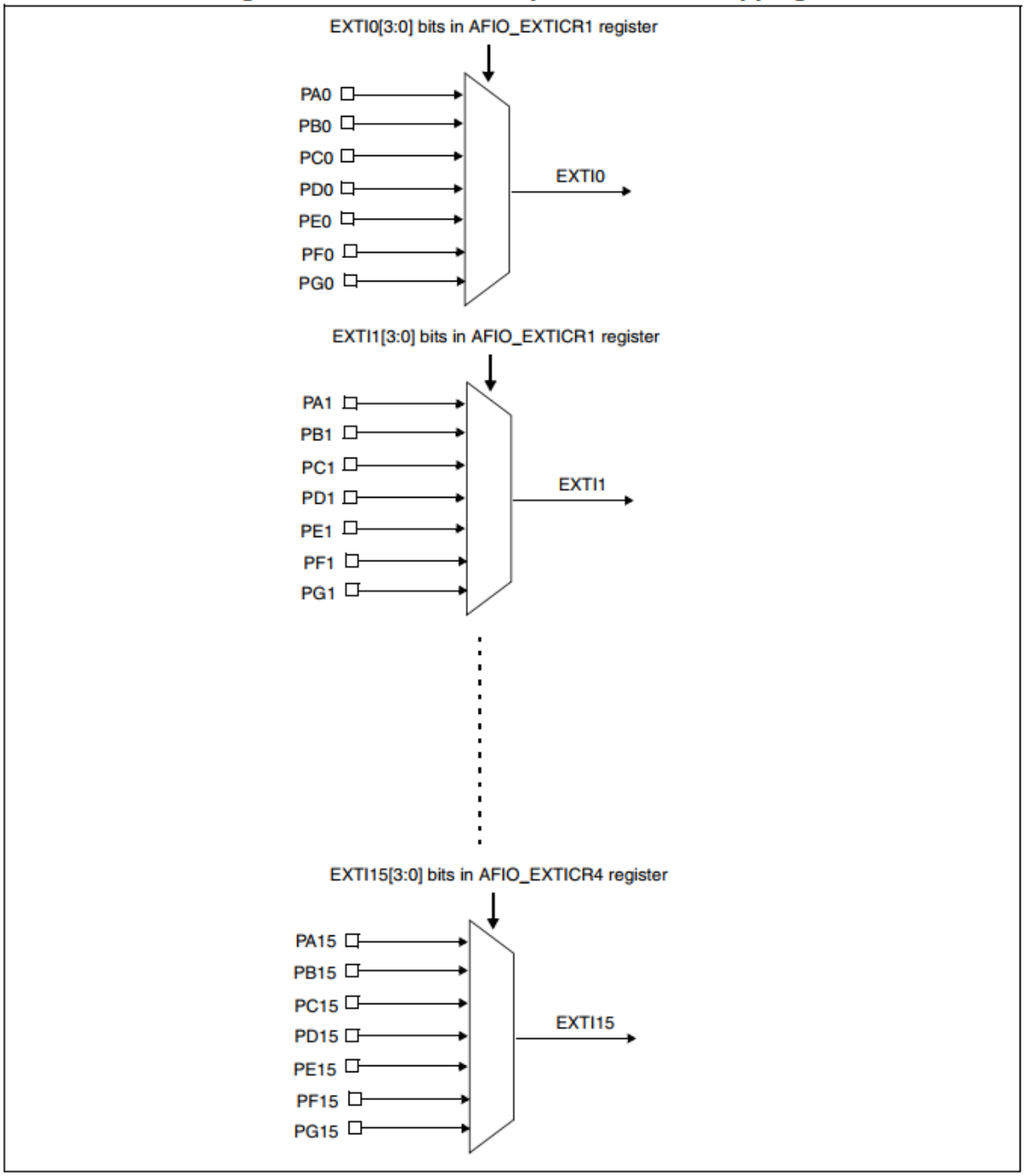
These bits are written by software to select the source input for EXTIx external interrupt.
Refer to *Section 10.2.5: External interrupt/event line mapping*

0000: PA[x] pin
0001: PB[x] pin
0010: PC[x] pin
0011: PD[x] pin
0100: PE[x] pin
0101: PF[x] pin
0110: PG[x] pin

对于EXTI0中断，它的来源可以选择下列引脚之一：
PA[0]、PB[0]、……、PG[0]

注意：从上图可知，EXTI0只能从PA0、……、PG0中选择一个，这也意味着PA0、……、PG0中只有一个引脚可以用于中断。这跟其他芯片不一样，很多芯片的任一GPIO引脚都可以同时用于中断。

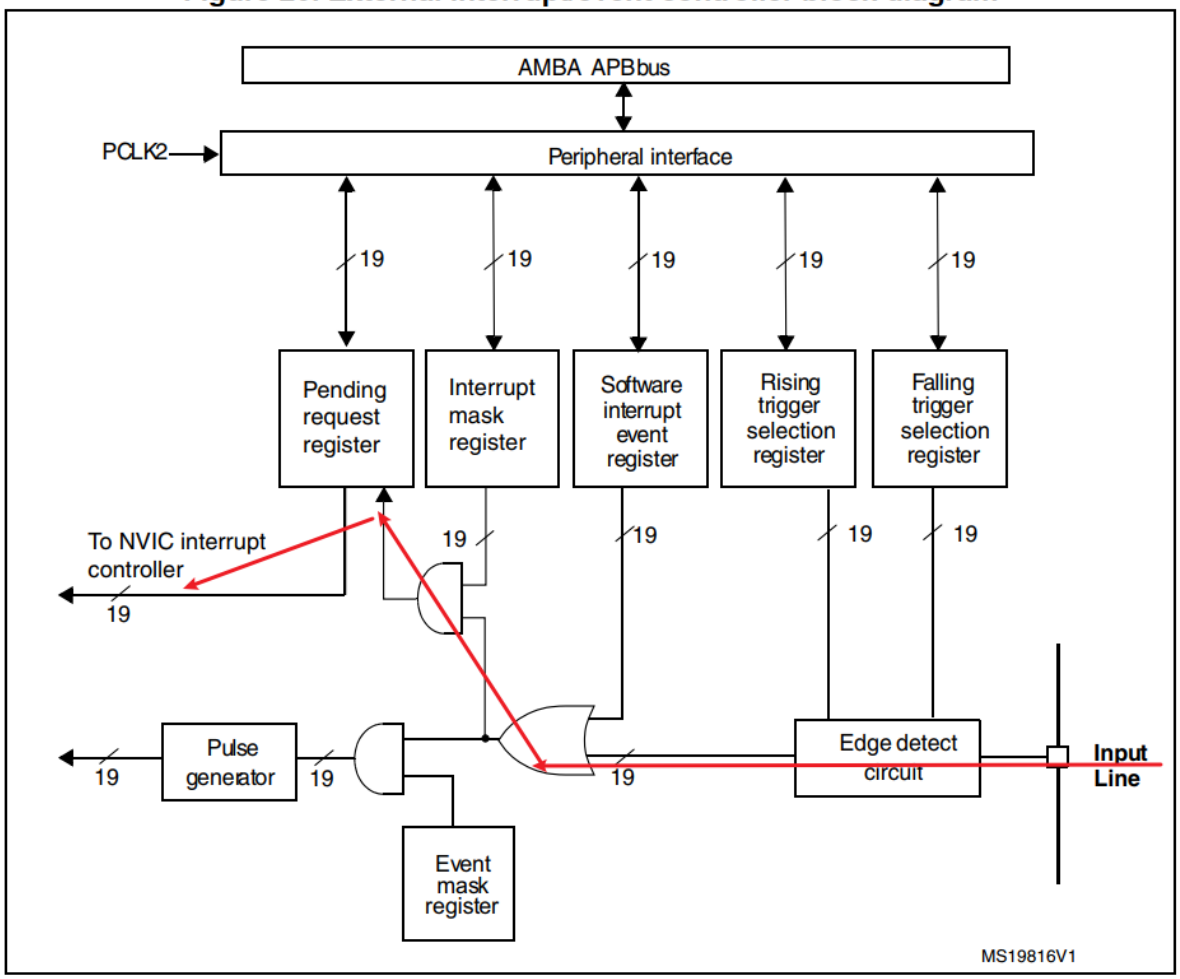
Figure 21. External interrupt/event GPIO mapping



1.2.2 EXTI

在GPIO控制器中，可以设置某个GPIO引脚作为中断源，给EXTI提供中断信号。
但是，这个中断的触发方式是怎么的？高电平触发、低电平触发、上升沿触发、下降沿触发？
这需要进一步设置。
EXTI框图如下：

Figure 20. External interrupt/event controller block diagram



沿着上面框图中的红线，我们要设置：

- Falling trigger selection register：是否选择下降沿触发
- Rising trigger selection register：是否选择上升沿触发
- Interrupt mask register：是否屏蔽中断

当发生中断时，可以读取下列寄存器判断是否发生了中断、发生了哪个中断：

- Pending reqeust register

要使用EXTI，流程如下：

Hardware interrupt selection

To configure the 20 lines as interrupt sources, use the following procedure:

- Configure the mask bits of the 20 Interrupt lines (EXTI_IMR)
- Configure the Trigger Selection bits of the Interrupt lines (EXTI_RTSR and EXTI_FTSR)
- Configure the enable and mask bits that control the NVIC IRQ channel mapped to the External Interrupt Controller (EXTI) so that an interrupt coming from one of the 20 lines can be correctly acknowledged.

翻译如下：

- 配置EXTI_IMR：允许EXTI发出中断
- 配置EXTI_RTSR、EXTI_FTSR，选择中断触发方式
- 配置NVIC中的寄存器，允许NVIC把中断发给CPU

1.2.3 NVIC

多个中断源汇聚到NVIC，NVIC的职责就是从多个中断源中取出优先级最高的中断，向CPU发出中断信号。
处理中断时，程序可以写NVIC的寄存器，清除中断。
涉及的寄存器：

表 7.9　用于中断控制的 NVIC 寄存器列表

地址	寄存器	CMSIS-Core 符号	功　能
0xE000E100～ 0xE000E11C	中断设置使能寄存器	NVIC->ISER [0]～ NVIC->ISER [7]	写 1 设置使能
0xE000E180～ 0xE000E19C	中断清除使能寄存器	NVIC->ICER [0]～ NVIC->ICER [7]	写 1 清除使能
0xE000E200～ 0xE000E21C	中断设置挂起寄存器	NVIC->ISPR [0]～ NVIC->ISPR [7]	写 1 设置挂起状态
0xE000E280～ 0xE000E29C	中断清除挂起寄存器	NVIC->ICPR [0]～ NVIC->ICPR [7]	写 1 清除挂起状态
0xE000E300～ 0xE000E31C	中断活跃位寄存器	NVIC->IABR [0]～ NVIC->IABR [7]	活跃状态位，只读
0xE000E400～ 0xE000E4EF	中断优先级寄存器	NVIC->IP [0]～ NVIC->IR [239]	每个中断的中断优先级(8 位宽)
0xE000EF00	软件触发中断寄存器	NVIC->STIR	写中断编号设置相应中断的挂起状态

我们暂时只需要关注：ISER(中断设置使能寄存器)、ICPR(中断清除挂起寄存器)。
要注意的是，这些寄存器有很多个，比如ISER0、ISER1等等。里面的每一位对应一个中断。
ISER0中的bit0对应异常向量表中的第16项(向量表从第0项开始)，如下图：

表 7.10　中断使能设置和清除寄存器

（0xE000E100～0xE000E11C，0xE000E180～0xE000E19C）

地址	名称	类型	复位值	描　　述
0xE000E100	NVIC->ISER[0]	R/W	0	设置中断 0～31 的使能 Bit[0]用于中断＃0(异常＃16) Bit[1]用于中断＃1(异常＃17) … Bit[31]用于中断＃31(异常＃47) 写 1 将位置 1，写 0 无作用 读出值表示当前使能状态
0xE000E104	NVIC->ISER[1]	R/W	0	设置中断 32～63 的使能 写 1 将位置 1，写 0 无作用 读出值表示当前使能状态
0xE000E108	NVIC->ISER[2]	R/W	0	设置中断 64～95 的使能 写 1 将位置 1，写 0 无作用 读出值表示当前使能状态
…	…	…	…	…
0xE000E180	NVIC->ICER[0]	R/W	0	清零中断 0～31 的使能 Bit[0]用于中断＃0(异常＃16) Bit[1]用于中断＃1(异常＃17) … Bit[31]用于中断＃31(异常＃47) 写 1 将位置 0，写 0 无作用 读出值表示当前使能状态
0xE000E184	NVIC->ICER[1]	R/W	0	清零中断 32～63 的使能 写 1 将位置 0，写 0 无作用 读出值表示当前使能状态
0xE000E188	NVIC->ICER[2]	R/W	0	清零中断 64～95 的使能 写 1 将位置 0，写 0 无作用 读出值表示当前使能状态
…	…	…	…	…

1.2.4 CPU

cortex M3/M4处理器内部有这几个寄存器：

1. PRIMASK

Priority mask register

The PRIMASK register prevents activation of all exceptions with configurable priority. See the register summary in *Table 2 on page 15* for its attributes. *Figure 5* shows the bit assignments.

Figure 5. PRIMASK bit assignments

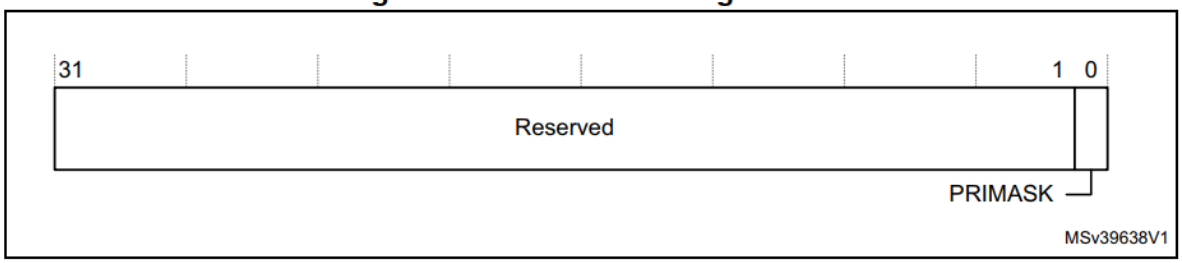


Table 7. PRIMASK register bit definitions

Bits	Description
Bits 31:1	Reserved
Bit 0	**PRIMASK**: 0: No effect 1: Prevents the activation of all exceptions with configurable priority.

把PRIMASK的bit0设置为1，就可以屏蔽所有**优先级可配置**的中断。
可以使用这些指令来设置它：

```
CPSIE I   ; 清除PRIMASK，使能中断
CPSID I   ; 设置PRIMASK，禁止中断

或者：
MOV R0, #1
MSR  PRIMASK R0   ; 将1写入PRIMASK禁止所有中断

MOV R0, #0
MSR PRIMASK, R0   ; 将0写入PRIMASK使能所有中断
```

2. FAULTMASK

Fault mask register

The FAULTMASK register prevents activation of all exceptions except for *Non-Maskable Interrupt* (NMI). See the register summary in *Table 2 on page 15* for its attributes. *Figure 6* shows the bit assignments.

Figure 6. FAULTMASK bit assignments

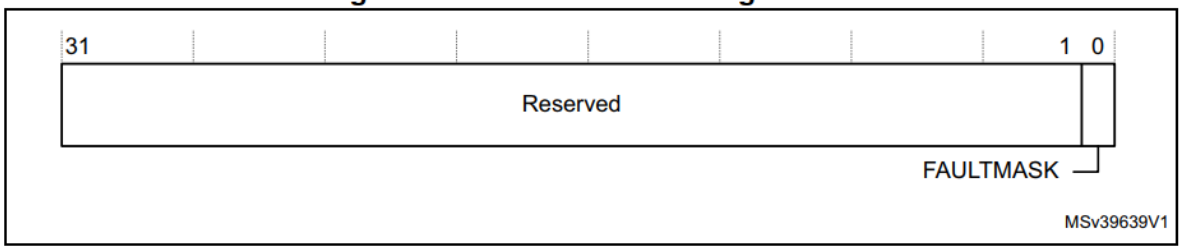


Table 8. FAULTMASK register bit definitions

Bits	Function
Bits 31:1	Reserved
Bit 0	**FAULTMASK**: 0: No effect 1: Prevents the activation of all exceptions except for NMI.

FAULTMASK和PRIMASK很像，它更进一步，出来一般的中断外，把HardFault都禁止了。
只有NMI可以发生。
可以使用这些指令来设置它：

```
CPSIE F  ; 清除FAULTMASK
CPSID F  ; 设置FAULTMASK

或者：
MOV R0, #1
MSR  FAULTMASK R0  ; 将1写入FAULTMASK禁止中断

MOV R0, #0
MSR FAULTMASK, R0  ; 将0写入FAULTMASK使能中断
```

3. BASEPRI

Base priority mask register

The BASEPRI register defines the minimum priority for exception processing. When BASEPRI is set to a nonzero value, it prevents the activation of all exceptions with same or lower priority level as the BASEPRI value. See the register summary in *Table 2 on page 15* for its attributes. *Figure 7* shows the bit assignments.

Figure 7. BASEPRI bit assignments

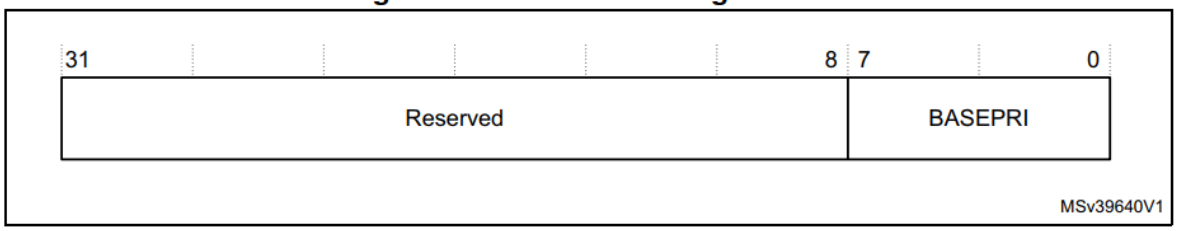


Table 9. BASEPRI register bit assignments

Bits	Function
Bits 31:8	Reserved
Bits 7:4	**BASEPRI[7:4]** Priority mask bits(1) 0x00: no effect Nonzero: defines the base priority for exception processing. The processor does not process any exception with a priority value greater than or equal to BASEPRI.
Bits 3:0	Reserved

1. This field is similar to the priority fields in the interrupt priority registers. See *Interrupt priority registers (NVIC_IPRx) on page 125* for more information. Remember that higher priority field values correspond to lower exception priorities.

BASEPRI用来屏蔽这些中断：它们的优先级，其值大于或等于BASEPRI。
可以使用这些指令来设置它：

```
MOVS R0, #0x60
MSR BASEPRI, R0   ; 禁止优先级在0x60~0xFF间的中断

MRS R0, BASEPRI   ; 读取BASEPRI

MOVS R0, #0
MSR BASEPRI, R0    ; 取消BASEPRI屏蔽
```