



SDK 使用手册

Version 2.6

仪星电子科技

2015-8-15



升级

V2.1 (2014.5.13) 全新的设计

V2.2 (2014.7.10) 修正说明书软件触发描述

V2.3 (2014.11.28) 修复”ReadVoltageDatas”读取时间过长问题

V2.4 (2014.12.5) 修复控制台类程序，无法检测 usb 拔插问题

V2.5 (2015.7.27) 增加 ISDS2602 设备支持

V2.6 (2015.8.15) 增加 Roll Mode 支持



目录

1. 简介.....	1
2. 初始化和结束.....	1
3. 设备监测.....	1
4. 采集范围设置.....	2
5. 采样率.....	2
6. 触发（硬件触发）.....	3
7. AC/DC.....	4
8. 采集.....	4
9. 采集完成通知.....	5
10. 数据读取.....	6
11. DDS.....	6



1. 简介

SDK 作为虚拟示波器配备的一个 Windows 标准 DLL 接口，通过这个接口可以直接控制虚拟示波器，并获得示波器采集的数据。该 SDK 支持 MDS0、MDS0-LA、HDS0、DDSO、ISDS205、ISDS210、ISDS220 和 ISDS2062 设备。

2. 初始化和结束

调用 InitDll() 来完成动态库的初始化，初始化的时候会分配内存和资源用于设备监测和数据读取用。

int InitDll(void);

Description Dll initialization

Input: -

Output: **Init Status**

Return value 1 Success

0 Failed

调用 FinishDll() 来完成动态库的结束，结束的时候，会时释放初始化中申请的内存和相关资源。

int FinishDll(void);

Description Dll finished

Input: -

Output: **-Finished Status**

Return value 1 Success

0 Failed

3. 设备监测

当 DLL 检测到有设备接入时，有 3 种方式通知主程序，回掉函数、触发 Event 和主程序循环检测。

3.1 回调函数

当检测到设备插入时，如果主程序注册了回掉函数 "addcallback"，它就会被调用；当检测到设备拔出时，如果主程序注册了回掉函数 "rmvcallback"，它就会被调用。Dll 有一个函数专门用于设置这个 2 个回掉函数

void SetDevNoticeCallBack(void* ppara, AddCallBack addcallback, RemoveCallBack rmvcallback);

Description This routines sets the callback function of equipment status changed.

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

addcallback a pointer to a function with the following prototype:
void AddCallBack(void * ppara)

rmvcallback a pointer to a function with the following prototype:
Void RemoveCallBack(void * ppara)

Output -

3.2 Event

当检测到设备插入时，如果主程序注册了 Event 句柄 "addevent"，它就会被设置；当检测到设备拔出时，如果主程序注册了回掉函数 "rmvevent"，它就会被设置。需要注意的是，主程序检测到 Event 后，需要将 Event 复位。Dll 有一个函数专门用于设置这 2 个 Event 句柄



void SetDevNoticeEvent(HANDLE addevent, HANDLE rmvevent);

Description This routines set the event handle, these will be set, when equipment status changed.

Input: **addevent** the event handle
rmvevent the event handle

Output -

3.3 循环检测

int IsDevAvailable();

Description This routines return the device is available or not.

Input: -

Output **Return value** 1 available
0 not available

说明：3 方式只要使用其中的一种就可以了，回掉函数和 Event 都是异步的处理方式，更加的高效；循环检测需要主程序过一定时间就检测设备是否插入或者拔出。

4.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。DII 会根据设置的采集范围，自动的调整前级的增益放大器。

int SetOscChannelRange(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 ISDS220 为 [-16000mV,16000mV]。

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。

5.采样率

int GetOscSupportSampleNum();

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

Input: -

Output **Return value** the support sample number

int GetOscSupportSamples(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

int SetOscSample(unsigned int sample);

Description This routines set the sample.

Input: **sample** the set sample

Output **Return value** 0 Failed
 other value new sample

unsigned int GetOscSample();

Description This routines get the sample.

Input: -

Output **Return value** sample

6.触发（硬件触发）

该功能需要设备硬件触发支持。硬件触发的触发点都是采集数据的最中间，比如采集128K 数据，触发点就是第 64K 的点。

触发模式

```
#define TRIGGER_MODE_AUTO 0
#define TRIGGER_MODE_LIANXU 1
```

触发条件

```
#define TRIGGER_STYLE_NONE 0x0000 //not trigger
#define TRIGGER_STYLE_RISE_EDGE 0x0001 //Rising edge
#define TRIGGER_STYLE_FALL_EDGE 0x0002 //Falling edge
#define TRIGGER_STYLE_EDGE 0x0004 //Edge
#define TRIGGER_STYLE_P_MORE 0x0008 //Positive Pulse width(>)
#define TRIGGER_STYLE_P_LESS 0x0010 //Positive Pulse width(>)
#define TRIGGER_STYLE_P 0x0020 //Positive Pulse width(<>)
#define TRIGGER_STYLE_N_MORE 0x0040 //Negative Pulse width(>)
#define TRIGGER_STYLE_N_LESS 0x0080 //Negative Pulse width(>)
#define TRIGGER_STYLE_N 0x0100 //Negative Pulse width(<>)
```

int IsSupportHardTrigger();

Description This routines get the equipment support hardware trigger or not .

Input: -

Output **Return value** 1 support hardware trigger
 0 not support hardware trigger

unsigned int GetTriggerMode();

Description This routines get the trigger mode.

Input: -

Output **Return value** TRIGGER_MODE_AUTO
 TRIGGER_MODE_LIANXU



void SetTriggerMode(unsigned int mode);

Description This routines set the trigger mode.

Input: **mode** TRIGGER_MODE_AUTO
TRIGGER_MODE_LIANXU

Output -

unsigned int GetTriggerStyle();

Description This routines get the trigger style.

Input: -

Output **Return value** TRIGGER_STYLE_NONE
TRIGGER_STYLE_RISE_EDGE
TRIGGER_STYLE_FALL_EDGE
TRIGGER_STYLE_EDGE
TRIGGER_STYLE_P_MORE
TRIGGER_STYLE_P_LESS
TRIGGER_STYLE_P
TRIGGER_STYLE_N_MORE
TRIGGER_STYLE_N_LESS
TRIGGER_STYLE_N

void SetTriggerStyle(unsigned int style);

Description This routines set the trigger style.

Input: **style** TRIGGER_STYLE_NONE
TRIGGER_STYLE_RISE_EDGE
TRIGGER_STYLE_FALL_EDGE
TRIGGER_STYLE_EDGE
TRIGGER_STYLE_P_MORE
TRIGGER_STYLE_P_LESS
TRIGGER_STYLE_P
TRIGGER_STYLE_N_MORE
TRIGGER_STYLE_N_LESS
TRIGGER_STYLE_N

Output -

unsigned int GetTriggerSource();

Description This routines get the trigger source.

Input: -

Output **Return value** 0 :channel 1
1 :channel 2

void SetTriggerSource(unsigned int source);

Description This routines set the trigger source.

Input: **source** 0 :channel 1
1 :channel 2



Output -

int GetTriggerLevel();

Description This routines get the trigger level.

Input: -

Output **Return value** level (mV)

void SetTriggerLevel(int level);

Description This routines set the trigger level.

Input: level (mV)

Output -

7.AC/DC

int IsSupportAcDc();

Description This routines get the device support AC/DC switch or not.

Input: -

Output **Return value** 0 :support AC/DC switch
1 :not support AC/DC switch

void SetAcDc(unsigned int channel, int ac);

Description This routines set the device AC coupling.

Input: channel 0 :channel 1
1 :channel 2
ac 1 : set AC coupling
0 : set DC coupling

Output -

int GetAcDc(unsigned int channel,);

Description This routines get the device AC coupling.

Input: channel 0 :channel 1
1 :channel 2

Output **Return value** 1 : AC coupling
0 : DC coupling

8.采集

调用**Capture**函数开始采集数据，**length**就是你想要采集的长度，以K为单位，比如**length=10**,就是10K 10240个点。对于采样率的大于等于存储深度的采集长度，取**length**和存储深度的最小值；对于采样率小于存储深度，取**length**和1秒采集数据的最小值。函数会返回实际采集数据的长度。

int Capture(int length);

Description This routines set the capture length and start capture.

Input: **length** capture length(KB)

Output **Return value** the real capture length(KB)

unsigned int GetMemoryLength();

Description This routines get memory depth of equipment (KB).

Input: -

Output memory depth of equipment(KB)

Roll Mode: 该模式下，采样率被固定的设置为最小采样率，采集长度也是固定的设置为 1 秒采集数据长度。正常的调用 **Capture**，把每次采集的数据连接在一起显示就是完整的波形。

int IsSupportRollMode();

Description This routines get the equipment support roll mode or not .

Input: -

Output **Return value** 1 support roll mode
0 not support roll mode

int SetRollMode(unsigned int en);

Description This routines enable or disenable the equipment into roll mode.

Input: -

Output **Return value** 1 success
0 failed

9.采集完成通知

当数据采集完成时，有 3 种方式通知主程序，回掉函数、触发 Event 和主程序循环检测。

9.1 回调函数

当数据采集完成时，如果主程序注册了回掉函数"**datacallback**"，它就会被调用。Dll 有一个函数专门用于设置这个回掉函数

void SetDataReadyCallBack(void* ppara, DataReadyCallBack 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 following prototype:
void **DataReadyCallBack** (void * **ppara**)

Output -

9.2 Event

当数据采集完成时，如果主程序注册了 Event 句柄"**dataevent**"，它就会被设置。需要注意的是，主程序检测到 Event 后，需要将 Event 复位。Dll 有一个函数专门用于设置这个 Event 句柄

void SetDevDataReadyEvent(HANDLE dataevent);

Description This routines set the event handle, these will be set, when capture complete

Input: **dataevent** the event handle

Output -

9.3 循环检测

int IsDataReady();

Description This routines return the capture is complete or not.



Input: -
Output **Return value** 1 complete
0 not complete

说明：3 方式只要使用其中的一种就可以了，回掉函数和 Event 都是异步的处理方式，更加的高效；循环检测需要主程序开始采集以后，过一定时间就检测是否采集完成。

10.数据读取

unsigned int ReadVoltageDats(char channel, double* buffer,unsigned int length);

Description This routines read the voltage datas. (V)

Input: **channel** read channel 0 :channel 1
1 :channel 2
buffer the buffer to store voltage datas
length the buffer length

Output **Return value** the read length

11.DDS

int IsSupportDDSDevice();

Description This routines get support dds or not

Input: -

Output **Return value** support dds or not

int GetDDSSupportBoxingStyle(int* style);

Description This routines get support wave styles

Input: **style** array to store support wave styles

Output **Return value** if style==NULL return number of support wave styles
else store the styles to array, and return number of wave styles

void SetDDSBBoxingStyle(unsigned int boxing);

Description This routines set wave style

Input: **boxing** BX_SINE 0x00 //Sine
BX_SQUARE 0x01 //Square
BX_TRIANGULAR 0x02 //Triangular
BX_UP_SAWTOOTH 0x03 //Up Sawtooth
BX_DOWN_SAWTOOTH 0x04 //Down Sawtooth

Output: -

void SetDDSPinlv(unsigned int pinlv);

Description This routines set frequency

Input: **pinlv** frequency

Output: -

void SetDDSDutyCycle(int cycle);

Description This routines set duty cycle

Input: **cycle** duty cycle



Output: -

void DDSOutputEnable(int enable);

Description This routines enable dds output or not

Input: **enable** 1 enable
0 not enable

Output: -

int IsDDSOutputEnable();

Description This routines get dds output enable or not

Input: -

Output **Return value** dds enable or not