19.12 深大实验-串口通讯(串口通讯，串口绘图，ADC 检测按键，ADC 热敏电阻测温)

19.12.1. 实验介绍

1、本实验项目主要目的：
- 掌握串口通讯，了解串口绘图工具
- 继续熟悉 ADC 的应用，熟悉 ADC 检测按键，熟悉 ADC 热敏电阻（NTC）测温

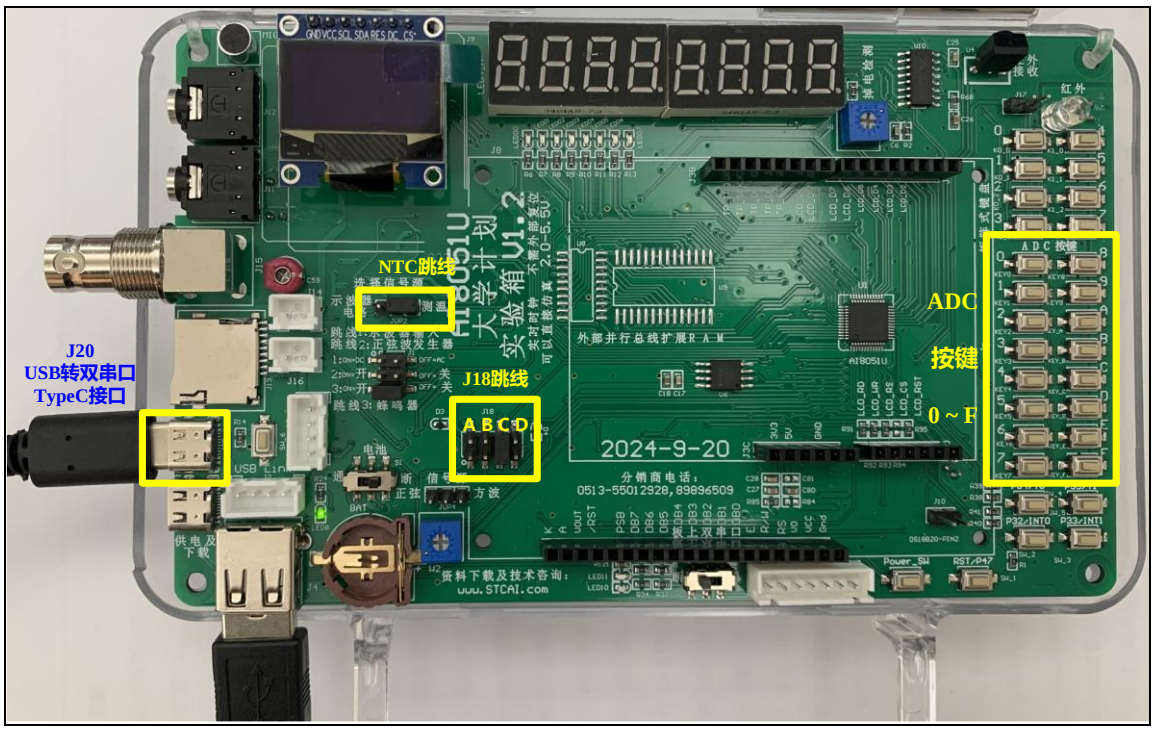
2、掌握 Ai8051U 实验箱原理图中的串口通信部分电路，ADC 应用部分电路

3、继续熟悉 AIapp-ISP 系统软件中串口助手工具，了解串口绘图工具

4、继续熟悉如何管理多文件项目

a) 认识实验箱：

Ai8051U 的实验箱正面图：



1、J20：　USB 转双串口 U2（Ai8H2K12U）TypeC 接口。本实验需要用 TypeA（连接电脑）-TypeC（连接实验箱 J20）线相连。（详见 J20-U2 原理图）

2、J18 跳线：本实验需要将实验箱上“J18 跳线 C”的跳线连上。
- J18.C：使主控芯片 UART1 的 P3.7/TxD_2 与双串口 U2（Ai8H2K12U）联通。详见 J18 跳线原理图

3、NTC 跳线：Negative Temperature Coefficient thermistor，即“负温度系数热敏电阻”，其电阻值随温度升高而减小，ADC 测温通过 ADC3 通道采集 NTC 温敏电阻的电压值来计算出温度值。详见 NTC 原理图所示，NTC 温敏电阻与 10K 精密电阻 R165 串联分压后的电压输入到 ADC3 脚位，而温敏电阻随温度变化，阻值会产生变化，从而导致分压电压跟着改变，于是通过 ADC 采集分压电压值进行 ADC 转换，将转换的 ADC 结果通过 UART1 串口发送到“串口

绘图”显示实时采样数据。

4、ADC 按键：按下不同的 ADC 按键可以改变 P1.0-ADC-KEY 脚的电压，ADC 通过采集 P1.0-ADC-KEY 脚的电压进行 ADC 转换，然后将转换的 ADC 结果通过 UART1 串口发送到“串口绘图”显示实时采样数据。

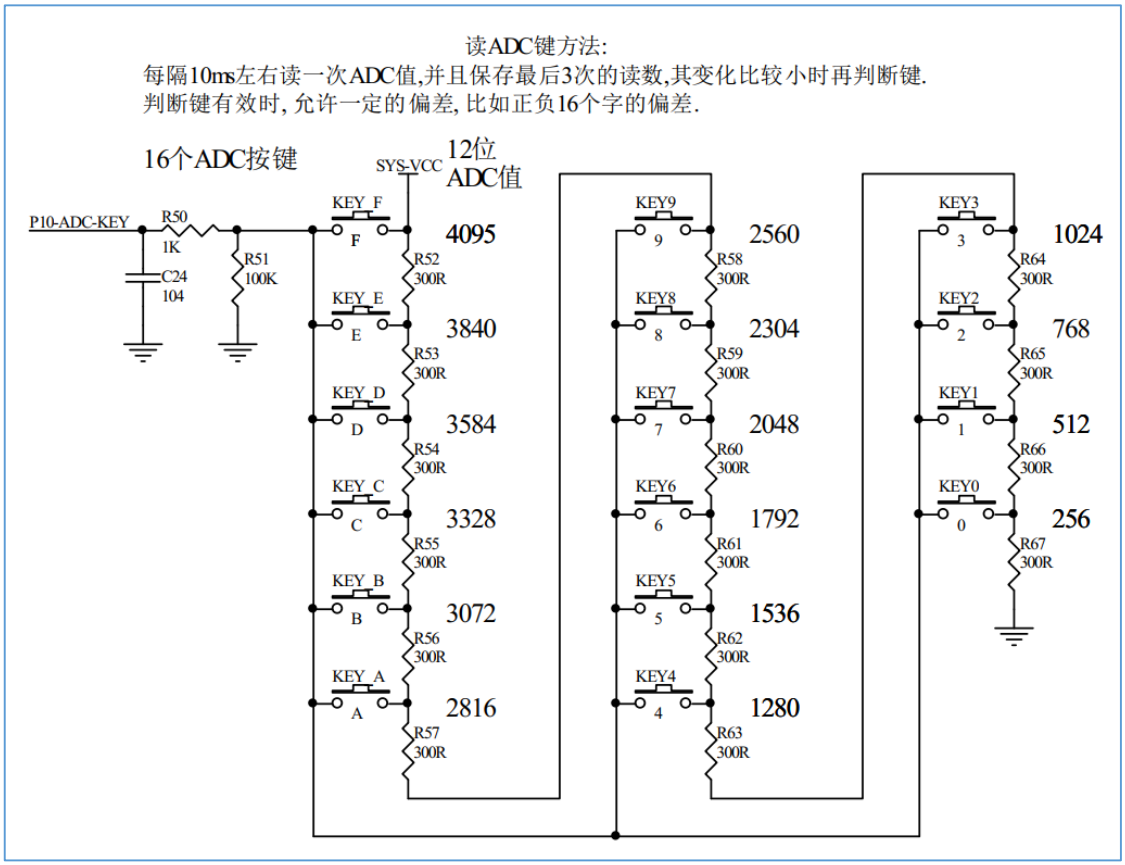
b) Keil 环境下多文件项目管理说明

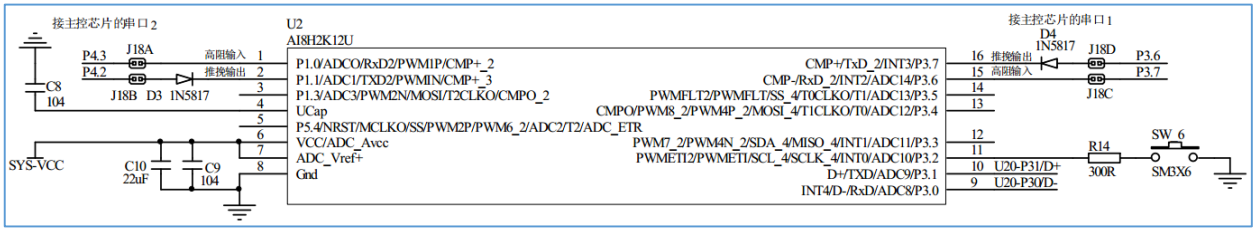
当项目的功能比较复杂时，就需要在 Keil 中建立多文件项目，以方便分工合作、代码复用、模块化管理、增强可读性和可维护性。

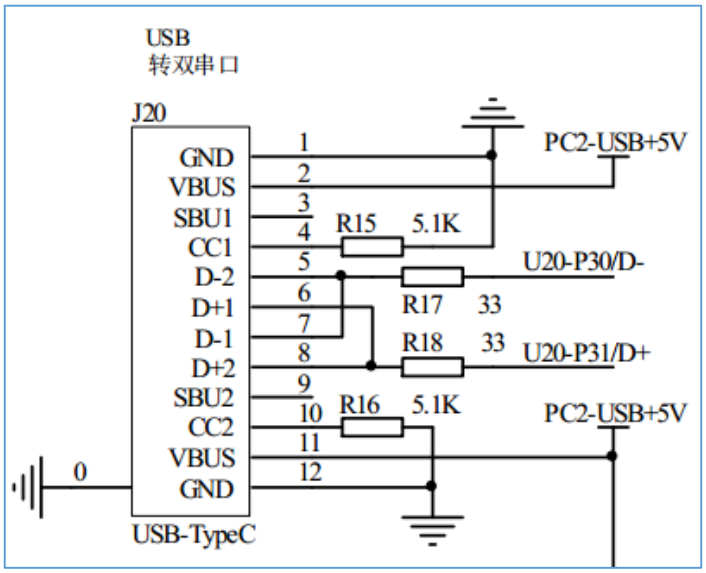
- 比较好的建议是将项目功能模块化，不同模块的实现代码放在不同的.c 文件中。
- 一般建议是一个模块对应一个.c 程序文件和一个.h 头文件
- 模块的初始化函数以及相关的数据处理函数都在.c 文件中实现
- 与模块相关的全局变量也必须在.c 文件中进行定义，一定不能在.h 文件中定义变量
- 如果有其他模块需要使用本模块定义的变量或函数，则这些函数和变量都需要在.h 文件中声明。
- 特别提醒：在.h 文件中声明外部变量必须使用 **extern** 关键字，否则就变成变量定义了，这样会出现变量重复定义的错误
- 为防止头文件被多次包含而产生错误或者警告，在头文件中使用类似如下的条件编译组合语句，可避免在同一个.c 文件中对同一个.h 头文件进行多次包含：

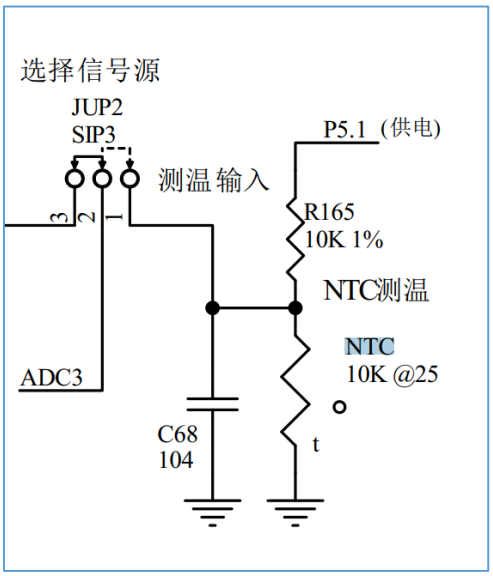
```
#ifndef XXXX
#define XXXX
...
#endif
```

19.12.2. 原理图









19.12.3. 实验程序代码

a) main.c -- 程序主函数

```
//main.c
//程序主函数
//短接 J18C 跳线
//烧录程序后，使用 USB Type-C 线连接电脑与实验箱 J20 接口
//AIapp-ISP 软件"CDC/HID-串口助手"页面，串口选择 CDC1 对应的 COM 口(先不打开串口)
//AIapp-ISP 软件"调试仿真接口" → "接口设置" 选择将所有调试接口绑定到 USB-CDC/串口助手
//AIapp-ISP 软件"调试仿真接口" → "串口绘图" 先点"设置"按钮，数据格式选择"双字节(高位在前)"
//点击"串口绘图"界面的"打开端口"按钮，开始进行串口绘图
#include "config.h"        //头文件中已包含 ai8051u.h 以及其他头文件

void main()
{
    u16 Res;

    SYS_Init();          //系统初始化

    while (1)
    {
        if (f100ms)     //判断 100ms 标志位
        {
```

```
            f100ms = 0;                     //清除 100ms 标志，并处理 100ms 事件

            Res = Get_ADC12bitResult(0);    //检测 ADC 按键值
            Res = Get_ADC12bitResult(0);    //检测 ADC 按键值
            Res = Get_ADC12bitResult(0);    //检测 ADC 按键值
            Uart1_Send((u8)(Res>>8));       //发送采样结果高 8 位数据
            Uart1_Send((u8)Res);            //发送采样结果低 8 位数据

            Res = Get_ADC12bitResult(3);    //采集 NTC 脚位 AD 值
            Res = Get_ADC12bitResult(3);    //采集 NTC 脚位 AD 值
            Res = Get_ADC12bitResult(3);    //采集 NTC 脚位 AD 值
            Uart1_Send((u8)(Res>>8));       //发送采样结果高 8 位数据
            Uart1_Send((u8)Res);            //发送采样结果低 8 位数据
        }
    }
}

void SYS_Init(void)     //系统初始化函数
{
    EAXFR = 1;          //使能访问扩展 XFR
    WTST = 0x00;        //设置最快速度访问程序代码
    CKCON = 0x00;       //设置最快速度访问内部 XDATA

    P0M0 = 0x00; P0M1 = 0x00;       //初始化 P0 口为准双向口模式
    P1M0 = 0x00; P1M1 = 0x09;       //初始化 P1 口为准双向口模式, P1.0,P1.3 高阻输入
    P2M0 = 0x00; P2M1 = 0x00;       //初始化 P2 口为准双向口模式
    P3M0 = 0x00; P3M1 = 0x00;       //初始化 P3 口为准双向口模式
    P4M0 = 0x00; P4M1 = 0x00;       //初始化 P4 口为准双向口模式
    P5M0 = 0x02; P5M1 = 0x00;       //初始化 P5 口为准双向口模式, P5.1 推挽输出
    P6M0 = 0x00; P6M1 = 0x00;       //初始化 P6 口为准双向口模式
    P7M0 = 0x00; P7M1 = 0x00;       //初始化 P7 口为准双向口模式

    S1_S1 = 0;          //UART1 通道选择, 00: P3.0 P3.1, 01: P3.6 P3.7, 10: P1.6 P1.7, 11: P4.3 P4.4
    S1_S0 = 1;          //UART1 选择  P3.6 P3.7  作为串口收发通道

    Timer0_Init();      //定时器 0 初始化,用于产生 100ms 的时间基准
    Uart1_Init();       //UART1 接口初始化
    ADC_Init();         //ADC 初始化

    P51 = 1;            //给 NTC 供电

    EA = 1;             //使能全局中断
}
```

b) uart.c -- 初始化 UART 接口

```
//uart.c
//初始化 UART 接口

#include "config.h"       //头文件中已包含 ai8051u.h 以及其他头文件

void Uart1_Init(void)          //115200bps@40.000MHz
{
    SCON = 0x50;               //8 位数据,可变波特率
    AUXR |= 0x01;              //串口 1 选择定时器 2 为波特率发生器
    AUXR |= 0x04;              //定时器时钟 1T 模式
    T2L = 0xA9;                //设置定时初始值
    T2H = 0xFF;                //设置定时初始值
    AUXR |= 0x10;              //定时器 2 开始计时
}

void Uart1_Send(u8 dat)       //通过串口 1 发送数据
{
    SBUF = dat;                //写入需要发送的数据
    while(!TI);                //判断发送是否完成
    TI = 0;                    //清除发送完成标志
}
```

c) adc.c -- ADC 初始化和 ADC 数据采集

```
//adc.c
//ADC 程序
//包括 ADC 初始化和 ADC 数据采集

#include "config.h"       //头文件中已包含 ai8051u.h, ai_usb.h 以及其他头文件

void ADC_Init()           //ADC 初始化函数
{
    ADCTIM = 0x3f;              //设置 ADC 内部时序，ADC 采样时间建议设最大值
    ADCCFG = 0x2f;              //设置 ADC 时钟为系统时钟/2/16
    ADC_CONTR = 0x80;           //使能 ADC 模块
}

u16 Get_ADC12bitResult(u8 channel)        //channel = 0~15
{
    ADC_RES = 0;
    ADC_RESL = 0;

    ADC_CONTR = (ADC_CONTR & 0xf0) | channel;    //设置 ADC 转换通道
```

```
    ADC_START = 1;              //启动 ADC 转换
    _nop_();
    _nop_();
    _nop_();
    _nop_();

    while(ADC_FLAG == 0);     //wait for ADC finish
    ADC_FLAG = 0;               //清除 ADC 结束标志
    return    (((u16)ADC_RES << 8) | ADC_RESL);
}
```

d) timer.c -- 定时器初始化以及定时器中断服务程序

```
//timer.c
//定时器初始化以及定时器中断服务程序
//定时器 0:100ms 定时,产生 100ms 的标志位

#include "config.h"       //头文件中已包含 ai8051u.h 以及其他头文件

bit f100ms;               //100ms 标志位

void Timer0_Init(void)  //100 毫秒@40.000MHz
{
    TM0PS = 0x3D;
        //设置定时器时钟预分频 ( 注意:并非所有系列都有此寄存器,详情请查看数据手册 )
    AUXR |= 0x80;     //定时器 0 时钟 1T 模式
    TMOD &= 0xF0;   //设置定时器 0 为模式 0
    TL0 = 0xFC;       //设置定时器 0 初始值
    TH0 = 0x03;       //设置定时器 0 初始值
    TF0 = 0;          //清除 TF0 标志
    TR0 = 1;          //定时器 0 开始计时
    ET0 = 1;          //使能定时器 0 中断

    f100ms = 0;       //初始化 100ms 标志位
}

void Timer0_Isr(void) interrupt 1       //定时器 0 中断服务程序
{
    f100ms = 1;       //设置 100ms 标志位
}
```

e) config.h -- 项目配置的头文件

```
#ifndef __CONFIG_H__          //防止头文件被重复包含
#define __CONFIG_H__
```

```
#define HIRC            40000000UL
#define FOSC            40000000UL
#define SYSCLK          FOSC
#define MAIN_Fosc       FOSC

#include <ai8051u.h>            //包含外部头文件
#include <stdio.h>
#include <intrins.h>

#include "def.h"                //包含项目头文件
#include "uart.h"
#include "adc.h"
#include "timer.h"

void SYS_Init(void);            //函数声明

#endif
```

f) uart.h -- 项目配置文件的头文件

```
#ifndef __UART_H__          //防止头文件被重复包含
#define __UART_H__

void    Uart1_Init();           //函数声明
void    Uart1_Send(u8 dat);     //函数声明

#endif
```

g) adc.h -- 项目配置文件的头文件

```
#ifndef __ADC_H__           //防止头文件被重复包含
#define __ADC_H__

void ADC_Init();            //函数声明
u16 Get_ADC12bitResult(u8 channel);     //函数声明

#endif
```

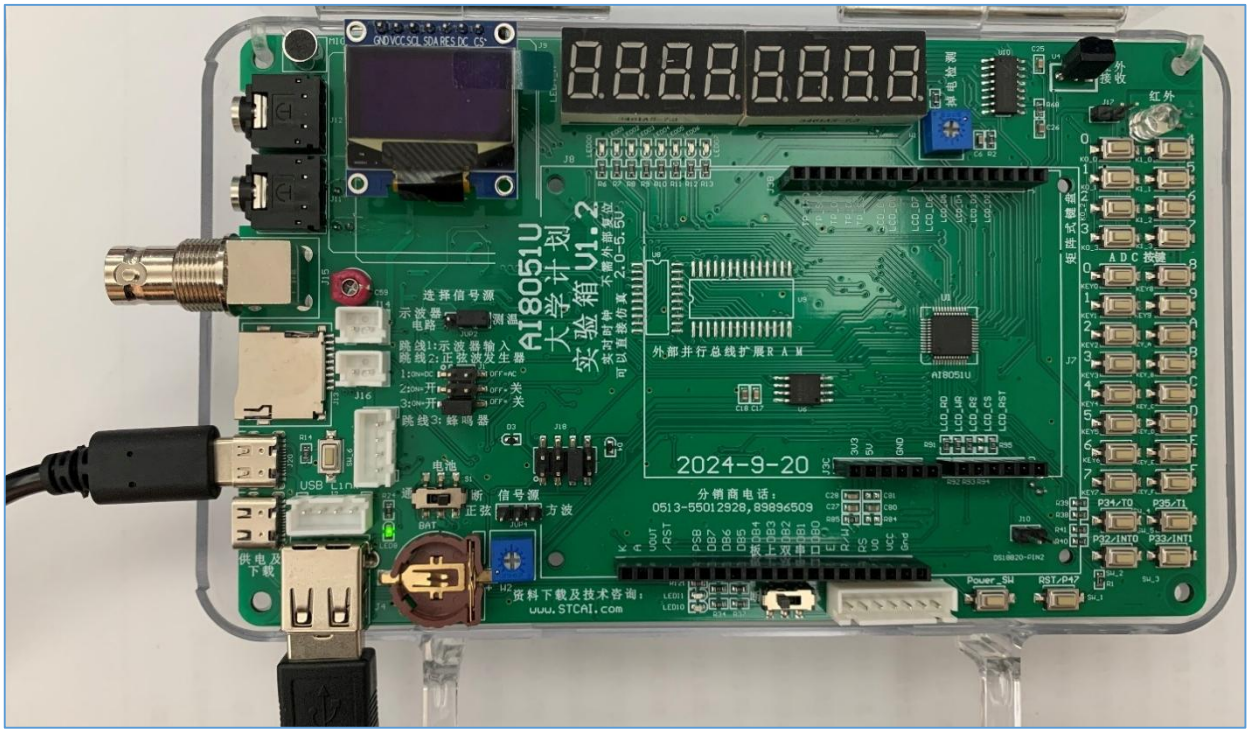
h) timer.h -- 项目配置文件的头文件

```
#ifndef __TIMER_H__         //防止头文件被重复包含
#define __TIMER_H__

void Timer0_Init(void);     //函数声明
```

```
extern bit f100ms;                //外部变量声明

#endif
```

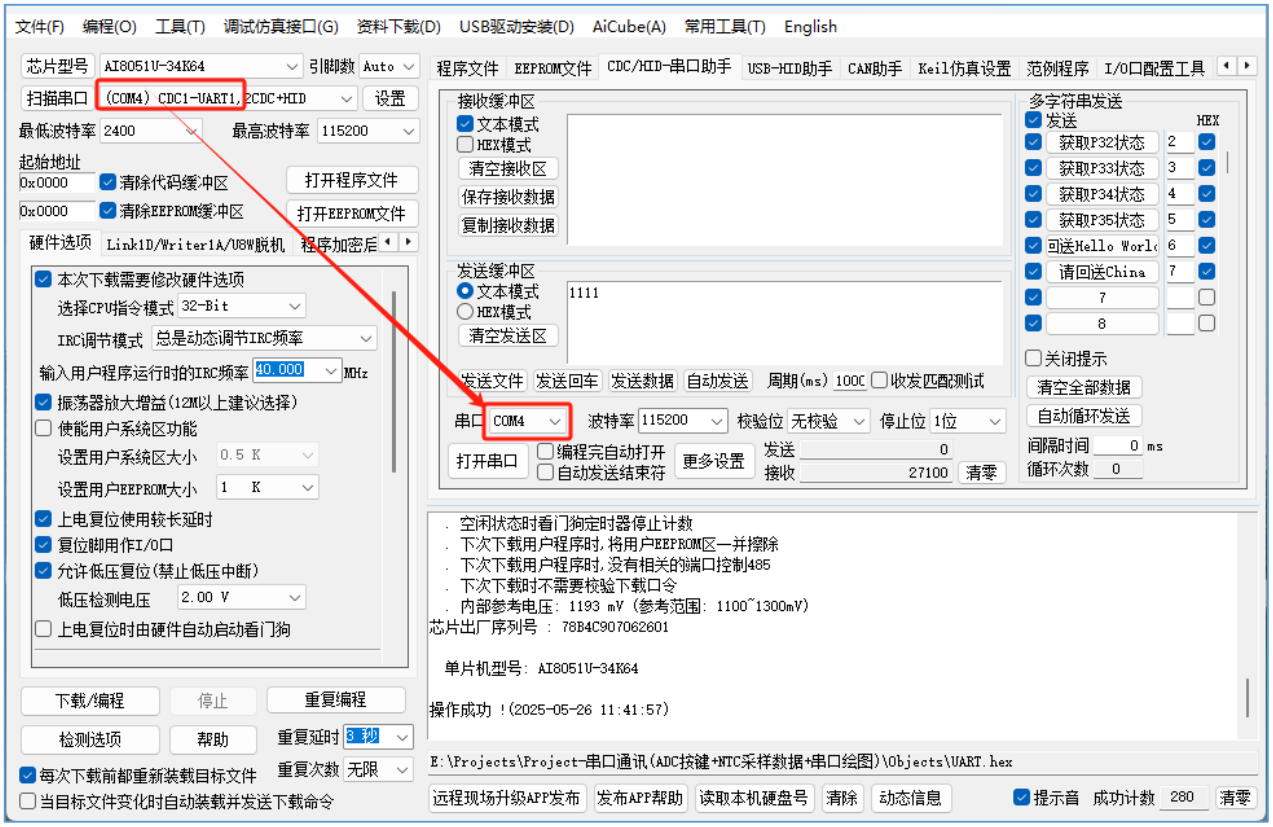
19.12.4. 程序下载

录入代码，保存，编译。按实验要求连接并设置好 Ai8051U 实验箱，如下图：

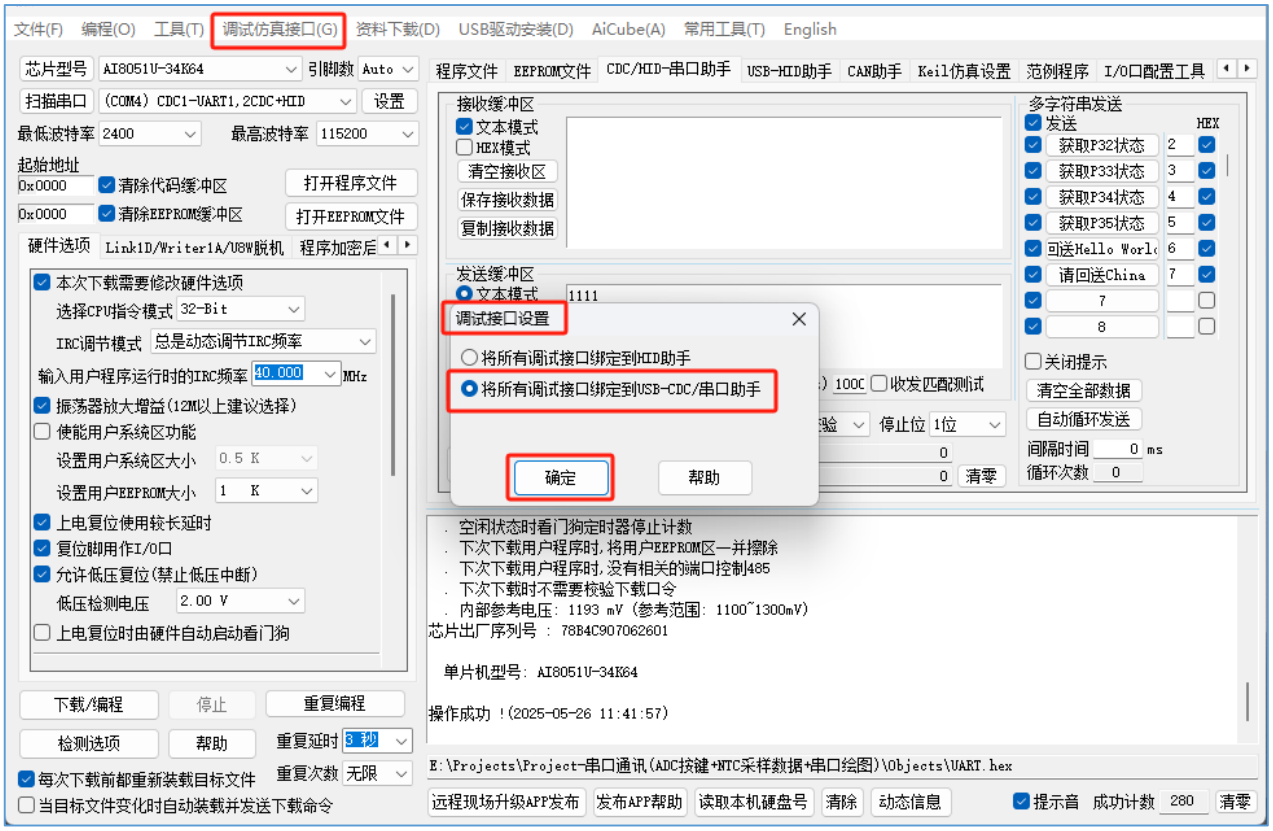


运行 AIapp-ISP 软件系统，按步骤打开并下载程序“uart.hex”，如下图：

在 AIapp-ISP 软件系统界面，点击“CDC/HID-串口助手”选项卡，串口选择 CDC1 对应的 COM 口，如下图：



在 AIapp-ISP 软件系统的菜单栏，点击“调试仿真接口”→点击“接口设置”，选择“将所有调试接口绑定到 USB-CDC/串口助手”，如下图：



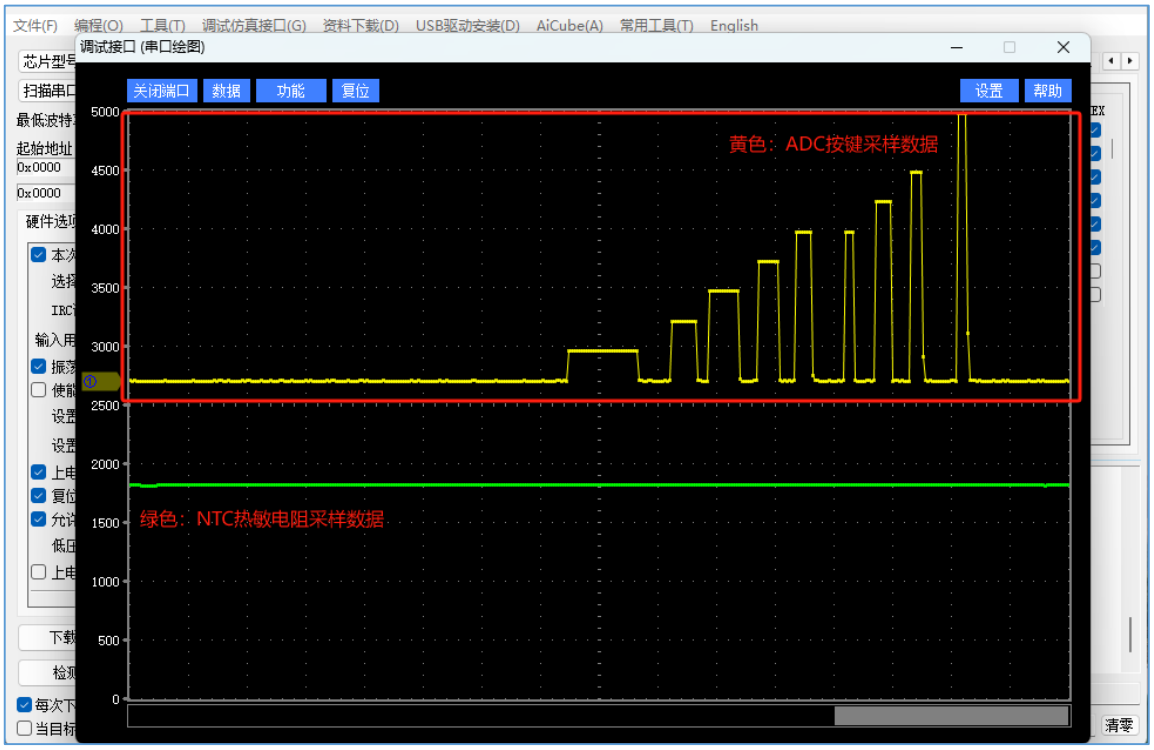
- 在 AIapp-ISP 软件系统的菜单栏，点击“调试仿真接口”→点击“串口绘图”;
- 在“串口绘图”浮窗，点击“设置”按钮;
- 在“设置”串口，数据格式：选择“双字节(高位在前)”。如下图：

一切准备就绪！

19.12.5. 验证实验效果

在“串口绘图”界面，点击“打开端口”按钮，开始进行串口绘图。

随意按 Ai8051U 实验箱上的“0～F”按键，串口绘图曲线如下图：



完成实验目标。