// Node2\_ESP32C6\_Receiver\_Display.ino

#include <Wire.h>

#include <Adafruit\_GFX.h>

#include <Adafruit\_SSD1306.h>

// OLED配置

#define SCREEN\_WIDTH 128

#define SCREEN\_HEIGHT 64

#define OLED\_RESET -1

Adafruit\_SSD1306 display(SCREEN\_WIDTH, SCREEN\_HEIGHT, &Wire, OLED\_RESET);

// ESP32-C6 I2C引脚

#define I2C\_SDA 6

#define I2C\_SCL 7

// Zigbee模块配置 (使用Serial1)

#define ZIGBEE\_RX 4     // ESP32-C6的GPIO4 - 连接Zigbee的TX

#define ZIGBEE\_TX 5     // ESP32-C6的GPIO5 - 连接Zigbee的RX

// 数据变量

float temperature = 0.0;

float humidity = 0.0;

int gasValue = 0;

int gasAlert = 0;

// 状态指示

unsigned long lastReceiveTime = 0;

unsigned long receiveCount = 0;

bool dataReceived = false;

void setup() {

  Serial.begin(115200);

  delay(1000);

  Serial.println("\n=== Node2 ESP32-C6 接收节点启动 ===");

  // 初始化I2C

  Wire.begin(I2C\_SDA, I2C\_SCL);

  Serial.printf("I2C初始化: SDA=GPIO%d, SCL=GPIO%d\n", I2C\_SDA, I2C\_SCL);

  // 初始化Zigbee串口 (ESP32-C6使用Serial1)

  Serial1.begin(9600, SERIAL\_8N1, ZIGBEE\_RX, ZIGBEE\_TX);

  Serial.printf("Zigbee串口初始化: 9600波特率\n");

  Serial.printf("  RX引脚: GPIO%d\n", ZIGBEE\_RX);

  Serial.printf("  TX引脚: GPIO%d\n", ZIGBEE\_TX);

  // 初始化OLED

  Serial.println("初始化OLED...");

  if(!display.begin(SSD1306\_SWITCHCAPVCC, 0x3C)) {

    Serial.println("0x3C失败，尝试0x3D...");

    if(!display.begin(SSD1306\_SWITCHCAPVCC, 0x3D)) {

      Serial.println("!!! OLED初始化失败 !!!");

      Serial.println("请检查接线:");

      Serial.println("  SDA -> GPIO6");

      Serial.println("  SCL -> GPIO7");

      Serial.println("  VCC -> 3.3V");

      Serial.println("  GND -> GND");

    } else {

      Serial.println("✓ OLED初始化成功 (地址0x3D)");

    }

  } else {

    Serial.println("✓ OLED初始化成功 (地址0x3C)");

  }

  // 显示初始画面

  displayWelcome();

  Serial.println("\n等待接收数据...\n");

  delay(2000);

}

void loop() {

  // 检查是否有数据接收

  if (Serial1.available()) {

    String receivedData = Serial1.readStringUntil('\n');

    receivedData.trim();  // 去除首尾空格和换行符

    if (receivedData.length() > 0) {

      receiveCount++;

      lastReceiveTime = millis();

      Serial.printf("[%lu] 接收到数据: %s\n", receiveCount, receivedData.c\_str());

      // 解析数据

      if (parseData(receivedData)) {

        dataReceived = true;

        // 更新显示

        updateDisplay();

        Serial.printf("     解析成功: T=%.1f°C, H=%.1f%%, G=%d, A=%d\n",

                      temperature, humidity, gasValue, gasAlert);

      } else {

        Serial.println("     ⚠ 数据解析失败");

      }

    }

  }

  // 如果超过10秒没收到数据，显示等待状态

  if (dataReceived && (millis() - lastReceiveTime > 10000)) {

    Serial.println("⚠ 超过10秒未收到数据，显示等待状态...");

    displayWaiting();

    dataReceived = false;

  }

  delay(50);  // 短暂延迟避免CPU占用过高

}

// 解析数据包

bool parseData(String data) {

  // 预期格式: T:25.5,H:60.0,G:123,A:0

  int tIndex = data.indexOf("T:");

  int hIndex = data.indexOf("H:");

  int gIndex = data.indexOf("G:");

  int aIndex = data.indexOf("A:");

  // 检查是否所有字段都存在

  if (tIndex == -1 || hIndex == -1 || gIndex == -1 || aIndex == -1) {

    return false;

  }

  // 解析温度

  int tEnd = data.indexOf(",", tIndex);

  if (tEnd == -1) return false;

  temperature = data.substring(tIndex + 2, tEnd).toFloat();

  // 解析湿度

  int hEnd = data.indexOf(",", hIndex);

  if (hEnd == -1) return false;

  humidity = data.substring(hIndex + 2, hEnd).toFloat();

  // 解析气体值

  int gEnd = data.indexOf(",", gIndex);

  if (gEnd == -1) return false;

  gasValue = data.substring(gIndex + 2, gEnd).toInt();

  // 解析警报状态

  gasAlert = data.substring(aIndex + 2).toInt();

  return true;

}

// 显示欢迎界面

void displayWelcome() {

  display.clearDisplay();

  display.setTextSize(2);

  display.setTextColor(SSD1306\_WHITE);

  display.setCursor(10, 20);

  display.println("iCEasy");

  display.setTextSize(1);

  display.setCursor(20, 45);

  display.println("Waiting...");

  display.display();

}

// 显示等待状态

void displayWaiting() {

  display.clearDisplay();

  display.setTextSize(1);

  display.setCursor(0, 0);

  display.println("iCEasy");

  display.drawLine(0, 10, SCREEN\_WIDTH, 10, SSD1306\_WHITE);

  display.setTextSize(1);

  display.setCursor(10, 28);

  display.println("No Signal...");

  display.display();

}

// 更新显示内容

void updateDisplay() {

  display.clearDisplay();

  // 第一行：iCEasy + 接收计数

  display.setTextSize(1);

  display.setCursor(0, 0);

  display.print("iCEasy");

  display.setCursor(80, 0);

  display.print("#");

  display.print(receiveCount);

  // 分隔线

  display.drawLine(0, 10, SCREEN\_WIDTH, 10, SSD1306\_WHITE);

  // 温度

  display.setTextSize(1);

  display.setCursor(0, 15);

  display.print("Temp: ");

  display.print(temperature, 1);

  display.print(" C");

  // 湿度

  display.setCursor(0, 28);

  display.print("Humi: ");

  display.print(humidity, 1);

  display.print(" %");

  // 气体传感器值

  display.setCursor(0, 41);

  display.print("Gas:  ");

  display.print(gasValue);

  // 气体警报状态

  display.setCursor(0, 54);

  if (gasAlert == 1) {

    // 反色显示警报

    display.setTextColor(SSD1306\_BLACK, SSD1306\_WHITE);

    display.print(" ALERT! ");

    display.setTextColor(SSD1306\_WHITE);

  } else {

    display.print("Normal");

  }

  display.display();

}