SDK HTHardDll.dll Guide

4 Channels PC Oscilloscope

Content

I Remarks

All DLL was generated by Vs2015.

WORD: unsigned 16bit integer, two bytes

BOOL: Boolean type, 32bit four bytes

ULONG: unsigned 32-bit integer, four bytes

All files in this DLL are compiled with the DLL_API symbols defined on the command line. This symbol should not be defined on any other project using this DLL. This way, any other project that contains this file in its source file will see the DLL_API functions as if they were imported from the DLL.

```
#ifndef DLL_API
#define DLL_API extern "C" __declspec(dllimport)
#endif
```

Define standard calls:

```
#define WIN_API __stdcall
```

char bGet All functions containing the bGet option. When bGet=1, it means getting the value from the device, 0 means settings.

II Function Introduction

1. dsoHTDeviceConnect

Function declaration: DLL_API WORD WINAPI dsoHTDeviceConnect (WORD DeviceIndex)

Return value:
Returns the instrument connection status. 0: Not connected; non-0: Connected.

Parameter:
DeviceIndex
WORD type variable, representing the index value of the current device.

Remark:
Get whether the PC is connected to the instrument.

Program example:

```
WORD DeviceIndex = 0;
//Call functions
if(0 = dsoHTSearchDevice(DeviceIndex))
{
//Do not connect
}
else
{
//There is device
}
```

2. dsoSetUSBBus

Function declaration: DLL_API WORD WINAPI dsoSetUSBBus(WORD DeviceIndex)

Return value:

Parameter:
DeviceIndex
WORD type variable, representing the index value of the current device.

Remark:
Set the bus trigger mode. This setting is reserved for use and has not been actually controlled yet.

3. dsoHTAttribsCHPos

Function declaration: DLL_API WORD dsoHTAttribsCHPos(WORD nDeviceIndex,
char nCH,
WORD *nPos,
char bGet = 1)

Return value:
0: failure, non-zero: success

Parameter:
DeviceIndex
WORD type variable, representing the index value of the current device

nCH
Channel 0-3

nPos
WORD type variable, indicating the current channel nCH vertical position range 0-255.

nCH
A char type variable represents the currently set channel. Range 0 - 3.

Remark:
Sets the vertical position of the channel. The range of the channel vertical position is 0-255, "0" means setting the channel position to the screen, "128" sets the channel to the middle of the screen; "255" means sets the channel to the top of the screen.

4. dsoHTAttribsCHEnable

Function declaration: DLL_API WORD dsoHTAttribsCHEnable (WORD nDeviceIndex,
char nCH,
char*bEnable,
char bGet = 1)

Return value:
0: failure, non-zero: success

Parameter:
DeviceIndex
WORD type variable, representing the index value of the current device

nCH
Channel 0-3

bEanbale

A char type variable, indicating that the current channel 1 is open and channel 0 is closed.

nCH
A char type variable represents the currently set channel. Range 0 - 3.

Remark:
Set the channel switch

5. dsoHTAttribsCHCouple

Function declaration: DLL_API WORD dsoHTAttribsCHCouple (WORD nDeviceIndex,
char nCH,
char*nCouple,
char bGet = 1)

Return value:
0: failure, non-zero: success

Parameter:
DeviceIndex
WORD type variable, representing the index value of the current device

nCH
Channel 0-3

nCouple
A char type variable, indicating the channel coupling mode. 0 is DC, 1 is AC.

nCH
A char type variable represents the currently set channel. Range 0 - 3.

Remark:
Set the channel coupling method

6. dsoHTAttribsCHVolt

Function declaration: DLL_API WORD dsoHTAttribsCHVolt (WORD nDeviceIndex,
char nCH, float *fVolt,char bGet,char nOption=0)

Return value:
0: failure, non-zero: success

Parameter:
DeviceIndex
WORD type variable, representing the index value of the current device

nCH
Channel 0-3

fVolt
Indicates that the channel voltage unit that can be set is mV.

nCH
A char type variable represents the currently set channel. Range 0 - 3

Remark:
Set the voltage range of the channel

7. dsoHTAttribsVTriggerPos

Function declaration: DLL_API WORD dsoHTAttribsVTriggerPos(WORD nDeviceIndex,
WORD nCH,
char bGet,
WORD *nPos1,
WORD *nPos2=NULL)

Return value:
0: failure, non-zero: success

Parameter:
DeviceIndex
WORD type variable, representing the index value of the current device.

nPos1
WORD type variable, indicating the vertical position of the trigger, ranging from 0-255.

nCH
WORD type variable, indicating the currently set channel. Range 0 - 3

nPos2
WORD type variable, set to NULL for secondary development.

Remark:
Sets the vertical position of the trigger.

8. dsoHTAttribsHTriggerPos

Function declaration: DLL_API WORD dsoHTAttribsHTriggerPos(WORD nDeviceIndex, char *nPos, char bGet = 1)

Return value:
0: failure, non-zero: success

Parameter:
DeviceIndex
WORD type variable, representing the index value of the current device.

nPos1
Char type variable, indicating the vertical position of horizontal trigger, ranging from 0-100.

Remark:
Set the horizontal trigger position, for example, when nPos=50, 50/100=0.5 means the horizontal trigger is in the center.

9.dsoHTSetSampleRateDiv

Function declaration: DLL_API WORD WINAPI dsoHTSetSampleRateDiv(WORD nDeviceIndex, WORD nTimeDIV)

Return value:
0: failure, non-zero: success

Parameter:
DeviceIndex

WORD type variable, representing the index value of the current device.

nTimeDIV,
WORD type variable, indicating that several commonly used time base values are used to indirectly set the sampling rate, that is, this function is called internally dsoHTSetSampleRate. In fact, this function sets several preset sampling rates. See Table 1 for details.

Remark:
Set the FGPA sampling rate.

10. dsoHTSetSampleRate

Function declaration: DLL_API WORD dsoHTSetSampleRate(WORD nDeviceIndex, double *fTimeDiv)

Return value:
0: failure, non-zero: success

Parameter:
DeviceIndex
WORD type variable, representing the index value of the current device.

fTimeDiv,
Double type variable pointer, used to set the sampling rate. Note that not all sampling rates can be set. The actual designed sampling rate is based on the pointer return.

Remark:
Set the FGPA sampling rate.

11. dsoHTStartCollectData

Function declaration: DLL_API WORD WINAPI dsoHTStartCollectData(WORD nDeviceIndex,WORD nStartControl)

Return value:
0: failure, non-zero: success

Parameter:
DeviceIndex
WORD type variable, representing the index value of the current device.

nStartControl
WORD type variable, indicating the mode of starting collection. 8 bits in total,
0:0 means AUTO, 1 means other means 0;
1:1 is rolling mode 0 is normal mode
2,3: Collection mode

Remark:
There are no special requirements. Just set nStartControl to 5 directly.

12. dsoHTGetState

Function declaration: DLL_API WORD WINAPI dsoHTGetState(WORD nDeviceIndex,PUCHAR pdate)

Return value:
Type 0: failure, non-zero: success

Parameter:
DeviceIndex
WORD type variable, representing the index value of the current device.

pDate
Unsigned char data pointer, its length is 128, secondary development only needs to understand the meaning of the 0th Byte
pdate[0]&1=1 means the oscilloscope has trigger =0 means no trigger
pdate[0]&2=2 means that the oscilloscope acquisition has ended =0 means that the acquisition has not been completed, and is usually used for secondary development.
pdate[0]&2==2 can collect data

Remark:
Get the collection status. The collected data can only be read when the collection is completed.

13. dsoHTGetData

Function declaration: DLL_API WORD WINAPI dsoHTCollectDataWave(WORD nDeviceIndex,
WORD*pCH1Data,
WORD*pCH2Data,
WORD*pCH3Data,
WORD*pCH4Data,
ULONG nReadLen=0)

Return value:
0: failure, non-zero: success

Parameter:
DeviceIndex
WORD type variable, representing the index value of the current device.

pCH%n%Data
WORD type variable pointer, used to store the collected data of channel n. The data range is 0-255. Length: when nReadlen is greater than 0, the length is nReadLen; when nReadLen is 0, the length is the memory length value set by dsoHTAttribsBufferLength.

nReadLen
ULONG variable, indicating that the length of the collection cannot be greater than the length set by dsoHTAttribsBufferLength. When it is 0, it means reading all.

Remark:
Data collection. The actual voltage value represented by the i-th point in the array pCH1Data is: (pCH1Data[i]-channel vertical position) x voltage/25,. For example, set the voltage range index of channel 1 to 5 (look up table 3 and get 20mV), dsoHTSetCHPos sets the vertical position of the channel to 128, sets the i-th point data to 65, and the actual voltage value of the i-th point on the side to be (65-128) *20mV/25=-50.4mV.

14. dsoHTAttribsBufferLength

Function declaration: DLL_API WORD dsoHTAttribsBufferLength(WORD nDeviceIndex,
ULONG *nBufferLength,
char bGet=1)

Return value:
0: failure, non-zero: success

Parameter:
DeviceIndex
WORD type variable, representing the index value of the current device.

nBufferLength
ULONG type variable pointer, data length of a single channel

Remark:
Note that this function sets the acquisition depth of a single channel, the range is 2.5K-64M (64*1024*1024), because the device's memory for storing data is only 64MB, it can only be set to 64MB when one channel is enabled. More specifically, if the number of channels opened is nCHEnable; the set collection depth is nLength, nCHEnable*nLength<=64KB. Note that when nCHEnable=3, 4 channels are actually opened.

15. dsoInitHard

Function declaration: DLL_API WORD dsoInitHard(WORD DeviceIndex)

Return value:
0: failure, non-zero: success

Parameter:
DeviceIndex
WORD type variable, representing the index value of the current device.

Remark:
Device initialization. It needs to be called promptly after the hardware is powered on and connected.

16. dsoHTAttribsTriggerMode

Function declaration: DLL_API WORD dsoHTAttribsTriggerMode(WORD nDeviceIndex, char*nTriggerMode, char bGet = 1)

Return value:
0: failure, non-zero: success

Parameter:
DeviceIndex
WORD type variable, representing the index value of the current device.

nTriggerMode
Char type variable, trigger mode. 0: Edge 1: Pulse 2: Video 17: Bus trigger

Remark:
Set trigger mode.

17. dsoHTAttribsTriggerSource

Function declaration: DLL_API WORD dsoHTAttribsTriggerMode(WORD nDeviceIndex, char*nTriggerSource, char bGet = 1)

Return value:
0: failure, non-zero: success

Parameter:
DeviceIndex
WORD type variable, representing the index value of the current device.

nTriggerSource
Char type variable, trigger source range 0-3

18. dsoHTAttribsTriggerSlop

Function declaration: DLL_API WORD dsoHTAttribsTriggerSlop(WORD nDeviceIndex, char*nTriggerSlop, char bGet = 1)

Return value:
0: failure, non-zero: success

Parameter:
DeviceIndex
WORD type variable, representing the index value of the current device.

nTriggerSlop
Char type variable, 0 means rising edge, 1 means falling edge

Remark:
Set the trigger source channel.

18. dsoHTAttribs AcqOption

Function declaration: DLL_API WORD dsoHTAttribsAcqOption (WORD nDeviceIndex, char * nOption, char bGet = 1)

Return value:
0: failure, non-zero: success

Parameter:
DeviceIndex
WORD type variable, representing the index value of the current device.

nOption
Char type variable, indicating that the collection mode has no special requirements and is set to 5

Remark:
Set the collection mode.

III Control Flow Chart

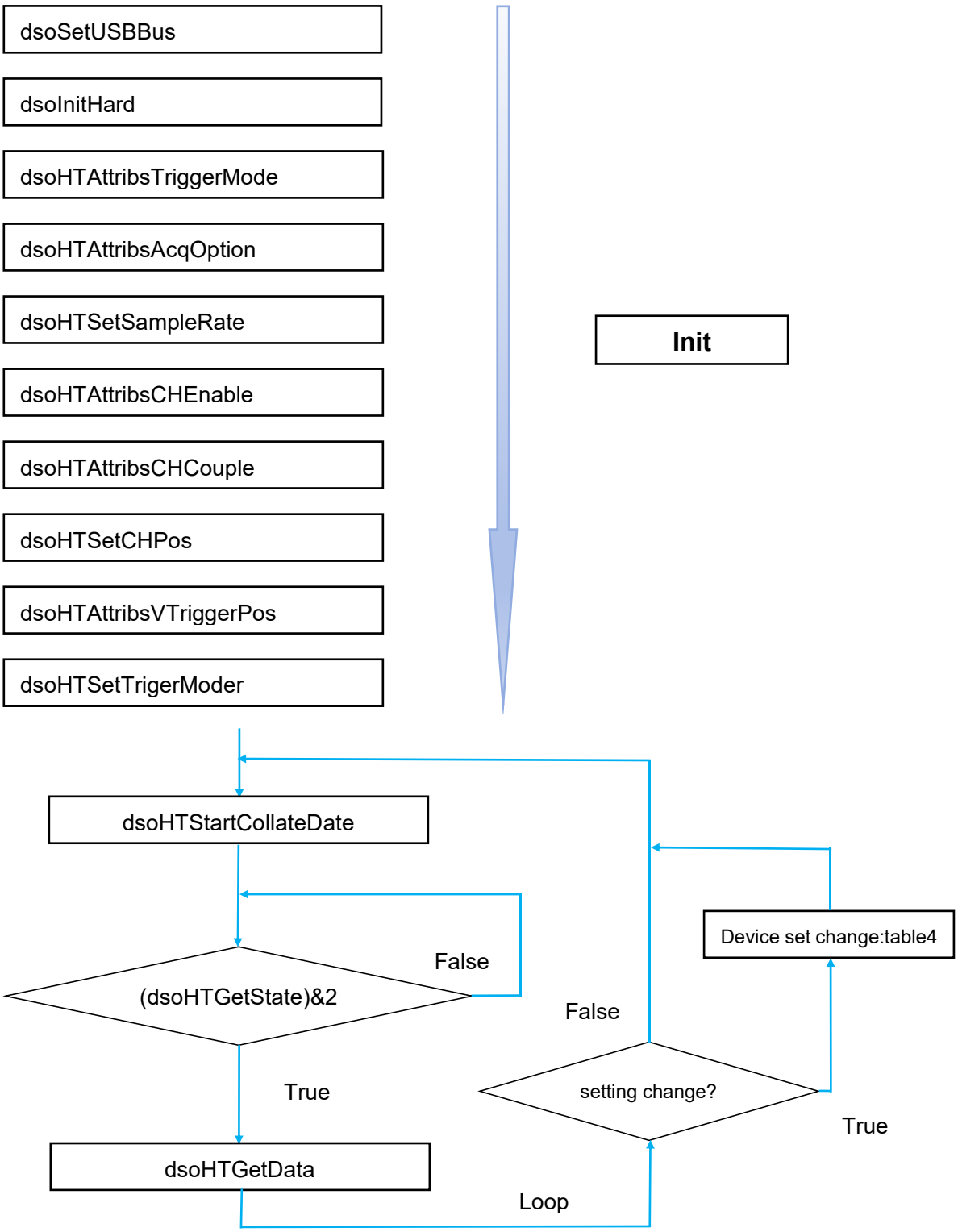

Table 1 Timebase and Sampling Rate

Index	Timebase	Sampling rate (Sa)	Index	Timebase	Sampling rate (Sa)
0	2nS	**S:1G D:0.5G Q:250M**	**19**	5mS	50K
1	5nS	**S:1G D:0.5G Q:250M**	**20**	10mS	25K
2	10nS	**S:1G D:0.5G Q:250M**	**21**	20mS	12.5K
3	20nS	**S:1G D:0.5G Q:250M**	**22**	50mS	5K
4	50nS	**S:1G D:0.5G Q:250M**	**23**	100mS	2.5K
5	100nS	**S:1G D:0.5G Q:250M**	**24**	200mS	1.25K
6	200nS	**S:1G D:0.5G Q:250M**	**25**	500mS	500
7	500nS	**S:1G D:0.5G Q:250M**	**26**	1S	250
8	1uS	250M	**27**	2S	125
9	2uS	125M	**28**	5S	50
10	5uS	50M	**29**	10S	25
11	10uS	25M	**30**	20S	12.5
12	20uS	12.5M	**31**	50S	5
13	50uS	5M	**32**	100S	2.5
14	100uS	2.5M	**33**	200S	1.25
15	200uS	1.25M	**34**	500S	0.5
16	500uS	500K	**35**	1000S	0.25
17	1mS	250K			
18	2mS	125K			

Note

1 Column " Sampling rate " shown in bold need Interpolation.

2 When the oscilloscope does not require interpolation, sample rate = 250 /timebase; 250 is number of points in a single grid, such as when the timebase is 1uS sampling rate = 250/(1e-6) = 250MSa.

3 "S" indicates single-channel mode; "D" means dual channel mode; "Q" means 3-4 channels turned on.

4 This table only indicates that when using the input parameter to set the sampling rate with dsoHTSetSampleRateDiv, the timebase index value can be used directly to set the sampling rate using the dsoHTSetSampleRate function.

Table 2 Range of each voltage base

Index	Voltage	**Range**	**Index**	Voltage	**Range**
0	0.5mV	4mV	**7**	100mV	800mV
1	1mV	8mV	**8**	200mV	1.6V
2	2mV	16mV	**9**	500mV	4V
3	5mV	40mV	**10**	1V	8V
4	10mV	80mV	**11**	2V	16V
5	20mV	160mV	**12**	5V	40V
6	50mV	400mV	**13**	10V	80V

Note

1 "Voltage division" means a large vertical grid voltage corresponding to the value, more precisely, is a waveform data acquisition data 32 corresponding to the difference value.

2 "Range" is represented by 1: 1 probe corresponding to the range. For example, 100mV with a 1: 1 scale probe is 800mV; 1:10 probe range is 8V.

Table 3 Common oscilloscope settings

Index	Setting	Functions
1	Voltage DIV	dsoHTAttribsCHVolt
2	Sampling Rate	dsoHTSetSampleRateDiv or dsoHTSetSampleRateDiv
3	Channel On/Off	dsoHTAttribsCHEnable
4	Vertical Trigger Position	dsoHTAttribsVTriggerPos
5	Horizontal Trigger Position	dsoHTAttribsHTriggerPos
6	Bandwidth Limitations	dsoHTAttribsCHBW
7	Input Coupling: AC/DC	dsoHTAttribsCHCouple
8	Trigger Mode	dsoHTAttribsTriggerMode
9	Trigger Source	dsoHTAttribsTriggerSource
10	Trigger Rising Edge/Falling Edge	dsoHTAttribsTriggerSlop
11	Channel Vertical Position	dsoHTAttribsCHpos

Notice

In the final analysis, setting is to change the value stored in the hardware register, so all settings can be delivered repeatedly or when delivery is needed.

Acquire data - voltage value calculation

```
WORD pCHData[4][4096];//Application space for each channel is 4096 data
dsoHTGetDataWave(WORD nDeviceIndex,WORD pCHData[0],WORD
pCHData[1],WORD pCHData[2],WORD pCHData[3]);//Call function for data acquisition
//Assume the vertical position of channel 1 is 64;
short pSrcData[4][4096];//The length is the same as the length of pCHData to store the
data minus the reference position
WORD nPos[4]; //vertical position of channel
for(int i=0;i<4;i++)
{
for(int j=0;j<4096;i++)
{
pSrcData[i][j]= pCHData[i][j]- nPos[i];
}
}
```

Assume that the actual voltage value corresponding to the 1000th point of channel 1 is calculated.
Assume nPos[0]=64; pCHData[0][999]=50; And pSrcData[0][999]=-14; Assume the voltage range index value set by channel 1 is "3". Looking up Table 2, we find that "3" corresponds to 5mV. Then the voltage value at this point is -14/25*5mV=-2.8mV.

Therefore, the formula for calculating the actual voltage at the jth point of channel 1 is: (pCHData[i][j]- nPos[i])/25.0f*the voltage value corresponding to the index.

Note: The point to be calculated pCHData[i][j] must be between [1-254], otherwise the data exceeds the range and the calculation will definitely be incorrect.