Mult MSO C SDK API 使用手册

Version 1.20

微目电子科技

2025-08-12

http://www.vimu.top/

升级记录

V1.20 (2025.8.1)

初始版本（跟随 MSO C SDK 版本）

目录

1. 简介

Mult MOS C SDK 是在 MSO C SDK 的基础上增加了多设备支持 API 的标准动态库。该接口支持 widows 系统(X86，X64 和 ARM64)和 linux 系统(X64，ARM32 和 ARM64)。

2. 与 MSO C SDK 的区别

设备控制、示波器、DDS 和 IO 的功能 API，命名和使用方法都跟 MSO C SDK 的相同，唯一的区别就是增加了，设备 ID 参数 unsigned int dev_id。

相比 MSO C SDK，Mult MOS C SDK 增加了流模式功能，该文档主要介绍流模式的 API 和使用方法。

3. 流模式使用

流模式需要更快的处理数据，防止数据堆积到内存中，所有流模式仅支持回调函数模式。

3.1. 主函数

```
 int main()
{
    //初始化
    InitDll(1, 1);
    //设置设备和数据回调函数
    SetDevNoticeCallBack(NULL, mDevNoticeAddCallBack,
mDevNoticeRemoveCallBack);
    SetStreamDataReadyCallBack(NULL, mStreamDataReadyCallBack);
   //扫描已经插入的设备
    ScanDevice();
   //主循环
    while (true)
    {
        std::this_thread::sleep_for(std::chrono::milliseconds(100));
        //采集完成就退出
        if(capture_ok)
            break;
    };
    //释放资源
    FinishDll();
    return 0;
}
```

3.2. 设备插入回调函数

```
void CALLBACK mDevNoticeAddCallBack(void* ppara, unsigned int dev_id)
{
    //DDS 初始化
    DDSInit(dev_id, 0, DDS_OUT_MODE_CONTINUOUS);

    //IO 初始化
    IOInit(dev_id);
```

```
    //选择工作模式
    SetOscCaptureMode(dev_id, 1);
    //设置采集范围
    SetStreamChannelRangemV(dev_id, 0, -10000, 10000);
    SetStreamChannelRangemV(dev_id, 1, -10000, 10000);

    //采样率设置
    int sample_num = GetStreamSupportSampleRateNum(dev_id);
    if (sample != NULL)
    {
        delete[]sample;
        sample = NULL;
    }
    sample = new unsigned int[sample_num];
    //读取支持的采样率
    if (GetStreamSupportSampleRates(dev_id, sample, sample_num))
    {
        for (int i = 0; i < sample_num; i++)
            std::cout << std::dec << sample[i] << '\n';
        std::cout << std::endl;
    }
    //记录设备和初始化采集状态
    m_dev_id = dev_id;
    capture_ok = false;
    capture_ok_mask = 0;
    //使用 1M 采样率，采集 10M 数据
    uint64_t length = 1024*1024*10;    //10M
    StreamCapture(m_dev_id, length/1024, capture_channel_mask, 1000000);//1M 采样率
}
```

3.3.数据回调函数

```
void CALLBACK mStreamDataReadyCallBack(void* ppara, unsigned int dev_id, unsigned char channel_index, double* buffer, unsigned int buffer_length, unsigned int failed, unsigned int success, unsigned long long int need_total_sample, unsigned long long int total_sample, unsigned long long int menoryuse)
{
    //Note：The callback function should not handle complex tasks, and if it takes too long,
    //it will cause the watchdog to reset and the USB to reconnect

    /*说明
        buffer 当次的缓冲区，回调完成缓冲区将销毁，所以需要将该缓冲区的数据自己拷贝走
        buffer_length 当次的缓冲区长度
        failed 采集是否失败
```

```
            success 采集是否完成
            need_total_sample 一共需要采集的数据
            total_sample 已经采集完成的数据
            menoryuse 目前 dll 使用了多少缓冲区（越多，代表 dll 堆积的数据越多）
        */

        //根据需求处理 buffer 中数据…….

        //最后一个通道回调完成，准备退出
        if(success)
        {
            capture_ok_mask |= (0x01<<channel_index);
            if(capture_ok_mask==capture_channel_mask)
                capture_ok = true; //标志完成，主函数退出
            }
        }
    }
```

4. 流模式 API 说明

4.1. 工作模式

int SetOscCaptureMode(unsigned int dev_id, int is_stream_mode);

Description This routines set the osc capture mode.

Input: dev_id the dev id

is_stream_mode enable the stream mode or not

Output Return value 1 Success

0 Failed

4.2. 采集范围设置

设备的前级带有程控增益放大器，当采集的信号小于 AD 量程的时候，增益放大器可以把信号放大，更多的利用 AD 的位数，提高采集信号的质量。Dll 会根据设置的采集范围，自动的调整前级的增益放大器。

int GetStreamRangeMinmV(unsigned int dev_id);

int GetStreamRangeMaxmV(unsigned int dev_id);

Description This routines set the range of input signal.

Output Return value

minmv the minimum voltage of the input signal (mV)

maxmv the maximum voltage of the input signal (mV)

int SetStreamChannelRangemV(unsigned int dev_id, int channel, int minmv, int maxmv);

Description This routines set the range of input signal.

Input: channel the set channel

0 channel 1

1 channel 2

minmv the minimum voltage of the input signal (mV)

maxmv the maximum voltage of the input signal (mV)

Output Return value 1 Success

0 Failed

说明：最大的采集范围为探头 X1 的时候，示波器可以采集的最大电压。比如 MSO20 为 [-12000mV,12000mV]。

注意：为了达到更好波形效果，一定要根据自己被测波形的幅度，设置采集范围。必要时，可以动态变化采集范围。

4.3. 采样率

int GetStreamSupportSampleRateNum(unsigned int dev_id);

Description This routines get the number of samples that the equipment support.

Input: -

Output Return value the sample number

int GetStreamSupportSampleRates(unsigned int dev_id, unsigned int* sample, int maxnum);

Description This routines get support samples of equipment.

Input: sample the array store the support samples of the equipment

maxnum the length of the array

Output Return value the sample number of array stored

4.4. 采集

int StreamCapture(unsigned int dev_id, unsigned long long int length_kb, unsigned short capture_channel, unsigned int sample_rate);

Description This routines set the capture length and start capture.

Input: length capture length(KB)

capture_channel: //ch1=0x0001 ch2=0x0020 ch3=0x0040 ch4=0x0080 logic=0x0100

ch1+ch2 0x03

ch1+ch2+ch3 0x07

ch1+ch2+ch3+ch4 0x0F

Output Return 1 success

-1 sample_rate error

-2 device id error

void StreamStopCapture(unsigned int dev_id);

Description This routines stop the capture

Input:

Output Return

4.5. 采集完成通知

当数据采集完成时，回掉函数会回调通知，并提供相应的采集缓冲区和采集状态。

Void SetStreamDataReadyCallBack(void* ppara, StreamDataReadyCallBack datacallback);

Description This routines sets the callback function of capture complete.

Input: **ppara** the parameter of the callback function

datacallback a pointer to a function with the

StreamDataReadyCallBack prototype:

Output -

void CALLBACK StreamDataReadyCallBack(void* ppara, unsigned int dev_id, unsigned char channel_index, double* buffer, unsigned int buffer_length, unsigned int failed, unsigned int success, unsigned long long int need_total_sample, unsigned long long int total_sample, unsigned long long int menoryuse);

说明：

buffer 当次的缓冲区，回调完成缓冲区将销毁，所以需要将该缓冲区的数据自己拷贝走

buffer_length 当次的缓冲区长度

failed 采集是否失败

success 采集是否完成

need_total_sample 一共需要采集的数据

total_sample 已经采集完成的数据

menoryuse 目前动态库使用了多少缓冲区（越多，代表dll堆积的数据越多）