代码使用说明

代码介绍

（1）T113设备上运行的代码：(运行前提：1.curl及ffmpeg移植，2.将以下的四个可执行文件都放到设备/root目录下)

1.**_bluez_part**：可执行文件名为"**bt_app**"，目前只有一个读服务功能和一个写服务功能，

2.**_edp_part**：可执行文件名为"**edp**"，T113连接onenet的应用部分代码，用于完成onenet与T113设备的联合

3.**_fork_edp_and_bluez**：可执行文件名为"**main**"，创建进程"**bt_app**"和"**edp**"

4.**_photo_upload_part**：可执行文件名为"**photo_upload**"，将设备拍摄的照片上传至onenet平台

（2）手机APP

_uniapp_part：uniapp开发的安卓APP，目前功能有：1.登录界面(账号密码都为1)、2.下拉onenet的api控制设备、3.用户退出界面、

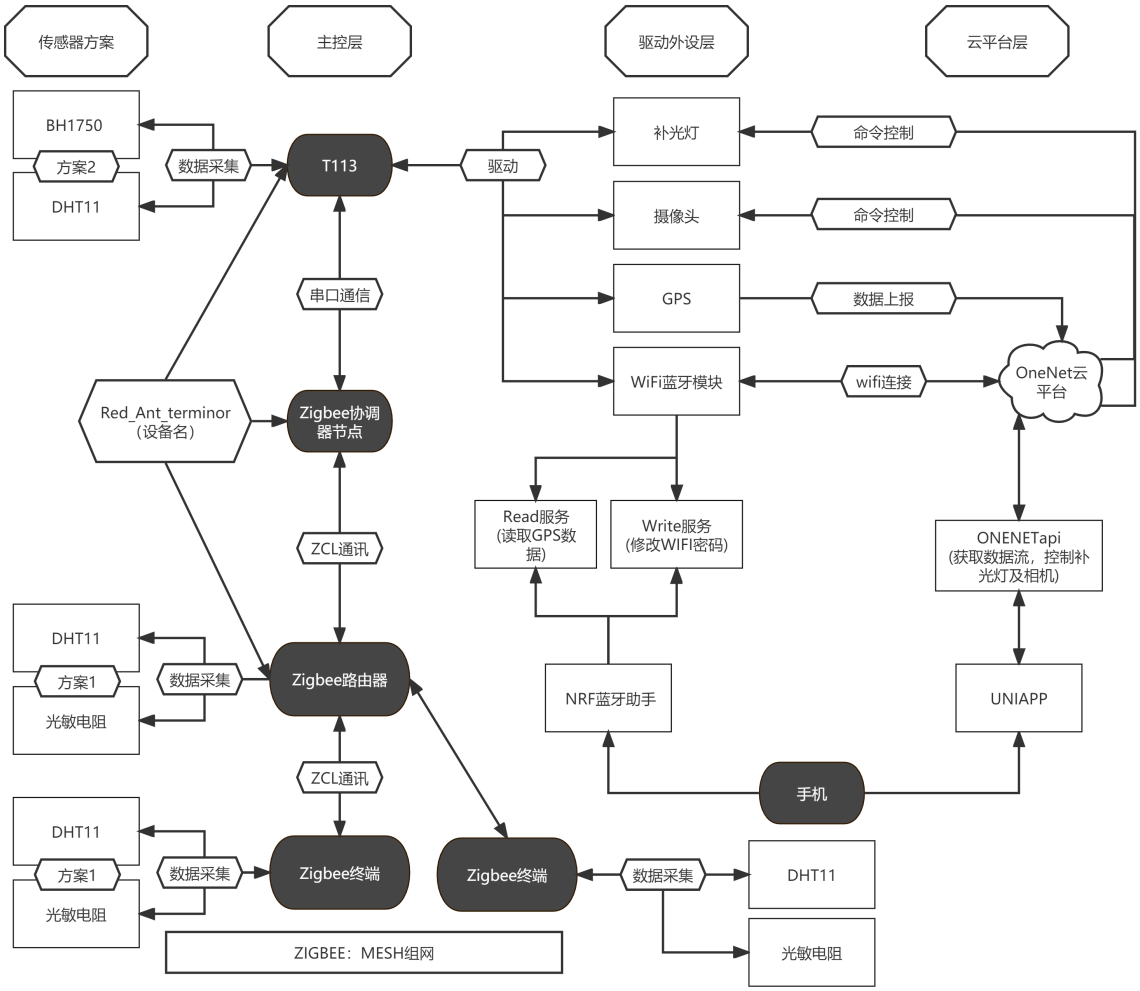
（3）ZIGBEE设备上运行的代码：

_zigbee_part：zigbee代码，这份代码包含协调器，路由器，终端设备的代码，用于路由器或终端设备采集温湿度光照数据发送回协调器，协调器再通过串口发送到T113开发板。具体怎么用得你们自己了解如何烧录代码了

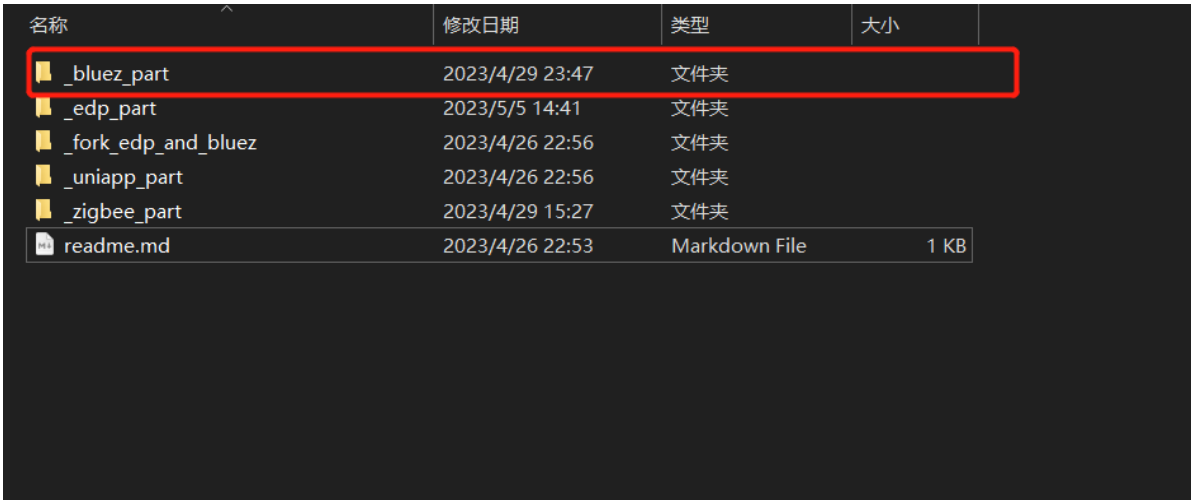
（4）PC上(Linux)运行的代码：

_photo_download_part：可执行文件名为"**photo**"，用于将onenet平台上的图片下载到PC端并恢复成图片

0.技术框图

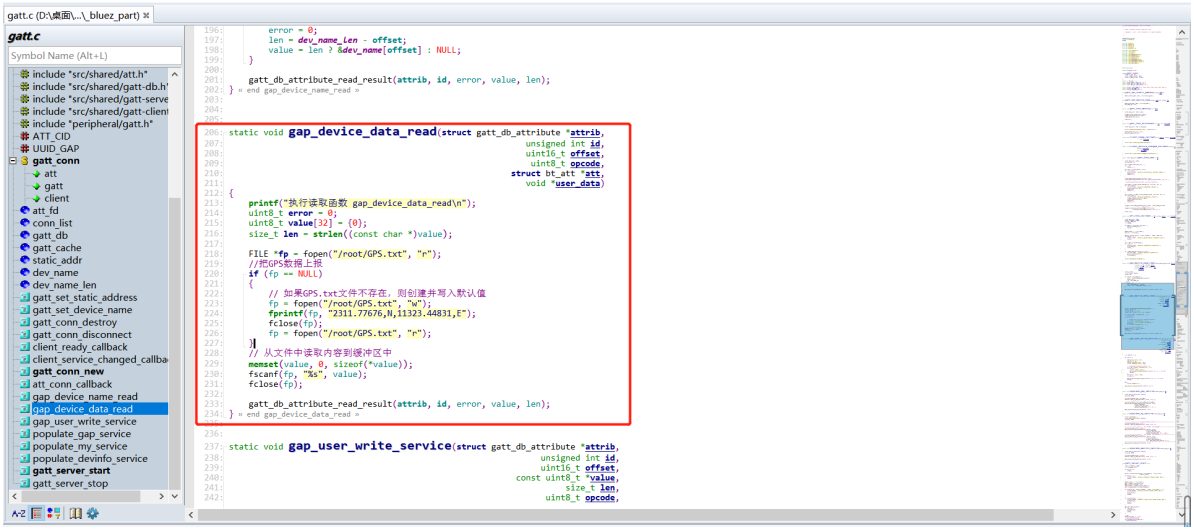


1._bluez_part



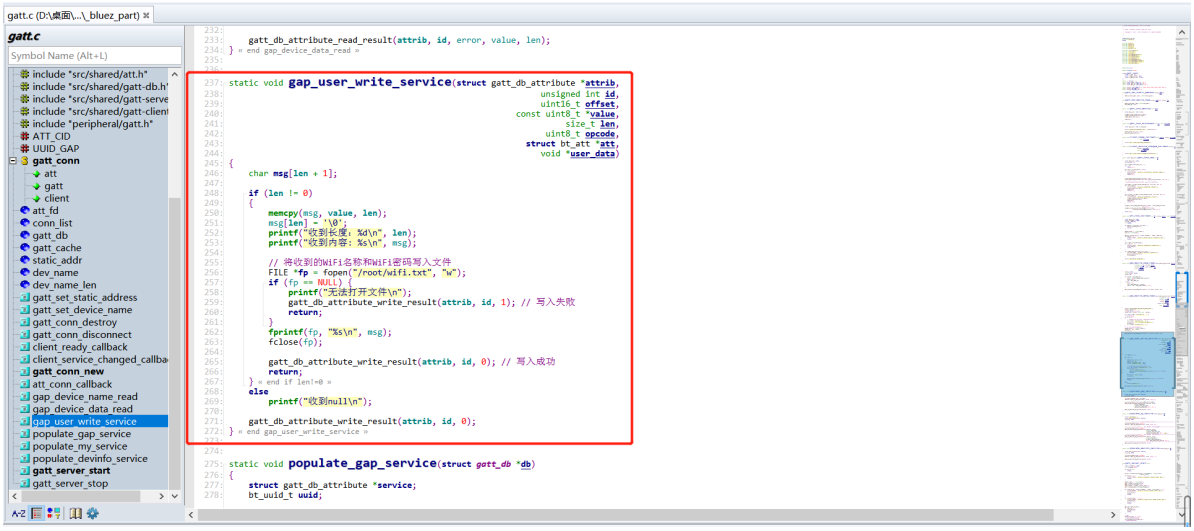
这部分代码，主要是实现了蓝牙模块的驱动，蓝牙模块主要实现了如下功能：

1.读服务，我是直接读取/root/GPS.txt的内容，后续可以想一下如何改进



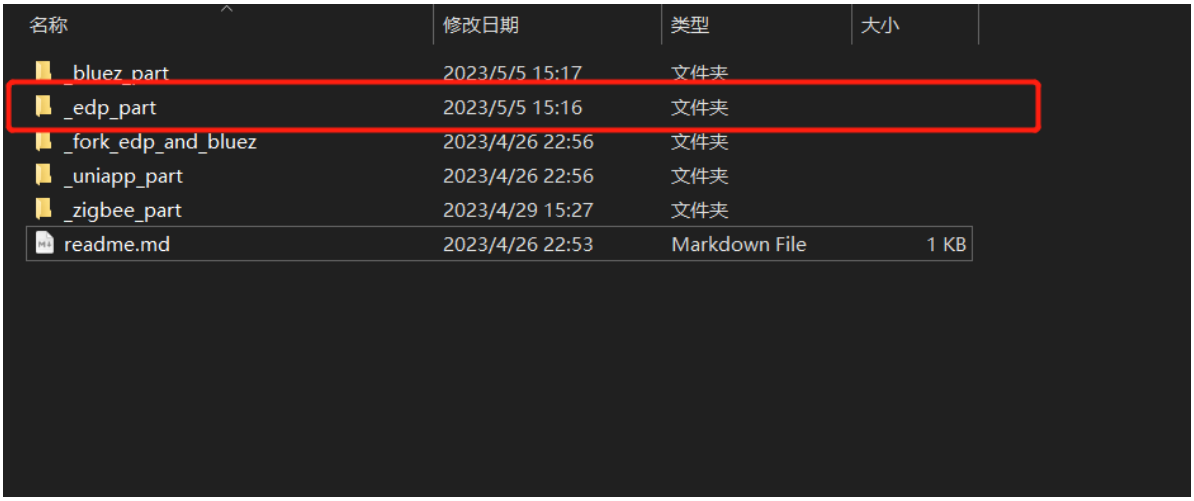
```
static void gap_device_data_read(struct gatt_db_attribute *attrib,
                                 unsigned int id,
                                 uint16_t offset,
                                 uint8_t opcode,
                                 struct bt_att *att,
                                 void *user_data)
{
    printf("执行读取函数 gap_device_data_read\n");
    uint8_t error = 0;
    uint8_t value[32] = {0};
    size_t len = strlen((const char *)value);

    FILE *fp = fopen("/root/GPS.txt", "r");
    //把GPS数据上报
    if (fp == NULL)
    {
        // 如果GPS.txt文件不存在，则创建并写入默认值
        fp = fopen("/root/GPS.txt", "w");
        fprintf(fp, "2311.77676,N,11323.44831,E");
        fclose(fp);
        fp = fopen("/root/GPS.txt", "r");
    }
    // 从文件中读取内容到缓冲区中
    memset(value, 0, sizeof(*value));
    fscanf(fp, "%s", value);
    fclose(fp);

    gatt_db_attribute_read_result(attrib, id, error, value, len);
}
```

1.写服务，更改WiFi密码的格式为**"WiFi名称,WiFi密码"**，例如:"Ace2,123780807"，Ace2为WiFi名称，123780807为WiFi密码



```
static void gap_user_write_service(struct gatt_db_attribute *attrib,
                                   unsigned int id,
                                   uint16_t offset,
                                   const uint8_t *value,
                                   size_t len,
                                   uint8_t opcode,
                                   struct bt_att *att,
                                   void *user_data)
{
    char msg[len + 1];

    if (len != 0)
    {
        memcpy(msg, value, len);
        msg[len] = '\0';
        printf("收到长度: %d\n", len);
        printf("收到内容: %s\n", msg);

        // 将收到的WiFi名称和WiFi密码写入文件
        FILE *fp = fopen("/root/wifi.txt", "w");
        if (fp == NULL) {
            printf("无法打开文件\n");
            gatt_db_attribute_write_result(attrib, id, 1); // 写入失败
            return;
        }
        fprintf(fp, "%s\n", msg);
        fclose(fp);

        gatt_db_attribute_write_result(attrib, id, 0); // 写入成功
        return;
    } // end if len!=0
    else
        printf("收到null\n");

    gatt_db_attribute_write_result(attrib, id, 0);
}
```

2._edp_part



这部分代码，主要是设备与云平台(OneNet)的交互中间层代码，主要实现了如下功能：

2.1.主线程

1.串口接收ZIGBEE协调器设备串口数据，协调器设备数据主要包含路由器设备及终端设备通过Mesh网传输过来的**温湿度**、**光照强度**数据

```
    /*3.把串口数据处理后放到buf数据缓冲区中上传到onenet*/
    /*3.1.准备要传到onenet上的数据*/
    /*3.1.1.准备串口0获得的温度（temp）、湿度（humi）、光照强度数据（light）*/
    onenet_value.temp = ((uart_buf0[0] - '0') * 10) + (uart_buf0[1] - '0');
    onenet_value.humi = ((uart_buf0[2] - '0') * 10) + (uart_buf0[3] - '0');
    if (!isdigit(uart_buf0[5])) { //如果第6位数据不为数字
        onenet_value.light = (uart_buf0[4] - '0');
    }else{
        onenet_value.light = ((uart_buf0[4] - '0') * 10) + (uart_buf0[5] - '0');
    }
```

2.串口接收GPS数据，GPS数据将被解析并存放至**gngga_buf**结构体中，结构体成员详见**gps_analyse.c**

```
    /*3.1.2.准备串口1获得的GPS经度、维度数据*/
    memset(&gngga_buf, 0, sizeof(gngga_buf));   //清空gngga_buf结构体
    gps_analyse(uart_buf1, &gngga_buf);         //将数据放进gngga_buf结构体，这个结构里暂时存储了解析后的GPS数据

    //2311.77676,N = 23+(11.77676/60) = 23.1962793°N
    //                  (float)(2311/100) =23.00000       (float)(2311%100) =11.00000       ((float)77676)/10000 = 0.77676
    onenet_value.latitude = (float)(gngga_buf.latitude/100) + ((((float)(gngga_buf.latitude%100)) + (((float)gngga_buf.latitude_x)/10000))/60);
    onenet_value.longitude = (float)(gngga_buf.longitude/100) + ((((float)(gngga_buf.longitude%100)) + (((float)gngga_buf.longitude_x)/10000))/60);
```

3.根据时间及光照强度，在17:00开始进行5分钟的拍照

```
    /*定时拍照功能*/
    timestamp = time(NULL);
    /*将时间戳转换为本地时间*/
    struct tm *local_time = localtime(&timestamp);
    /*判断是否是下午五点钟*/
    if ((local_time->tm_hour == 17 && local_time->tm_min >= 0 && local_time->tm_min <= 5 && local_time->tm_sec >= 0)) {//下午五点开始拍照，拍个5分钟
        if(onenet_value.light<50)
        {
            /* 点灯 */
            sprintf(command, "echo 1 > /sys/class/leds/red/brightness");
            ret = system(command);
            /* 拍照 */
            sprintf(command, "ffmpeg -y -i /dev/video0 -vframes 1 -s 1920x1080 -q:v 0 -f image2 ./test_photo.jpg");
            ret = system(command);
            sleep(1);
            /* 关灯 */
            sprintf(command, "echo 0 > /sys/class/leds/red/brightness");
            ret = system(command);
        }else{
            /* 拍照 */
            sprintf(command, "ffmpeg -y -i /dev/video0 -vframes 1 -s 1920x1080 -q:v 0 -f image2 ./test_photo.jpg");
            ret = system(command);
            sleep(1);
        }
    } « end if (local_time->tm_hour=... »
```

4.将温度、湿度、光照强度、经纬度数据上传至OneNet

```
    /*3.2.清空原本的onenet数据流缓冲区，把新数据写进去，发送到onenet平台*/
    switch(onenet_send_state)
    {
    case 0://状态0发送TEMP数据
        memset(buf ,0 ,sizeof(buf));
        OneNet_SendData(buf,"Temp",onenet_value.temp);
        sleep(1);
        printf("Send Temp: %d \n", onenet_value.temp); // 打印
        onenet_send_state = 1;
    case 1://状态1发送HUMI数据
        memset(buf ,0 ,sizeof(buf));
        OneNet_SendData(buf,"Humi",onenet_value.humi);
        sleep(1);
        printf("Send Humi: %d \n", onenet_value.humi); // 打印
        onenet_send_state = 2;
    case 2://状态2发送LIGH数据
        memset(buf ,0 ,sizeof(buf));
        OneNet_SendData(buf,"ligh",onenet_value.light);
        sleep(1);
        printf("Send LIGH: %d \n", onenet_value.light); // 打印
        onenet_send_state = 3;
    case 3://状态3发送维度数据
        OneNet_SendData_float(1,onenet_value.latitude);
        sleep(1);
        printf("latitude: %f \n", onenet_value.latitude); // 打印
        onenet_send_state = 4;
    case 4://状态4发送经度数据
        OneNet_SendData_float(2,onenet_value.longitude);
        sleep(1);
        printf("longitude: %f \n", onenet_value.longitude); // 打印
        onenet_send_state = 0;
        printf("*************Send UART End*************\n"); // 打印
        printf("*************Send UART End*************\n"); // 打印
    } « end switch onenet_send_state »
```

2.1.副线程

1.根据OneNet云平台下发的数据，实现补光灯的远程控制，命令格式为**{"LEDSET":"1"}/*{"LEDSET":"0"}***

```
    } else {
        /* 如果没有找到"{\"CAMERA\":\""字符串，则继续查找"{\"LEDSET\":\""字符串 */
        cmdStart = strstr(RecvBuffer, "{\"LEDSET\":\"");
        if (cmdStart == NULL) {
            /* 如果命令部分不存在，则执行默认操作 */
            continue;
        }

        cmdStart += strlen("{\"LEDSET\":\"");
        int cmdLen = strcspn(cmdStart, "\"}");

        /* 复制命令部分到一个新的缓冲区 */
        char cmdBuffer[20]; // 假设命令部分的最大长度为20
        strncpy(cmdBuffer, cmdStart, cmdLen);
        cmdBuffer[cmdLen] = '\0';

        /* 根据提取出的命令执行相应的操作 */
        if (strcmp(cmdBuffer, "0") == 0) {
            /* 2.灭灯 */
            sprintf(command, "echo 0 > /sys/class/leds/red/brightness");
            ret = system(command);
        } else if (strcmp(cmdBuffer, "1") == 0) {
            /* 1.点灯 */
            sprintf(command, "echo 1 > /sys/class/leds/red/brightness");
            ret = system(command);
        }
    } « end else »
```

2.根据OneNet云平台下发的数据，实现相机的远程控制，命令格式为**{"CAMERA":"1"}/{"CAMERA":"0"}**

```
    /* 查找命令部分的起始位置和长度 */
    char* cmdStart = strstr(RecvBuffer, "{\"CAMERA\":\"");
    if (cmdStart != NULL) {
        /* 如果找到了"{\"CAMERA\":\""字符串，则执行相应的操作 */
        cmdStart += strlen("{\"CAMERA\":\"");
        int cmdLen = strcspn(cmdStart, "\"}");

        /* 复制命令部分到一个新的缓冲区 */
        char cmdBuffer[20]; // 假设命令部分的最大长度为20
        strncpy(cmdBuffer, cmdStart, cmdLen);
        cmdBuffer[cmdLen] = '\0';

        /* 执行相应的操作 */
        if (strcmp(cmdBuffer, "1") == 0) {
            /* 3.拍照 */
            sprintf(command, "ffmpeg -y -i /dev/video0 -vframes 1 -s 1920x1080 -q:v 0 -f image2 ./test_photo.jpg");
            ret = system(command);
        }
    } else {
```