

SDK - HTHardDll.dll Manual

VB 6.0 IDE

Note:

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, whereas 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[4];
    WORD nCHVoltDIV[4];
    WORD nCHCoupling[4];
    BOOL bCHBWLlimit[4];
    WORD nTrigSource;
    BOOL bTrigFilt;
    WORD nALT;
}RELAYCONTROL,*PRELAYCONTROL;
```

Parameters Remarks:

`bCHEnable[4]`: The channel is enable/disable. Value: 1 is enable; 0 is disable.

`nCHVoltDIV[4]`: The VOLT/DIV index of the channel.

`nCHCoupling[4]`: The coupling index of the channel .

Value: DC is 0; AC is 1; GND is 2;

bCHBWLimit[4]: The bandwidth of the channel.

Value: be enable is 1; be disable is 0.

nTrigSource: The index of the trigger source. If the channel has 4 channels, the value is
: CH1 is 0, CH2 is 1, CH3 is 2,

bTrigFilt: The High Frequency Rejection. Be enable is 1, disable is 0.

nALT: If the trigger is alternate trigger, the value is 1, otherwise is 0.

Example:

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

_HT_CONTROL_DATA contains some controls.

```
typedef struct _HT_CONTROL_DATA
{
    WORD nCHSet; //Channel Enable/Disable
                //The 0 bit: 0 is enable, 1 is disable.
                // The 1 bit: 0 is enable, 1 is disable.
                // The 2 bit: 0 is enable, 1 is disable.
                // The 3 bit: 0 is enable, 1 is disable.
    WORD nTimeDIV; //The index of time Base
    WORD nTriggerSource; // The index of the trigger source
    WORD nHTriggerPos; //The Horizontal Trigger Pos (Value: 0 ~ 100)
    WORD nVTriggerPos; //The Vertical Trigger Pos (Value: 0 ~ 255)
    WORD nTriggerSlope; //The Edge trigger Slope (Rise slope is 0,Fall slope is 1)
    ULONG nBufferLen; //The buffer Length
    ULONG nReadDataLen; //The Data Length of have be read.
    WORD nALT; // Is alternate trigger or not

}CONTROLDATA,*PCONTROLDATA;
```

Example:

```
Declare a variable: CONTROLDATA    myControlData;
Declare a pointer: PCONTROLDATA    pControlData;
```

Functions:

1. DLL_API WORD WINAPI dsoHTSearchDevice(short* pDevInfo)

Return Value:

The number of have connected devices.

Parameters:

pDevInfo

pointer to devices information that have connected to PC.

Remarks:

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

Example:

```
short DevInfo[32];
WORD nConnectedDevNum = 0;
//DevInfo Initial....
//...
//Call this function
nConnectedDevNum = dsoHTSearchDevice(DevInfo);
if (nConnectedDevNum > 0)
{
    For(int i=0 ;i<32 i++)
    {
        if (DevInfo[i] == 0)
        {
            // has device. The device is not used.
        }
        else if (DevInfo[i] == 1)
        {
            // has device. The device is used.
        }
        else
        {
            //No device was found.
        }
    }
}
```

2. DLL_API WORD WINAPI dsoHTDeviceConnect(WORD nDeviceIndex)

Return value:

If the device was connected is 1, otherwise is 0.

Parameters:

nDeviceIndex

The device index.

Remarks:

Whether the device was connected.

Example:

```
WORD nDeviceIndex = 0;
WORD nRe = 0;
```

```

//Call the function
nRe = dsoHTDeviceConnect(nDeviceIndex);
if (nRe == 1)
    ;//Connected
else
    ;//Not Connected

```

3. DLL_API WORD WINAPI dsoHTSetCHPos(

```

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

```

Return Value:

If the function succeeds, the return value is zero.

If the function fails, the return value is zero.

Parameters:

nDeviceIndex

index of the device

pLevel

pointer to calibration level.

nVoltDIV

index of channel

nPos

the position of channel, range: 0 ~ 255 (8bit)

nCH

the target channel (CH1 is 0, CH2 is 1, CH3 is 2.)

Remarks:

Set channel level (Zero Level)

Example:

```

WORD nDeviceIndex = 0;
WORD CHLevel[256]; // the calibration level, see dsoHTReadCalibrationData function
WORD nVoltDIV = 6; //
WORD nPos = 128; //
WORD nCH = 0; //
//
if ( 0 == dsoHTSetCHPos(nDeviceIndex,CHLevel,nVoltDIV,nPos,nCH) )
    ;//Fail
else
    ;//Succeed

```

4. DLL_API WORD WINAPI dsoHTSetVTriggerLevel (
WORD nDeviceIndex,
WORD* pLevel,
WORD nPos)

Return value:

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero.

Parameters:

nDeviceIndex

index of current device

pLevel

pointer to calibration level

nPos

the position of the trigger level, range : 0 ~ 255 (8bit)

Remarks:

Set the position of the trigger level.

Example:

```
WORD nDeviceIndex = 0;
```

```
WORD CHLevel[256]; //the calibration level, see dsoHTReadCalibrationData function
```

```
WORD nPos = 128;
```

```
//call function
```

```
if ( 0 == dsoHTSetVTriggerLevel( nDeviceIndex, CHLevel, nPos) )
```

```
    ; //Fail
```

```
else
```

```
    ; //Succeed
```

5. DLL_API WORD WINAPI dsoHTSetHTriggerLength(
WORD nDeviceIndex,
ULONG nBufferLen,
WORD HTriggerPos,
WORD nTimeDIV,
,WORD nYTFormat
)

Return value:

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero.

Parameters:

nDeviceIndex

index of current device.

nBufferLen

the buffer length

HTriggerPos

the horizontal trigger position(percent), range : 0 ~ 100

nTimeDIV

index of the TIME/IDV

nYTFormat

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

Remarks:

Set the trigger length

Example:

```
WORD nDeviceIndex = 0;
ULONG nBufferLen = 10240;//10K
WORD HTriggerPos = 50; //50%
WORD nTimeDIV = 12; //40us
//Call function
if ( 0 == dsoHTSetHTriggerLength(nDeviceIndex,nBufferLen,HTriggerPos,
                                nTimeDIV) )
    ; //Fail
else
    ; //Succeed
```

6. DLL_API WORD WINAPI dsoHTSetCHAndTriggerVB (WORD nDeviceIndex,
WORD* pCHEnable,
WORD* pCHVoltDIV,
WORD* pCHCoupling,
WORD* pCHBWLlimit,
WORD nTriggerSource,
WORD nTriggerFilt,
WORD nALT
);

Return value:

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero.

Parameters:

nDeviceIndex

index of current device

WORD CHEnable[4];//CH enable/sisable. 0 is Enable, 1 is disable.

WORD CHVoltDIV[4];//index of CH VOLT/DIV

WORD CHCoupling[4];//channel coupling

WORD CHBWLlimit[4];//channel bandwidth enable/disable

WORD nTrigSource;// the trigger source

WORD bTrigFilt;// the high frequency rejection

WORD nALT;// Is alternate trigger or not. 0 is ALT, otherwise is 1.

Remarks

Set channel and trigger

Example:

```
WORD nDeviceIndex = 0;
WORD CHEnable[4];
WORD CHVoltDIV[4];
WORD CHCoupling[4];
WORD CHBWLimit[4];
WORD nTriggerSource = 0;
WORD nTriggerFilt = 0;
WORD nALT = 0;
//Initial array
//.....
//Call function
if ( 0 == dsoHTSetCHAndTrigger(nDeviceIndex,
    CHEnable,
    CHVoltDIV,
    CHCoupling,
    CHBWLimit,
    nTriggerSource,
    nTriggerFilt,
    nALT) )
    ; //Fail
else
    ; //Succeed
```

```
7. DLL_API WORD WINAPI dsoHTSetTriggerAndSyncOutput(
    WORD nDeviceIndex,
    WORD nTriggerMode,
    WORD nTriggerSlope,
    WORD nPWCondition,
    ULONG nPW,
    USHORT nVideoStandard,
    USHORT nVedioSyncSelect,
    USHORT nVideoHsyncNumOption,
    WORD nSync
)
```

Return value:

If the function succeeds, the value is nonzero.

If the function fails, the value is zero.

Parameters:

nDeviceIndex

index of current device

nTriggerMode

the mode of the trigger. Edge trigger is 0, Pulse Trigger is 1.

nTriggerSlope

the slope of the edge trigger, Rise slope is 0, Fall slope is 1.

nPWCondition

the condition of the pulse trigger

value:

+ Less is 0

+ Equal is 1

+ More is 2

- Less is 3

- Equal is 4

- More is 5

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)

nVideoStandard

Set video standard, value : 0 is PAL/SECAM ,1 is NTSC

nVedioSyncSelect

Set video sync, value:

All Lines: 0

All Field: 1

Odd Field: 2

Even Field; 3

Line Num: 4

nVideoHsyncNumOption

Set video line number, the min value is 1.

nSync

The synchro output is enable or disable. 1 is enable, 0 is disable.

Remarks:

set trigger

Exmample:

```
WORD nDeviceIndex = 0;
WORD nTriggerMode = 0; //Edge Trigger
WORD nTriggerSlope = 0; //Rise Slope
WORD nPWCondition = 0; //
ULONG nPW = 2; //10ns, nPW = 10ns/5。
WORD nSync = 0; //Closed
//Call function
if (0 == dsoHTSetTriggerAndSyncOutput(nDeviceIndex, nTriggerMode,
                                     nTriggerSlope, nTriggerCondition, nPW, nSync) )
    ; //Fail
else
    ; //succeed
```



```
; //Succeed
```

10. DLL_API WORD WINAPI dsoHTStartTrigger(WORD nDeviceIndex)

Return value:

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero.

Parameters:

nDeviceIndex

index of the current device

Remarks:

Notify the device to detect trigger signal.

Example:

```
WORD nDeviceIndex = 0;
if (0 == dsoHTStartTrigger(nDeviceIndex) )
    ; //Fail
else
    ; //Succeed
```

11. DLL_API WORD WINAPI dsoHTForceTrigger(WORD nDeviceIndex)

Return value:

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero.

Parameters:

nDeviceIndex

index of the device

Remarks:

Force to trigger the signal

Example:

```
WORD nDeviceIndex = 0;
if (0 == dsoHTForceTrigger(nDeviceIndex) )
    ; //Fail
else
    ; //Succeed
```

12. DLL_API WORD WINAPI dsoHTGetState(WORD nDeviceIndex)

Return value:

If the device prepares to collect data, the return value is 1.

If the device is detecting trigger signal, the return value is 3.

If the device has collected data, the return value is 7.

You can read the data only the return value is 7.

Parameters:

nDeviceIndex

Index of the device

Remarks:

Retrieve the collect state of the device.

Example:

```
WORD nDeviceIndex = 0;
while ( dsoHTGeState(nDeviceIndex) != 0x07)
{
    continue;
}
//Read data from the device
```

13. DLL_API WORD WINAPI dsoSDGetData (

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

Return value:

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero.

Parameters:

nDeviceIndex

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

Set _HT_CONTROL_DATA

nInsertMode

the mode of interpolation, value: 0 is step, 1 is line, 2 is Sinx/x.

the parameter is used for the 4 minimum TIME/DIV

Remarks:

Read the data to PC buffer from the device.

Example:

```
WORD nDeviceIndex = 0;
WORD nInsertMode = 2; //Sinx /x
if ( 0 == dsoSDGetData ( nDeviceIndex, pCH1Data, pCH2Data, pCH3Data, pCH4Data
                        , pControl, nInsertMode) )
    ; //Fail
else
    ; //Succeed
```

14. DLL_API ULONG WINAPI dsoHTGetHardFC(WORD nDeviceIndex)**Return value:**

Retrieve the hardware counter value.

Parameters:

nDeviceIndex

index of the device

Remarks:

If the counter type is frequency, the return value is frequency value (Units:Hz), otherwise the value is the trigger signal counter.

**15. DLL_API WORD WINAPI dsoHTSetHardFC (WORD nDeviceIndex,
WORD nType
)****Return value:**

Set the hardware counter type.

Parameters:

nDeviceIndex

index of the device

nType

If the value is 0, it opens the hardware frequency counter.

If the value is 1, it opens the hardware trigger signal counter.

If the value is 2, it closes the hardware counter.

**16. DLL_API WORD WINAPI dsoHTReadCalibrationData (WORD nDeviceIndex,
WORD* pLevel,
WORD nLen
)****Return value:**

If the function succeeds, the return value is 1.

If the function fails, the return value is zero.

Parameters:

nDeviceIndex

index of the device

pLevel

pointer to buffer, save the device calibration data
nLen
the calibration buffer size

Remarks:

This function must be called when initializing the device.

17. DLL_API BOOL WINAPI dsoSetUSBBus (WORD nDeviceIndex)

Return value:

If the function succeeds, the return value is TRUE.

If the function fails, the return value is FALSE.

Parameters:

nDeviceIndex
index of the device

Remarks:

If you change to USB communication, you must call this function firstly.

Flow-Chart:

