2025 年 CIMC“西门子杯”中国智能制造挑战赛

智能制造工程设计与应用类赛项：工业嵌入式系统开发方向（筹）

全国总决赛 测评流程说明

一、 测评流程说明

各参赛队根据抽签结果和赛程安排，在规定时间内到场检录，检录完成后上交随身携带的所有物品。检录完成后，各参赛队员根据指引，领取封存提交的作品，在边裁的监督下开箱，取出作品。各参赛队有 15 分钟的时间恢复电路，等候测评。测评开始前，各参赛队成员根据指引，到达指定赛位，将参赛作品与测评环境进行连接，连接完成后示意边裁开始测评。测评完毕后请参赛队全体成员在测评表上签字，确认后即可携带作品离场。

现场要求及详细说明请参考《全国总决赛竞赛细则》。

二、 测评步骤

以下为部分测评步骤，用于参赛队赛前进行参考和练习，具体步骤请以决赛现场要求为准。

1、 上位机下发读取设备 ID 号，MCU 返回
   【下发】command:get_device_id
   【上报】report:device_id=0x0001
2、 上位机下发读取 RTC 时间命令，MCU 返回
   【下发】command:get_RTC
   【上报】report:currentTime=2025-01-01 12:00:00
3、 上位机下发修改 RTC 时间命令，MCU 返回
   【下发】command:set_RTC=2025-01-01 12:00:00
   【上报】report:ok
4、 上位机下发读取 RTC 时间命令，MCU 返回
   【下发】command:get_RTC
   【上报】report:currentTime=2025-01-01 12:00:00
5、 上位机下发变比，MCU 返回，修改完成后直接保存到 config.ini 中
   【下发】command:set_ratio:ch0=xx.xx,ch1=xx.xx,ch2=xx.xx
   【上报】report:ok
6、 上位机读取变比，MCU 返回
   【下发】command:get_ratio
   【上报】report:ch0ratio =xx.xx,ch1ratio =xx.xx,ch2ratio =xx.xx
7、 上位机下发单次采集，MCU 返回（乘以变比，保留两位小数）
   【下发】command:get_data
   【上报】report:ch0=xx.xx,ch1=xx.xx,ch2=xx.xx

8、 上位机下发连续采集，MCU 返回

【下发】command:start_sample

【上报】report:2025-01-01 12:00:00 ch0=xx.xx,ch1=xx.xx,ch2=xx.xx

9、 上位机下发停止采集，MCU 返回

【下发】command:stop_sample

【上报】report:ok

10、 上位机下发阈值，MCU 返回，修改完成后直接保存到 config.ini 中

【下发】command:set_limit:ch0=xx.xx,ch1=xx.xx,ch2=xx.xx

【上报】report:ok

11、 上位机读取阈值，MCU 返回

【下发】command:get_limit

【上报】report:ch0limit=xx.xx,ch1limit=xx.xx,ch2limit=xx.xx

12、 上位机下发单次采集，MCU 返回（超阈值的在通道后面加*）

【下发】command:get_data

【上报】report:ch0*=xx.xx,ch1=xx.xx,ch2=xx.xx

13、 上位机下发读取设备 ID 号，MCU 返回

【下发】command:FFFF0200080163FA

报文解析：设备 ID（2 字节）：0xFFFF 代表广播发送
系统报文类型（1 字节）：0x02 代表获取设备 ID
报文长度（2 字节）：0x0008 代表报文长度 8 字节
协议版本（1 字节）：0x01 代表报文协议版本为 1
CRC 校验（2 字节）：0x63FA （按小端序传输，即高位在左低位在右）

【上报】report:000102000A010001F1C2

报文解析：设备 ID（2 字节）：0x0001 代表设备 ID 为 0001
系统报文类型（1 字节）：0x02 代表应答
报文长度（2 字节）：0x000A 代表报文长度 10 字节
协议版本（1 字节）：0x01 代表报文协议版本为 1
报文内容（2 字节）：0x0001 此处为获取设备 ID，所以报文内容为设备 ID
CRC 校验（2 字节）：0xF1C2 （按小端序传输，即高位在左低位在右）

14、 上位机下发修改设备 ID 号，MCU 返回确认指令

【下发】command:000101000A010002C382

报文解析：设备 ID（2 字节）：0x0001 代表接收的目标设备 ID 为 0001
系统报文类型（1 字节）：0x01 代表修改设备 ID
报文长度（2 字节）：0x000A 代表报文长度 10 字节
协议版本（1 字节）：0x01 代表报文协议版本为 1
报文内容（2 字节）：0x0002 代表把设备 ID 修改为 0002
CRC 校验（2 字节）：0xC382 （按小端序传输，即高位在左低位在右）

【上报】report:000202000A018000F151

报文解析：设备 ID（2 字节）：0x0002 代表设备 ID 为 0002
（这里要注意，修改成功之后是 0002，修改不成功应当是 0001）
系统报文类型（1 字节）：0x02 代表应答
报文长度（2 字节）：0x000A 代表报文长度 10 字节
协议版本（1 字节）：0x01 代表报文协议版本为 1
报文内容（2 字节）：0x8000 代表返回应答数据，操作成功
0x7000~0x7FFF 代表返回应答数据，操作不成功
CRC 校验（2 字节）：0xF151 （按小端序传输，即高位在左低位在右）

15、 上位机下发单次采集，MCU 返回

【下发】command:000221000801E7B5

报文解析：设备 ID（2 字节）：0x0002 代表接收的目标设备 ID 为 0002

系统报文类型（1 字节）：0x21 单次读取通道数据

报文长度（2 字节）：0x0008 代表报文长度 8 字节

协议版本（1 字节）：0x01 代表报文协议版本为 1

CRC 校验（2 字节）：0xE7B5 （按小端序传输，即高位在左低位在右）

【上报】report:00020100180168904 81E4060A3D74164CCCD46959C00DF4F

报文解析：设备 ID（2 字节）：0x0002 代表设备 ID 为 0002

系统报文类型（1 字节）：0x01 代表数据

报文长度（2 字节）：0x0018 代表报文长度 24 字节

协议版本（1 字节）：01 代表报文协议版本为 1

报文内容（16 字节）：时间戳（4 字节） UNIX 时间戳（秒）

通道 0 数据（4 字节） IEEE 754 浮点数格式

通道 1 数据（4 字节） IEEE 754 浮点数格式

通道 2 数据（4 字节） IEEE 754 浮点数格式

CRC 校验（2 字节）：DF4F （按小端序传输，即高位在左低位在右）

该条示例报文为：6890481E **2025 13:41:50 （GMT+0800）**

4060A3D7 **通道 0 = 3.51 V**

4164CCCD **通道 1 = 14.30 mA**

46959C00 **通道 2 = 19150 Ω**

16、 上位机下发连续采集，MCU 返回

【下发】command:000222000801A3B5

报文解析：设备 ID（2 字节）：0x0002 代表接收的目标设备 ID 为 0002

系统报文类型（1 字节）：0x22 连续读取通道数据

报文长度（2 字节）：0x0008 代表报文长度 8 字节

协议版本（1 字节）：0x01 代表报文协议版本为 1

CRC 校验（2 字节）：0xA3B5 （按小端序传输，即高位在左低位在右）

【上报】report:00020100180168904 81E4060A3D74164CCCD46959C00DF4F

报文解析：设备 ID（2 字节）：0x0002 代表设备 ID 为 0002

系统报文类型（1 字节）：0x01 代表数据

报文长度（2 字节）：0x0018 代表报文长度 24 字节

协议版本（1 字节）：01 代表报文协议版本为 1

报文内容（16 字节）：时间戳（4 字节） UNIX 时间戳（秒）

通道 0 数据（4 字节） IEEE 754 浮点数格式

通道 1 数据（4 字节） IEEE 754 浮点数格式

通道 2 数据（4 字节） IEEE 754 浮点数格式

CRC 校验（2 字节）：DF4F （按小端序传输，即高位在左低位在右）

17、 上位机下发停止采集，MCU 返回

【下发】command:00022F0008010FB7

报文解析：设备 ID（2 字节）：0x0002 代表接收的目标设备 ID 为 0002

系统报文类型（1 字节）：0x2F 停止连续读取通道数据

报文长度（2 字节）：0x0008 代表报文长度 8 字节

协议版本（1 字节）：0x01 代表报文协议版本为 1

CRC 校验（2 字节）：0x0FB7 （按小端序传输，即高位在左低位在右）

【上报】report:000202000A018000F151

报文解析：设备 ID（2 字节）：0x0002 代表设备 ID 为 0002

系统报文类型（1 字节）：0x02 代表应答

报文长度（2 字节）：0x000A 代表报文长度 10 字节

协议版本（1 字节）：0x01 代表报文协议版本为 1

报文内容（2 字节）：0x8000 代表返回应答数据，操作成功

CRC 校验（2 字节）：0xF151 （按小端序传输，即高位在左低位在右）

18、 通过按下不同的按键修改 OLED 显示功能。

按键 1：显示 Ch0 原始数据

按键 2：显示 Ch1 原始数据

按键 3：显示 Ch2 原始数据

按键 4：显示 Ch0 变比后数据

按键 5：显示 Ch1 变比后数据

按键 6：显示 Ch2 变比后数据

19、 检查 TF 卡中 config.ini 存储的数据，是否保存了修改后的变比和阈值

三、 IEEE 754 浮点数格式

IEEE 754 是表示浮点数的国际标准，在本次比赛报文的设计中要求使用单精度 32 位浮点数格式来表示测量值。

32 位单精度浮点数由三部分组成：

符号位(S)：1 位（最高位）

指数位(E)：8 位（中间）

尾数位(M)：23 位（最低位）

结构表示：[S] [EEEEEEEE] [MMMMMMMMMMMMMMMMMMMMMMM]

浮点数值计算公式为 $(-1)^S \times (1.M) \times 2^{E-127}$

其中：

S：0 表示正数，1 表示负数

M：小数部分的二进制表示

E：指数字段的无符号整数值

127：单精度浮点数的偏移值

以 40533333 为例，将其转换为二进制为 0 10000000 10100110011001100110011，其中符号位 S=0（正数），指数 E=10000000(二进制)=128，尾数 M=10100110011001100110011。

代入公式 $(-1)^S \times (1.M) \times 2^{E-127}$ 计算得到结果 = 1.10100110011001100110011×2^1 ≈ 3.3V。

在 C 语言中可以按如下方式进行转换：

```c
#include "stdint.h"
#include "stdio.h"

// 使用联合体实现浮点数与 IEEE 754 格式的转换
typedef union {
    float f;
    uint32_t u;
} Float_IEEE754;

// 浮点数转换为 IEEE 754 格式
uint32_t floatToIEEE754(float value) {
    Float_IEEE754 converter;
    converter.f = value;
    return converter.u;
}


int main(void) {
    float testValue = 100.0f;

    // 转换为 IEEE 754 格式
    uint32_t ieee = floatToIEEE754(testValue);

    printf("0x%08x\r\n",ieee);
}
```

四、 CRC 校验

循环冗余校验（CRC 校验）是一种常用的错误检测算法，广泛应用于数据通信和存储领域。核心思想是通过对数据进行多项式除法运算，生成一个固定长度的校验值，将其附加到原始数据中传输。接收端再次执行相同的算法，通过对接收到的数据进行校验值计算，可以检测出数据在传输过程中是否发生了错误。CRC 校验具有多种方式，在本设计工作中双端采用 CRC-16/MODBUS 的参数模型进行计算，在实际程序设计过程中可以通过查表法可以简化计算，即通过预先计算的结果简化 CRC 的计算过程，进而提高效率。以下为在 C 语言中的计算方法示例。

```c
#include "stdint.h"
#include "stdio.h"

```

```c
// CRC16 MODBUS 查表法表项
const uint16_t crc16_table[256] = {
    0x0000, 0xC0C1, 0xC181, 0x0140, 0xC301, 0x03C0, 0x0280, 0xC241,
    0xC601, 0x06C0, 0x0780, 0xC741, 0x0500, 0xC5C1, 0xC481, 0x0440,
    0xCC01, 0x0CC0, 0x0D80, 0xCD41, 0x0F00, 0xCFC1, 0xCE81, 0x0E40,
    0x0A00, 0xCAC1, 0xCB81, 0x0B40, 0xC901, 0x09C0, 0x0880, 0xC841,
    0xD801, 0x18C0, 0x1980, 0xD941, 0x1B00, 0xDBC1, 0xDA81, 0x1A40,
    0x1E00, 0xDEC1, 0xDF81, 0x1F40, 0xDD01, 0x1DC0, 0x1C80, 0xDC41,
    0x1400, 0xD4C1, 0xD581, 0x1540, 0xD701, 0x17C0, 0x1680, 0xD641,
    0xD201, 0x12C0, 0x1380, 0xD341, 0x1100, 0xD1C1, 0xD081, 0x1040,
    0xF001, 0x30C0, 0x3180, 0xF141, 0x3300, 0xF3C1, 0xF281, 0x3240,
    0x3600, 0xF6C1, 0xF781, 0x3740, 0xF501, 0x35C0, 0x3480, 0xF441,
    0x3C00, 0xFCC1, 0xFD81, 0x3D40, 0xFF01, 0x3FC0, 0x3E80, 0xFE41,
    0xFA01, 0x3AC0, 0x3B80, 0xFB41, 0x3900, 0xF9C1, 0xF881, 0x3840,
    0x2800, 0xE8C1, 0xE981, 0x2940, 0xEB01, 0x2BC0, 0x2A80, 0xEA41,
    0xEE01, 0x2EC0, 0x2F80, 0xEF41, 0x2D00, 0xEDC1, 0xEC81, 0x2C40,
    0xE401, 0x24C0, 0x2580, 0xE541, 0x2700, 0xE7C1, 0xE681, 0x2640,
    0x2200, 0xE2C1, 0xE381, 0x2340, 0xE101, 0x21C0, 0x2080, 0xE041,
    0xA001, 0x60C0, 0x6180, 0xA141, 0x6300, 0xA3C1, 0xA281, 0x6240,
    0x6600, 0xA6C1, 0xA781, 0x6740, 0xA501, 0x65C0, 0x6480, 0xA441,
    0x6C00, 0xACC1, 0xAD81, 0x6D40, 0xAF01, 0x6FC0, 0x6E80, 0xAE41,
    0xAA01, 0x6AC0, 0x6B80, 0xAB41, 0x6900, 0xA9C1, 0xA881, 0x6840,
    0x7800, 0xB8C1, 0xB981, 0x7940, 0xBB01, 0x7BC0, 0x7A80, 0xBA41,
    0xBE01, 0x7EC0, 0x7F80, 0xBF41, 0x7D00, 0xBDC1, 0xBC81, 0x7C40,
    0xB401, 0x74C0, 0x7580, 0xB541, 0x7700, 0xB7C1, 0xB681, 0x7640,
    0x7200, 0xB2C1, 0xB381, 0x7340, 0xB101, 0x71C0, 0x7080, 0xB041,
    0x5000, 0x90C1, 0x9181, 0x5140, 0x9301, 0x53C0, 0x5280, 0x9241,
    0x9601, 0x56C0, 0x5780, 0x9741, 0x5500, 0x95C1, 0x9481, 0x5440,
    0x9C01, 0x5CC0, 0x5D80, 0x9D41, 0x5F00, 0x9FC1, 0x9E81, 0x5E40,
    0x5A00, 0x9AC1, 0x9B81, 0x5B40, 0x9901, 0x59C0, 0x5880, 0x9841,
    0x8801, 0x48C0, 0x4980, 0x8941, 0x4B00, 0x8BC1, 0x8A81, 0x4A40,
    0x4E00, 0x8EC1, 0x8F81, 0x4F40, 0x8D01, 0x4DC0, 0x4C80, 0x8C41,
    0x4400, 0x84C1, 0x8581, 0x4540, 0x8701, 0x47C0, 0x4680, 0x8641,
    0x8201, 0x42C0, 0x4380, 0x8341, 0x4100, 0x81C1, 0x8081, 0x4040
};

// CRC16 计算函数
uint16_t Calculate_CRC16(uint8_t *data, uint16_t length)
{
    uint16_t crc = 0xFFFF; // MODBUS CRC16 的初始值为0xFFFF

    while(length--)
    {
        crc = (crc >> 8) ^ crc16_table[(crc ^ *data++) & 0xFF];
    }

```

```
        return crc;
    }

    // 报文处理函数
    void ProcessMessage(uint8_t *message, uint16_t length)
    {
        uint16_t crc;

        // 计算CRC16
        crc = Calculate_CRC16(message, length);

        printf("计算得到的 CRC=%x\r\n",crc);

        // 在原始报文后添加CRC16
        message[length] = (uint8_t)(crc >> 8);      // CRC 高字节
        message[length + 1] = (uint8_t)(crc);       // CRC 低字节

        //打印完整报文
        for(int i=0;i<=length+1;i++)
        {
            printf("%02x ",message[i]);
        }
        printf("\r\n");
    }

    // 测试
    int main(void)
    {
        //00 02 01 00 18 01 68 90 48 1E 40 60 A3 D7 41 64 CC CD 46 95 9C 00
        //结果应该是 DF 4F

        uint8_t testMessage[] = {
            0x00, 0x02, 0x01, 0x00, 0x18, 0x01, 0x68, 0x90,
            0x48, 0x1E, 0x40, 0x60, 0xA3, 0xD7, 0x41, 0x64,
            0xCC, 0xCD, 0x46, 0x95, 0x9C, 0x00
        };

        uint16_t messageLength = sizeof(testMessage);
        ProcessMessage(testMessage, messageLength);

    }
```

五、 设计资料提交说明

各参赛队伍需在 8 月 11 日晚 19:00 前提交本参赛队伍的采样板设计资料，包含原理图、PCB

图、以及说明文件（写明设计软件及版本等信息），将上述资料打包命名为【队伍编号.zip】上传，未按要求提交的队伍不参与决赛评奖。

提交地点：http://inbox.weiyun.com/qhTXVYbm

截止时间：2025-08-11 19:00:00

六、 其他

比赛期间根据赛程安排有不同的任务环节，请各参赛队伍及时注意官网及 QQ 群中的通知。

具体竞赛细则要求请参考《全国总决赛竞赛细则》