I2C Communication

Introduction

In this tutorial, we will learn how to use the I2C interface of K230 to communicate

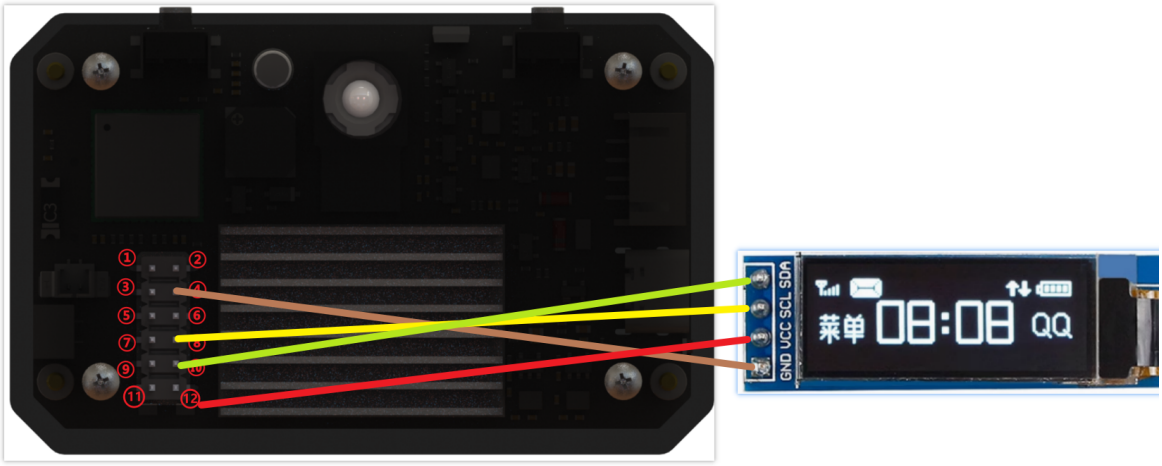
We will use K230's IIC1 to light up a 128*32 OLED screen as an example of IIC usage.

> What is IIC (I2C, I²C)?

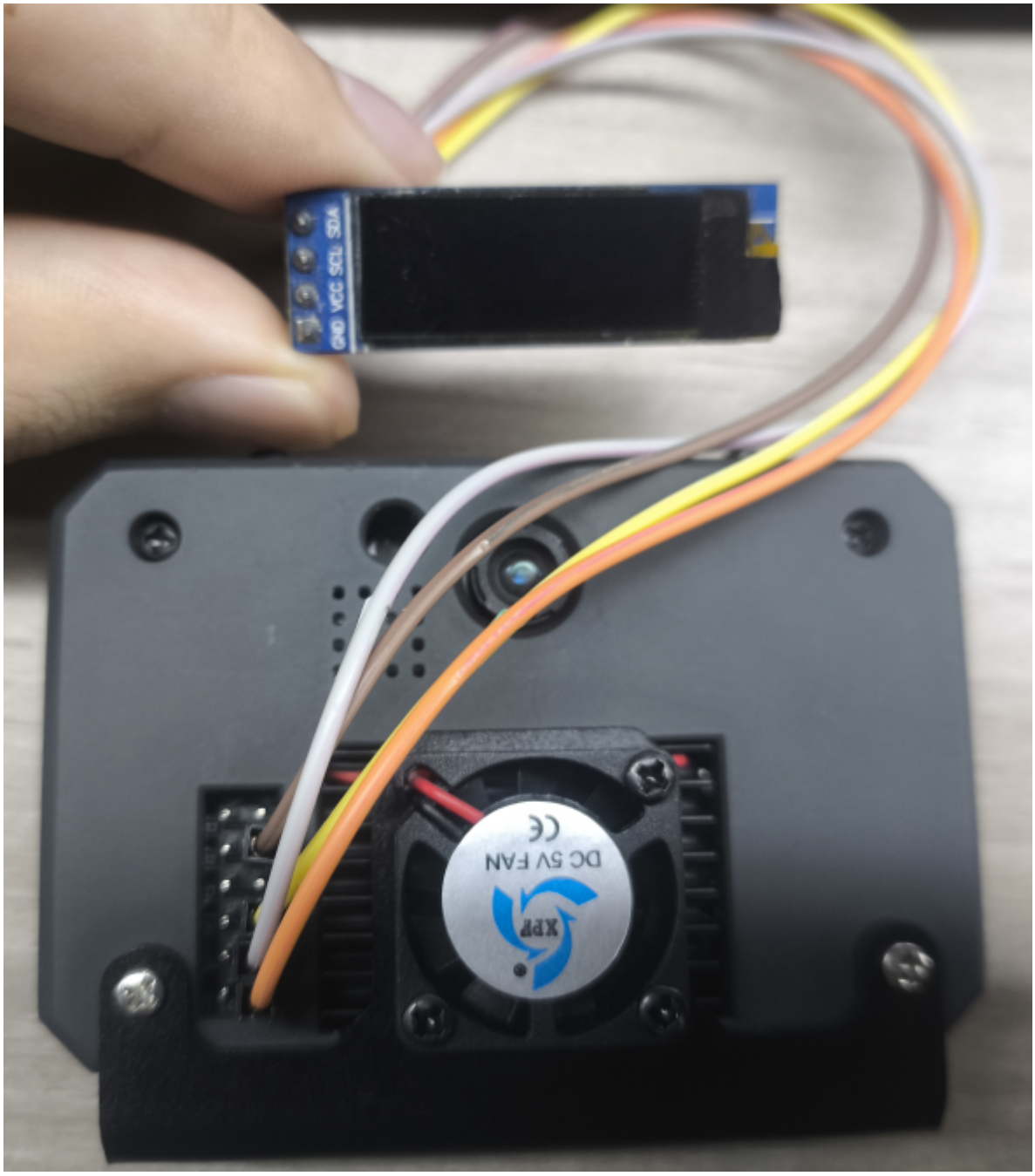
> Like UART, which we learned in the previous section, IIC (I²C) is also a serial communication protocol. At the end of this tutorial, we will briefly understand the difference between these two communication protocols.

wiring

We wire as shown in the figure:



The actual pictures are as follows:



Driving OLED

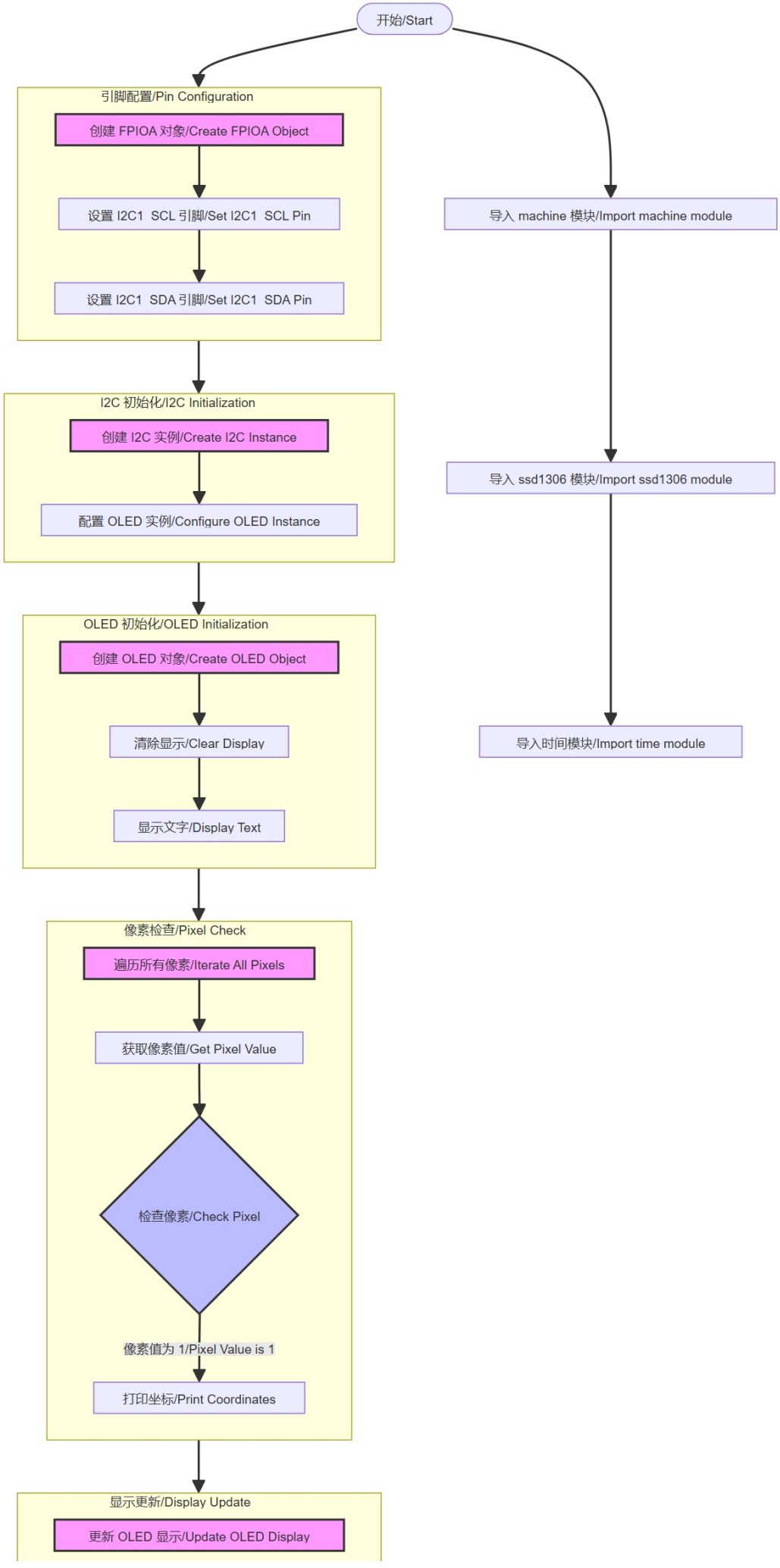
Copy the following code into CanMV IDE and run

```
# 导入 machine 模块中的 I2C 和 FPIOA 类
# (Import the I2C and FPIOA classes from the machine module)
from machine import I2C
from machine import FPIOA

# 导入 ssd1306 模块中的 SSD1306_I2C 类
# (Import the SSD1306_I2C class from the ssd1306 module)
from ybUtils.ssd1306 import SSD1306_I2C

# 导入时间模块
# (Import the time module)
import time
```

```
# 实例化 FPIOA 对象
# (Create an instance of the FPIOA object)
fpioa = FPIOA()

# 创建 I2C 实例，使用 I2C 通道 1
# (Create an I2C instance, using I2C channel 1)
i2c = I2C(1)

# 配置引脚功能:
# - 将 GPIO 引脚 34 映射到 I2C1_SCL 功能
# - 将 GPIO 引脚 35 映射到 I2C1_SDA 功能
# (Configure pin functions:
# - Map GPIO pin 34 to I2C1_SCL function
# - Map GPIO pin 35 to I2C1_SDA function)
fpioa.set_function(34, FPIOA.IIC1_SCL, oe=1, ie=1, pu=1, st=1, ds=15)
fpioa.set_function(35, FPIOA.IIC1_SDA, oe=1, ie=1, pu=1, st=1, ds=15)

# 创建 SSD1306 OLED 显示器实例
# (Create an SSD1306 OLED display instance)
# 设置 OLED 显示器的分辨率为 128x32
# (Set the OLED display resolution to 128x32)
oled = SSD1306_I2C(128, 32, i2c)

# 清除 OLED 显示器的内容
# (Clear the content of the OLED display)
oled.fill(0)

# 在 OLED 显示器上显示文字 "Hello Yahboom"，在坐标 (0, 10) 处
# (Display the text "Hello Yahboom" on the OLED display at coordinates (0, 10))
oled.text('Hello Yahboom', 0, 10)

# 遍历 OLED 显示器的所有像素
# (Iterate through all the pixels of the OLED display)
for x in range(128):
    for y in range(32):
        # 获取指定像素的值
        # (Get the value of the specified pixel)
        a = oled.pixel(x, y)

        # 如果像素值为 1（亮起），则打印坐标
        # (If the pixel value is 1 (lit up), print the coordinates)
        if a == 1:
            print(x, y)

# 更新 OLED 显示器的内容
# (Update the content of the OLED display)
oled.show()
```

You can see the effect shown in the picture on the OLED screen.

The flowchart of code execution is as follows:


开始/Start
引脚配置/Pin Configuration
创建 FPIOA 对象/Create FPIOA Object
设置 I2C1 SCL 引脚/Set I2C1 SCL Pin
设置 I2C1 SDA 引脚/Set I2C1 SDA Pin
I2C 初始化/I2C Initialization
创建 I2C 实例/Create I2C Instance
配置 OLED 实例/Configure OLED Instance
OLED 初始化/OLED Initialization
创建 OLED 对象/Create OLED Object
清除显示/Clear Display
显示文字/Display Text
像素检查/Pixel Check
遍历所有像素/Iterate All Pixels
获取像素值/Get Pixel Value
检查像素/Check Pixel
像素值为 1/Pixel Value is 1
打印坐标/Print Coordinates
显示更新/Display Update
更新 OLED 显示/Update OLED Display
导入 machine 模块/Import machine module
导入 ssd1306 模块/Import ssd1306 module
导入时间模块/Import time module

Expanded content

SSD1306

SSD1306 is a monochrome OLED display controller chip, which is widely used in small display modules.

> Driving SSD1306 to display Chinese requires additional processing, and the SSD1306 driver library in the K230 firmware does not support it.

UART and IIC

IIC (I^2C) is a serial communications bus developed by Philips for connecting low-speed peripherals to microcontrollers.

It only requires two lines: serial data line (SDA) and serial clock line (SCL), and can support multiple master devices and multiple slave devices.

The main differences between IIC and UART:

- IIC uses two wires (SDA and SCL) while UART uses two wires (TX and RX)
- IIC supports multi-device bus, UART is usually point-to-point communication
- IIC has master and slave devices, and the devices at both ends of UART have equal status
- IIC has an address addressing mechanism, but UART does not
- IIC is synchronized by clock, UART uses fixed baud rate

How to choose to use IIC or UART ?

Scenarios for using UART:

- Need simple point-to-point communication
- Long communication distance (up to tens of meters)
- Full-duplex communication is required (sending and receiving data at the same time)
- Does not require many connected devices
- The communication speed requirement is high but not critical (usually within 115200bps)
- For example: communicate with PC, debug output, connect GPS module, Bluetooth module, etc.

Scenarios for using IIC:

- Need to connect multiple devices on a single board (multiple masters and multiple slaves)
- PCB space is limited and IO ports need to be saved
- Short communication distance (usually within a board or between adjacent boards)
- Connect low-speed peripherals such as sensors, EEPROM, real-time clock, etc.
- The communication speed requirement is not high (100kbps for standard mode, 400kbps for fast mode)
- For example: connecting temperature and humidity sensors, accelerometers, OLED displays, etc.