Programmer’s Guide

DSO1200

Content

1 What's the DSO1200DLL.DLL?

The DSO1200DLL.DLL is a dynamic-link library for Windows OS. It provides several function calls to control the DSO1200. You may use some language that support DLL link function, such as Visual C++, Visual Basic or Labview to control DSO1200 with DSO1200DLL.DLL library. Here, we illustrate some examples using Visual C++, Visual Basic and Labview. The other languages please refer to their description about DLL link application.

2 Defination

2.1 Data Types

2.1.1 BOOLEAN

Boolean variable (1: TRUE, 0:FALSE).

2.1.2 INT8U

8-bit unsigned integer

2.1.3 INT8S

8-bit signed integer

2.1.4 INT16U

16-bit unsigned integer

2.1.5 INT16S

16-bit signed integer

2.1.6 INT32U

32-bit unsigned integer

2.1.7 INT32S

32-bit signed integer

2.1.8 FP32

Single float point variable, 4 bits

2.1.9 FP64

Double float point variable, 8 bits

2.1.10 HTSTATUS

32-bit unsigned integer

Function return status. 0: Success, 1:Error:

2.1.11 HT_DEVICE_ID

32-bit unsigned integer
0~15: the index of machine. The machine which connected to PC first will be set as 0, the second is 1, and so on.
16~31: the communiate mode. 0: USB, 1:COM, 2:LAN.

2.2 Struct

2.2.1 HT_DEVICE_INFO

The device Information
typedef struct _HT_DEVICE_INFO
{
    char szName[20];
    char szSerial[20];
    INT16U iFirmVersion;
    INT16U iHardVersion[4];
    INT32U iDate;
} HT_DEVICE_INFO,*PHT_DEVICE_INFO;

szName[20]
The name of the machine, always it is "DSO1200"
szSerial[20]
The serial number of machine, which match the number behined the machine.
iFirmVersion
The firmware's version
iHardVersion[4]
the hardware verison,
HardVersion[0]-PCB version
HardVersion[1]-mcu 1# version
HardVersion[2]-mcu 2# version
HardVersion[3]-Lan version.
iDate
Factory data, 0~7 bits: day, 8~15 bits: month, 16~31 bits: year.
For example: 0x07D90102 must be 2009 year 1 month 2 day.

2.2.2 TRIGGER

The trigger system
typedef struct _TRIGGER
{
    INT16U Mode;

```
    INT16U Source;
    INT16U Sweep;
    INT16U Slope;
    INT16U HFReject;
    INT16U PWCondition;
    INT32U pluseValue;
}TRIGGER;
```

Mode
Trigger mode: 0:Edge trigger, 1:pulse trigger
Source
Trigger source: 0:CH1, 1: CH2
Sweep
Trigger sweep: 0: Auto, 1:Normal, 2:Single
Slope
Trigger Slope: 0: Rising, 1:Falling
HFReject
HF Reject: 0: OFF, 1: ON
PWCondition
Pulse when condition: 0: +Less, 1: +Equal, 2: +More, 3: -Less, 4: -Equal, 5: -More
pluseValue
Pulse setting value: the unit is ns/5, for example: if the value is 1000, the pulse setting is 5ms.

2.2.3 MATH

The math parameters

```
typedef struct _MATH
{
    INT16U    MathOperate;
    INT16U    MathSource[2];
    INT16U    fftWindow;
    INT16U    fftSource;
    INT16U    fftScale;
    INT16U    fftSplite;
    INT16U    fftDB;
}MATH;
```

MathOperate
Math operator: 0: +, 1: -, 2: *, 3: /, 4: FFT
MathSource[2]
The two sources of the math, 0:CH1, 1:CH2
fftWindow
FFT Window, 0:Rectangle, 1:Hanning, 2:Hamming, 3:Blackman
fftSource
FFT Source, 0: CH1, 1:CH2

fftScale
FFT Scale, 0:Vrms, 1:dBVrms
fftSplite
FFT Splite, 0:normal, 1:splite window
fftDB
db index in the array, {1, 2, 5, 10, 20, 50, 100} (dB/div)

2.2.4 CHANNEL

The channel's parameters

```
typedef struct _CHANNEL
{
    INT16U      enable;
    INT16U      voltDivIndex;
    INT16U      couple;
    INT16U      probe;
    INT16U      invert;
    INT16U      BW20M;
    INT16U      BW100M;
    INT32U      voltDivValue;
    INT16S      vertPos;
    INT16S      vTrigPos;
    INT16U      nData;
    INT16S      horiPos;
    INT16U      hTriggerPos;
    INT16U      timebase;
    INT16S      voltDivType;
    INT16U      bHasEmpty;
    INT16S      emptyPos[2];
}CHANNEL;
```

enable
Show/hide the channel on the screen, 0:hide, 1:show
voltDivIndex
0: 5.00mV, 1: 10.0mV,2: 20.0mV, 3: 50.0mV, 4: 100mV,5: 200mV, 6: 500mV, 7: 1V, 8: 2V, 9: 5V,
couple
Channel's couple: 0: AC, 1: DC, 2: GND
probe
Channel's probe: 0: 1X, 1: 10X, 2: 100X, 3: 1000X
invert
Channel's Invert: 0: OFF, 1: ON
BW20M
20M Band Width limit: 0: OFF, 1: ON
BW100M
100M Band Width limit: 0: OFF, 1: ON.

voltDivValue
the 100 times of the voltage. For example: if the voltDiv is 123, the display string will be "1.23mV"
vertPos
The channel’s position displayed on the screen, the top is 0, and bottom is 199
vTrigPos
The channel's trigger postion displayed on the screen, the top is 0, and the bottom is 199.
nData
The number of data acquired from the hardware
horiPos
The channel's horizontal trigger position displayed on the screen, the left is 0, and the right is 299
hTriggerPos
The channel's horizontal trigger position in the hardware, 0 ~ 18383(16K).
timebase
The channel's timebase, 0: 2.000ns, 1: 5.000ns, 2: 10.00ns3: 20.00ns, 4: 50.00ns, 5: 100.0ns, 6: 200.0ns,7: 500.0ns, 8: 1.000us, 9: 2.000us, 10: 5.000us, 11: 10.00us, 12: 20.00us, 13: 50.00us, 14: 100.0us, 15: 200.0us, 16: 500.0us, 17: 1.000ms, 18: 2.000ms, 19: 5.000ms, 20: 10.00ms, 21: 20.00ms, 22: 50.00ms, 23: 100.0ms, 24: 200.0ms, 25: 500.0ms, 26: 1.000s, 27: 2.000s, 28: 5.000s, 29: 10.00s, 30: 20.00s, 31: 50.00s, 32: 100.0s, 33: 200.0s, 34: 500.0s, 35: 1000s.
voltDivType
Corase or fine, 0: Corase, 1: fine
bHasEmpty
Be avilible when the DSO work in ROLL or scan mode. You should draw the points in the empty area. The area is defined by the next parameter -- emptyPos.
0: Don't have empty area,
1: From emptyPos[0] to emptyPos[1] points are empty area
emptyPos[2]
See the parameter bHasEmpty

2.2.5 SCREEN_DISPLAY

The screen display’s parameter

```
typedef struct _SCREEN_DISPLAY
{
    INT32U      xDotSpace;
    INT32U      xDisLeft;
    INT32U      nDisData;
    INT16U      bDrawType;
}SCREEN_DISPLAY;
```

xDotSpace

The 1000 times of the distance between the each point display on the LCD screen. If is 250, it means the point space is 2.5 pixel on the LCD screen. The space is between 0 to 300.

xDisLeft

The 1000 times of the left position, If is 13000, it means the left position is 13 on the LCD screen.The position is between 0 to 300.

nDisData

The number of the points display on the screen, (0 ~ 1200)

bDrawType

Draw Type, 1: Normal, : Only draw Dot

2.2.6 HOLDOFF

The holdoff's parameters

```
typedef struct _HOLDOFF
{
    INT16U      time;
    INT16S      bOn;
}HOLDOFF;
```

Time

Channge the holdoff time.

bOn

Turn on/off the holdoff, 0: OFF, 1: ON

2.2.7 STOPSTATUS

The status when stop the machine.

```
typedef struct _STOPSTATUS
{
    INT16S      horiLevel;
    CHANNEL  ch[4];
    INT16U      timebase;
}STOPSTATUS;
```

horiLevel

The horizontal trigger position displayed on the screen, the left is 0, and the right is 299

ch[4]

The four channel's parameters, 0:CH1, 1:CH2, 2:MATH, 3:REF

Timebase

The timebase index

2.2.8 UPLOAD_DATA

All of the parameters of the hardware.

```
typedef struct _UPLOAD_DATA
{
    CHANNEL          ch[4];
    STOPSTATUS       stopStatus;
    MATH             math;
    TRIGGER          trigger[3];
    SCREEN_DISPLAY   display[4];
    INT16U           timebase;
    INT16U           function;
    INT16U           status;
    INT16S           horiPos;
    INT16U           triggerMode;
    INT16U           chSel;
    INT16U           horiFormat;
    INT16U           SamplingMode;
    INT16U           hTriggerPos;
    HOLDOFF          holdOff;
}UPLOAD_DATA;
```

ch[4]

The four channels' parameters, 0:CH1, 1:CH2, 2:MATH, 3:REF, See the struct “CHANNEL”

stopStatus

The status when pressing 'stop', See the sturct “STOPSTATUS”

Math

Math parameters see the struct “MATH”

trigger[3]

Trigger’s system, When the trigger is edge or pulse, 0 is available, When the trigger is ALT, trigger[1] is Ch1’s trigger and trigger[2] is Ch2’s trigger.

display[4]

Draw parameters of the four channels.

timebase

The timebase of the device. It is avaible when the trigger is not ALT.

function

The current type, 0: Digital scope, 1: Digital Meter measure

status

the status of the device, 0: Stop, 1: Run, 2: Auto, 3: Tri'd, 4: Wait

horiPos

The horizontal trigger position displayed on the screen, the left is 0, and the right is 299. It ‘s available when the trigger is not ALT.

triggerMode

Trigger type, 0: Edge Trigger, 1: Pulse Triger, 2: ALT Trigger

chSel

The current selected channel, 0:CH1, 1:CH2, 2:MATH, 3:REF

horiFormat

0: Y-T Mode, 1: X-Y Mode, 2: ROLL Mode

SamplingMode

0: Real Sample, 1: ETS

hTriggerPos

Trigger Position in the hardware. It 's available when the trigger is not ALT.

holdOff

Holdoff parameter, see the struct "HOLDOFF".

2.2.9 DMM_VALUE

The parameters of the digital meter measure's value

```
typedef struct _DMM_VALUE
{
    INT8U  value[4];
    float    flValue;
    INT8U  iDotPos;
    INT8U  iUnit;
    INT8U  iUnitType;
    INT8U    iSign;
}DMM_VALUE;
```

value[4]

The char value displayed on the screen. For example,if the value is {1,2,3,4} and the iDotPos is 0, the string "1234" will display on the screen.
If the iDotPos is 1, the string is "1.234",
If the iDotPos is 2, the string is "12.34".
If the iDotPos is 3, the string is "123.4"

flValue

the value format float displayed on the screen

iDotPos

see the parameter value[4]

iUnit

The unit of the value, 0:p, 1:n, 2:u, 3:m, 4: K, 5:M, 6:G, 7:NULL.

iUnitType

The unit type of the value, 0: V, 1: A, 2: Ω , 3: F
For example: if the iUnit is 3 and the iUnitType is 1 and the value is 1.234, the string display on the screen will be "1.234mA"

iSign

the sign of the value, 0: Postive, 1: Negtive

2.2.10 DMM_INFO

The digital meter measure's parameter.

```
typedef struct _DMM_INFO
{
    INT8U            iType;
```

```
    BOOLEAN     bOverflow;
    INT8U       iDCAC;
    INT8U       iMode;
    BOOLEAN     bRel;
    INT8U       iRange;
    INT8U       bAma;
    INT8U       iBar;
    INT8U       iBarSign;
    DMM_VALUE   value;
    DMM_VALUE   relValue;
}DMM_INFO, *PDMM_INFO;
```

iType

0: Voltage, 1: Current, 2: Resistance, 3: Diode, 4: Conti, 5: Capacitance

bOverflow

whether the value is overflow or not

iDCAC

The couple of the meter, 0:DC, 1:AC

iMode

0: AUTO, 1: Manual

bRel

0: Normal, 1: REL

iRange

the range index in the range array

Voltage: {60.00mV, 600.0mV, 6.000V, 60.00V, 600.0V, 6000V}

Crrent: {60.00mA, 600.0mA}

Resistance:{600.0, 6.000K, 60.00K, 600.0K, 6.000M, 60.00M}

bAma

Only available when the type is Current, 0: A, 1: mA

iBar

Bar value, from 0 to 60

iBarSign

Bar sign: 0: Positive, 1: Negative

Value

The current value acquired

relValue

The rel value when press rel.

3 Function Call Reference

The DSO1200DLL.DLL function call reference is in C language.

3.1 Machine Control Function

3.1.1 HTGetUSBDeviceList

Initialize the hardware and get the current usb device list connecting to the PC
INT32U HTGetUSBDeviceList(INT8U* usbList);

Parameters
INT8U* usbList: the array of the usb device.

Return values
the number of equipments connecting to the PC.

Remarks
The size of the usb list array is 16. The parameter in the array show the device status. 0 means the usb index was never connecting to PC, 1 means one deivce is connecting, 2 means a device was connecting but is disconnect now. for example: if the array return as {1, 2, 1, 0, 0, 0, 0...}, it means there are 2 equipments connecting to the PC, and the index of the there is 0 , 2. The index 2 is disconnect.
You must call this function first to initial the machine.

3.1.2 HTGetDeviceInfo

Get the information of the device.
HTSTATUS HTGetDeviceInfo(
HT_DEVICE_ID iDeviceID,PHT_DEVICE_INFO pHTDeviceInfo);

Parameters
HT_DEVICE_ID iDeivceID:
the id of the device which is made by CREATE_DEVID. (This parameter won't be introduced again)
PHT_DEVICE_INFO pHTDeviceInfo:
the pointer to the device information struct.

Return values
0: Success, 1:Fail.

Remarks
Get the device's information from hardware. The information struct is defined in the struct HT_DEVICE_INFO..

3.1.3 HTScreenShoot

Shoot the current screen of the machine.
HTSTATUS HTScreenShoot(HT_DEVICE_ID iDeviceID,char* pszPath);

Parameters
char* pszPath:
The location of the screen capture you want to save. The path should be translate with ansi string..

Return values
0: Success, 1:Fail.

Remarks
Shoot the current screen of the machine. For example:
HTSTATUS status = HTScreenShoot(0, "C:\\test.bmp");

3.1.4 HTShutDown

Shut down the machine.
HTSTATUS HTShutDown(HT_DEVICE_ID iDeviceID, INT8U iMode);

Parameters
INT8U iMode:
0: Shut down the machine.
1: Restart the machine..

Return values
0: Success, 1:Fail.

Remarks
With this function, you can shut down or restart the machine.

3.1.5 HTGetFunction

Return the current function of the machine.
int HTGetFunction(HT_DEVICE_ID iDeviceID);

Parameters

Return values
0: Digital Scope,
1: Digital Meter measure

Remarks
.

3.1.6 HTChangeFunction

Change the current function between Digital Scope and Digital Meter measure.
HTChangeFunction(HT_DEVICE_ID iDeviceID,INT8U iFunction);

Parameters
INT8U iFunction:
    0: Digital Scope.
1: Digital Meter Measure

Return values
0: Success, 1:Fail.

Remarks
.

3.1.7 HTShowMenu

Show or hide the menu in the lcd screen.
HTSTATUS HTShowMenu(HT_DEVICE_ID iDeviceID,BOOLEAN bShow);

Parameters
INT8U bShow:
    0: hide.
1: show

Return values
0: Success, 1:Fail.

Remarks
This function is only validate in the Digital Scope mode. The menu always show in the Digital Meter Measure mode.

3.1.8 HTUSBCheckConnect

Check wether the usb is connecting or not.
HTSTATUS WINAPI HTUSBCheckConnect(HT_DEVICE_ID iDeviceID);

Parameters

Return values
0: Connect, 1:disconnect.

Remarks
For plug and play, you should always call this function.

3.2 Digital Scope Function

3.2.1 DSOSetStatus

Set the current running status, Run or Stop
HTSTATUS DSOSetStatus(HT_DEVICE_ID iDeviceID,BOOLEAN bStatus);

Parameters
BOOLEAN bStatus
0: stop, 1:run

Return values
0: Success, 1: Fail

Remarks

3.2.2 DSOAutoSetup

Start autoset function.
HTSTATUS DSOAutoSetup(HT_DEVICE_ID iDeviceID);

Parameters

Return values
0: Success, 1: Fail

Remarks
Start autoset to channge the waveform to the optimal status.

3.2.3 DSOFactorySetup

Reset all of the setup to factory
HTSTATUS DSOFactorySetup(HT_DEVICE_ID iDeviceID);

Parameters

Return values
0: Success, 1: Fail

Remarks

3.2.4 DSOSetTimeBase

Set the current time/div
HTSTATUS DSOSetTimeBase(HT_DEVICE_ID iDeviceID,
INT8U iChannel,INT8U iTimeBase);

Parameters
INT8U iChannel
the channel to set timebase.
0: If the trigger mode is ALT, set ch1's timebase, otherwise set public timebase.
1: If the trigger mode is ALT, set Ch2's timebase, otherwise set public timebase.
2: If the trigger mode is ALT, nothing to do, otherwise set public timebase.
3: Set REF's timebase.
INT8U iTimeBase
The time/div index, see the struct CHANNEL.

Return values
0: Success, 1: Fail

Remarks
If the trigger mode is ALT, each channel's timebase is independent
If the trigger mode is not ALT, ch1 and ch2's timebase is the public timebase.

3.2.5 DSOSetHoriFormat

Set the horizontal format.
HTSTATUS DSOSetHoriFormat(HT_DEVICE_ID iDeviceID,
INT8U iFormat);

Parameters
INT8U iFormat
0: Y-T format, 1: X-Y Format, 2: ROLL format

Return values

0: Success, 1: Fail

Remarks

3.2.6 DSOSetHTriggerPos

Set horizontal trigger position
HTSTATUS DSOSetHTriggerPos(HT_DEVICE_ID iDeviceID, INT8U iChannel, INT16U iHTriggerPos);

Parameters
INT8U iChannel
0: If the trigger mode is ALT, set CH1; Otherwise,set public
1:If the trigger mode is ALT, set CH2; Otherwise, set public..
2:Set public
3:Set REF.
INT16U iHTriggerPos:
Horizontal trigger position

Return values
0: Success, 1: Fail

Remarks
If the trigger mode is ALT, each channel's horizontal trigger position is independent
If the trigger mode is not ALT, the horizontal trigger position is public.

3.2.7 DSOSetCHEnable

Enable or disable the channel
HTSTATUS DSOSetCHEnable(HT_DEVICE_ID iDeviceID,INT8U iChannel, BOOLEAN bEnable);

Parameters
INT8U iChannel
0: CH1, 1:CH2, 2:MATh, 3:REF
BOOLEAN bEnable
1: Enable, 0: Disable

Return values
0: Success, 1: Fail

Remarks

If the channel is enable, it's visible.
If the channel is disable, it's hide.

3.2.8 DSOSetVOLTDIV

Set the channel's volt/div
HTSTATUS DSOSetVOLTDIV(HT_DEVICE_ID iDeviceID, INT8U iChannel,INT8U iVoltDIV);

Parameters
INT8U iChannel
0: CH1, 1:CH2, 2:MATh, 3:REF
INT8U iVoltDiv
See the voltDivIndex in the struct CHANNLE

Return values
0: Success, 1: Fail

Remarks

3.2.9 DSOSetCoupling

Set the channel's couple
HTSTATUS DSOSetCoupling(HT_DEVICE_ID iDeviceID, INT8U iChannel,INT8U iCoupling);

Parameters
INT8U iChannel
0: CH1, 1:CH2, 2:MATh, 3:REF
INT8U iCoupling
0: AC, 1:DC, 2:GND.See the couple in the struct CHANNLE

Return values
0: Success, 1: Fail

Remarks

3.2.10 DSOSetProbe

Set the channel's probe
HTSTATUS DSOSetProbe(HT_DEVICE_ID iDeviceID, INT8U iChannel, INT8U iProbe);

Parameters
INT8U iChannel
0: CH1, 1:CH2, 2:MATh, 3:REF
INT8U iProbe
0: 1X, 1:10X, 2:100X, 3:1000X..See the probe in the struct CHANNLE

Return values
0: Success, 1: Fail

Remarks

3.2.11 DSOSetBW20M

Set the channel's bandwidth limit.
HTSTATUS DSOSetBW20M(HT_DEVICE_ID iDeviceID,
INT8U iChannel ,BOOLEAN bEnable);

Parameters
INT8U iChannel
0: CH1, 1:CH2, 2:MATH, 3:REF
BOOLEAN bEnable
1: Enable 20M bandwidth limit, 0:Disable

Return values
0: Success, 1: Fail

Remarks

3.2.12 DSOSetBW100M

Set the channel's bandwidth limit.
HTSTATUS DSOSetBW100M(HT_DEVICE_ID iDeviceID,
INT8U iChannel ,BOOLEAN bEnable);

Parameters
INT8U iChannel
0: CH1, 1:CH2, 2:MATH, 3:REF
BOOLEAN bEnable
1: Enable 100M bandwidth limit, 0:Disable

Return values
0: Success, 1: Fail

Remarks

3.2.13 DSOSetCoarseOrFine

Set the channel's volt/div type, coarse of fine.
HTSTATUS DSOSetCoarseOrFine(HT_DEVICE_ID iDeviceID, INT8U iChannel , INT8U iCoarse);

Parameters
INT8U iChannel
0: CH1, 1:CH2, 2:MATh, 3:REF
INT8U iCoarse
0: Coarse, 1: Fine

Return values
0: Success, 1: Fail

Remarks

3.2.14 DSOSetInvert

Set the channel invert or not
HTSTATUS DSOSetInvert(HT_DEVICE_ID iDeviceID, INT8U iChannel ,BOOLEAN bInvert);

Parameters
INT8U iChannel
0: CH1, 1:CH2, 2:MATh, 3:REF
BOOLEAN bInvert
0: Normal, 1: Invert

Return values
0: Success, 1: Fail

Remarks

3.2.15 DSOResetChannel

Move the channel to the vertical center of the screen
HTSTATUS DSOResetChannel(HT_DEVICE_ID iDeviceID, INT8U iChannel);

Parameters
INT8U iChannel
0: CH1, 1:CH2, 2:MATh, 3:REF

Return values
0: Success, 1: Fail

Remarks
The vertical trigger of the channel isn't change.

3.2.16 DSOSetChannelLeverPos

Channge the channel's vertical position.

HTSTATUS DSOSetChannelLeverPos(HT_DEVICE_ID iDeviceID,INT8U iChannel,INT16U iLeverPos);

Parameters
INT8U iChannel
0: CH1, 1:CH2, 2:MATh, 3:REF
INT16U iLeverPos
The channel's position in the screen, top is 0 and the bottom is 199.

Return values
0: Success, 1: Fail

Remarks

3.2.17 DSOSetMathOperator

Change the math operate type
HTSTATUS DSOSetMathOperator(HT_DEVICE_ID iDeviceID, INT8U iOperator);

Parameters
INT8U iOperator
0: +, 1: -, 2: *, 3: /, 4: FFT

Return values
0: Success, 1: Fail

Remarks
If channge to FFT, the math menu will change to FFT menu.
See the struct MATH

3.2.18 DSOSetMathSource

Change the math sourceA or SourceB
HTSTATUS DSOSetMathSource(HT_DEVICE_ID iDeviceID,INT8U iType, INT8U iSource);

Parameters
INT8U iType
0: Source A, 1: Source B.
INT8U iSource
0: CH1, 1:CH2

Return values
0: Success, 1: Fail

Remarks
If want to change FFT source, you should use the DSOSetFFTSource
0: Success, 1: Fail

Remarks
See the struct MATH

3.2.19 DSOSetFFTWindow

Change the FFT window function
HTSTATUS DSOSetFFTWindow(HT_DEVICE_ID iDeviceID,INT8U iFFTWindow);

Parameters
INT8U iFFTWindow
0:Rectangle, 1:Hanning, 2:Hamming, 3:Blackman.

Return values
0: Success, 1: Fail

Remarks
See the struct MATH

3.2.20 DSOSetFFTScale

Set the FFT Scale

**HTSTATUS DSOSetFFTScale(HT_DEVICE_ID iDeviceID,
INT8U iFFTScale);**

Parameters
INT8U iFFTScale
0:Vrms, 1:dBVrms

Return values
0: Success, 1: Fail

Remarks
See the struct MATH

3.2.21 DSOSetVoltDIVChange

Channge the voltage/div when the volt/div type is fine.
**HTSTATUS DSOSetVoltDIVChange(HT_DEVICE_ID iDeviceID,
INT8U iType,INT8U iChannel);**

Parameters
INT8U iType
0: Decrease, 1: Increase
INT8U iChannel
0: CH1, 1: CH2, 2: MATH, 3: REF

Return values
0: Success, 1: Fail

Remarks
The function is validate when the volt/div type is fine.You can change the volt/div small steps by this function.

3.2.22 DSOSetTriggerMode

Change the trigger mode
HTSTATUS DSOSetTriggerMode(HT_DEVICE_ID iDeviceID,INT8U iMode);

Parameters
INT8U iMode
0: Edge, 1: Pulse, 2: ALT

Return values
0: Success, 1: Fail

Remarks
Change the current trigger mode.

3.2.23 DSOSetTriggerHFReject

Set the trigger HF Reject or not
HTSTATUS DSOSetTriggerHFReject(HT_DEVICE_ID iDeviceID,INT8U iCh, BOOLEAN bEnable);

Parameters
INT8U iCh
    0: CH1, 1:CH2
BOOLEAN bEnable
    0: Disable, 1: Enable

Return values
0: Success, 1: Fail

Remarks

3.2.24 DSOSetTriggerSource

Set the trigger source.
HTSTATUS DSOSetTriggerSource(HT_DEVICE_ID iDeviceID, INT8U iSource);

Parameters
INT8U iSource
    0: CH1, 1:CH2

Return values
0: Success, 1: Fail

Remarks
This function is only validate when the trigger is not ALT

3.2.25 DSOSetTriggerSweep

Set the trigger's sweep
HTSTATUS DSOSetTriggerSweep(HT_DEVICE_ID iDeviceID, INT8U iSweep);

Parameters
INT8U iSweep

0: Auto, 1: Normal, 2: Single

Return values
0: Success, 1: Fail

Remarks

3.2.26 DSOSetVTriggerLeverPos

Change the vertical trigger position
HTSTATUS DSOSetVTriggerLeverPos(HT_DEVICE_ID iDeviceID, INT8U iChannel,INT16U iPos);

Parameters
INT8U iChannel
0:CH1, 1:CH2
INT16U iPos
Vertical trigger position, the top is 0 and the bottom is 199

Return values
0: Success, 1: Fail

Remarks

3.2.27 DSOSetTriggerSlope

Set the trigger slope
HTSTATUS DSOSetTriggerSlope(HT_DEVICE_ID iDeviceID, INT8U iChannel, INT8U iSlope);

Parameters
INT8U iChannel
When the trigger is ALT, 0 is CH1 and 1 is CH2; otherwise, the parameter is ingored.
INT16U iSlope
0: Rise, 1: Fall

Return values
0: Success, 1: Fail

Remarks
This function is validate when the trigger isn't pulse.

3.2.28 DSOSetPulseTriggerCondition

Set the trigger condition when the trigger is pulse.
HTSTATUS DSOSetPulseTriggerCondition(HT_DEVICE_ID iDeviceID, INT8U iChannel, INT8U iCondition);

Parameters
INT8U iChannel
When the trigger is ALT, 0 is CH1 and 1 is CH2; otherwise, the parameter is ingored.
INT16U iCondition
0: +Less, 1: +Equal, 2: +More, 3: -Less, 4: -Equal, 5: -More

Return values
0: Success, 1: Fail

Remarks
This function is validate when the trigger isn't edge.

3.2.29 DSOSetPulseTriggerTime

Set the pulse trigger setting time
HTSTATUS DSOSetPulseTriggerTime(HT_DEVICE_ID iDeviceID, INT8U iChannel, INT32U iTime);

Parameters
INT8U iChannel
When the trigger is ALT, 0 is CH1 and 1 is CH2; otherwise, the parameter is ingored.
INT16U iTime
the unit is ns/5, for example: if the value is 1000, the pulse setting is 5ms

Return values
0: Success, 1: Fail

Remarks
This function is validate when the trigger isn't edge.
see the 'pluseValue' parameter in the TRIGGER struct

3.2.30 DSOSetALTTrigType

Set the ALT trigger type
HTSTATUS DSOSetALTTrigType(HT_DEVICE_ID iDeviceID, INT8U iCh,

INT8U iMode);

Parameters
INT8U iCh
0:CH1, 1:CH2
INT16U iMode
0: Edge trigger, 1: pulse trigger.

Return values
0: Success, 1: Fail

Remarks
This function is validate when the trigger is ALT.

3.2.31 DSOGetAllSetting

Get all of the setting and data to display
HTSTATUS DSOGetAllSetting(HT_DEVICE_ID iDeviceID,UPLOAD_DATA* upData，USHORT* pCh1DisData, USHORT* pCh2DisData, USHORT* pMathDisData, USHORT* pRefDisData);

Parameters
UPLOAD_DATA* upData
See the sturct UPLOAD_DATA;
USHORT* pCh1DisData
Ch1's data to display, the size of which is 1200.
USHORT* pCh2DisData
Ch2's data to display, the size of which is 1200.
USHORT* pMathDisData
Math's data to display, the size of which is 1200.
USHORT* pRefDisData
Ref's data to display, the size of which is 1200.

Return values
0: Success, 1: Fail

Remarks
Before draw the waveform, you should get all of the waveform data and the setup from the machine.

3.2.32 DSOGetCh12Data

Get the CH1 and Ch2 memory data from hardware
HTSTATUS WINAPI DSOGetCh12Data(HT_DEVICE_ID iDeviceID,
INT8U* pCh1Data, INT8U* pCh2Data,
ULONG nCh1DataLen, ULONG nCh2DataLen);

Parameters
INT8U* pCh1Data
CH1 data
INT8U* pCh2Data
CH2 data
ULONG nCh1Datalen
the length of the CH1 data which specified by nData in the struct CHANNEL.
ULONG nCh2Datalen
the length of the CH2 data which specified by nData in the struct CHANNEL.

Return values
0: Success, 1: Fail

Remarks
The data in the struct UPLOAD_DATA is the display data, the max size of which is 1200.
Use this function, you can get all of the memory data from the hardware, the size of which is 16K or 32K.
The length of the data is specified in by nData in the struct CHANNEL

3.2.33 DSOGetChREFData

Get the REF memory data from hardware
HTSTATUS DSOGetChREFData(HT_DEVICE_ID iDeviceID,
INT8U* pREFData, ULONG nDataLen);

Parameters
INT8U* pREFData
REF data
ULONG nDatalen
the length of the REF data which specified by nData in the struct CHANNEL.

Return values
0: Success, 1: Fail

Remarks

See the remarks of the function DSOGetCh12Data

3.3 Digital Meter Measure Function

3.3.1 DMMGetInfo

Get the measurement information
HTSTATUS DMMGetInfo(HT_DEVICE_ID iDeviceID, PDMM_INFO info);

Parameters
PDMM_INFO info
DMM information, specified in the struct DMM_INFO

Return values
0: Success, 1: Fail

Remarks
Get all of the digital meter measurement's information.

3.3.2 DMMSetMeasureMode

Set the meausrement's type
HTSTATUS DMMSetMeasureMode(HT_DEVICE_ID iDeviceID,
INT8U iMode);

Parameters
INT8U iMode
0: Voltage, 1: Current, 2: Resistance, 3: Diode, 4: CONTI, 5:Capacitance

Return values
0: Success, 1: Fail

Remarks
Chagne the digital meter mesurement type

3.3.3 DMMSetVoltACDC

Set the voltage's couple
HTSTATUS DMMSetVoltACDC(HT_DEVICE_ID iDeviceID, INT8U iACDC);

Parameters
INT8U iACDC
0: DC, 1:AC

Return values
0: Success, 1: Fail

Remarks

3.3.4 DMMSetVoltRel

Set the voltage's REL or not
HTSTATUS DMMSetVoltRel(HT_DEVICE_ID iDeviceID, BOOLEAN bRel);

Parameters
BOOLEAN bRel
0: normal, 1:REL

Return values
0: Success, 1: Fail

Remarks

3.3.5 DMMSetVoltMode

Set the voltage's mode, auto or manual.
HTSTATUS DMMSetVoltMode(HT_DEVICE_ID iDeviceID, INT8U iMode);
Parameters
INT8U iMode
0: AUTO, 1: Manual

Return values
0: Success, 1: Fail

Remarks

3.3.6 DMMSetVoltRange

Change the voltage's range.
HTSTATUS DMMSetVoltRange(HT_DEVICE_ID iDeviceID, INT8U iRange);

Parameters
INT8U iRange
See the parameter 'iRange' in the struct DMM_INFO

Return values
0: Success, 1: Fail

Remarks

3.3.7 DMMSetCurrentACDC

Set the current's couple
HTSTATUS DMMSetCurrentACDC(HT_DEVICE_ID iDeviceID, INT8U iACDC);

Parameters
INT8U iACDC
0: DC, 1: AC

Return values
0: Success, 1: Fail

Remarks

3.3.8 DMMSetCurrentRel

Set the current's REL or not
HTSTATUS DMMSetCurrentRel(HT_DEVICE_ID iDeviceID, BOOLEAN bRel);

Parameters
BOOLEAN bRel
0: normal, 1:REL

Return values
0: Success, 1: Fail

Remarks

3.3.9 DMMSetCurrentMode

Set the current's mode
HTSTATUS DMMSetCurrentMode(HT_DEVICE_ID iDeviceID, INT8U iMode);

Parameters
INT8U iMode
0: AUTO, 1: Manual

Return values
0: Success, 1: Fail

Remarks

3.3.10 DMMSetCurrentRange

Change the current's range.
HTSTATUS DMMSeCurrentRange(HT_DEVICE_ID iDeviceID, INT8U iRange);

Parameters
INT8U iRange
See the parameter 'iRange' in the struct DMM_INFO

Return values
0: Success, 1: Fail

Remarks

3.3.11 DMMSetCurrentAmA

Set the current type
HTSTATUS WINAPI DMMSetCurrentAmA(HT_DEVICE_ID iDeviceID, INT8U iAmA);

Parameters
INT8U iAmA
0: A, 1: mA

Return values
0: Success, 1: Fail

Remarks

3.3.12 DMMSetOHMRel

Set the resistance's REL or not

HTSTATUS DMMSetOHMRel(HT_DEVICE_ID iDeviceID, BOOLEAN bRel);

Parameters
BOOLEAN bRel
0: Normal, 1: REL

Return values
0: Success, 1: Fail

Remarks

3.3.13 DMMSetOHMRange

Set the resistance's range
HTSTATUS DMMSetOHMRange(HT_DEVICE_ID iDeviceID, INT8U iRange);

Parameters
INT8U iRange
See the parameter 'iRange' in the struct DMM_INFO

Return values
0: Success, 1: Fail

Remarks

3.3.14 DMMSetOHMMode

Set the resistance's mode
HTSTATUS DMMSetOHMMode(HT_DEVICE_ID iDeviceID, INT8U iMode);

Parameters
INT8U iMode
0: AUTO, 1: Manual

Return values
0: Success, 1: Fail

Remarks

3.3.15 DMMSetCapRel

Set the capacitance's rel or not

HTSTATUS DMMSetCapRel(HT_DEVICE_ID iDeviceID, BOOLEAN bRel);

Parameters
BOOLEAN bRel
0: Normal, 1: REL

Return values
0: Success, 1: Fail

Remarks