南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

Embedded System and Microcomputer Principle

LAB13 IIC Communication

2024 Fall
wangq9@mail.sustech.edu.cn

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

CONTENTS

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

01

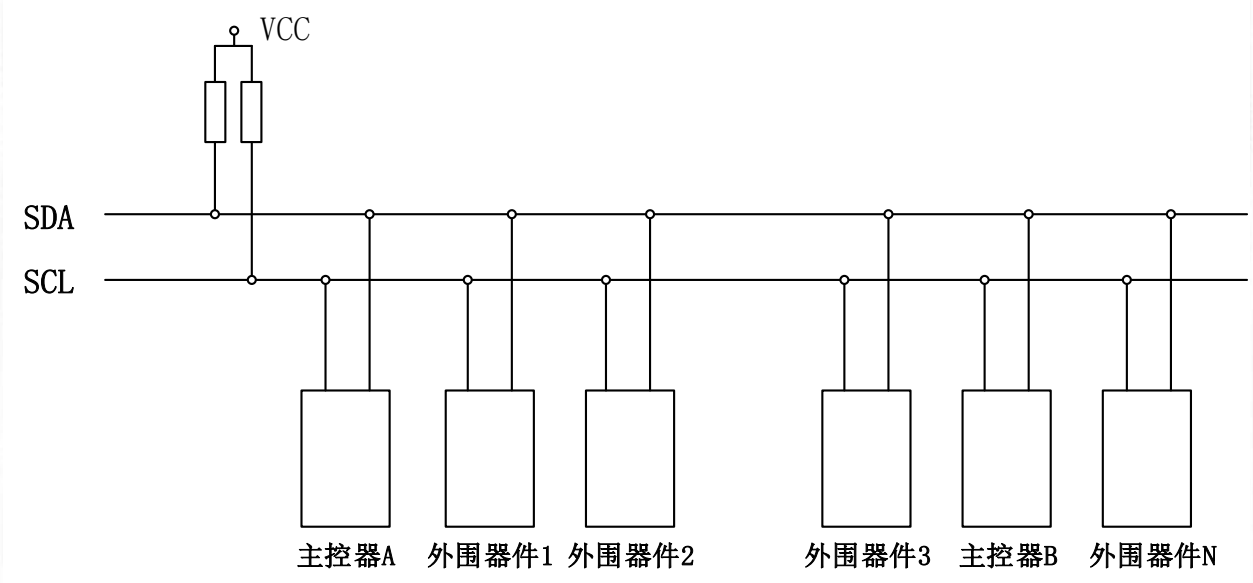
IIC Principle Description

1. IIC Principle Description

-- Introduction

- IIC: Inter Integrated Circuit
- Developed by PHILIPS, used to connect MCU and peripheral devices
- Synchronous serial half duplex communication bus
- Two lines: SCL (Serial clock), SDA (Serial data)
- Data transmission speed: Standard mode 100k bit/s, Fast mode 400k bit/s, High speed mode 3.4Mbit/s
- Mainly used for communication between chips at close range and low speed
- Low cost, simple structure, poor anti-interference ability

1. IIC Principle Description
-- Bus structure



- Consisting of a clock line SCL and a data line SDA, both of which are connected with pull-up resistors to ensure that the bus is idle at a high level.
- Supports multiple device connections and allows for the existence of multiple hosts, with each device having a unique address.

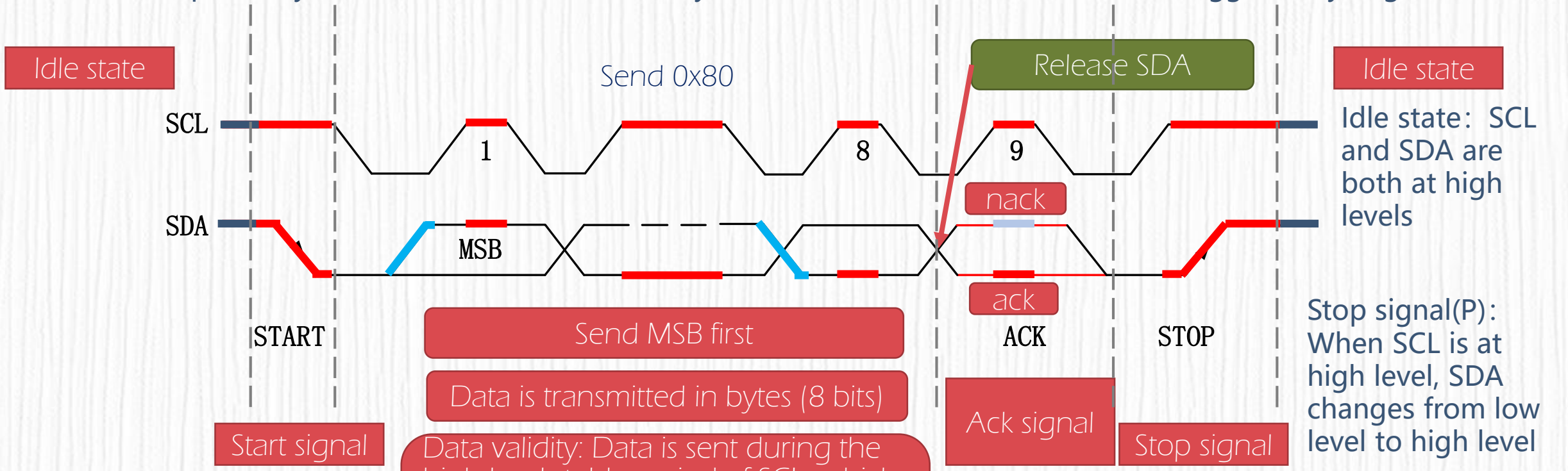
1. IIC Principle Description

-- Main features

- Three signals
  - Start signal
  - Stop signal
  - Acknowledge signal (ACK/NACK)
- Two notes
  - Data validity
  - Data transmission sequence
- One state
  - Idle state

1. IIC Principle Description -- Work timing

Data transmission: Each bit of data transmitted on the IIC bus corresponds to a clock pulse (or synchronous control), that is, with the coordination of the SCL serial clock, each bit of data is sequentially transmitted on the SDA bit by bit. The transmission of data bits is triggered by edges.



Idle state: SCL and SDA are both at high levels

Stop signal(P): When SCL is at high level, SDA changes from low level to high level

Start signal(S): When SCL is at high level, SDA changes from high level to low level

Data validity: Data is sent during the high-level stable period of SCL, which means that the data needs to be prepared before the rising edge of SCL arrives and must be stable before the falling edge arrives.

Acknowledge signal: ACK: While the slave pulls down SDA in the 9th clock cycle to confirm receipt of data. Otherwise, sends NACK signal.

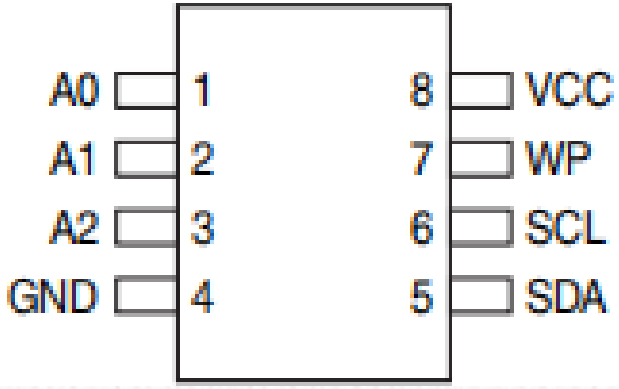
南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

02

AT24C02 Principle Description

2. AT24C02 Principle Description -- AT24C02 chip

- 2K bit serial CMOS EEPROM memory
- 256 8-bit bytes
- Operated through the IIC bus, with a dedicated write protection function
- EEPROM is a type of storage device that does not lose data after power off. It is commonly used to store configuration information and can be loaded when the system is powered on again.



AT24C02 pin diagram

Pin	Function
A0/1/2	Device address determination pin
WP	Write protection pin
SCL	Clock line
SDA	Data line

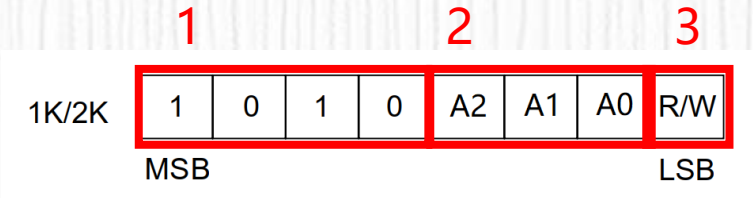
2. AT24C02 Principle Description
-- AT24C02 address

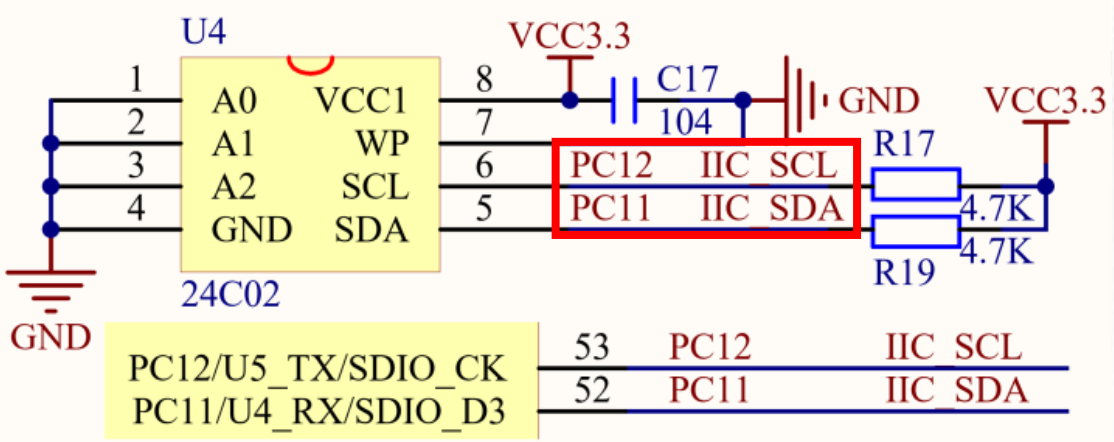
- In MiniSTM32, A0, A1, and A2 are all grounded
- Write operation address: 0xA0
- Read operation address: 0xA1

1： Unprogrammable part

2： Programmable part, determined by hardware pins A0/1/2

3: Data transmission direction, read data '1', write data '0'

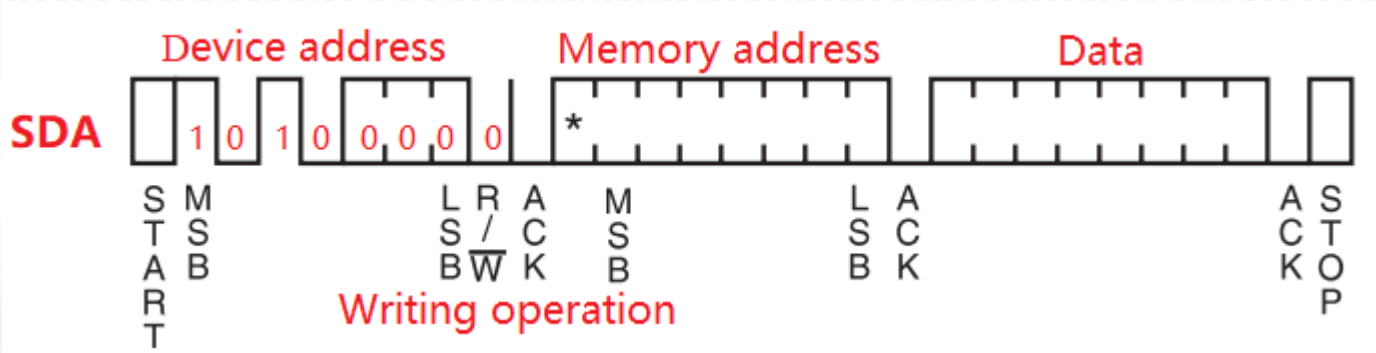




Circuit connection diagram between AT24C02 and STM32

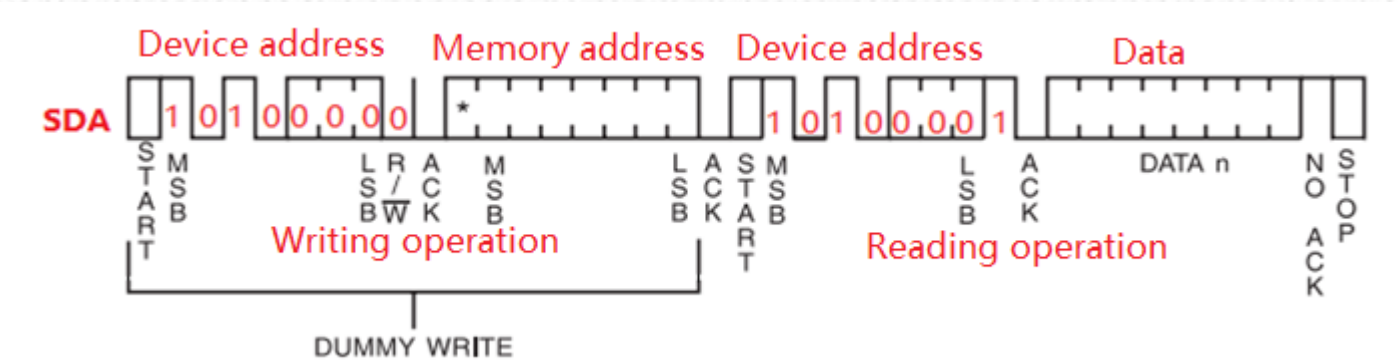
2. AT24C02 Principle Description
-- AT24C02 write timing

- The host sends the first byte of data on the IIC bus with device address 0xA0 of 24C02.
- After receiving the acknowledge signal of 24C02, send the memory address of 24C02.
- After waiting for the ACK signal of 24C02, send the data written to the memory address.
- After the host completes the writing operation, it can issue a stop signal to terminate data transmission.



2. AT24C02 Principle Description
-- AT24C02 read timing

- The process of reading includes writing and reading timing.
- Write first. After the start signal is generated, the host sends the 24C02 device address 0xA0, and sends the memory address that needs to be read after obtaining the slave acknowledge signal.
- In the read timing, after the start signal is generated, the host sends the device address 0xA1, after obtaining the slave acknowledge signal, and then the slave returns the data in the write timing.
- If the host returns a ACK signal after obtaining data, the slave will continue to transmit data.
- When the host sends a NACK signal, the slave ends the transmission.



南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

03

How to Program

3. How to Program

- Our Goal
  - Establish communication between 24C02 and MCU using IIC protocol through software simulation.
  - Use KEY0 and KEY1 to trigger read/write operations.
  - Each time KEY1 is pressed, the MCU writes data to 24C02 through the IIC bus.
  - Each time KEY0 is pressed, MCU read data from 24C02.
  - Use LCD screen for operation prompts and output display.
  - The hardware IIC of STM32F1 is not used in this demo. In order to avoid the Philips IIC patent issue, ST designed the hardware IIC of STM32 more complicated.

3. How to Program

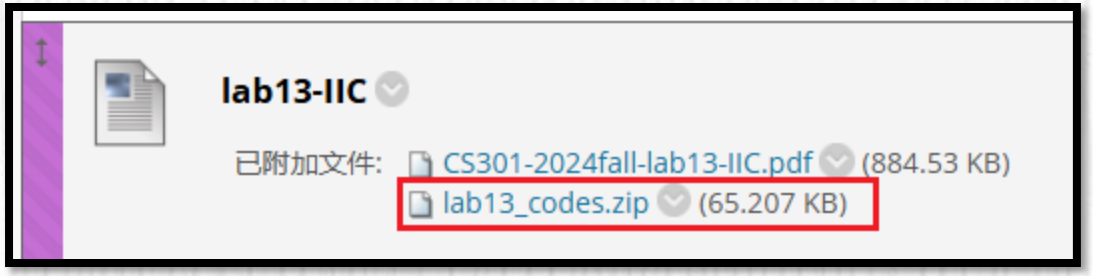
南方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

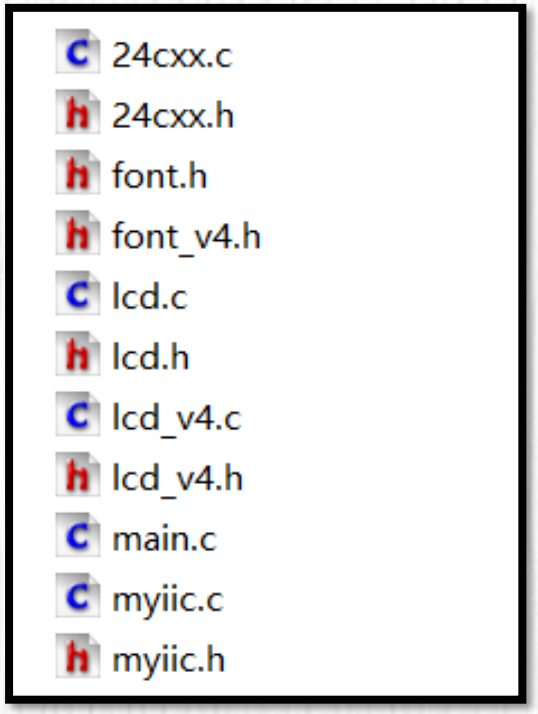
- Configure GPIO
  - Set PC11 as output open drain
  - Set PC12 as output push pull
  - Set KEY0, KEY1, LED0, LED1

Pin ...	Signal...	GPIO ...	GPIO mode	GPIO Pull-up/Pull-down	Maximum o...	User Label	Modified
PA8	n/a	High	Output Push Pull	No pull-up and no pull-down	Low	LED0	☑
PA15	n/a	n/a	Input mode	Pull-up	n/a	KEY1	☑
PC5	n/a	n/a	Input mode	Pull-up	n/a	KEY0	☑
PC11	n/a	Low	Output Open Drain	No pull-up and no pull-down	Low	IIC_SDA	☑
PC12	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	IIC_SCL	☑
PD2	n/a	High	Output Push Pull	No pull-up and no pull-down	Low	LED1	☑

3. How to Program

- Download lab13_codes.zip from Blackboard.
- V3: put 24cxx.c, myiic.c, lcd.c, main.c in Src folder, and 24cxx.h, myiic.h, lcd.h, font.h in Inc folder.
- V4: put 24cxx.c, myiic.c, lcd_v4.c, main.c in Src folder, and 24cxx.h, myiic.h, lcd_v4.h, font_v4.h in Inc folder.





3. How to program

- Functions
  - void IIC_Init(void)
  - void IIC_Start(void)
  - void IIC_Stop(void)
  - uint8_t IIC_Wait_Ack(void)
  - void IIC_Ack(void)
  - void IIC_NAck(void)
  - void IIC_Send_Byte(uint8_t txd)
  - uint8_t IIC_Read_Byte(unsigned char ack)

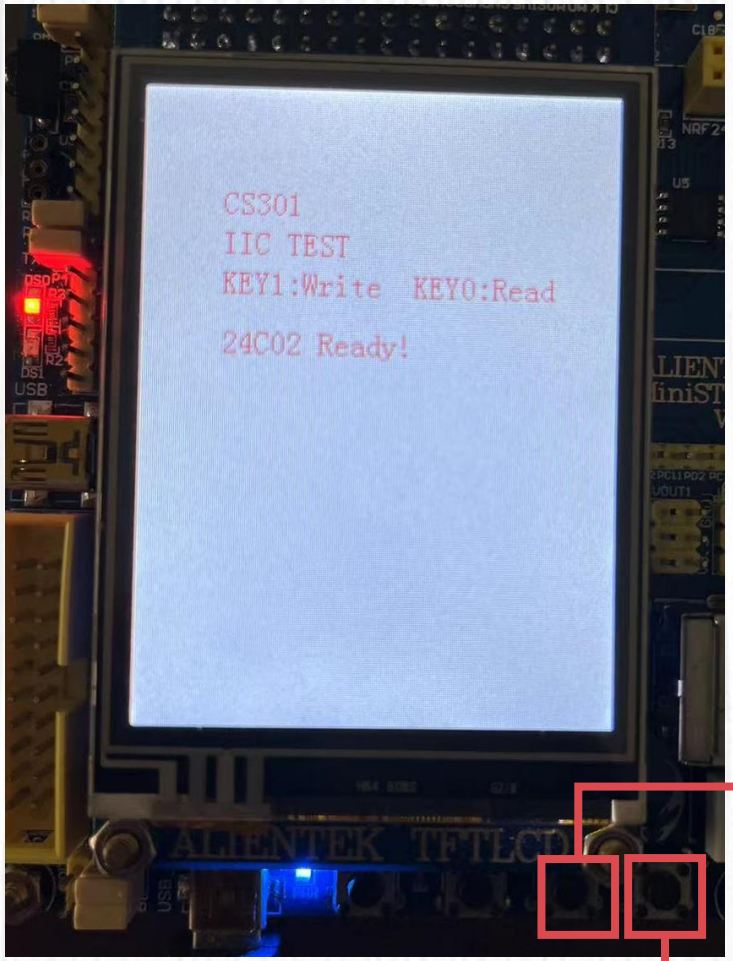
3. How to program

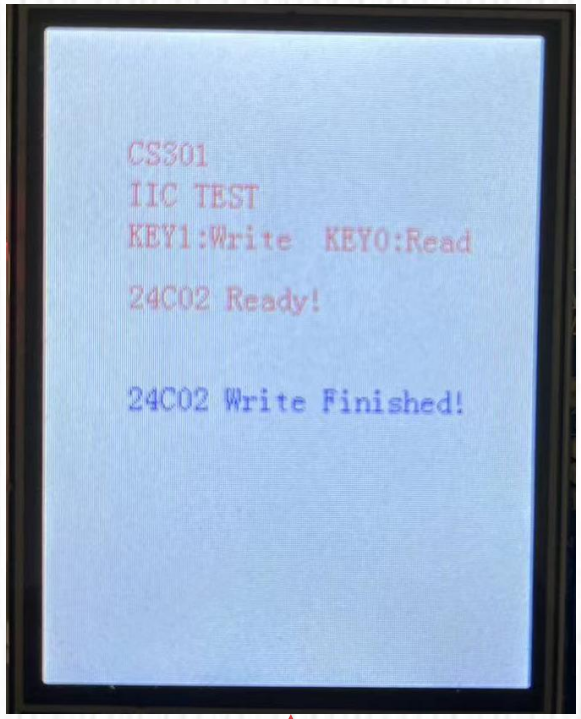
- Main program

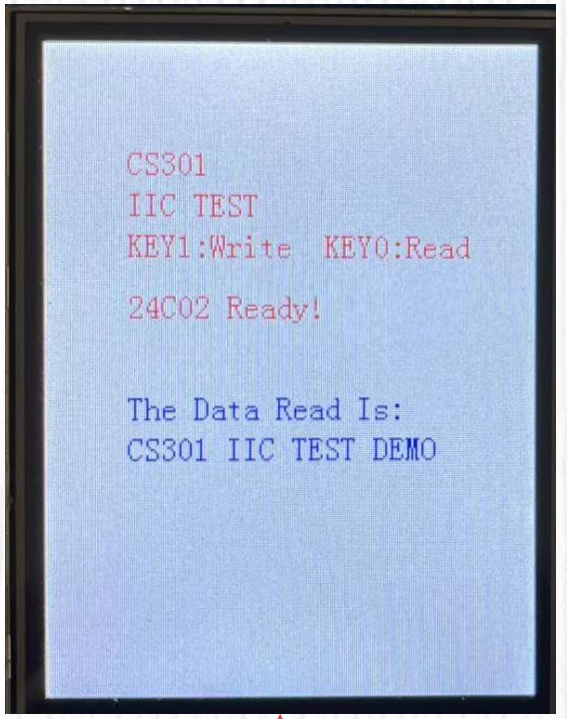
```
AT24CXX_Init();                    //IIC initialization
……
while(AT24CXX_Check()){
        ……
}
LCD_ShowString(40,120,200,16,16,"24C02 Ready!"); //Use lcd_show_string() on V4

while (1){
        if(HAL_GPIO_ReadPin(KEY1_GPIO_Port, KEY1_Pin) == GPIO_PIN_RESET) { //KEY1 is pressed, write
                ……
                AT24CXX_Write(0,(uint8_t*)TEXT_Buffer,SIZE);
        }
        if(HAL_GPIO_ReadPin(KEY0_GPIO_Port, KEY0_Pin) == GPIO_PIN_RESET) { //KEY0 is pressed, read
                ……
                AT24CXX_Read(0,datatemp,SIZE);
                LCD_ShowString(40,190,200,16,16,datatemp); //Display the string
        }
}
```

3. How to program
-- Results







04

Practice

4. Practice

- Run the demo on MiniSTM32 board.
- Use USART1 to send the data for writing into AT24C02 chip, and then read the new data from AT24C02, show the new data on LCD screen or send the new data to PC using USART1.
- Writing sequence: PC -> MiniSTM32 -> 24C02
- Reading sequence: 24C02 -> MiniSTM32 (-> PC)