

USB emulated U disk

USB emulated U disk

- 1、software to hardware
- 2、Brief principle
 - 2.1、Hardware schematic diagram
 - 2.2、Physical connection diagram
 - 2.3、Principle of control
- 3、Engineering configuration
 - 3.1、Notes
 - 3.2、Pin configuration
- 4、Main Function
 - 4.1、User function
 - 4.2、HAL library function parsing
- 5、Experimental phenomenon

This tutorial demonstrates: using **USB Full Speed Device interface** to simulate USB flash drive to store files (OLED display W25Q64, STM32 USB connection information).

1、software to hardware

- **STM32F103CubeIDE**
- **STM32 robot expansion board**

USB: Chip internal peripheral

W25Q64: Onboard

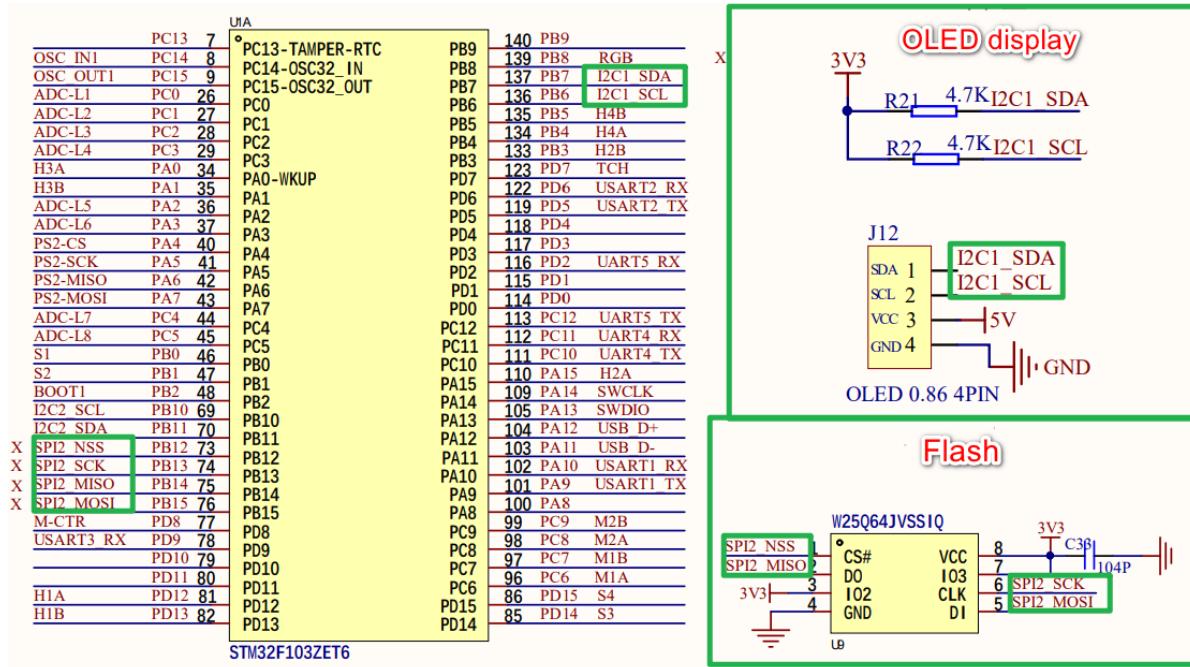
OLED: External

- **Type-Cv cable or ST-Link**

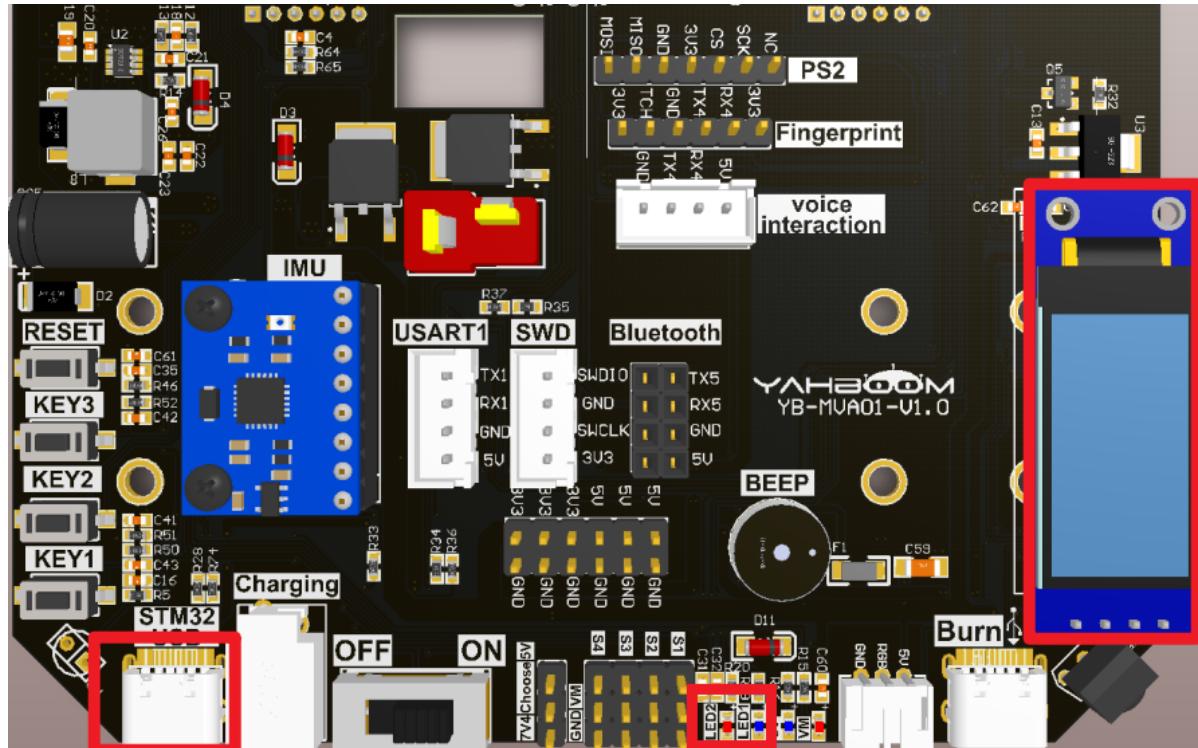
Download or simulate the program of the development board

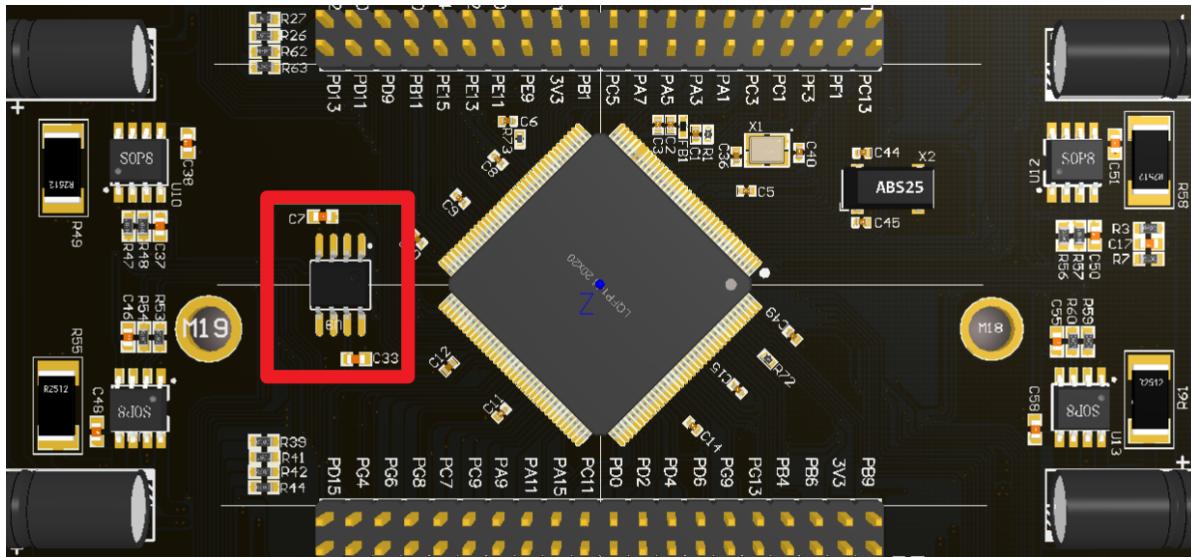
2、Brief principle

2.1、Hardware schematic diagram



2.2、Physical connection diagram





OLED	STM32 board
VCC	VCC
SCL	SCL
SDA	SDA
GND	GND

STM32 USB	PC
Type-C interface	USB interface

The Type-C data cable was used to connect the computer to the STM USB interface

2.3. Principle of control

- **USB:** (Universal Serial Bus)

USB	Characteristics
Class of speed	Low speed (USB1.0: 1.5Mbps) Full speed (USB1.1: 12Mbps) High speed (USB2.0: 480Mbps) Super high speed (USB3.0: 5Gbps, USB3.1: 10Gbps) The development board is in full speed mode
Communication interface	Development board uses 16-pin Type-C interface (see schematic)
Data transfer	Control transmission Interrupt transmission Bulk transfer Isochronous transmission

USB	Characteristics
Advantages	<ol style="list-style-type: none"> 1. Support hotplugging and PNP (Plug-and-Play); 2. Provide USB peripherals with various speed levels to adapt to different requirements; 3. The port has very flexible expansibility; 4. Provides a single, easy-to-use standard connection type; 5. To meet the requirements of various types of peripheral devices, USB provides four different data transfer types.

- **USB Protocol**

The USB bus uses differential signal transmission, where D+ and D- represent the positive and negative data lines of the differential signal respectively.

USB1.1 and USB2.0 use a differential signal level of 3.3V and USB3.0 uses a differential signal level of 5V.

USB related knowledge can go to the "USB Encyclopedia of USB Chinese network" for understanding and learning, the following only introduces the common concepts

- **Full speed/low speed USB bus device connection method**

High speed/full speed equipment: D+ connected with 1.5k pull-up resistor;

Low speed equipment: D-connected 1.5k pull-up resistor;

- **Bus status**

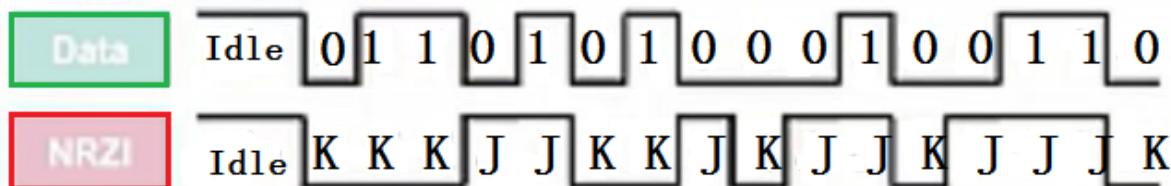
Level type	Scope
Difference 1	D+>2.8V, D-<0.3V
Difference 0	D+<0.3V, D-<0.3V
Single ended 0 (SE0)	D+<0.3V, D-<0.3V
Single ended 1 (SE1)	D+>2.8V, D->2.8V
J state: Low speed Full speed High speed	Difference 0 Difference 1 Difference 1
K state Low speed Full speed High speed	Difference 1 Difference 0 Difference 0
Restore status	K state
Packet start (SOF)	The USB data bus switches from the idle state to the K state

Level type	Scope
Packet end (EOP)	The SEO lasts for two basic time units, and the J state lasts for one time unit

- **Encoding of USB signals**

The data in USB is encoded by NRZI (none-return to zero inverted), and the different states of D+ and D- lines are defined as J and K.

Encoding/decoding of data (reverse non-return-to-zero code)



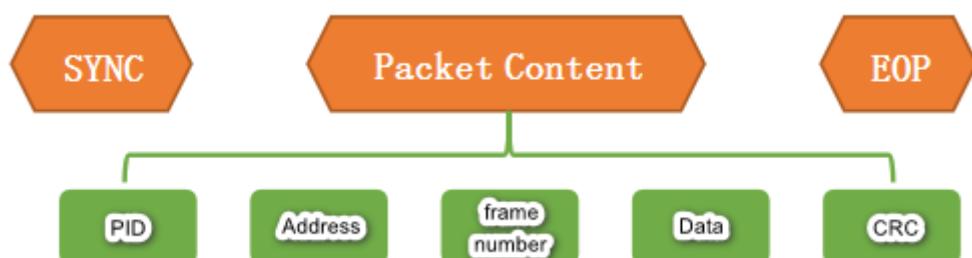
Note: The edge that meets 0 changes its state. If it is 1, the original state remains

Before the data is NRZI encoded, a 0 signal is inserted after every six consecutive 1 signals so as not to lose synchronization.

- **USB Protocol layer**

Packet is the basic unit of information transmission in USB system. All data is transmitted on the bus after being packaged.

Each packet consists of six parts: synchronization field (SYNC), packet identifier (PID), address field (ADDR), DATA field (including frame number), validation field (CRC), End of Packet (EOP).



- **Practical Applications**

The STM32F103ZET6 has an embedded device controller compatible with full speed USB, which complies with the standard for full speed USB devices (12 MBPS).

The USB-specific 48MHz clock is generated directly from the internal main PLL (the clock source must be a HSE crystal oscillator).

source file	function
usb_device.c	USB device initialization function
usbd_conf.c	USB protocol parameters, GPIO initialization, and other functions
usbd_storage_if.c	USB storage device related functions

source file	function
usbd_desc.c	USB descriptor and USB enumeration processing

USB related knowledge does not do too much introduction, you can consult the information on the Internet, mainly introduces the functions used in engineering

3、Engineering configuration

Project Configuration: Prompts for configuration options in the STM32CubeIDE project configuration process

3.1、Notes

Omitted project configuration: **New project, chip selection, project configuration, SYS for pin configuration, RCC configuration, clock configuration, and project configuration** content

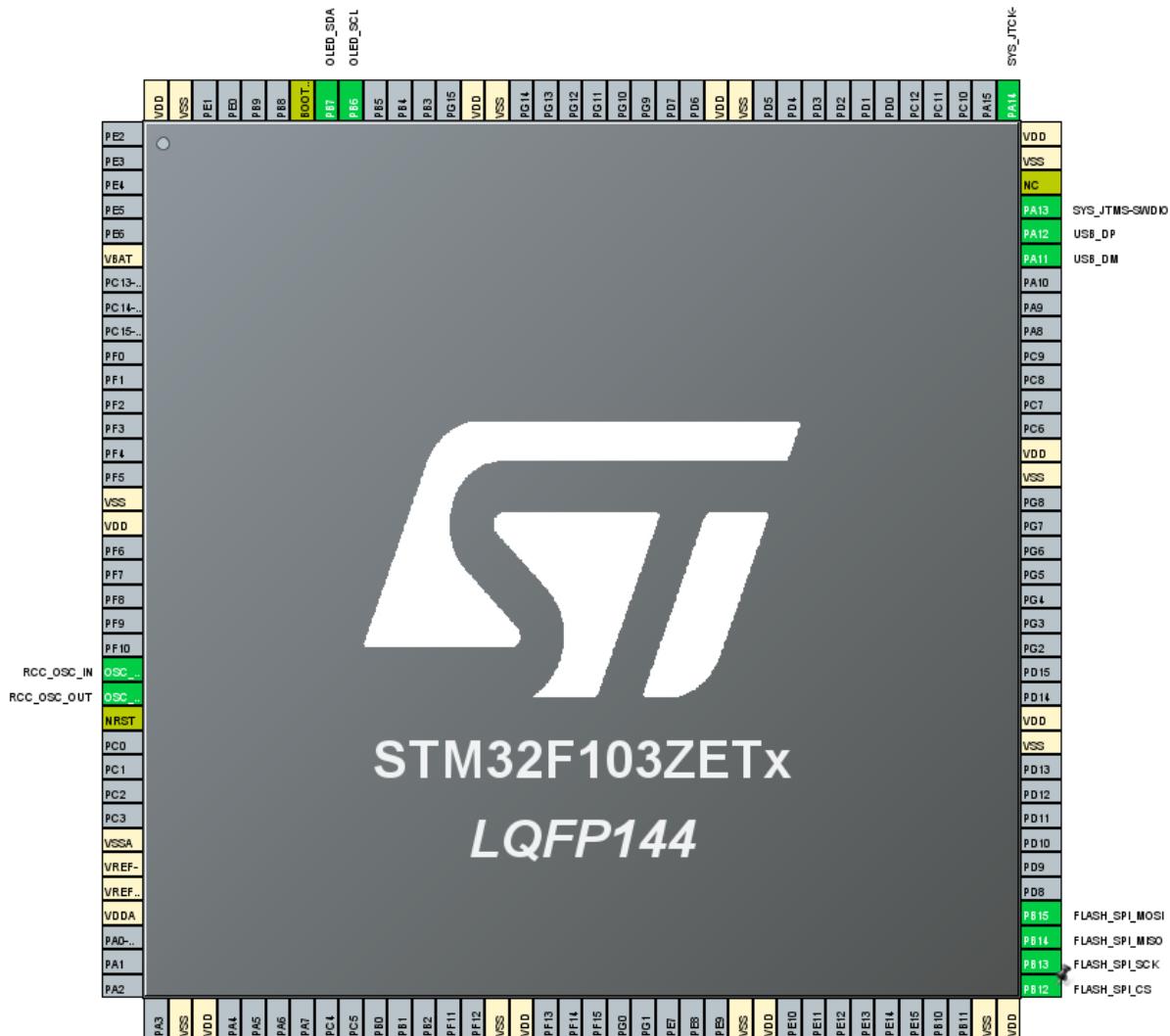
The project configuration part, which is not omitted, is the key point to configure in this tutorial.

Please refer to [2. Development environment construction and use: STM32CubeIDE installation and use] to understand how to configure the omitted parts of the project.

3.2、Pin configuration

- **Configure the specified pin function**

You can directly select the corresponding pin number in the pin view, and the corresponding option will appear when the mouse is left clicked



- I2C

Pinout & Configuration

Clock Configuration

Software Packs

Pinout

I2C1 Mode and Configuration

Mode

I2C I2C

Configuration

Reset Configuration

Parameter Settings **User Constants** **NVIC Settings** **DMA Settings** **GPIO Settings**

Configure the below parameters :

Master Features

- I2C Speed Mode
- I2C Clock Speed (Hz)
- Fast Mode Duty Cycle

Slave Features

- Clock No Stretch Mode
- Primary Address Length selection
- Dual Address Acknowledged
- Primary slave address
- General Call address detection

Fast Mode
400000
Duty cycle Tlow/Thigh = 2
Disabled
7-bit
Disabled
0
Disabled

- SPI

Pinout & Configuration

Clock Configuration

SPI2 Mode and Configuration

Mode

Mode: Full-Duplex Master
Hardware NSS Signal: Disable

Configuration

Parameter Settings **User Constants** **NVIC Settings** **DMA Settings** **GPIO Settings**

Configure the below parameters :

Basic Parameters

- Frame Format: Motorola
- Data Size: 8 Bits
- First Bit: MSB First

Clock Parameters

- Prescaler (for Baud Rate): 2
- Baud Rate: 18.0 MBits/s
- Clock Polarity (CPOL): Low
- Clock Phase (CPHA): 1 Edge

Advanced Parameters

- CRC Calculation: Disabled
- NSS Signal Type: Software

- **USB**

Pinout & Configuration

Clock Configuration

USB Mode and Configuration

Mode

Device (FS)

Configuration

Parameter Settings **User Constants** **NVIC Settings** **DMA Settings** **GPIO Settings**

Configure the below parameters :

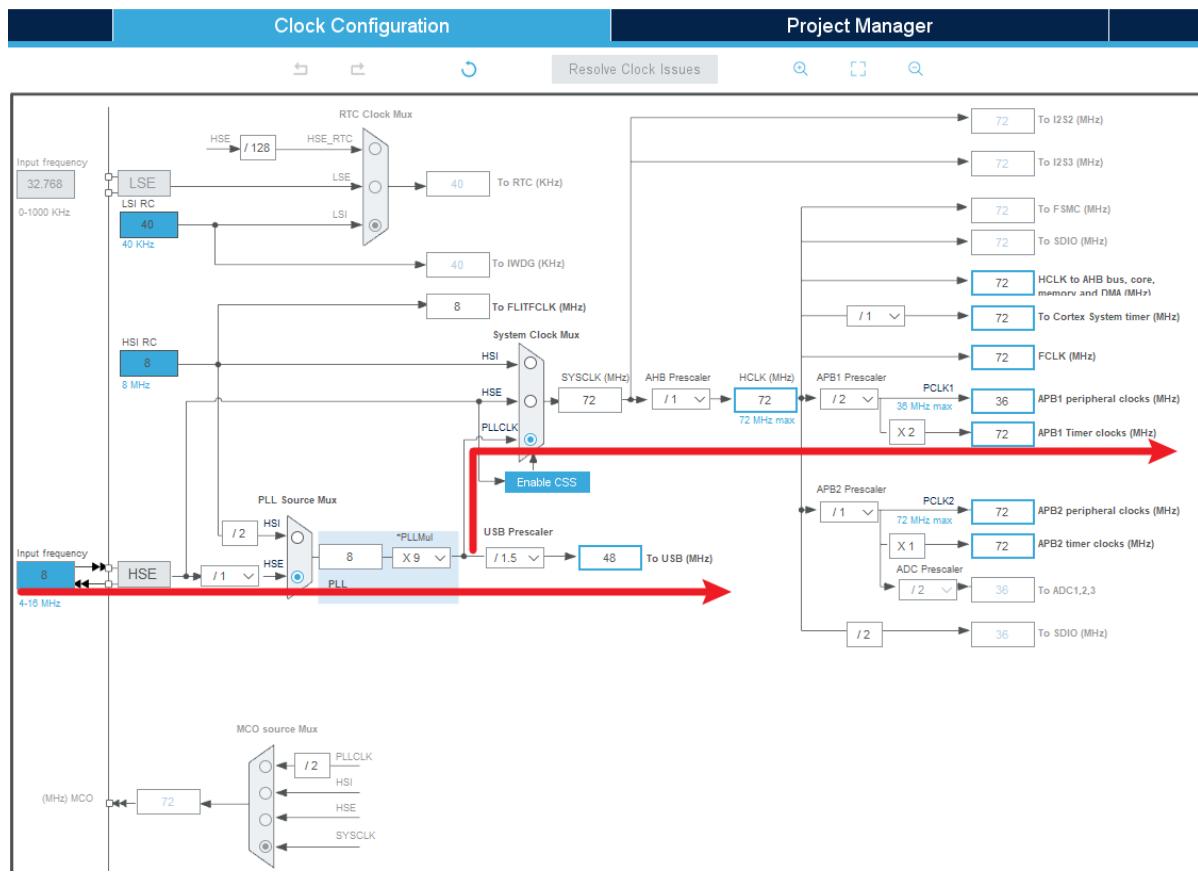
Basic Parameters

- Speed: Full Speed 12MBit/s

Power Parameters

- Low Power: Disabled
- Link Power Management: Disabled
- Battery Charging: Disabled

- **Clock Configuration**



- Advanced Settings

Pinout & Configuration **Clock Configuration** **Project Manager**

Driver Selector	
RCC	HAL
GPIO	HAL
> I2C	HAL
> SPI	HAL
USB_DEVICE	HAL

Generated Function Calls

Generate Code	Rank	Function Name	Peripheral Instance Name	<input type="checkbox"/> Do Not Generate Function Call	Visibility (Static)
✓	1	SystemClock_Config	RCC	<input type="checkbox"/>	<input type="checkbox"/>
✓	2	MX_GPIO_Init	GPIO	<input type="checkbox"/>	<input checked="" type="checkbox"/>
✓	3	MX_I2C1_Init	I2C1	<input type="checkbox"/>	<input checked="" type="checkbox"/>
✓	4	MX_SPI2_Init	SPI2	<input type="checkbox"/>	<input checked="" type="checkbox"/>
✓	5	MX_USB_DEVICE_Init	USB_DEVICE	<input type="checkbox"/>	<input type="checkbox"/>

- Generating code



4、Main Function

This section mainly introduces the functional code written by users. **Detailed code can be opened by yourself in the project file we provide, and enter the Bsp folder to view the source code.**

4.1、User function

Functions that were covered in the previous tutorial will not be covered

function: VCP_Status

Function prototypes	void VCP_Status(void)
Functional Description	Check the USB connection status
Input parameters	None
Return value	None

4.2、HAL library function parsing

The HAL library functions that were covered in the previous tutorial will not be covered

If you want to find the HAL library and LL library function analysis involved in the entire tutorial, you can view the documents in the folder [8. STM32 Manual: STM32F1_HAL Library and LL Library_User Manual]

function: MX_USB_DEVICE_Init

Function prototypes	void MX_USB_DEVICE_Init(void)
Functional Description	Initialize the USB device
Input parameters	None
Return value	None

function: MX_USB_DEVICE_Init

Function prototypes	USBD_StatusTypeDef USBD_Init (USBD_HandleTypeDef *pdev, USBD_DescriptorsTypeDef *pdesc, uint8_t id)
Functional Description	Initialize the configuration and parameters of the USB device
Input parameters1	pdev : Handle to a USB device
Input parameters2	pdesc : USB device descriptor information
Input parameters3	id : The identifier of the USB device
Return value	USB status: USBD_OK, USBD_BUSY, USBD_FAIL

function: USB_LP_CAN1_RX0_IRQHandler

Function prototypes	void USB_LP_CAN1_RX0_IRQHandler(void)
Functional Description	Handle USB_LP_CAN1_RX0_IRQHandler interrupt (USB low priority and CAN1 receive 0 interrupt)
Input parameters	None
Return value	None

5、Experimental phenomenon

After downloading the program successfully, press the RESET button of the development board to observe the OLED display and LED display

For program download, please refer to [2. Development environment construction and use: program download and simulation]

phenomenon:

The STM32 USB interface is not connected:

The first line of OLED shows: "W25Q64 find! "

OLED second line: "USB disconnect"

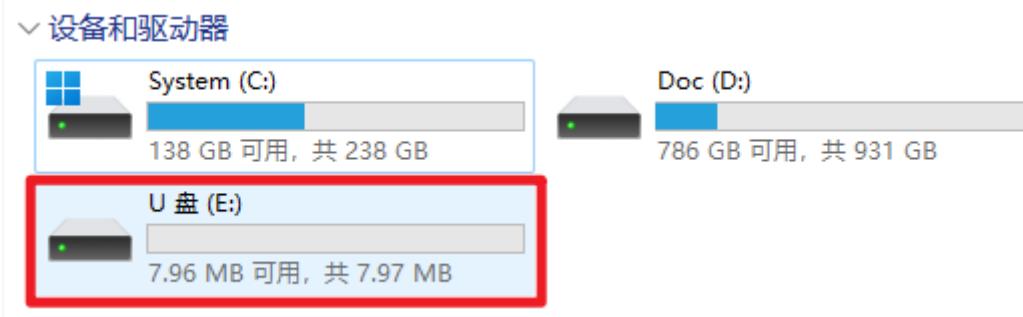
Connect the STM32 USB port:

The first line of OLED shows: "W25Q64 find! "

OLED second line: "USB connect"

The USB flash drive recognized by the system for the first time, needs to be formatted to open it;

Format U disk must carefully check the label, do not format other disks, otherwise the data in the disk will be lost



SPI FLASH has a limit to write data, it is best not to write data to it frequently, otherwise it is easy to write SPI FLASH bad

Note the Type-C data line interface location for the following experimental phenomena

