E-Series 录制回放 API 文档

涉及文件

antsdrDevice.cpp

antsdrDevice.h

函数

int antsdrDevice::open(bool is_ref_pll)

打开 SDR 设备

bool antsdrDevice::set_rx_samprate(double fs)

设置 rx 端采样率，取值范围 2.09M~61.44M

bool antsdrDevice::set_rx_freq(double fs)

设置 rx 端中心频率，取值范围 70M~6G

bool antsdrDevice::set_rx_gain(double gain, int chanels)

设置 rx 端增益，取值范围-3~71，步进 1

bool antsdrDevice::set_tx_samprate(double fs)

设置 tx 端采样率，取值范围 2.09M~61.44M

bool antsdrDevice::set_tx_freq(double freq)

设置 tx 端中心频率，取值范围 70M~6G

bool antsdrDevice::set_tx_attenuation(double attenuation)

设置 tx 端衰减，取值范围 0-89.75，步进 0.25

void antsdrDevice::create_socket_fd()

创建与 SDR 设备的网络连接

void antsdrDevice::config_record_data(int64_t recordersize)

设置单次采集的字节数

void antsdrDevice::config_recorder_data_key(int64_t recordersize)

设置按键录制所采集的字节数

uint32_t antsdrDevice::recorder_data_file(FILE * file, int filesize)

读取 filesize 字节长度的数据到 file 文件中

void antsdrDevice::sample_stop_replay()

SDR 设备停止回放，未停止就进行其他操作可能造成下位机卡死！

void antsdrDevice::sample_start_replay()

SDR 设备开始回放

void antsdrDevice::send_replay_len(uint32_t length)

设置回放数据字节长度

uint32_t antsdrDevice::send_once_block_data(char *tx_buffer)

发送 4MB 数据，数据的起始地址为 tx_buffer

void antsdrDevice::release_socket()

关闭与 SDR 设备的网络连接

使用模板

打开并设置参数

```
antsdrDevice device;
if(device.open(true) < 0){
    printf('Failed to open device\n');
    return;
}
device.set_rx_samprate(61440000);
device.set_rx_freq(5766.5e6);
device.set_rx_gain(65, 1);
device.set_tx_samprate(61440000);
device.set_tx_freq(5766.5e6);
device.set_tx_attenuation(20);
```

录制

```
device.create_socket_fd();
device.config_recorder_data(CHUNK_SIZE*32);
FILE *file = fopen('recorder_data.cs16','wb');
if (!file) {
    perror('Failed to open file for writing');
    device.release_socket();
    return -1;
}
device.recorder_data_file(file, CHUNK_SIZE*32);
fclose(file);
device.release_socket();
```

注意

CHUNK_SIZE 的值为 4*1024*1024，录制和发送的文件大小必须为 4MB 的整数倍

回放

```
device.create_socket_fd();
device.sample_stop_replay();
device_.release_socket();
device_.create_socket_fd();

FILE *file_r = fopen('recorder_data.cs16', 'rb');
if (!file_r) {
    perror('Failed to open file');
    device.release_socket();
    exit(EXIT_FAILURE);
}
fseek(file_r, 0, SEEK_END);
uint64_t file_size = ftell(file_r);
rewind(file_r);
int16_t blockcnt = file_size /CHUNK_SIZE;
file_size= blockcnt*CHUNK_SIZE;
device.send_replay_len(file_size);

char *tx_buffer = (char*)malloc(CHUNK_SIZE);
if (!tx_buffer) {
    printf('Failed to allocate memory for tx buffer\n');
    fclose(file_r);
    device.release_socket();
    exit(EXIT_FAILURE);
}
while ((bytes_read = fread(tx_buffer, 1, CHUNK_SIZE, file_r)) > 0) {
    one_block_sent = device.send_once_block_data(tx_buffer);
    total_sent += one_block_sent;
    if (one_block_sent == 0) {
        printf('Warning: send_once_block_data returned 0, this might indicate an error\n');
        break;
    }
    if(total_sent == file_size)
        break;
}
fclose(file_r);
free(tx_buffer);
device.sample_start_replay();
device.release_socket();
```

按键录制

```
device.create_socket_fd( );
device.config_recorder_data_key(TOTAL_SIZE);
FILE *file_key = fopen('recorder_data_key.cs16', 'wb' );
if ( !file_key) {
    perror('Failed to open file for writing' );
    device.release_socket( );
    return -1;
}
device.recorder_data_file(file_key, TOTAL_SIZE);
fclose(file_key);
device.release_socket( );
printf('Replay completed successfully.\n' );
```

注意

在程序结束时一定要调用 device.release_socket( )，否则会报错 Connection refused

例程 E-Series_example

功能：实现简单的录制与回放功能，接收 128M 的频谱数据，并将录制得到的 recorder_data.cs16 数据文件发送给 E310/E200 进行回放，回放 2s 后自动停止回放；E310 可以进入按键触发模式，按下按键后自动录制 512M 的数据并进行回放

运行：

在 E-Series_example 目录的终端下输入命令

```
mkdir build && cd build

cmake ..

make

./record_replay
```

```
find device 0 iq channel.
find device 1 iq channel.
请按下按键触发录制
Press the button to start data collection.
```