第4讲

中断

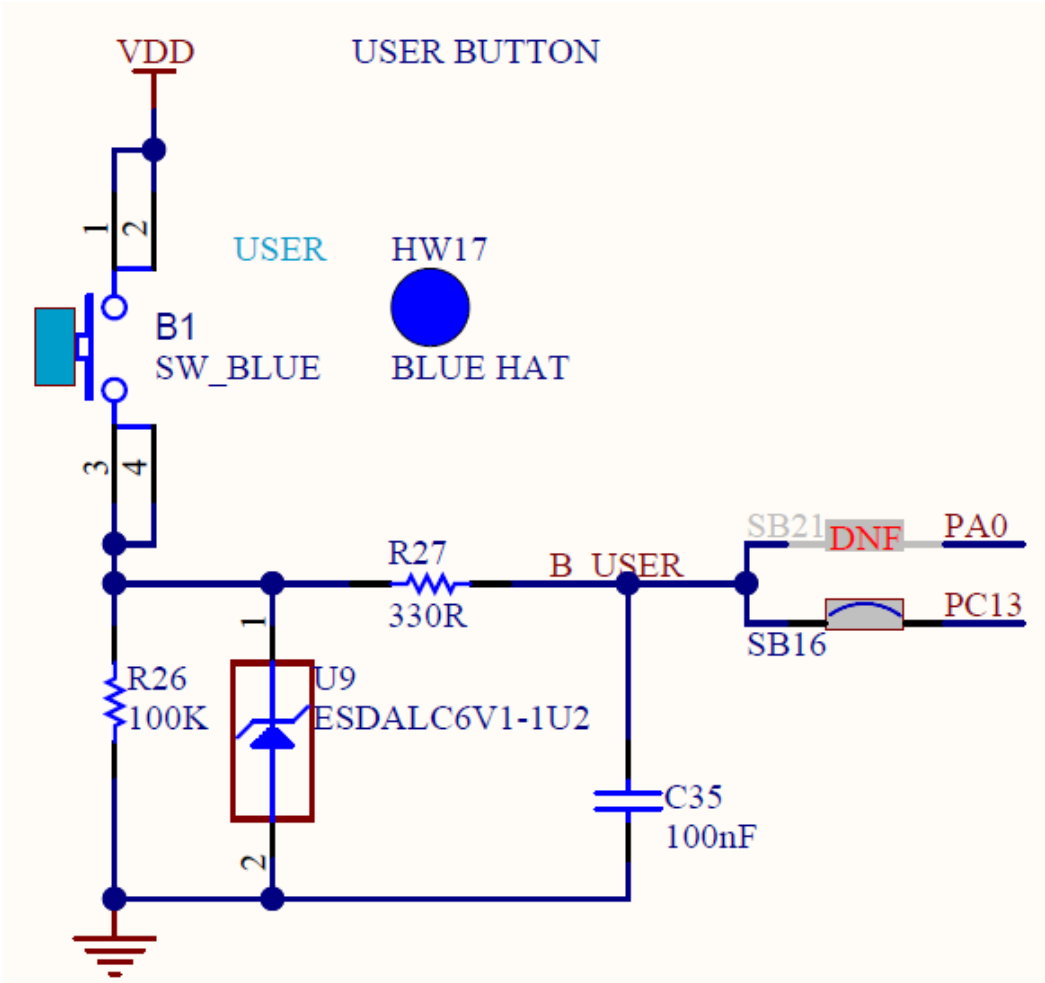
主要内容

- STM32的中断管理
- STM32的外部中断
- 动手练习4：用中断方式实现

  按键控制LED和蜂鸣器

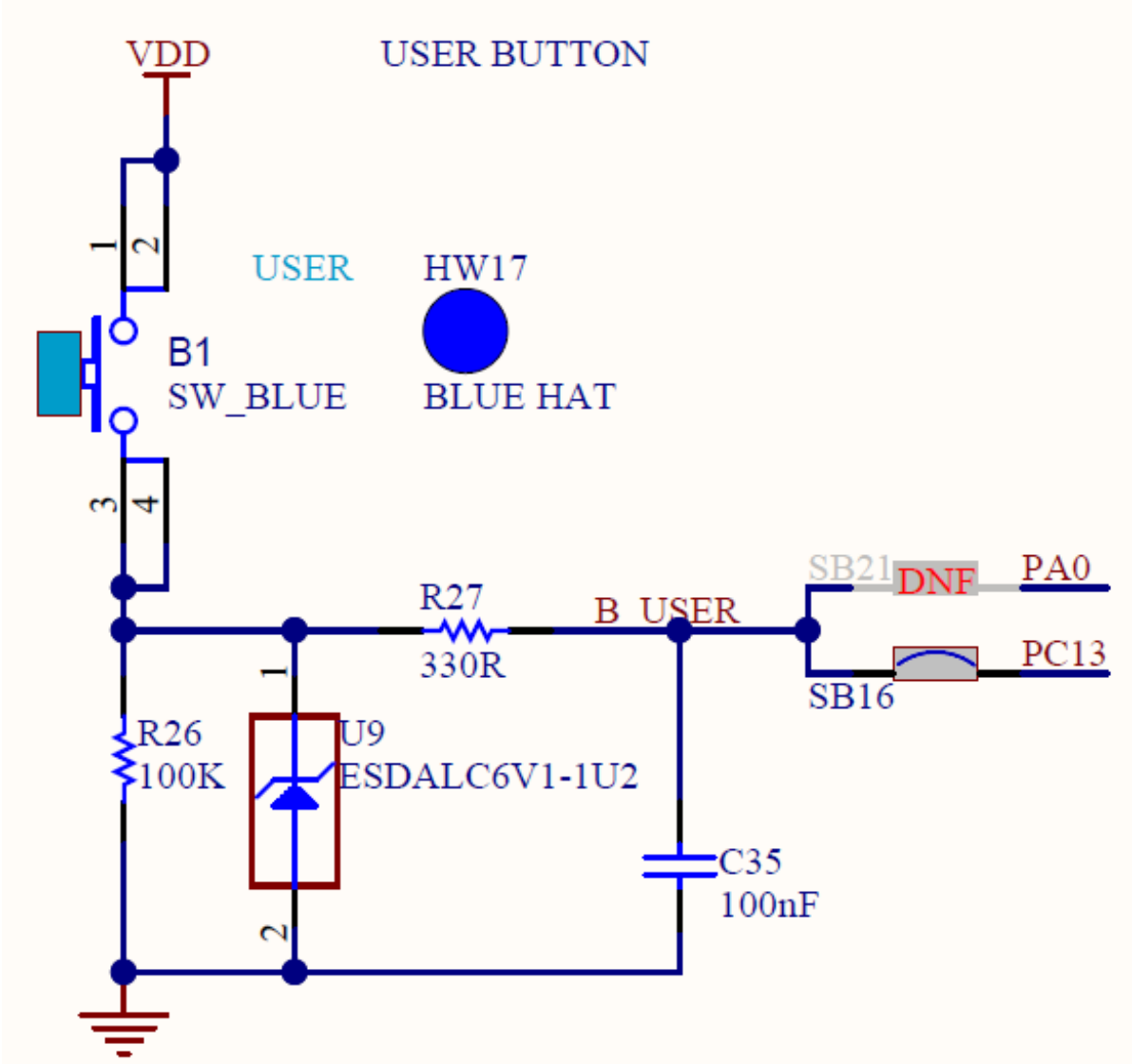
按键识别：查询方式

- 按键何时被按下是一个不确定事件
- CPU通过不停地查询，来判断按键的实时状态
- CPU很多时间做的工作，都是在判断某一寄存器是否为某个数值
- 效率低



按键识别：中断方式

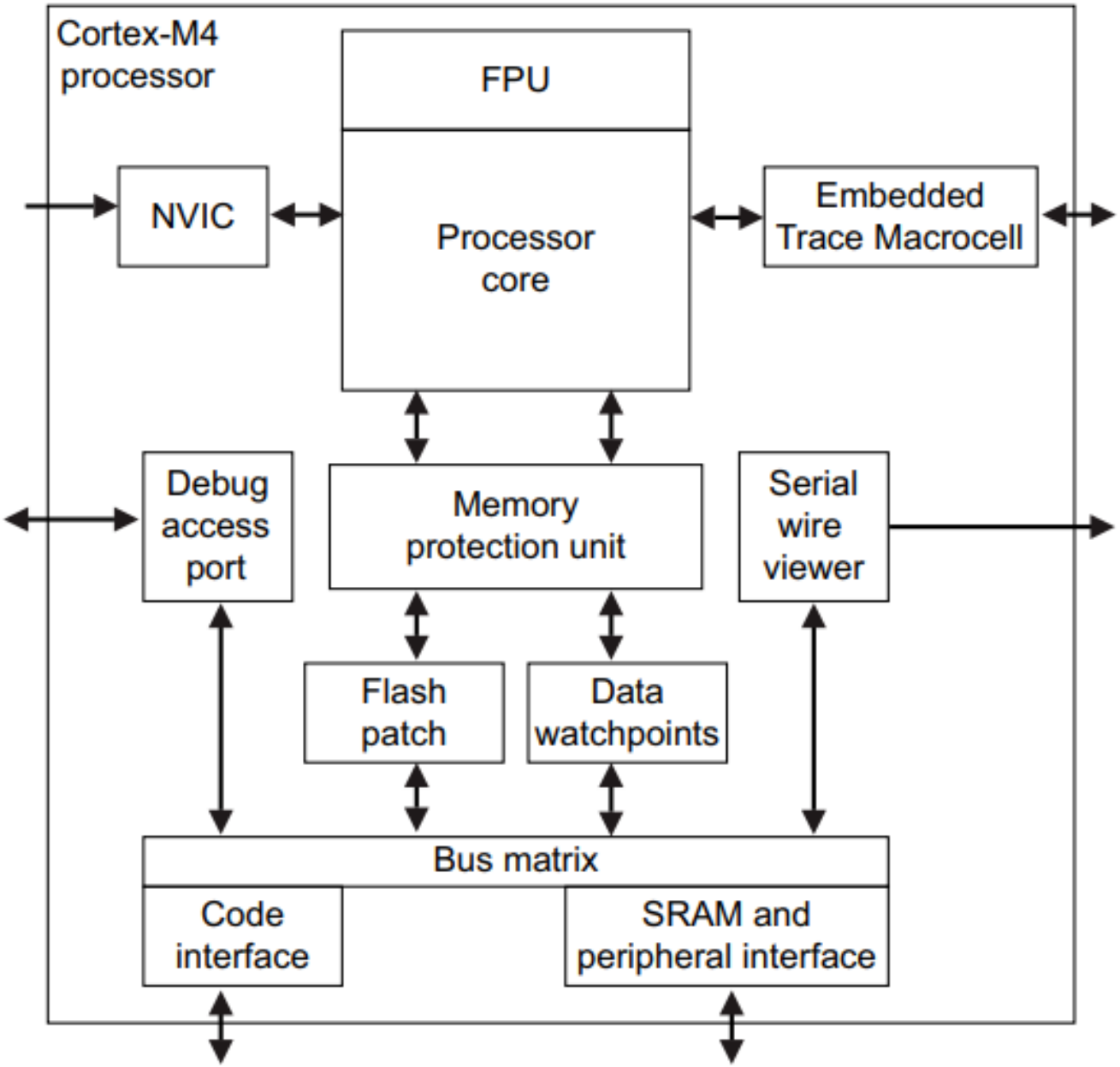
- 按键何时被按下是一个不确定事件
- CPU可正常运行，直到有中断请求发生
- 中断发生后，CPU才转去执行中断事件
- 效率高



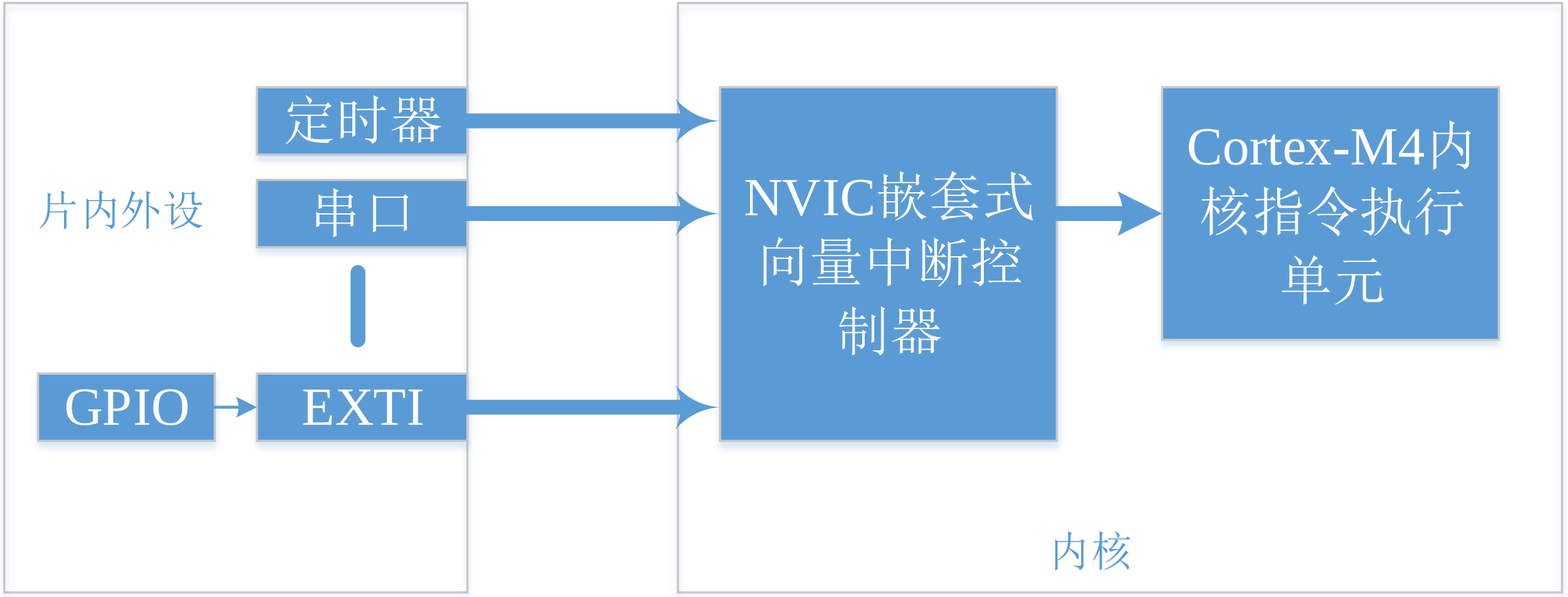
STM32 NVIC中断管理

NVIC: Nested Vectored Interrupt Controller

STM32 Cortex-M4 implementation



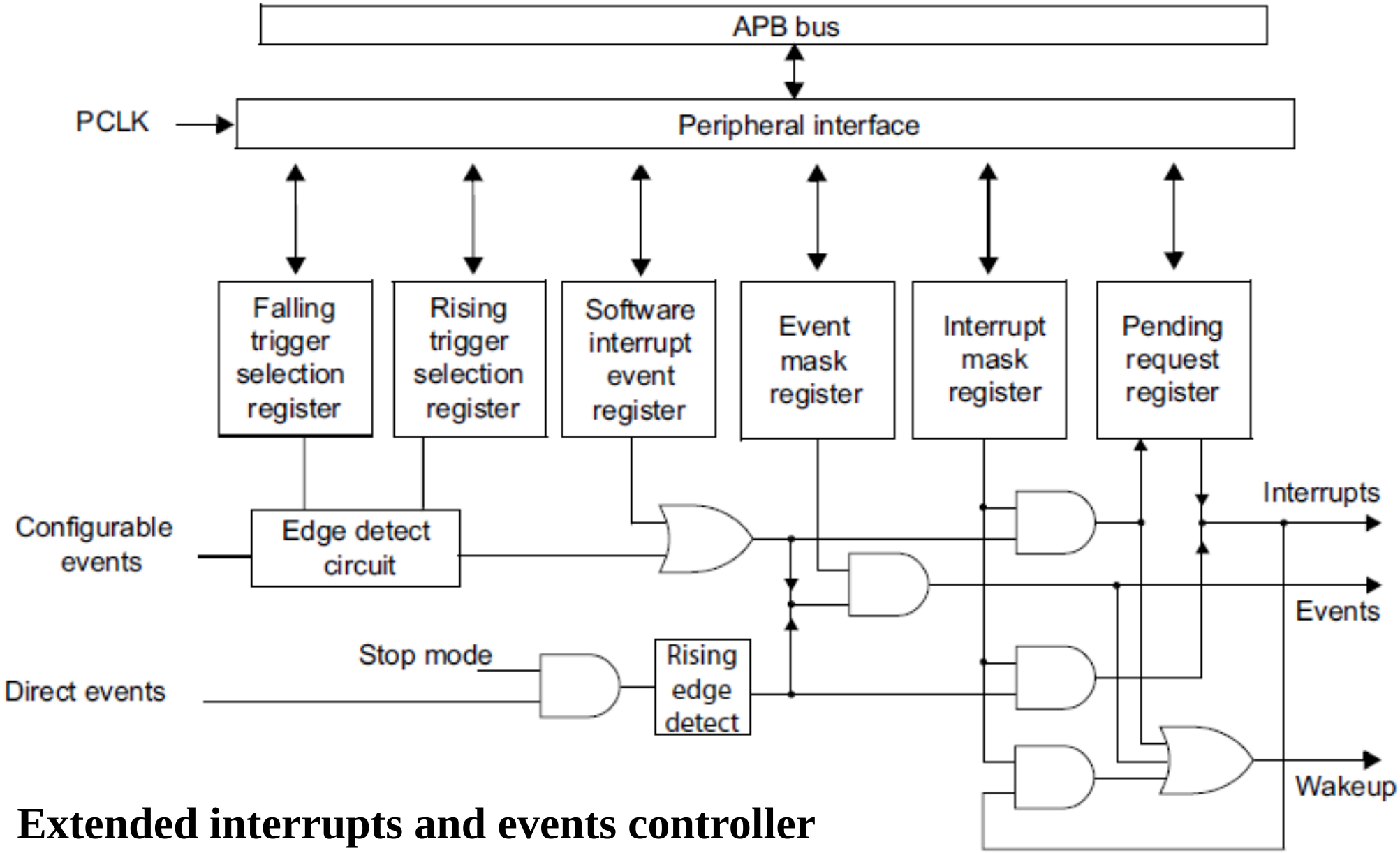
中断结构



EXTI: Extended interrupts and events controller

EXTI 控制器

EXTI block diagram



上升沿触发选择

下降沿触发选择

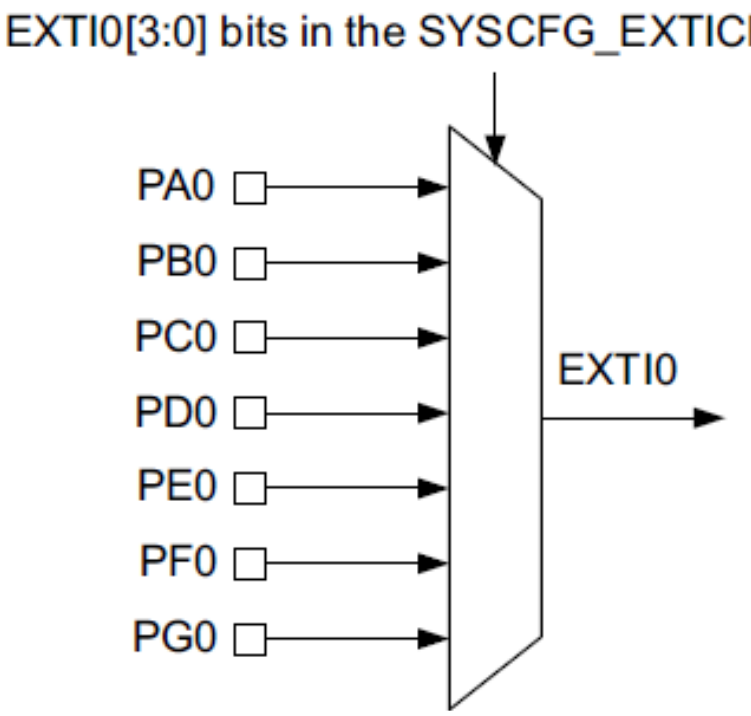
软件中断事件

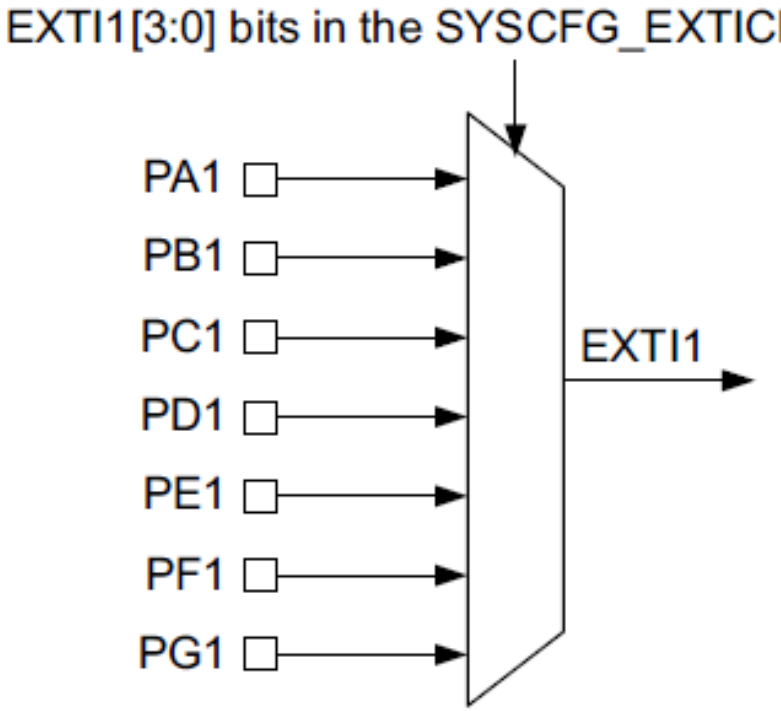
中断屏蔽

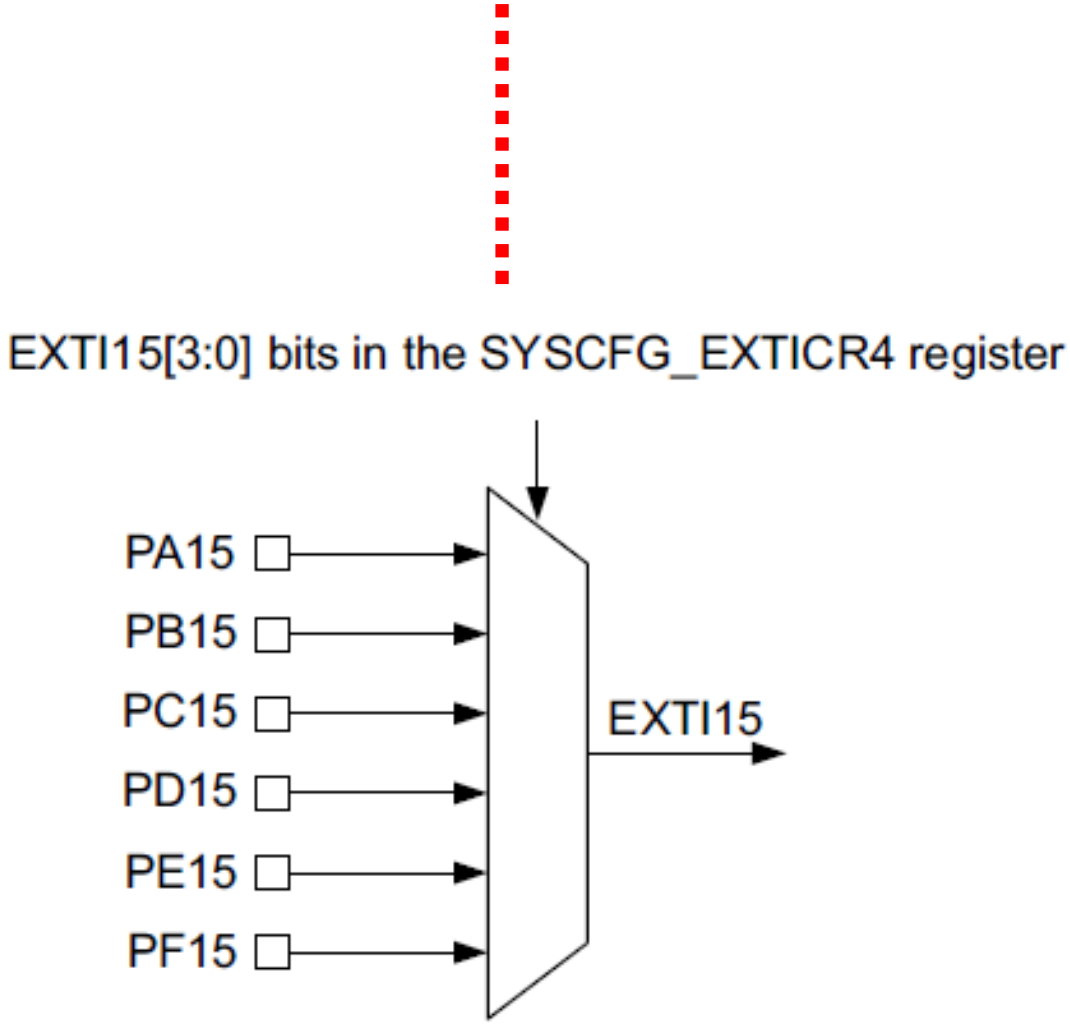
挂起请求

事件屏蔽

Extended interrupts and events controller







- STM32G4xx的EXTI有44个中断线（line），其中EXTI line 0~15分配给了GPIO。

SYSCFG external interrupt configuration register 1: SYSCFG_EXTICR1

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res

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 3:0 EXTI0[3:0]: EXTI 0 configuration bits
These bits are written by software to select the source input for the EXTI0 external interrupt.
0000: PA[0] pin
0001: PB[0] pin
0010: PC[0] pin
0011: PD[0] pin
0100: PE[0] pin
0101: PF[0] pin
0110: PG[0] pin

练习4：外部中断

练习4：外部中断

- 完成功能: 中断方式实现
  - ✓ 用B1控制LD2的亮灭
  - ✓ NUCLEO-G431RB板:

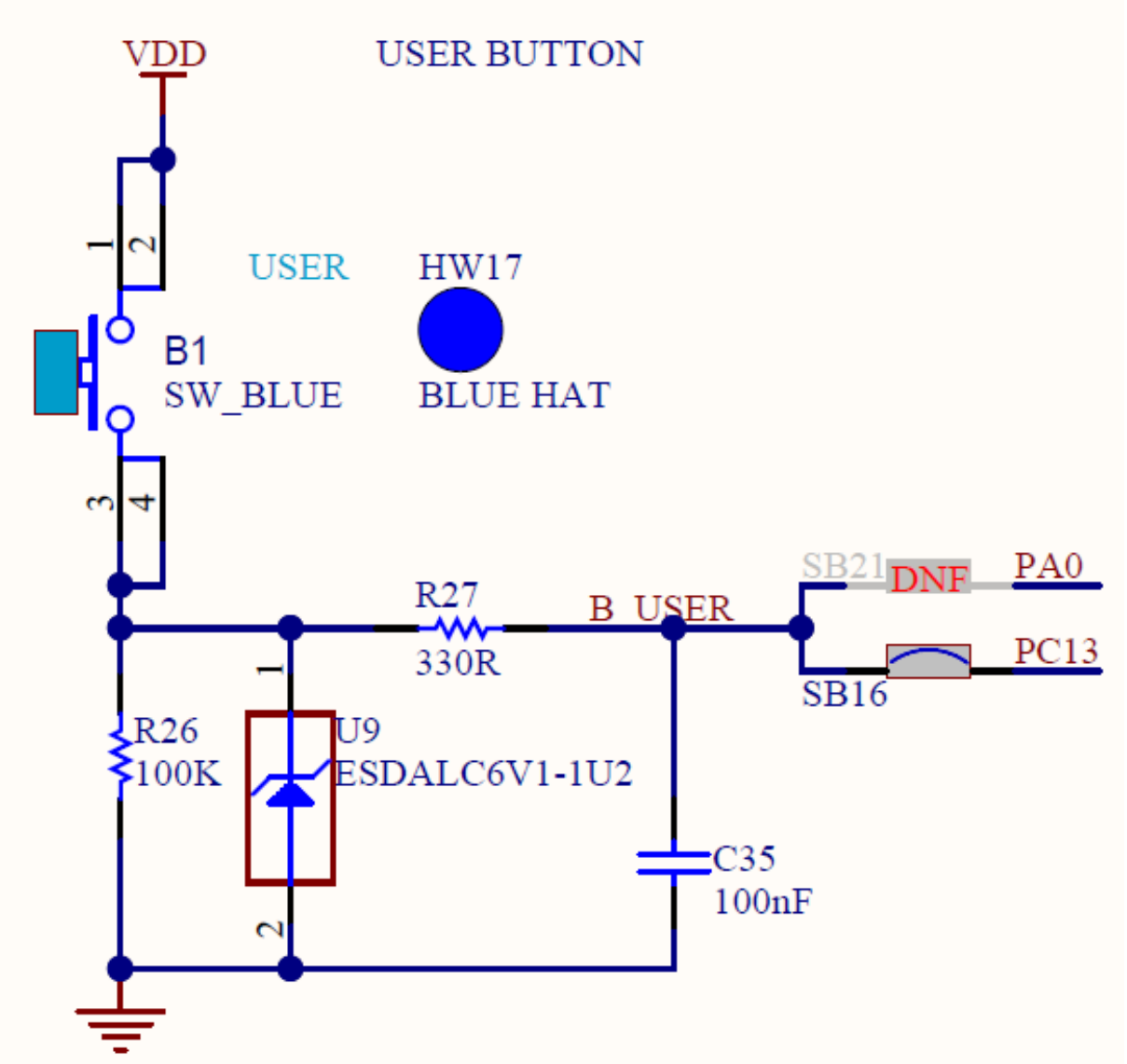
    PA5 -> LD2

    PC13 <- B1

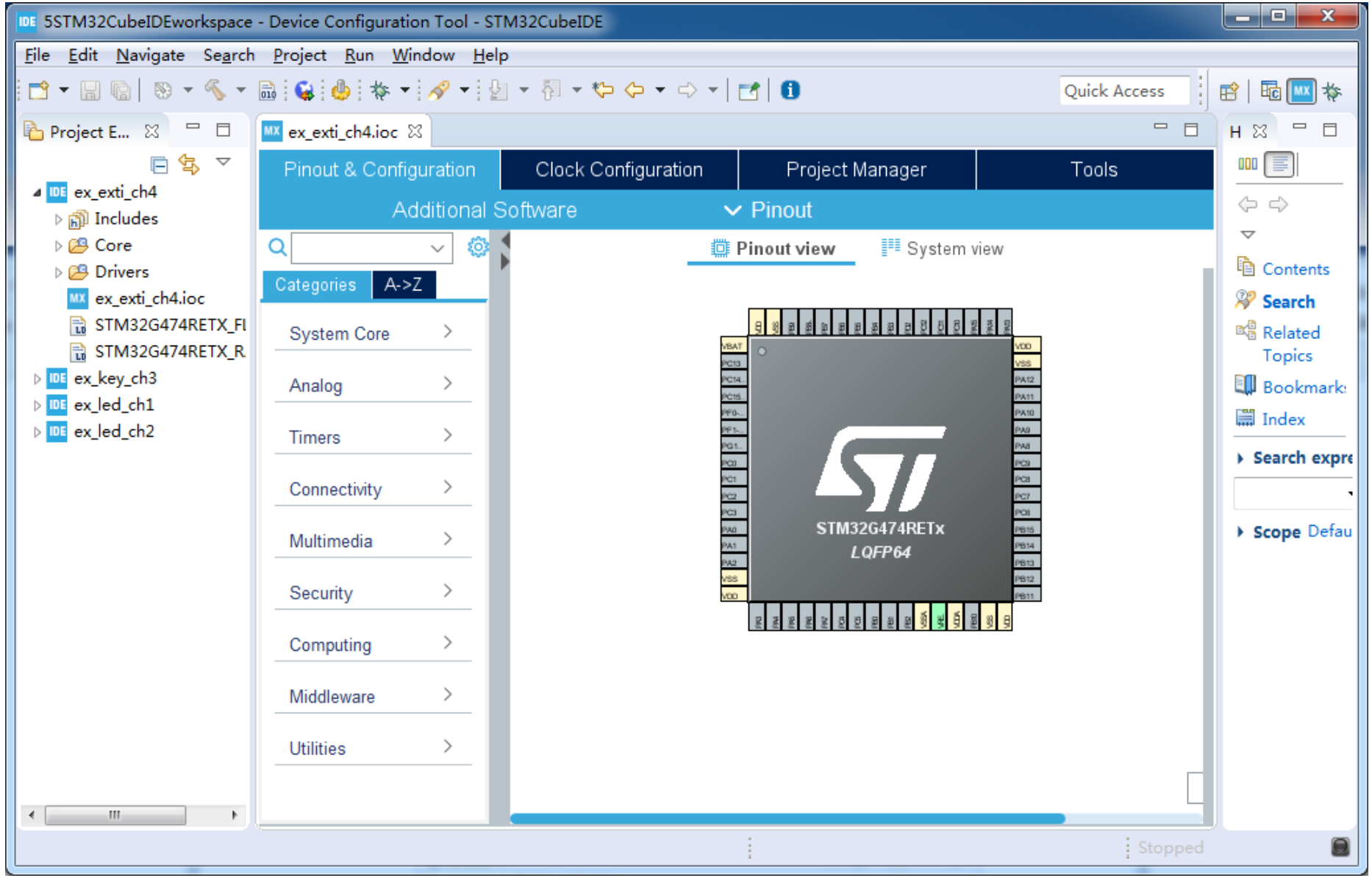
    PA4 ->蜂鸣器

NUCLEO64 STM32G4

en.mb1367-g431rb-c04_schematic.pdf

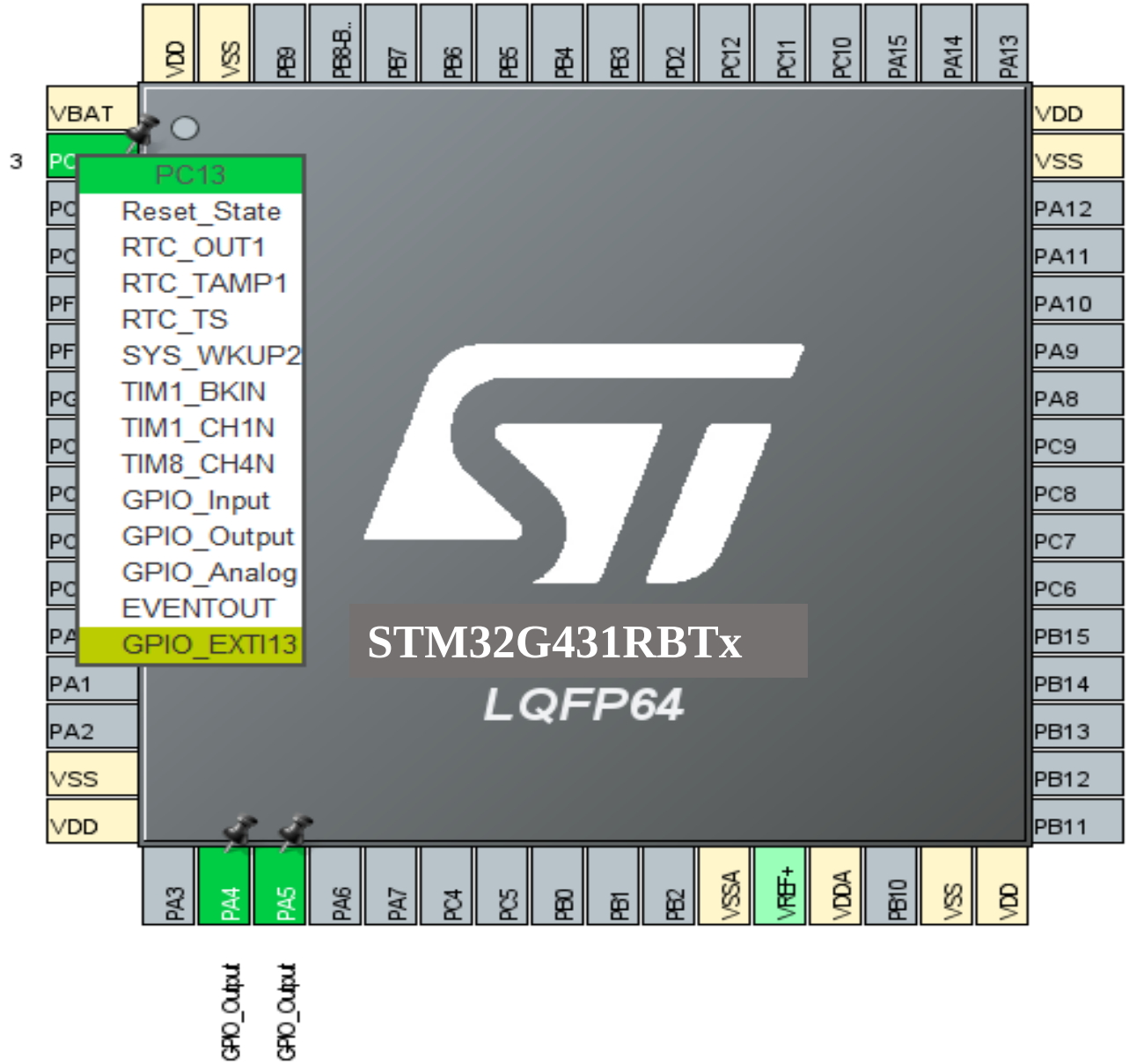


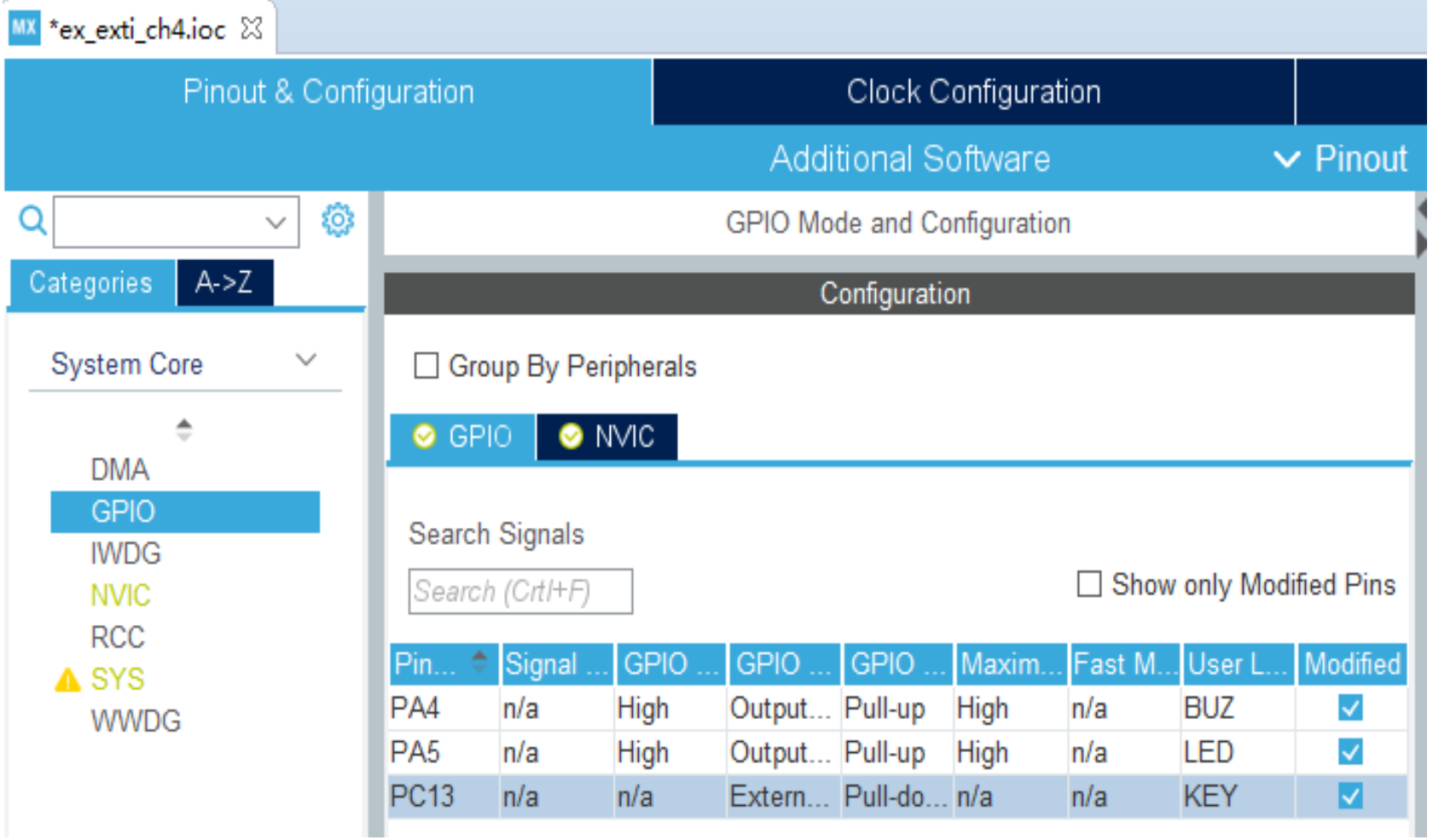
建立新工程

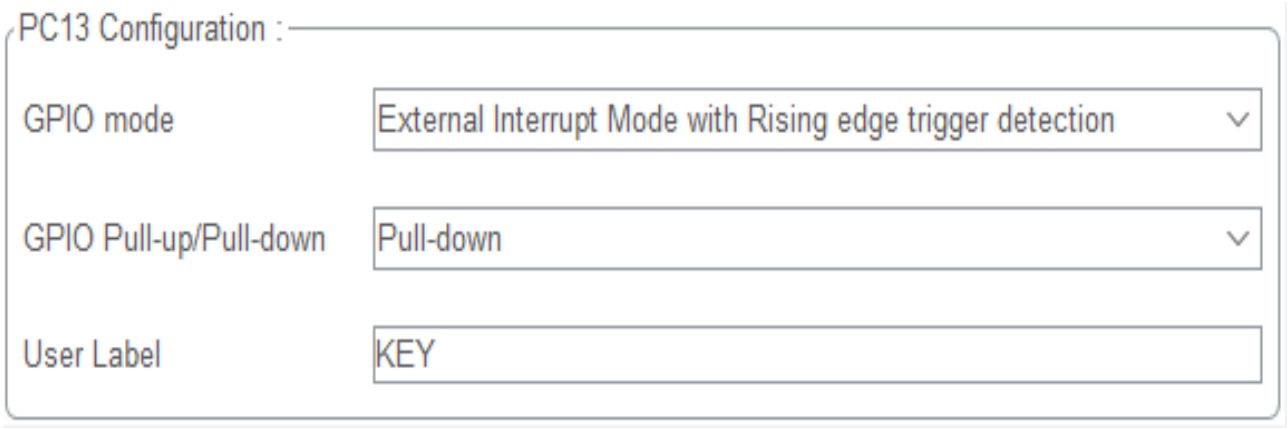

5STM32CubeIDEworkspace - Device Configuration Tool - STM32CubeIDE
File Edit Navigate Search Project Run Window Help
Quick Access
Project E...
ex_exti_ch4
Includes
Core
Drivers
ex_exti_ch4.ioc
STM32G474RETX_FI
STM32G474RETX_R
ex_key_ch3
ex_led_ch1
ex_led_ch2
ex_exti_ch4.ioc
Pinout & Configuration
Clock Configuration
Project Manager
Tools
Additional Software
Pinout
Categories
A->Z
System Core
Analog
Timers
Connectivity
Multimedia
Security
Computing
Middleware
Utilities
Pinout view
System view
STM32G474RETx
LQFP64
Contents
Search
Related Topics
Bookmarks
Index
Search expre
Scope Defau
Stopped

配置GPIO

PA4
PA5
PC13





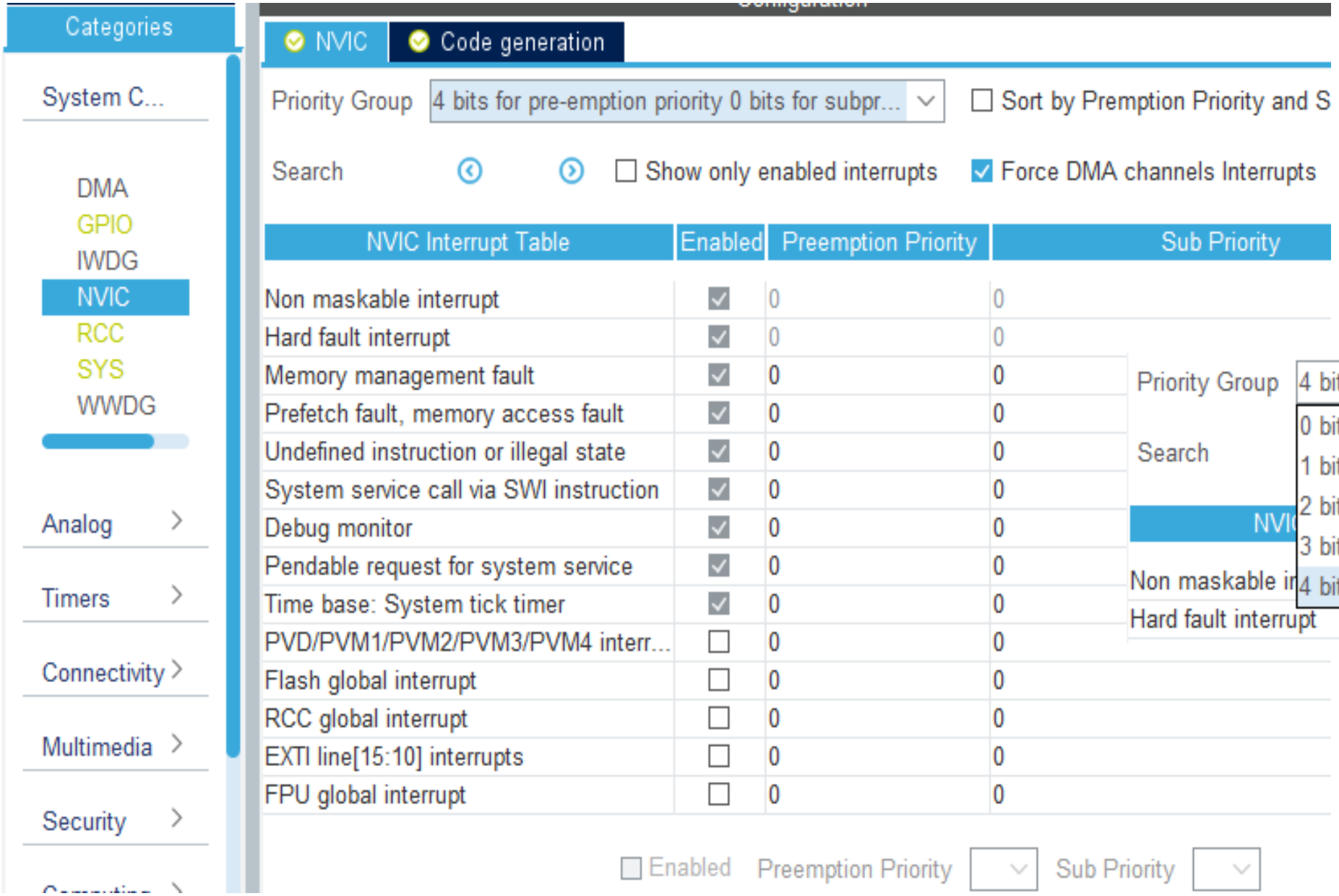


上升沿触发（Rising edge trigger detection）

下降沿触发（Falling edge trigger detection）

上升/下降沿触发（Rising/Falling edge trigger detection）

配置中断


Categories
System C...
DMA
GPIO
IWDG
NVIC
RCC
SYS
WWDG
Analog
Timers
Connectivity
Multimedia
Security
NVIC
Code generation
Priority Group
4 bits for pre-emption priority 0 bits for subpr...
Sort by Premption Priority and S
Search
Show only enabled interrupts
Force DMA channels Interrupts
NVIC Interrupt Table
Enabled
Preemption Priority
Sub Priority
Non maskable interrupt
Hard fault interrupt
Memory management fault
Prefetch fault, memory access fault
Undefined instruction or illegal state
System service call via SWI instruction
Debug monitor
Pendable request for system service
Time base: System tick timer
PVD/PVM1/PVM2/PVM3/PVM4 interr...
Flash global interrupt
RCC global interrupt
EXTI line[15:10] interrupts
FPU global interrupt
Enabled
Preemption Priority
Sub Priority
Priority Group
4 bits for pre-emption priority 0 bits for subprio...
0 bits for pre-emption priority 4 bits for subpriority
1 bits for pre-emption priority 3 bits for subpriority
2 bits for pre-emption priority 2 bits for subpriority
3 bits for pre-emption priority 1 bits for subpriority
4 bits for pre-emption priority 0 bits for subpriority

配置中断

- 通过NVIC，进行中断分组，分配抢占式优先级（Preemption Priority）和响应优先级（Sub Priority）
- 两种优先级组合起来，决定多个中断的执行次序。抢占式优先级的级别要高于响应优先级。优先级用数字表示，数字值越小优先级越高。
- 高抢占式优先级的中断，能打断低抢占式优先级的中断。
- 如果两个中断的抢占式优先级相同，响应优先级不同，它们同时发生时，则响应优先级高的中断先执行；不过，如果不是同时发生，它们是不能相互打断的

配置中断

- GPIO 有16条中断线，但在固件库中，前5条中断线（line0~line4）都有独立的中断服务函数，而中断线5~9，10~15都是分别共用一个函数的。具体有哪些中断服务函数，可以在固件库的startup_stm32g4xx.s文件中查到。这里给出GPIO相关的EXTI中断服务函数，总共7个：

EXTI0_IRQHandler
EXTI1_IRQHandler
EXTI2_IRQHandler
EXTI3_IRQHandler
EXTI4_IRQHandler
EXTI9_5_IRQHandler
EXTI15_10_IRQHandler

配置中断

NVIC Interrupt Table	Enabled	Preemption Priority	Sub Priority
Non maskable interrupt	☑	0	0
Hard fault interrupt	☑	0	0
Memory management fault	☑	0	0
Prefetch fault, memory access fault	☑	0	0
Undefined instruction or illegal state	☑	0	0
System service call via SWI instruction	☑	0	0
Debug monitor	☑	0	0
Pendable request for system service	☑	0	0
Time base: System tick timer	☑	0	0
PVD/PVM1/PVM2/PVM3/PVM4 interrup...	☐	0	0
Flash global interrupt	☐	0	0
RCC global interrupt	☐	0	0
EXTI line[15:10] interrupts	☑	1	0
FPU global interrupt	☐	0	0

☑ Enabled Preemption Priority 1 ⌄ Sub Priority 0 ⌄

其他硬件参数配置

■选择时钟源和Debug模式

✓ System Core->RC->将高速时钟（HSE）选择为Crystal/Ceramic Resonator

✓ SYS->Debug选择为Serial Wire

■配置系统时钟

✓ 在“Clock Configuration”中，将系统时钟（SYSCLK）配置为170Mhz

■ 保存硬件配置界面（*.ioc），启动代码生成

代码分析

- Core->Src，打开main.c
- MX_GPIO_Init()

```
/* EXTI interrupt init*/
HAL_NVIC_SetPriority(EXTI15_10_IRQn, 1, 0);
HAL_NVIC_EnableIRQ(EXTI15_10_IRQn);
```

在stm32g431xx.h中定义

- 中断线EXTI15_10的中断服务函数为：

```
void EXTI15_10_IRQHandler(void)
{
  /* USER CODE BEGIN EXTI15_10_IRQn 0 */
  /* USER CODE END EXTI15_10_IRQn 0 */
  HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_13);
  /* USER CODE BEGIN EXTI15_10_IRQn 1 */
  /* USER CODE END EXTI15_10_IRQn 1 */
}
```

stm32g4xx_it.c

代码分析

HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_13);

```
void HAL_GPIO_EXTI_IRQHandler(uint16_t GPIO_Pin)
{
  /* EXTI line interrupt detected */
  if (__HAL_GPIO_EXTI_GET_IT(GPIO_Pin) != 0x00u)
  {
    __HAL_GPIO_EXTI_CLEAR_IT(GPIO_Pin);
    HAL_GPIO_EXTI_Callback(GPIO_Pin);
  }
}
```

stm32g4xx_hal_gpio.c

代码分析

HAL_GPIO_EXTI_Callback(GPIO_Pin)

```
__weak void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
  /* Prevent unused argument(s) compilation warning */
  UNUSED(GPIO_Pin);
  /* NOTE: This function should not be modified, when the callback is
needed, the HAL_GPIO_EXTI_Callback could be implemented in the
user file
   */
}
```

stm32g4xx_hal_gpio.c

代码修改

- 可以将HAL_GPIO_EXTI_Callback()重定义在main.c中，stm32g4xx_hal_gpio.c中的这个弱函数不用删除。

```
/* USER CODE BEGIN 4 */
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
    HAL_Delay(10);
    if (GPIO_Pin == KEY_Pin)
    {
            HAL_GPIO_TogglePin(LED_GPIO_Port, LED_Pin);
            HAL_GPIO_TogglePin(BUZ_GPIO_Port, BUZ_Pin);
    }
}
/* USER CODE END 4 */
```

编译、下载与运行

■ Build Project

- 在**下载**之前，先进入主菜单Run->Debug Configurations，在弹出的“创建、管理和运行配置”（Create, manage, and configurations）界面中，用鼠标右键点击左侧栏目中的“STM32 Cortex-M C/C++ Application”，新建一个新的配置（New Configuration），.....
- 配置完毕后，点击配置界面右下角的“Debug”，即可以自动完成下载
- 下载完成后，点击主菜单上的**运行**（Resume）按钮，就可以运行程序
- 每按一次B1键，LD2和蜂鸣器的状态就会变化一次

练习4：按键输入

任务4.1、用中断方式方式实现：

按下B1后，蜂鸣器以1Hz的频率发出响声；松开B1后，蜂鸣器不响。

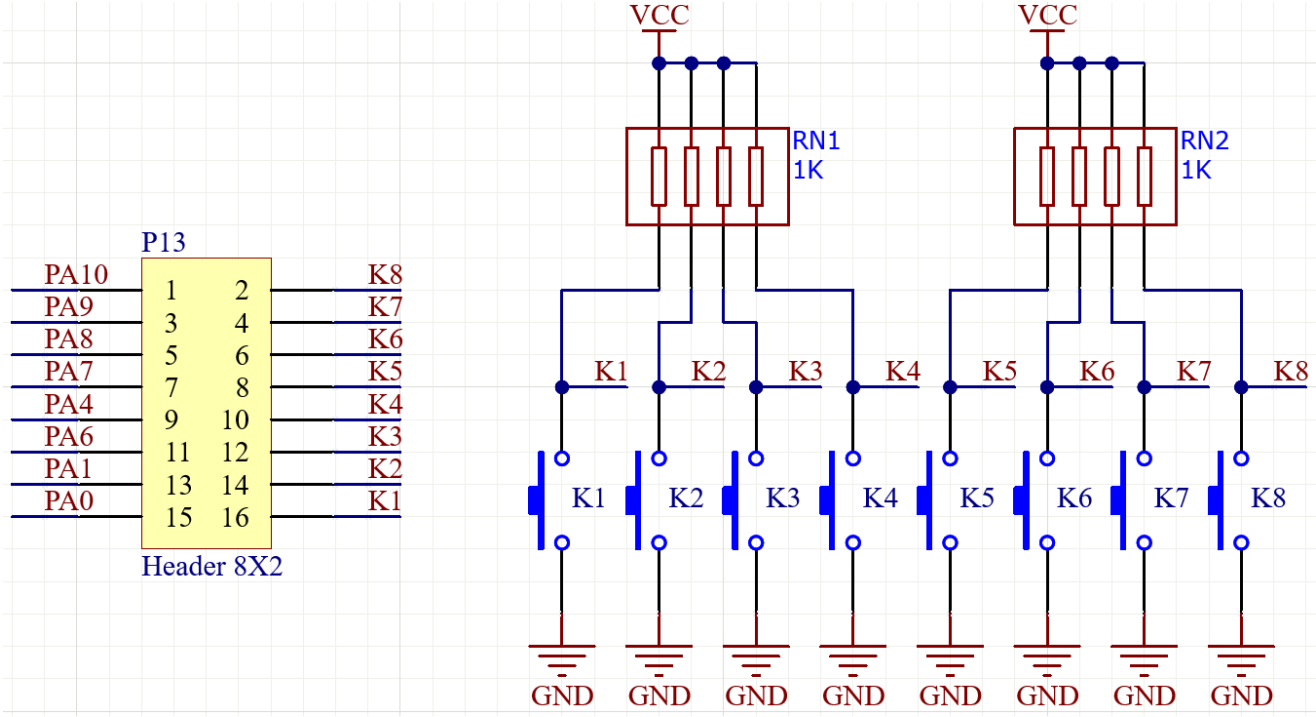
尝试上升沿触发中断和下降沿触发中断。

任务4.2、用中断方式实现：

第1次按下B1，LD2以0.25Hz频率闪烁；第2次按下B1，LD2以1Hz频率闪烁；第3次按下B1，LD2以2Hz频率闪烁；再按B1，重复上述过程。

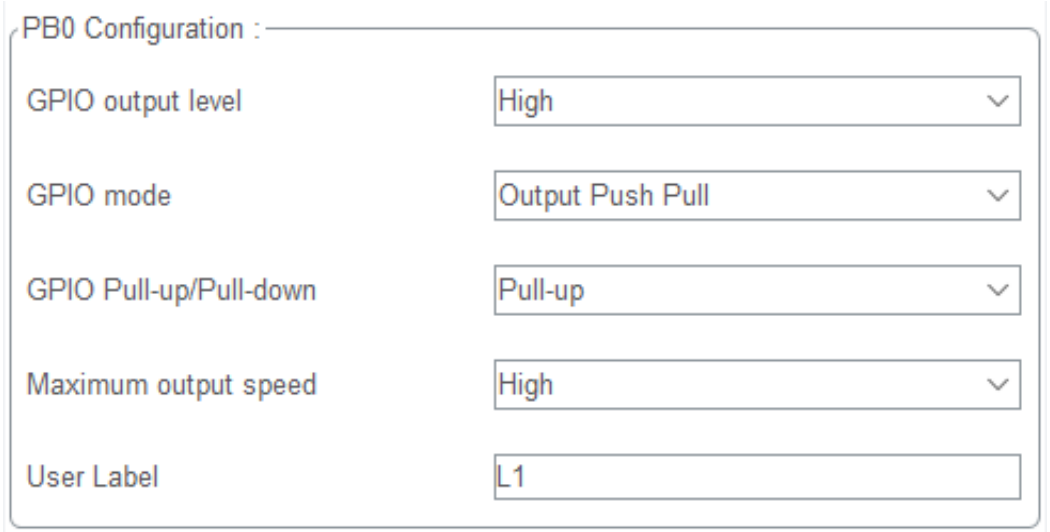
用多个按键，控制多个发光二极管和蜂鸣器

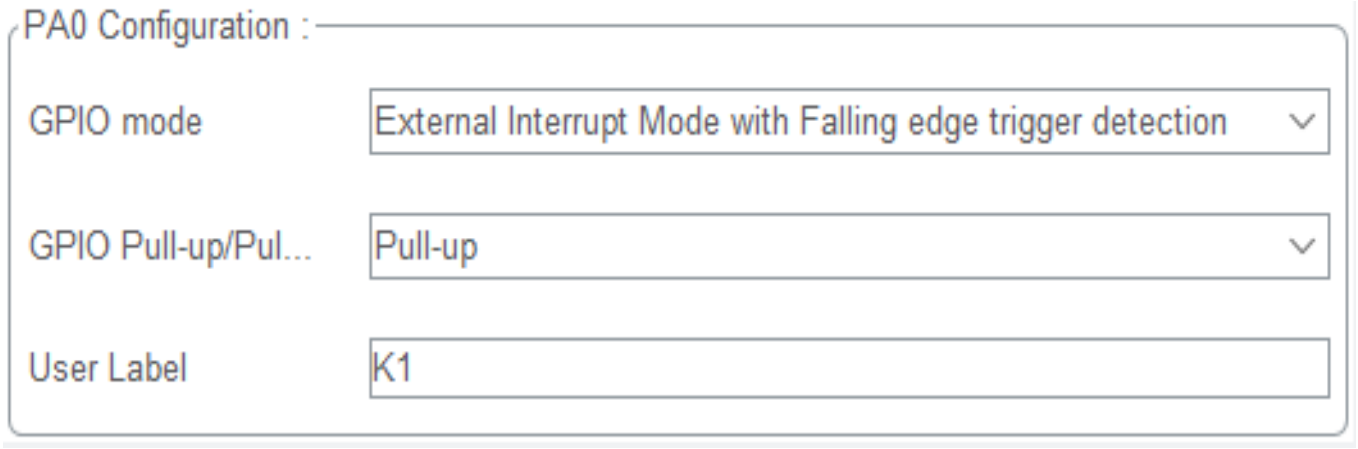
- 利用扩展板上的按键、发光二极管和蜂鸣器
- PA4控制蜂鸣器，用PC13接收按键B1的状态；PA5控制LD2
- 用扩展板上的K1控制LD2，K2控制扩展板上的L1~L4，K3控制扩展板上的L5~L8
- K1、K2和K3的状态分别用PA0、PA1和PA6接收



配置GPIO

- 打开*.ioc文件
- 将PB0~PB7、PA4和PA5配置为输出。
- PA5的User Label： nucleo_LED
- 将PA0、PA1、PA6和PC13(nucleo_KEY)分别配置为：
- GPIO_EXTI0、GPIO_EXTI1、GPIO_EXTI6和GPIO_EXTI13。





中断配置

NVIC | Code generation

Priority Group: 4 bits for pre-emption priority 0 bits for subpr... | Sort by Premption Priority and

Search | Show only enabled interrupts | Force DMA channels Interrupt

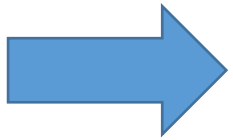
NVIC Interrupt Table	Enabled	Preemption Priority	Sub Priority
Non maskable interrupt	☑	0	0
Hard fault interrupt	☑	0	0
Memory management fault	☑	0	0
Prefetch fault, memory access fault	☑	0	0
Undefined instruction or illegal state	☑	0	0
System service call via SWI instruction	☑	0	0
Debug monitor	☑	0	0
Pendable request for system service	☑	0	0
Time base: System tick timer	☑	0	0
PVD/PVM1/PVM2/PVM3/PVM4 inte...	☐	0	0
Flash global interrupt	☐	0	0
RCC global interrupt	☐	0	0
EXTI line0 interrupt	☑	4	0
EXTI line1 interrupt	☑	3	0
EXTI line[9:5] interrupts	☑	2	0
EXTI line[15:10] interrupts	☑	1	0
FPU global interrupt	☐	0	0

代码修改

- MX_GPIO_Init()
- 中断服务函数

```
void EXTI0_IRQHandler(void)
{
  HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_0);
}
void EXTI1_IRQHandler(void)
{
  HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_1);
}
void EXTI9_5_IRQHandler(void)
{
  HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_6);
}
```

stm32g4xx_hal_gpio.c



HAL_GPIO_EXTI_Callback()

重定义回调函数HAL_GPIO_EXTI_Callback()

```
/* USER CODE BEGIN 4 */
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
        HAL_Delay(10);
        switch (GPIO_Pin)
        {
                ....
        }
}
/* USER CODE END 4 */
```

练习4：外部中断

任务4.3、用中断方式方式实现：

按下NUCLEO-G431RB板上的B1键，蜂鸣器响；

按扩展板上的K1、K2和K3键，LD2及扩展板上的L1~L8的状态改变。

任务4.4、查看stm32g4xx_it.c文件中的外部中断函数；使用设置断点、单步运行等调试手段，分析中断执行过程。

EXTI15_10_IRQHandler → HAL_GPIO_EXTI_IRQHandler → HAL_GPIO_EXTI_Callback

练习4：外部中断

任务4.5、K1,K2,K3分别代表一个3位十进制数的个、十，百分位；譬如：K1连续按下2次，K2连续按下2次，K3按下1次，表示此次输入的数为122。编写程序，识别按键表示的数，并通过L1~L8以二进制方式显示出来。

任务4.6、在任务4.5的基础上，实现：

通过扩展板上的数码管显示所输入的数值。

提交网络学堂：每个子任务的工程文件（压缩），代码有简单注释

谢　谢！