**Microcontroller Experiment**

The Buzzer Experiment

Name: Han Yichen 韩一尘

Student ID: 22722051

Date: 2023/11/1

1. **Experimental principle**

When this embedded C code runs on an STM32 microcontroller, it controls two LEDs by configuring GPIO ports and clocks, and utilizes delay functions for control. Now, let's provide a comprehensive analysis from the circuit perspective:

1. Clock Enable Section:

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOB| RCC\_APB2Periph\_GPIOE, ENABLE);`: Enables the clock for GPIOB and GPIOE, ensuring the functionality of these two GPIO ports.

2. LED Hardware Initialization Section:

The GPIO\_InitStructure` structure is used to configure various parameters for GPIO.

GPIO\_Pin\_5`: LED0 is connected to PB5, and LED1 is connected to PE5.

GPIO\_Mode\_Out\_PP`: Configured as push-pull output, allowing current to flow when outputting low or high levels.

GPIO\_Speed\_50MHz`: Configures GPIO output speed to 50MHz.

3. LED Initialization Section:

GPIO\_Init(GPIOB,&GPIO\_InitStructure);and

GPIO\_Init(GPIOE, &GPIO\_InitStructure);`:Initializes GPIOB and GPIOE with the above configuration to control LED0 and LED1.

GPIO\_SetBits(GPIOB, GPIO\_Pin\_5);andGPIO\_SetBits(GPIOE, GPIO\_Pin\_5);`: Sets PB5 and PE5 to output high levels, initially turning off LED0 and LED1.

4. \*\*LED Control Section:

In the main loop, it achieves the alternation of LED0 and LED1 illumination by changing the corresponding GPIO output states.

LED0=0; and `LED1=1; Illuminates LED0 and turns off LED1.

LED0=1; and `LED1=0; Turns off LED0 and illuminates LED1.

5. Delay Section:

delay\_ms(300);`: Implements the time during which LED stays on or off using delay functions. In this case, LED remains on or off for 300 milliseconds.

In summary, this code controls the on-off state of LEDs at the hardware circuit level using GPIO on an STM32 microcontroller, and the blink rate and on-off time of LEDs are controlled through delay functions. Such a control structure provides a simple yet effective implementation for basic LED control in embedded systems.

**2. Main program analysis**

This experiment requires the use of code to drive LED operation, so it is necessary to import the led. h file, which is shown as following:

#include "led.h" //初始化 PB5 和 PE5 为输出口.并使能这两个口的时钟

//LED IO 初始化

void LED\_Init(void)

{

GPIO\_InitTypeDef

GPIO\_InitStructure;

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOB| RCC\_APB2Periph\_GPIOE, ENABLE);//使能 PB,PE 端口时钟 GPIO\_InitStructure.GPIO\_Pin=GPIO\_Pin\_5;//LED0-->PB.5 推挽输出 GPIO\_InitStructure.GPIO\_Mode=GPIO\_Mode\_Out\_PP;//推挽输出 GPIO\_InitStructure.GPIO\_Speed=GPIO\_Speed\_50MHz;

GPIO\_Init(GPIOB, &GPIO\_InitStructure);

GPIO\_SetBits(GPIOB,GPIO\_Pin\_5); //PB.5 输出高

GPIO\_InitStructure.GPIO\_Pin=GPIO\_Pin\_5; //LED1-->PE.5 推挽输出 GPIO\_Init(GPIOE,&GPIO\_InitStructure);

GPIO\_SetBits(GPIOE,GPIO\_Pin\_5); /PE.5 输出高

}

Next, write the main function responsible for the main execution:

#include "led.h"

#include "delay.h"

#include "sys.h"

int main(void)

{

delay\_init(); //延时函数初始化

LED\_Init(); //初始化与 LED 连接的硬件接口

while(1)

{

LED0=0;

LED1=1;

delay\_ms(300); //延时 300ms

LED0=1;

LED1=0;

delay\_ms(300); //延时 300ms

}

}

It is worth mentioning that in STM's own library, the flag bit set for the LED to light up is 0 and the flag bit set for the LED to turn off is 1, which is somewhat different from our general understanding.

**3.Experimental result**

After compiling and importing the code into STM32, it was found that LED0 (red) and LED1 (green) flickered alternately for the set parameter time (300 milliseconds). By attempting to modify the set parameters, it can be seen that the flashing speed of the two LED lights has changed.

**4. Improvement and perfection**

Due to being the first time conducting experiments on microcontrollers and the outdated compiler, it has caused considerable trouble in compiling and importing programs. Complete the experiment by flipping through the experimental manual and asking classmates.