**Microcontroller Experiment**

External Interrupt Experiment

Name: Han Yichen 韩一尘

Student ID: 22722051

Date: 2023/11/1

**1.Experimental principle**

External interrupt line of STM32

Each IO of the STM32 can be used as an external interrupt input.

STM32's interrupt controller supports 19 external interrupt/event requests:

Lines 0 to 15: The input to the external I/O port is interrupted.

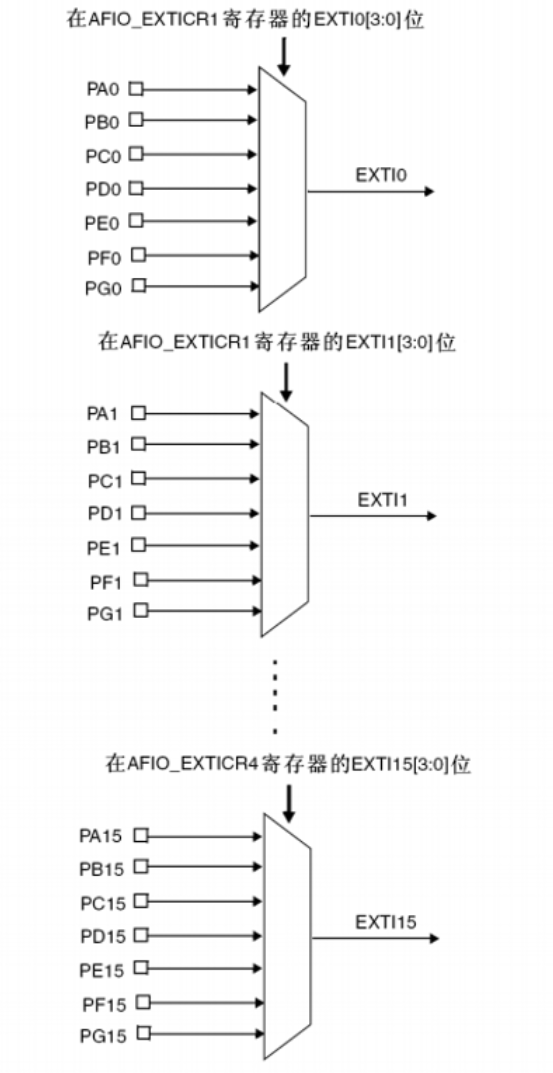
Line 16: Connects to the PVD output.

Line 17: Connects to the RTC alarm event.

Line 18: Connect to USB Wake up event.

Each external interrupt can be independently configured with trigger mode (rising edge, falling edge or double edge trigger), trigger/mask, and dedicated status bit. 2. Mapping between external interrupt cables and I/O pins

The corresponding relationship between the external interrupt line and the IO port of STM32: For each interrupt line, we can set the corresponding trigger mode (rising edge trigger, falling edge trigger, edge trigger) and enable.



typedef struct

{

uint32\_t EXTI\_Line;

EXTIMode\_TypeDef EXTI\_Mode;

EXTITrigger\_TypeDef EXTI\_Trigger;

FunctionalState EXTI\_LineCmd;

}EXTI\_InitTypeDef;

As you can see from the definition, there are four parameters that need to be set. The first parameter is the label of the trunk line. The value ranges from EXTI\_Line0 to EXTI\_Line15. This function configures interrupt parameters for an interrupt line. The second parameter is the interrupt mode, with optional values EXTI\_Mode\_Interrupt and EXTI\_Mode\_Event. The third parameter is the triggering method, which can be EXTI\_Trigger\_Falling triggered by falling edge, EXTI\_Trigger\_Rising triggered by rising edge, or EXTI\_Trigger\_Rising\_Falling triggered by any level (rising and falling edge). The last parameter is to enable the interrupt line.

We set the interrupt line and GPIO mapping relationship, and then set the initialization parameters such as the trigger mode of the interrupt. Even with external interrupts, when it comes to interrupts we of course have to set the NVIC interrupt priority. Set the interrupt priority of trunk 2.

General steps to use an external interrupt with an IO port:

1) Initialize the I/O port as input.

2) Turn on AFIO clock

3) Set the mapping relationship between the I/O port and the interrupt line.

4) Initialize online interrupts, set trigger conditions, etc.

5) Configure Interrupt packet (NVIC) and enable interrupt.

6) Write interrupt service functions.

**2.Main program analysis**

Common library function

①void GPIO\_EXTILineConfig(uint8\_t

GPIO\_PortSource, uint8\_t GPIO\_PinSource);

// Set the mapping between the I/O port and the interrupt line

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

②void EXTI\_Init(EXTI\_InitTypeDef\* EXTI\_InitStruct);

// Initialize the interrupt line: trigger mode, etc.

③ITStatus EXTI\_GetITStatus(uint32\_t EXTI\_Line);

// Check whether the interrupt line is interrupted.

④void EXTI\_ClearITPendingBit(uint32\_t EXTI\_Line);

// Clear the interrupt flag bit on the interrupt line These four functions are commonly used to configure external interrupt library functions, the first function is well understood, is to configure the mapping relationship between the I/O port and the interrupt line. Let's focus on the second function:

EXTI\_Init function: void EXTI\_Init(EXTI\_InitTypeDef\* EXTI\_InitStruct);

General procedure for configuring library functions 1 Initialize the I/O port as input. GPIO\_Init();

② Enable the I/O port multiplexing clock.

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_AFIO,ENABLE);

③ Set the mapping relationship between the I/O port and the interrupt line.

void GPIO\_EXTILineConfig();

④ Initialize online interrupts, set trigger conditions, etc.

EXTI\_Init();

⑤ Configure interrupt packet (NVIC) and enable interruption.

NVIC\_Init();

⑥ Write interrupt service functions.

EXTIx\_IRQHandler();

⑦ Clear the interrupt flag bit

EXTI\_ClearITPendingBit();

**3.Experimental result**

When WK\_UP is pressed, the status of the buzzer is reversed. When KEY\_0 is pressed, the status of LED1 is reversed. When KEY\_0 is pressed, the state of LED0 and LED1 reverses simultaneously.

**4. Improvement and perfection**

In the external interrupt experiment, more interrupt sources can be added, such as timer interrupt, serial interrupt, etc., to realize more complex functions.

The interrupt handler is optimized to improve the interrupt response speed and processing efficiency. For example, the interrupt vector table can be used to reduce the number of jumps of interrupt handlers and improve the execution speed.

In the experiment, data processing and storage functions are added, such as the use of SD card, Flash and other memory, to achieve long-term data storage and fast reading.