SDK - HTHardDll.dll Manual

VC++ 6.0 IDE

Remarks:

HTHardDll.dll was compiled under VC++6.0..

WORD: unsigned short

BOOL: bool

ULONG: unsigned long

The following ifdef block is the standard way of creating macros which make exporting from a DLL simpler. All files within this DLL are compiled with the DLL_API symbol defined on the command line. This symbol should not be defined on any project that uses this DLL. This way any other project whose source files include this file see DLL_API functions as being imported from a DLL, wheras this DLL sees symbols defined with this macro as being exported.

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

Define _stdcall:

```
#define WIN_API __stdcall
```

Defines the struct for the application.

_HT_RELAY_CONTROL contains the relay control information.

```
typedef struct _HT_RELAY_CONTROL
{
    BOOL bCHEnable[MAX_CH_NUM];
    WORD nCHVoltDIV[MAX_CH_NUM];
    WORD nCHCoupling[MAX_CH_NUM];
    BOOL bCHBWLimit[MAX_CH_NUM];
    WORD nTrigSource;
    BOOL bTrigFilt;
    WORD nALT;
}RELAYCONTROL,*PRELAYCONTROL;
```

MAX_CH_NUM: Detail in DefMacro.h.

Parameters Remarks:

bCHEnable[MAX_CH_NUM]: Chanel Switch Array(Size to MAX_CH_NUM). Value: 1 ON, 0 OFF.

nCHVoltDIV[MAX_CH_NUM]: Chanel Voltage Array(Size to MAX_CH_NUM).

nCHCoupling[MAX_CH_NUM]: Chanel Couple Array(Size to MAX_CH_NUM). Value: 0 DC, 1 AC, 2 GND

bCHBWLimit[MAX_CH_NUM]:Chanel Band Limit Array(Size to MAX_CH_NUM). Value: 1 ON, 0 OFF.
nTrigSource: Trigger Source. 0 CH1, 1 CH2, 2 CH3, 3 CH4,5 EXT ,6 EXT/10.
bTrigFilt: High Frequency Rejection. Value: 1 ON, 0 OFF.
nALT: Whether is alternate. Value: 1 is alternate ,0 is non-alternate.
Example:

Declare a variable: RELAYCONTROL myRelayControl;
Declare a pointer: PRELAYCONTROL pRelayControl;

Functions

1. DLL_API WORD WINAPI dsoHTSearchDevice(short* pDevInfo)

Return Value:

The number of Having Connected devices..

Parmeter:

pDevInfo

Pointer to devices that have connected to PC.

Remarks:

You should call this function to know how many device that have connected to PC. One PC support 32 devices to be connected.

Example:

```
short DevInfo[32];
WORD nConnectedDevNum = 0;
//DevInfo Initial
//…
//Call this function
nConnectedDevNum = dsoHTSearchDevice(DevInfo);
```

When DevInfo[n] is 0 , there are devices. When DevInfo[n] is -1, there is no device.

2. DLL_API WORD dsoInitHard(WORD DeviceIndex)

Return Value:

0：Fail. Non 0: Succeed..

Parmeter:

nDeviceIndex

The device index.

Remarks:

You should call this function after connected

3. DLL_API WORD WINAPI dsoHTDeviceConnect(WORD nDeviceIndex)

Return Value:

1 : connect, 0: disconnect.

Parameter:

nDeviceIndex

The device index.

Remarks:

Whether the device is connected.

Example:

If there are two devices On PC, and you check whether to connect to the second device, you Need to do:

```
WORD nDeviceIndex = 1;     //0 : the first device
WORD nRe = 0;
//Call this function
nRe = dsoHTDeviceConnect(nDeviceIndex);
if (nRe = = 1)
    ;//the device si connected
else
    ;//the device is disconnected
```

4. DLL_API WORD WINAPI dsoGetFPGAVersion (WORD DeviceIndex)

Return Value:

FPGA Version

Parameter:

DeviceIndex

Index of the device.

Remarks:

get FPGA Version

Example:

```
WORD DeviceIndex = 0;
//…
WORD nFPGVersion = dsoGetFPGAVersion (DeviceIndex)
```

5. DLL_API WORD WINAPI dsoHTSetCHPos(

```
WORD nDeviceIndex,
WORD* pLevel,
WORD nVoltDIV,
WORD nPos,
WORD nCH
WORD nCHMode
)
```

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

nDeviceIndex

Index of the device.

pLevel

Pointer to calibration level.

nVoltDIV

Index of channel voltage.

nPos

The position of chnanel, range :0~255.

nCH

The targer channel.0 CH1,1 CH2,2 CH3 ,3 CH4.

nCHMode

channel working mode(1,2,4)

Remarts:

Set channel level.

Example:

```
WORD nDeviceIndex = 0;
WORD CHLevel[288]; //the calibration level, See in dsoHTReadCalibrationData.
WORD nVoltDIV = 6; //voltage 1V/div, index is 6.
WORD nPos = 128; //Zero level is 128.
WORD nCH = 0; //CH1.
WORD nCHMod=4;
//Call this function.
if ( 0 = = dsoHTSetCHPos(nDeviceIndex,CHLevel,nVoltDIV,nPos,nCH,nCHMod) )
    ;//Fail
else
    ;//Succeed
```

6. DLL_API WORD WINAPI dsoHTSetVTriggerLevel(

WORD nDeviceIndex,

WORD nPos

WORD nSensitivity)

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

nPos

The position of the trigger level.

nSensitivity

Trigger Sensitivity,if the lines is thick try it bigger

Remarks:

Set the position of the trigger level.

Exmaple:

```
WORD nDeviceIndex = 0;
WORD nPos = 128;
//call this function
if ( 0 = = dsoHTSetVTriggerLevel( nDeviceIndex, nPos, 4) )
        ; //Fail
else
        ; //Succeed
```

7. DLL_API WORD WINAPI dsoHTSetHTriggerLength(
WORD nDeviceIndex,
PCONTROLDATA pControl,
WORD nCHMod
)

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

pControl

Pointer to Struct PCONTROLDATA, read "HTSoftDll.h" for detail.

nCHMode

Channel working mode(1,2,4).

Remarks:

Set trigger length.

8. DLL_API WORD WINAPI dsoHTSetCHAndTrigger(
WORD nDeviceIndex,
RELAYCONTROL RelayControl,
WORD nTimeDIV)

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

RelayControl

See in the structure HT_RELAY_CONTROL.

nTimeDIV

The index of time base 0-35.

Remarks:

Set channel and trigger.

Example:

```
WORD nDeviceIndex = 0;
RELAYCONTROL RelayControl;
WORD nTimeDIV=12;
//RelayControl
//Call this function
```

```
if ( 0 = = dsoHTSetCHAndTrigger(nDeviceIndex,RelayControl,nTimeDIV) )
    ; //Fail
else
    ; //succeed
```

9. DLL_API WORD WINAPI dsoHTSetSampleRate(
WORD nDeviceIndex,
WORD *pAmpLevel,
WORD nYTFormat,
PRELAYCONTROL pRelayControl,
PCONTROLDATA pControl))

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

pAmpLevel

Pointer to amplitude calibration.

nYTFormat

The mode of horizontal format. 0: Normal, 1: Scan, 2:Roll.

pRelayControl

See in the structure HT_RELAY_CONTROL.

pControl

Pointer to Struct PCONTROLDATA, read "HTSoftDll.h" for detail.

Remarks:

Set Sampling Rate.

10. DLL_API WORD dsoHTSetSampleRateVi(WORD nDeviceIndex,
WORD *pAmpLevel,
WORD* pCHEnable,
WORD* pCHVoltDIV,
WORD* pCHCoupling,
WORD* pCHBWLimit,
WORD nTriggerSource,
WORD nTriggerFilt,
WORD nALT,
PCONTROLDATA pControl);

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

pAmpLevel,

Pointer to amplitude calibration.

pCHVoltDIV

See in the structure HT_RELAY_CONTROL.nCHVoltDIV.

pCHEnable

See in the structure HT_RELAY_CONTROL.bCHEnable.

pCHCoupling

See in the structure HT_RELAY_CONTROL.nCHCoupling.

pCHBWLimit

See in the structure HT_RELAY_CONTROL.bCHBWLimit.

nTriggerSource, nTriggerFilt,nALT

pControl

Pointer to Struct PCONTROLDATA, read "HTSoftDll.h" for detail.

Remarks:

transform of dsoHTSetSampleRate

11. DLL_API WORD WINAPI dsoHTStartCollectData(WORD nDeviceIndex
WORD nStartControl)

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

nDeviceIndex

The index of the device.

nStartControl

8bit use index 0-2 3bit

0:1 AUTO Trigger

1:1 ROLL Mode

2:1 stop after this collect

Remarks:

This inform the device start collecting.

Example:

```
WORD nDeviceIndex = 0;
WORD nStartControl=1;
if (0 = = dsoHTStartCollectData(nDeviceIndex, nStartControl) )
    ; //Fail
else
    ; //Succeed
```

12. DLL_API WORD WINAPI dsoHTGetState(WORD nDeviceIndex)

Return value:

8bit use index 0-1 2bit

0:if triggered this bit is 1

1:if date collection is finished this bit is 1

Parameter:

nDeviceIndex

The index of the device.

Remarks:

Retrieve the collect state of the device.

Example:

```
WORD nDeviceIndex = 0;
while ( dsoHTGeState(nDeviceIndex) != 0x02)
{
    continue;
}
//Read data from the device.
```

13. DLL_API WORD WINAPI dsoHTGetData (

```
WORD nDeviceIndex,
WORD* pCH1Data,
WORD* pCH2Data,
WORD* pCH3Data,
WORD* pCH4Data,
PCONTROLDATA pControl,
);
```

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

nDeviceIndex

The index of the device.

pCH1Data

Pointer to CH1 data buffer.

pCH2Data

Pointer to CH2 data buffer.

pCH3Data

Pointer to CH3 data buffer.

pCH4Data

Pointer to CH4 data buffer.

pControl

Pointer to Struct PCONTROLDATA, read "HTSoftDll.h" for detail.

Remarks:

Read the Normal Mode data to PC buffer from the device.

Example:

```
WORD nDeviceIndex = 0;
if ( 0 = = dsoSDGetData ( nDeviceIndex, pCH1Data, pCH2Data, pCH3Data, pCH4Data
                          , pControl) )
    ; //Fail
else
```

; //Succeed

14. DLL_API WORD dsoHTGetScanData(WORD nDeviceIndex,
WORD* pCH1Data,
WORD* pCH2Data,
WORD* pCH3Data,
WORD* pCH4Data,
PCONTROLDATA pControl);

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

nDeviceIndex

The index of the device.

pCH1Data

Pointer to CH1 data buffer.

pCH2Data

Pointer to CH2 data buffer.

pCH3Data

Pointer to CH3 data buffer.

pCH4Data

Pointer to CH4 data buffer.

pControl

Pointer to Struct PCONTROLDATA, read "HTSoftDll.h" for detail.

Remarks:

Read the SCAN Mode data to PC buffer from the device.

15. DLL_API WORD dsoHTGetRollData(WORD nDeviceIndex,
WORD* pCH1Data,
WORD* pCH2Data,
WORD* pCH3Data,
WORD* pCH4Data,
PCONTROLDATA pControl)

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

nDeviceIndex

The index of the device.

pCH1Data

Pointer to CH1 data buffer.

pCH2Data

Pointer to CH2 data buffer.

pCH3Data

Pointer to CH3 data buffer.

pCH4Data

Pointer to CH4 data buffer.

pControl

Pointer to Struct PCONTROLDATA, read "HTSoftDll.h" for detail.

Remarks:

Read the ROLL Mode data to PC buffer from the device.

16. DLL_API ULONG WINAPI dsoHTGetHardFC(WORD nDeviceIndex)

Return value:

nIndata ,Retrieve the hardware counter value.

Parameter:

nDeviceIndex

The index of the device.

Remarks:

frequency= nIndata*1E9/(8* nTime).

17. DLL_API WORD WINAPI dsoHTSetHardFC (WORD nDeviceIndex,
ULONG nTime
)

Return value:

0：Fail. Non 0: Succeed.

Parameter:

nDeviceIndex

The index of the device.

nTime

set the interval (uint ns) during after which you can call dsoHTGetHardFC.We recomment 0.1S-1S,

18. DLL_API WORD WINAPI dsoHTWriteCalibrationData (
WORD DeviceIndex,
WORD* pLevel,
WORD nLen
)

19. DLL_API WORD WINAPI dsoHTReadCalibrationData (WORD nDeviceIndex,
WORD* pLevel,
WORD nLen
)

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

pLevel nType

Pointer to buffer that save the device calibration data.

nLen

The calibration buffer size.

Remarks:

The function must be call when read/write zero calibrate.

20. DLL_API BOOL WINAPI dsoSetUSBBus (WORD nDeviceIndex)

Return value:

FALSE: Fail , TRUE : Succeed

Parameter:

nDeviceIndex

The index of the device.

Remarks:

call this function firstly after connect

21. DLL_API WORD WINAPI dsoHTRDAmpCali(WORD nDeviceIndex,
WORD * pLevel,
WORD nLen)

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

pLevel：

Pointer to amp calibration level.

nLen

Length of pLevel

Remarks:

dsoHTRDAmpCali read read amplitude calibrate

22. DLL_API WORD dsoHTSetCHAndTriggerVB(WORD nDeviceIndex,
WORD* pCHEnable,
WORD* pCHVoltDIV,
WORD* pCHCoupling,
WORD* pCHBWLimit,
WORD nTriggerSource,
WORD nTriggerFilt,
WORD nALT,
WORD nTimeDIV);

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

pCHVoltDIV

See in the structure HT_RELAY_CONTROL.nCHVoltDIV.

pCHEnable

See in the structure HT_RELAY_CONTROL.bCHEnable.

pCHCoupling

See in the structure HT_RELAY_CONTROL.nCHCoupling.

pCHBWLimit

See in the structure HT_RELAY_CONTROL.bCHBWLimit.

nTriggerSource,nTriggerFilt,nALT

see struct RELAYCONTROL

The index of time base 0-35.。

Remarks:

transform of dsoHTSetCHAndTrigger.

23. DLL_API WORD ddsSetOnOff(WORD DeviceIndex,short nOnOff)

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

nOnOff

0 for Open and 1 for Close

Remarks:

DDS switch

24. DLL_API WORD ddsSetCmd(WORD DeviceIndex, USHORT nSingle)

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

nSingle

Set it to 1 will get single wave and 0 get continual wave

Remarks:

Set output wave form

25. DLL_API WORD ddsEmitSingle(WORD DeviceIndex)

Return Value:

0：Fail. Non 0: Succeed.

Remarks:

If you set nSingle to 1 in function ddsSetCmd you should call this function to gennerate wave

26. DLL_API WORD ddsSetFrequency(WORD DeviceIndex,
double dbFre,
WORD* pWaveNum,
WORD* pPeriodNum)

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

dbFre

The frequency of wave

pWaveNum

Actually this parameter is a return value when you call ddsDownload

pPeriodNum

Actually this parameter is a return value when you call ddsDownload

Remarks:

set wave frenquency

27. DLL_API WORD ddsDownload(WORD DeviceIndex,
WORD iWaveNum,
WORD* pData)

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

iWaveNum

come from function ddsSetFrequency.

pData

Pointer to wave pots array you will send to the Device

Remarks:

Send wave data

28. DLL_API WORD dsoInitADCOnce(WORD DeviceIndex)

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

Remarks:

Call this function after dsoSetUSBBus

29. DLL_API WORD dsoHTADCCHModGain(WORD DeviceIndex,
WORD nCHMod)

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

nCHMod

Channel working mode(1,2,4).

Remarks:

call this function when channel mode changge

30. DLL_API WORD dsoHTSetAmpCalibrate(WORD nDeviceIndex,
WORD nCHSet,
WORD nTimeDIV,
WORD *pLevel,
WORD *nVoltDiv,
WORD *pCHPos)

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

nCHSet

same as PCONTROLDATA->nCHSet.

nTimeDIV

The index of time base 0-35.

pLevel

Pointer to amplitude calibration.。

nVoltDiv

See in the structure HT_RELAY_CONTROL.nCHVoltDIV.

pCHPos

Ponter to 4 lenth array which store all channels vertical position

Remarks:

Amplitude calibration

31. DLL_API WORD dsoHTSetRamAndTrigerControl(WORD DeviceIndex,
WORD nTimeDiv,
WORD nCHset,
WORD nTrigerSource,
WORD nPeak)

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

nTimeDiv

The index of time base 0-35.

nCHSet

same as PCONTROLDATA->nCHSet.nTimeDIV

nTrigerSource

Trigger Source 0-3

nPeak

Set it to 1 to open peak collection

Remarks:

Set Trigger Source

32. DLL_API WORD dsoHTSetTrigerMode(WORD m_nDeviceIndex,
WORD nTriggerMode,
WORD nTriggerSlop,
WORD nTriggerCouple)

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

nTriggerMode

Trigger mode .0:Edge 1: Pulse 2: Video

nTriggerSlop

Trigger slope 0 :increasing slope 1:decreasing slope

nTriggerCouple

Triger couple 0：DC 1：AC 2:low-frequency suppression 3:high-frequency suppression 4:Noise Suppression

Remarks:

Set trigger mode

33. DLL_API WORD WINAPI dsoHTSetVideoTriger(WORD m_nDeviceIndex,
USHORT nStand,
USHORT nVedioSyncSelect,
USHORT nVideoHsyncNumOption,
USHORT nVideoPositive,
WORD nLevel,
WORD nLogicTriggerSource)

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

nStand

Analog television system 0:PALSECAM 1:NTSC。

nVedioSyncSelect

The video sync.

Value:

All Lines: 0

Line Num: 1

Odd Field: 2

Even Field: 3

All Field: 4

nVideoHsyncNumOption,

The video line number, the min value is 1.

nVideoPositive

Video Polarity 0:Positive 1:Negative

nLevel

Trigger level

nLogicTriggerSource

Logical Trigger source

Remarks:

Call this function after dsoHTSetTrigerMode when trigger mode is VIDEO

34. DLL_API WORD WINAPI dsoHTSetPulseTriger(WORD m_nDeviceIndex,
ULONG nPW,
WORD nPWCondition)

Return Value:

0：Fail. Non 0: Succeed.

Parameter:

nPW

The pulse width value. The time range of the pulse width is 10ns~10s, the pulse width value range is 2 ~ 2000000000（10ns/5 ~ 10000000000ns/5）

nPWCondition

The condition fo the pulse trigger

Value :

Equal 0 .

NoEqual 1 .

More 2 .

Less 3 .

Remarks:

Call this function after dsoHTSetTrigerMode when trigger mode is PULSE

Flow Chart:

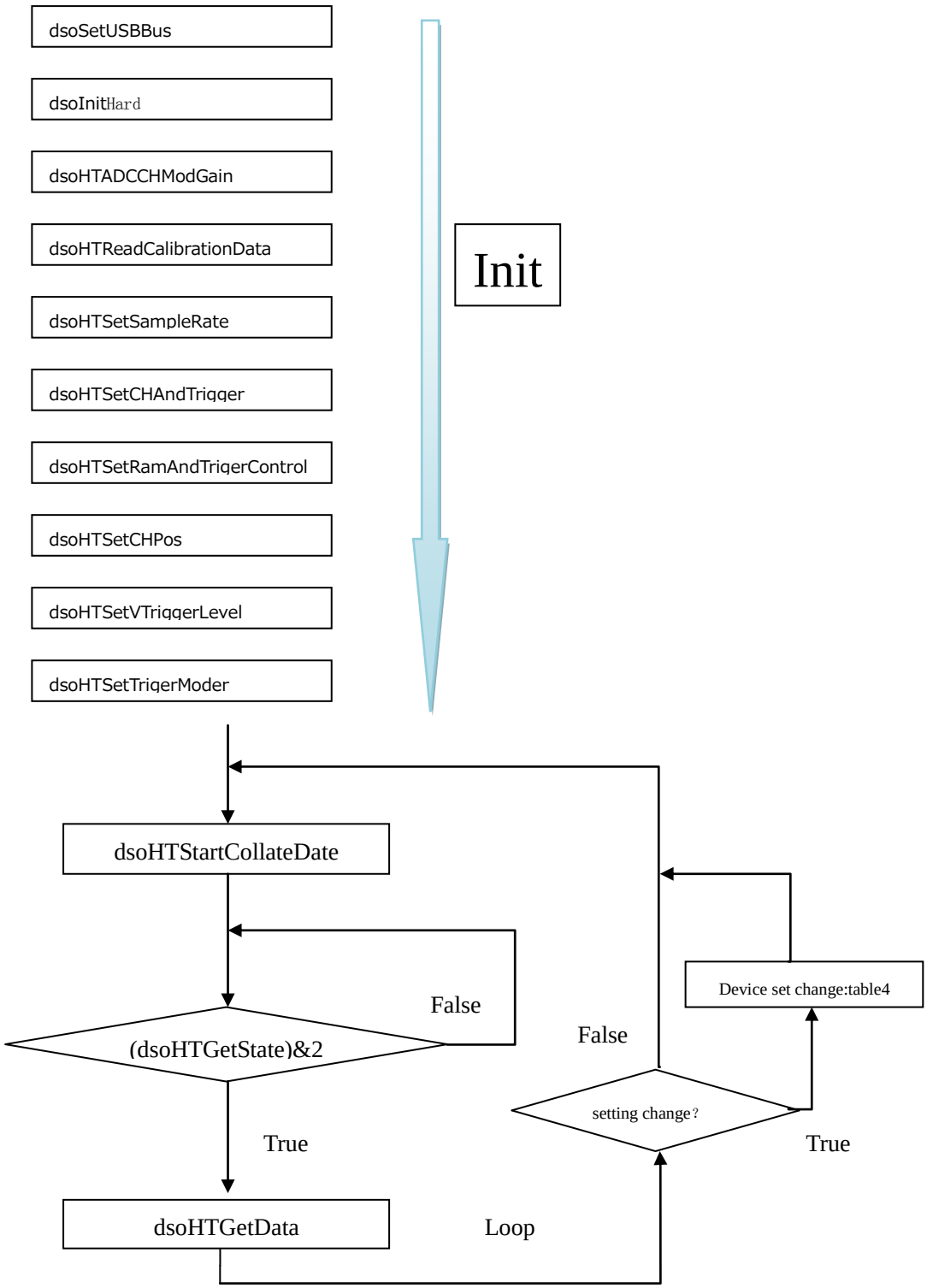
dsoSetUSBBus
dsoInitHard
dsoHTADCCHModGain
dsoHTReadCalibrationData
dsoHTSetSampleRate
dsoHTSetCHAndTrigger
dsoHTSetRamAndTrigerControl
dsoHTSetCHPos
dsoHTSetVTriggerLevel
dsoHTSetTrigerModer
Init
dsoHTStartCollateDate
(dsoHTGetState)&2
False
True
dsoHTGetData
Loop
setting change?
False
True
Device set change:table4

Table 1：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 双: 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 four-channel mode. About channel mode shown in Table 2: timebase and channel mode

Table 2：timebase and channel mode

Index	TimeBase	number of channel enabled / channel mode
0	2nS	1/S；2/D；3、4/Q
…	…	1/S；2/D；3、4/Q
6	200nS	1/S；2/D；3、4/Q
7	500nS	1、2/D；3、4/Q
8	1uS	*/Q
…	…	*/Q
35	1000S	*/Q

Note:

1 For example when time base is 200nS, Single channel enabled will be the single-channel mode; two for dual channel mode; three and four for four-channel mode

Table 3：Range of each voltage base

Index	voltage base	Range	Index	voltage base	Range
0	2mV	16mV	6	200mV	1.6V
1	5mV	40mV	7	500mV	4V
2	10mV	80mV	8	1V	8V
3	20mV	160mV	9	2V	16V
4	50mV	400mV	10	5V	40V
5	100mV	800mV	11	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 4：Common oscilloscope settings

Index	Setting	Functions
1	Voltage DIV	dsoHTSetCHAndTrigger dsoHTSetAmpCalibrate：call if channel mode has changged
2	Time DIV	dsoHTSetSampleRate dsoHTSetRamAndTrigerControl SetADCCHModGain：call if channel mode has changged SetAmpCalibrate：call if channel mode has changged
3	Channel enable/disable	dsoHTSetCHAndTrigger SetADCCHModGain：call if channel mode has changged SetAmpCalibrate：call if channel mode has changged
4	Vertical trigger position	dsoHTSetVTriggerLevel
5	Horizontal trigger position	dsoHTSetHTriggerLength
6	Bandwidth limitations	dsoHTSetCHAndTrigger
7	input coupling: AC /DC	dsoHTSetCHAndTrigger
8	Trigger Mode	dsoHTSetTrigerMode dsoHTSetVideoTriger：call if Trigger Mode is Video dsoHTSetPulseTriger：call if Trigger Mode is Pulse
9	Trigger source	dsoHTSetRamAndTrigerControl
10	Trigger slope Rise / Fall	dsoHTSetTrigerMode