Acute Digital Storage Oscilloscope Software development kit (SDK) Programming guide

Version: 1.10
Publish: 2018/07/23

內容

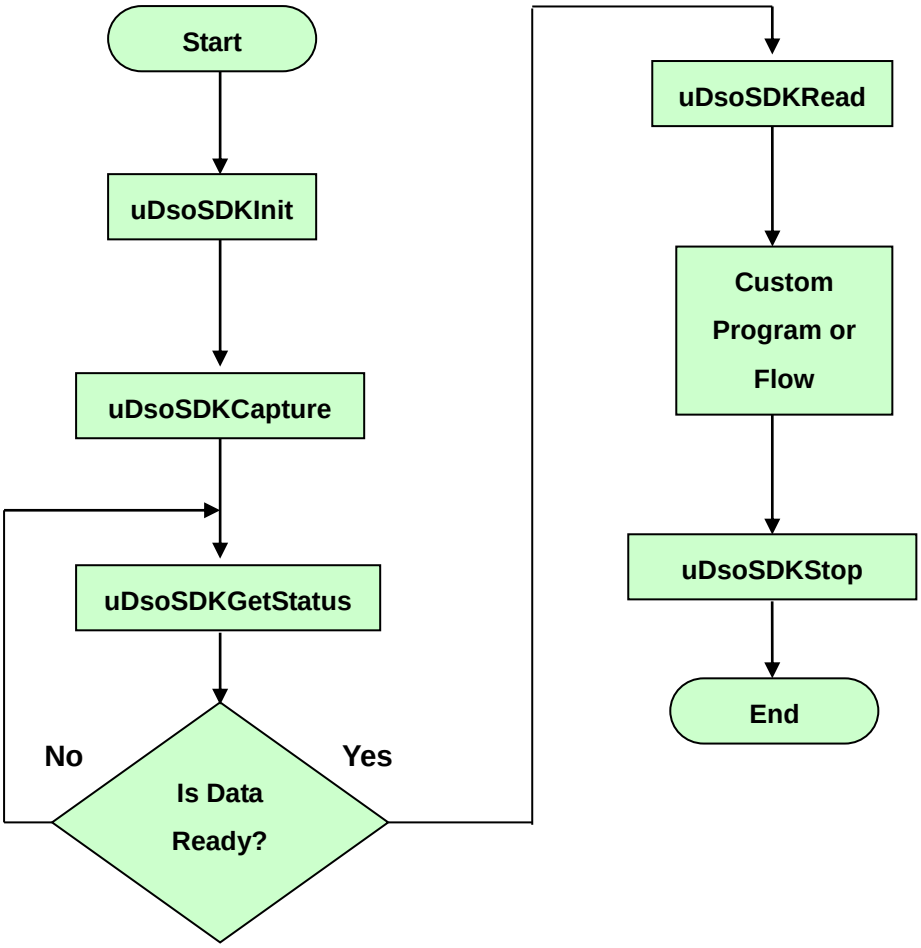
DSOSDK 簡介

DSOSDK.DLL 基於 Win32 規格提供開放的 API，使用時須同時連接上 Acute DSO 及搭配 DSORun.DLL 1.3.0.6 或之後的版本才可正常使用。DSOSDK.DLL 功能是延續並擴充自 DSORun.DLL，諸如量測計算功能，訊號發生器控制功能等等.可使 Acute DSO 的應用更加的廣泛。

在使用 DSOSDK.DLL 之前，若您是使用 DS-1000 系列 DSO，請先執行 DSO AP 來校準 DSO 硬體後再使用 SDK 的功能。

配合 DSOSDK.dll 軟體版本: 1. 0. 0. 37

DSOSDK.DLL 流程



DSOSDK.DLL 函式說明

DSO 初始化函式

int uDsoSDKSetWorkDir(LPSTR lpszDirectory)

功能:

指定 SDK 函式所使用 Dsorun.dll 位置。

若不呼叫此函式, 則系統預設讀取 SDK 同目錄下的 Dsorun.dll。

參數:

lpszDirectory [in]:

Type:LPSTR

SDK 的工作目錄。

回傳值:

若回傳值為 0 表示在該目錄下偵測不到Dsorun.dll, 回傳值為 1 代表設定成功。

範例:

傳入 "D:\\MyWorkDir\\", 將 Dsorun.dll 的讀取目錄改到 D:\MyWorkDir 下。

int uDsoSDKInit()

功能:

偵測並初始化 DSO 硬體，堆疊使用時需連接觸發線。

回傳值:

此回傳值所代表的意義為有幾部DSO被偵測到，也就是

說如果回傳值為 0 表示偵測不到 DSO。

int uDsoSDKInitStack(LPSTR lpszSN)

功能:

偵測並依指定序號初始化 DSO 硬體，堆疊使用時需連接觸發線。

參數:

lpszSN[in]:

Type:LPSTR

依序排列指定 DSO 的序號，各 DSO 序號以逗號做區隔，字串須以 0 作為結尾。

回傳值:

此回傳值所代表的意義為有幾部DSO被偵測到，也就是

說如果回傳值為 0 表示偵測不到 DSO。

範例:

傳入 "TSA22120001,TSA22120002,TSA22120003"

TSA22120001 – CH1, CH2

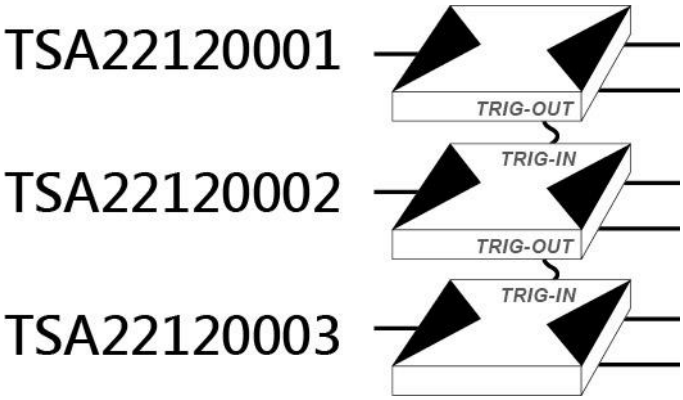
TSA22120002 – CH3, CH4

TSA22120003 – CH5, CH6

第一條堆疊線從 TSA22120001 的 Trig-Out 連接至 TSA22120002 的 Trig-In

第二條堆疊線從 TSA22120002 的 Trig-Out 連接至 TSA22120003 的 Trig-In

堆疊線連接方式請參考下圖：



BOOL uDsoSDKSelectGroup(int iGroup)

功能:

使用時若要同時控管多個以堆疊模式建立之 DSO 群組時，可以透過切換群組後，再進行初始化及指定參數等動作。各群組內的 DSO 各群組內的 DSO 若非單台時，則呈堆疊狀態，各群組間的 DSO 則為非堆疊狀態。如果只使用單一群組時則不必呼叫此函式。

參數:

iGroup[in]:

Type: int

指定選擇的群組，最大值為 19，最小值為 0，初始預設選擇群組 0。

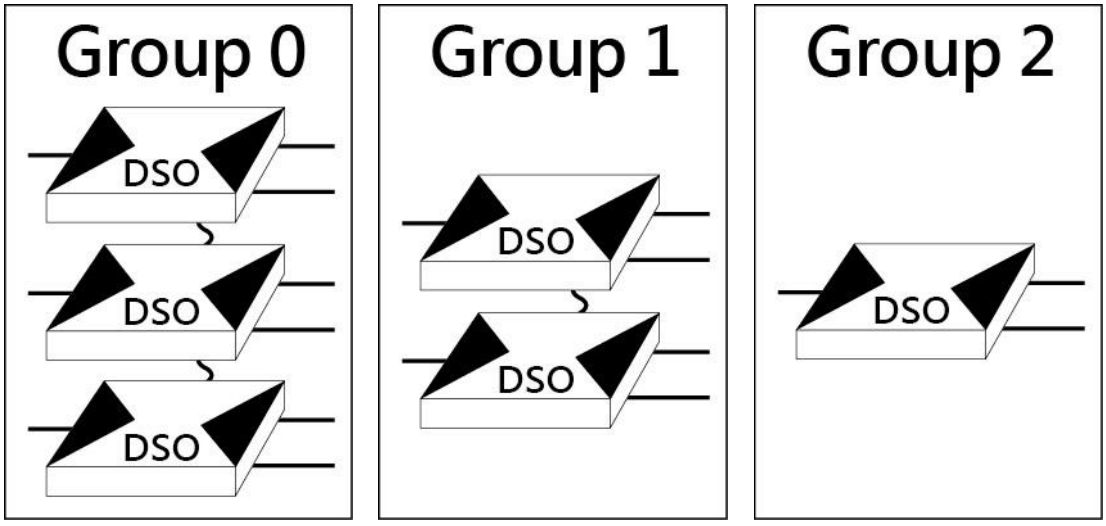
回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

範例:

以下為將 6 台 DSO 分別指定到 3 個群組的範例

```
uDsoSDKSelectGroup(0);                          選擇群組 0
uDsoSDKInitStack("TSA22120001,TSA22120002,TSA22120003");指定並初始群組 0
uDsoSDKSelectGroup(1);                          選擇群組 1
uDsoSDKInitStack("TSA22120004,TSA22120005");    指定並初始群組 1 的 DSO
uDsoSDKSelectGroup(2);                          選擇群組 2
uDsoSDKInitStack("TSA22120006");  指定並初始群組 2 的 DSO
```



DSO 硬體參數讀取

BOOL uDsoSDKGetVendorName(int iDev, LPSTR lpszData)

功能:

取得 DSO 廠商名稱。

參數:

iDev[in]:

Type: int

指定 DSO 裝置的編號，從 0 為開始為第一台。

lpszData[out]:

Type: LPSTR

用以存放 DSO 廠商名稱字串。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKGetProductName(int iDev, LPSTR lpszData)

功能:

取得 DSO 產品名稱。

參數:

iDev[in]:

Type: int

指定 DSO 裝置的編號，從 0 為開始為第一台。

lpszData[out]:

Type: LPSTR

用以存放 DSO 產品名稱字串。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKGetSerialNum(int iDev, LPSTR lpszData)

功能:

取得 DSO 產品序號。

參數:

iDev[in]:

Type: int

指定 DSO 裝置的編號，從 0 為開始為第一台。

lpszData[out]:

Type: LPSTR

用以存放 DSO 產品序號字串。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKGetHwVer(int iDev, int * piHwVer)

功能:

取得 DSO 硬體版本。

參數:

iDev[in]:

Type: int

指定 DSO 裝置的編號，從 0 為開始為第一台。

piHwVer[out]:

Type: int *

用以存放 DSO 硬體版本。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKGetFwVer(int iDev, int * piFwVer)

功能:

取得 DSO 韌體版本。

參數:

iDev[in]:

Type: int

指定 DSO 裝置的編號，從 0 為開始為第一台。

piFwVer[out]:

Type: int *

用以存放 DSO 韌體版本。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKGetProductID(int iDev, int * piProductID)

功能:

取得 DSO 裝置 ID。

參數:

iDev[in]:

Type: int

指定 DSO 裝置的編號，從 0 為開始為第一台。

piProductID[out]:

Type: int *

用以存放 DSO 裝置 ID。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKGetUsbDeviceHandle(int iDev, HANDLE hUsbHandle)

功能:

取得 DSO USB Handle，可用以進行 USB 拔除的偵測。

參數:

iDev[in]:

Type: int

指定 DSO 裝置的編號，從 0 為開始為第一台。

hUsbHandle[out]:

Type: HANDLE

用以存放 DSO USB Handle。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

範例:

```
//註冊 USB handle 到主收 Windows Message 的 windows
DEV_BROADCAST_HANDLE NotificationFilter;
ZeroMemory( &NotificationFilter, sizeof(NotificationFilter) );
NotificationFilter.dbch_size = sizeof(DEV_BROADCAST_HANDLE);
NotificationFilter.dbch_devicetype = DBT_DEVTYP_HANDLE;
NotificationFilter.dbch_handle = m_hDsoDev;
NotificationFilter.dbch_eventguid = MYGUID;
m_hDeviceNotify = RegisterDeviceNotification( hMainWin,
    &NotificationFilter, DEVICE_NOTIFY_WINDOW_HANDLE );

// 當 Device change 時, Windows 會發送 WM_DEVICECHANGE 給註冊的視窗
LRESULT MsgDeviceChange( HWND hDlg, UINT uMessage, WPARAM wParam,
LPARAM lParam )
{
    if ( DBT_DEVICEARRIVAL == wParam ||
        DBT_DEVICEREMOVECOMPLETE == wParam )
    {
        PDEV_BROADCAST_HDR pHdr = (PDEV_BROADCAST_HDR)lParam;
        switch ( pHdr->dbch_devicetype )
        {
            case DBT_DEVTYP_HANDLE:
                pDevHnd = (PDEV_BROADCAST_HANDLE)pHdr;
```

```
                    // Plug-out check here !!
                    break;
                ......
            }
        }
        return true;
}
```

BOOL uDsoSDKGetCalibrationData(int iDev, double pdbValue[6])

功能:

取得 DSO 資料校正資訊。

參數:

iDev[in]:

Type: int

指定 DSO 裝置的編號，從 0 為開始為第一台。

pdbValue[out]:

Type: double *

用以存放 DSO 校正資訊，須準備 6 個 double 的暫存空間。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

備註:

當使用者讀取讀取未經校正運算的 RAW Data 波形資料時，若要還原出正確的電壓值時，必須以校正資料套用公式計算後，才能得到正確的電壓值。也可以透過 SDK 提供的函式 uDsoSDKRawToDouble_mv 或是 uDsoSDKRawToDouble_uv 一次將整個波形資料轉換完成。

Mul , OffsetA, OffsetB 為波形還原成實際電壓值所需的資料。

```
double dMul[2], dOfsA[2], dOfsB[2];
dMul[0] = pdbValue[0];
dOfsA[0] = pdbValue[1];
dOfsB[0] = pdbValue[2];
dMul[1] = pdbValue[3];
dOfsA[1] = pdbValue[4];
dOfsB[1] = pdbValue[5];
```

```
RealVolt = ((RawValue - dOfsA) * dMul + dOfsB – VOffset) * Probe;
Volt_CH1 = (RawValue_CH1 – dOfsA[0] ) * dMul[0] + dOfsB[0] – VOffset
RealVolt_CH1 = (Volt_CH1 – VOffset) * Probe;   // CH1 真實電壓值(uV)
Volt_CH2 = (RawValue_CH2 – dOfsA[1] ) * dMul[1] + dOfsB[1] – VOffset)
RealVolt_CH2 = (Volt_CH2 – VOffset) * Probe;   // CH2 真實電壓值(uV)
```

int uDsoSDKGetDeviceCount( )

功能:

在裝置初始完畢後回傳當前群組內的裝置數。

回傳值:

回傳當前群組內的裝置數。

擷取參數設定

BOOL uDsoSDKSetSampleRate(__int64 i64SampleRate)
BOOL uDsoSDKGetSampleRate(__int64 & i64SampleRate)

功能:

設定/讀取DSO取樣率。預設值為10MS/s。

參數:

i64SampleRate[in]:

Type: int

i64SampleRate 是 DSO 的取樣率，可提供輸入的數值請參考下表，取樣率單位為 Samples/per second (S/s)。

DSO 型號	支援的取樣率
DS1002	2.5G S/s, 100M S/s, 50MS/s, 20MS/s, 10MS/s, 5MS/s, 2MS/s, 1MS/s, 500KS/s, 200KS/s, 100KS/s, 50KS/s, 20KS/s, 10KS/s, 5KS/s, 2KS/s, 1KS/s,500S/s, 200S/s, 100S/s.
DS1102 DS1202 DS1302	5GS/s, 200MS/s, 100MS/s, 50MS/s, 20MS/s, 10MS/s, 5MS/s, 1MS/s, 500KS/s, 200KS/s, 100KS/s, 50KS/s, 20KS/s, 10KS/s, 5KS/s, 2KS/s, 1KS/s,500S/s, 200S/s, 100S/s.
TravelScope series	25GS/s, 10GS/s, 5GS/s, 2.5GS/s, 1GS/s, 500MS/s, 250MS/s, 100MS/s, 50MS/s, 25MS/s, 10MS/s, 5MS/s, 2.5MS/s, 1MS/s, 500KS/s, 250KS/s, 100KS/s, 50KS/s, 25KS/s, 10KS/s, 5KS/s, 2.5KS/s 1KS/s, 500S/s, 200S/s, 100S/s.

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKSetRecordLength(int iRecordLength)
BOOL uDsoSDKSetRecordLength(int & iRecordLength)

功能:

設定/讀取DSO記憶深度。預設記憶深度為2k點。

參數:

iRecordLength[in]:

Type: int

iRecordLength為擷取波形的長度。最大值請參考 DSO 硬體規格書中最大記憶深度的部分。如果DSO當時使用的取樣率為2.5GS/s(DS1002)或是5GS/s (DS1102, DS1202, DS1302) 則iRecordLength不能大於25k。 TravelScope 系列則無此限制。

不論是DS1000或是TravelScope，設定記憶深度都必須為8的倍數。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKSetWaitMode(int iWaitMode, __int64 i64CustomWaitTime_ps)
BOOL uDsoSDKGetWaitMode(int iWaitMode, __int64 &i64CustomWaitTime_ps)

功能:

設定/讀取DSO觸發等待模式，如果在該時間內都沒有發生觸發事件的話，則DSO將會自動擷取當前的波形。預設等待模式為Quick。

參數:

iWaitMode[in]:

Type: int

參數		TravelScope系列	DS1000系列
WAIT_QUICK	0	Quick, 等待時間約為1.28ms	Quick, 等待時間約為6.25ms
WAIT_SLOW	1	Slow, 等待時間約為1s	Slow, 等待時間約為4.88s
WAIT_FOREVER	2	Forever	
WAIT_CUSTOM	3	提供使用者設定等待時間	

i64CustomWaitTime_ps[in]:

Type: int64

此數值只有在 iWaitMode 為 WAIT_CUSTOM 時才有作用，單位為 ps。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKSetDelayTime(__int64 i64DelayTime_ps)
BOOL uDsoSDKGetDelayTime(__int64 & i64DelayTime_ps)

功能:

設定/讀取DSO在收到觸發後延遲多少時間才開始擷取資料。預設值為0 ps。

參數:

i64DelayTime_ps[in]:

Type: int64

觸發延遲時間，單位為 ps。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKSetHoldoffTime(__int64 i64HoldOffTime_ps)
BOOL uDsoSDKGetHoldoffTime(__int64 & i64HoldOffTime_ps)

功能:

設定/讀取 DSO 在收到觸發後，需延滯多少時間才會重新啟動觸發。預設值為0 ps。

(僅適用於TravelScope系列)

參數:

i64HoldOffTime_ps[in]:

Type: int64

觸發延滯時間，單位為 ps。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKSetTrigPos(__int64 i64TrigPosition)
BOOL uDsoSDKGetTrigPos(__int64 & i64TrigPosition)

功能:

設定/讀取DSO觸發位置。預設值為0，代表觸發點為波形中央。

參數:

i64TrigPosition[in]:

Type: int64

設定觸發位置，單位為 ps。當 i64TrigPosition 該值為 0 時，指的是觸發位置位於波形擷取緩衝區的正中央(50%)的位置。若為負值，則觸發位置在緩衝區開始到中央的位置。若為正值，則觸發位置在中央到緩衝區結束的位置。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKSetBWL(int iCh, int iBwlFlag)
BOOL uDsoSDKGetBWL(int iCh, int & iBwlFlag)

功能:

設定/讀取通道頻寬限制功能。預設為不使用頻寬限制。

參數:

iCh[in]:

Type: int

選擇設定通道，傳入0代表通道一。

iBwlFlag[in]:

Type: int

請參考以下表格填入所需參數。

參數		說明
BWL_FULL	0	不使用頻寬限制，即使用DSO最高頻寬
BWL_20M	1	開啟20MHz頻寬限制
BWL_100M	2	開啟100MHz頻寬限制 ***(僅適用於TravelScope系列)***

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKSetCoupling(int iCh, int iCouplingFlag)
BOOL uDsoSDKGetCoupling(int iCh, int & iCouplingFlag)

功能:

設定/讀取通道耦合功能。預設值為COUPLING_DC直流耦合模式。

參數:

iCh[in]:

Type: int

選擇設定通道，傳入0代表通道一。

iCouplingFlag[in]:

Type: int

請參考以下表格填入所需參數。

參數		說明
COUPLING_DC	0	通道耦合模式：直流

COUPLING_AC	1	通道耦合模式：交流
COUPLING_GND	2	通道耦合模式：接地

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKSetChOnOff(int iCh, bool bChOn)
BOOL uDsoSDKGetChOnOff(int iCh)

功能:

設定/讀取通道開啟關閉功能。通道狀態預設為開啟。

TravelScope系列雙通道時最高取樣率為500MHz，單通道時最高取樣率為1GS/s。

DS1302 / DS1202 / DS1102系列雙通道時最高取樣率為100MS/s，單通道時最高取樣率為200MS/s。

DS1002系列雙通道詞最高取樣率為50MS/s，單通道時最高取樣率為100MS/s。

參數:

iCh[in]:

Type: int

選擇設定通道，傳入0代表通道一。

bChOn[in]:

Type: bool

傳入TRUE代表開啟通道，傳入FALSE代表關閉通道。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKSetAcquireMode(int iCh, int iAcquireMode)
BOOL uDsoSDKGetAcquireMode(int iCh, int & iAcquireMode)

功能:

設定/讀取通道擷取模式。預設值為ACQ_SAMPLE取樣模式。

參數:

iCh[in]:

Type: int

選擇設定通道，傳入0代表通道一。

iAcquireMode[in]:

Type: int

請參考以下表格填入所需參數。

參數		說明
ACQ_SAMPLE	0	取樣模式
ACQ_PEAKDETECT	4	峰值檢測模式
ACQ_HIRES	5	高解析度模式

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKSetVoltDiv(int iCh, int iVoltDiv_uV)
BOOL uDsoSDKGetVoltDiv(int iCh, int & iVoltDiv_uV)

功能:

設定/讀取通道電壓刻度。預設值為100mV (相對探棒設定為x1時)。

參數:

iCh[in]:

Type: int

選擇設定通道，傳入0代表通道一。

iVoltDiv_uV[in]:

Type: int

填入所需之電壓刻度，可供輸入的數值如下表，輸入單位為uV。需要注意的是，當使用的探棒的設定倍率非x1時，傳入的iVoltDiv必須除以探棒的倍率。例如探棒設定倍率是x10，設定通道電壓刻度2V/Div時，則傳入的iVoltDiv應該是2000000/10。

支援的電壓刻度
10V, 5V, 2V, 1V, 500mV, 200mV, 100mV, 50mV, 20mV, 10mV, 5mV, 2mV

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKSetVoltPos(int iCh, int iPosition)
BOOL uDsoSDKGetVoltPos(int iCh, int & iPosition)

功能:

設定/讀取通道電壓位置。預設值為8000，即波形中央。

參數:

iCh[in]:

Type: int

選擇設定通道，傳入0代表通道一。

iPosition[in]:

Type: int

填入所需之電壓0V位置，填入0代表波形最底端，填入16000代表波形最頂端。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKSetVoltOfs(int iCh, int iVoltOfs_uV)
BOOL uDsoSDKGetVoltOfs(int iCh, int & iVoltOfs_uV)

功能:

設定/讀取DSO通道電壓偏移。預設值為偏移0V。***(僅適用於TravelScope系列)***

參數:

iCh[in]:

Type: int

選擇設定通道，傳入0代表通道一。

iVoltOfs_uV[in]:

Type: int

填入所需之電壓偏移值，單位為uV，若填入 1000 uV則代表波形向下移動 1mV，若填入 -1000uV 則代表波形向上移動 1mV。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKSetHwBit(int iHwBit)

功能:

設定DSO電壓解析度。預設值為8Bit。***(僅適用於TravelScope高解析度系列)***

參數:

iHwBit[in]:

Type: int

選擇欲使用的電壓解析度，可填入8、12、14、15、16，解析度為16bit時僅支援單一通道的資料擷取

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

觸發設定

DSO 剛初始化完成時會自動切換為預設的邊緣觸發模式，此時觸發源選擇為

Ch1，斜率選擇為上升緣，觸發電壓設定為 1.6V。(相對探棒倍率 x10)
當使用多機堆疊功能時，僅可選擇主機的通道 1、2 及外部觸發做為觸發訊號源。
其餘從機無法設定觸發選項，只能被主機或是其他從機觸發。

BOOL uDsoSDKSetEdgeTrig(int iSrc, int iSlope, __int64 i64Threshold_uV)

功能:

設定觸發模式為邊緣觸發。

參數:

iSrc[in]:

Type: int

觸發來源通道。

參數		說明
TRIG_SOURCE_CH1	0	通道 1
TRIG_SOURCE_CH2	1	通道 2
TRIG_SOURCE_EXT	2	外部觸發

iSlope[in]:

Type: int

請參考以下表格填入所需參數。

參數		說明
TRIG_EDGE_RISING	0	上升緣
TRIG_EDGE_FALLING	1	下降緣
TRIG_EDGE_EITHER	2	任意邊緣
TRIG_EDGE_ALTERNATE	3	交替式邊緣

i64Threshold_uV[in]:

Type: __int64

觸發電壓，單位為 uV。需要注意的是，當使用的探棒的設定倍率非 x1 時，傳入的 i64Threshold_uV 必須除以探棒的倍率。例如探棒設定倍率是 x10，設定臨界電壓 1.6V 時，則傳入的 i64Threshold_uV 應該是 1600000/10。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKSetVideoTrig(int iSrc, int iMode, int iScanline)

功能:

設定觸發模式為視頻觸發。

參數:

iSrc[in]:

Type: int

觸發來源通道。

參數		說明
TRIG_SOURCE_CH1	0	通道 1
TRIG_SOURCE_CH2	1	通道 2
TRIG_SOURCE_EXT	2	外部觸發

iMode[in]:

Type: int

請參考以下表格填入所需參數。

參數		說明
TRIG_VIDEO_MODE_SCANLINE	0	視頻線
TRIG_VIDEO_MODE_ANYFIELD	1	任意圖場
TRIG_VIDEO_MODE_ODDFIELD	2	奇數圖場
TRIG_VIDEO_MODE_EVENFIELD	3	偶數圖場

iScanline[in]:

Type: int

視頻線編號，僅在 iMode 選擇為視頻線觸發時才有用。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKSetRuntTrig(int iSrc, int iPolarity, int iCompareType, int iEqualRange, __int64 i64Width_ps, __int64 i64ThresholdA_uV, __int64 i64ThresholdB_uV)

功能:

設定觸發模式為矮波觸發。***(僅適用於TravelScope系列)***

參數:

iSrc[in]:

Type: int

觸發來源通道。

參數	說明

TRIG_SOURCE_CH1	0	通道 1
TRIG_SOURCE_CH2	1	通道 2
TRIG_SOURCE_EXT	2	外部觸發

iPolarity[in]:

Type: int

請參考以下表格填入所需參數。

參數		說明
TRIG_RUNT_POLARITY_HIGHPULSE	0	正矮波
TRIG_RUNT_POLARITY_LOWPULSE	1	負矮波
TRIG_RUNT_POLARITY_EITHER	2	任意矮波

iCompareType[in]:

Type: int

請參考以下表格填入所需參數。

參數		說明
TRIG_RUNT_CMP_LESS	0	寬度比較小於
TRIG_RUNT_CMP_GREAT	1	寬度比較大於
TRIG_RUNT_CMP_EQUAL	2	寬度比較等於
TRIG_RUNT_CMP_NOT_EQUAL	3	寬度比較不等於

iEqualRange[in]:

Type: int

寬度設定比較容忍範圍，標準設定為 5 (5%)，在寬度設定為 10ms 時，可容許 10ms x 5% = 500ns 的誤差。

i64Width_ps[in]:

Type: __int64

矮波寬度設定，單位為 ps，填入零代表不指定寬度。

i64ThresholdA_uV[in]:

Type: __int64

第一組觸發電壓，單位為 uV。需要注意的是，當使用的探棒的設定倍率非 x1 時，傳入的 i64ThresholdA_uV 必須除以探棒的倍率。例如探棒設定倍率是 x10，設定臨界電壓 1.6V 時，則傳入的 i64ThresholdA_uV 應該是 1600000/10。

i64ThresholdB_uV[in]:

Type: __int64

第二組觸發電壓，單位為 uV。同上，當使用的探棒的設定倍率非 x1 時，傳入的 i64ThresholdB_uV 必須除以探棒的倍率。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKSetWidthTrig(int iSrc, int iPolarity, int iCompareType, int iEqualRange, __int64 i64Width_ps, __int64 i64Threshold_uV)

功能:

設定觸發模式為寬度觸發。***(僅適用於TravelScope系列)***

參數:

iSrc[in]:

Type: int

觸發來源通道。

參數		說明
TRIG_SOURCE_CH1	0	通道 1
TRIG_SOURCE_CH2	1	通道 2
TRIG_SOURCE_EXT	2	外部觸發

iPolarity[in]:

Type: int

請參考以下表格填入所需參數。

參數		說明
TRIG_WIDTH_POLARITY_HIGHPULSE	0	正脈波
TRIG_WIDTH_POLARITY_LOWPULSE	1	負脈波
TRIG_WIDTH_POLARITY_EITHER	2	任意脈波

iCompareType[in]:

Type: int

請參考以下表格填入所需參數。

參數		說明
TRIG_WIDTH_CMP_LESS	0	寬度比較小於
TRIG_WIDTH_CMP_GREAT	1	寬度比較大於
TRIG_WIDTH_CMP_EQUAL	2	寬度比較等於
TRIG_WIDTH_CMP_NOT_EQUAL	3	寬度比較不等於

iEqualRange[in]:

Type: int

寬度設定比較容忍範圍，標準設定為 5 (5%)，在寬度設定為 10ms 時，可容許 10ms x 5% = 500ns 的誤差。

i64Width_ps[in]:

Type: __int64

矮波寬度設定，單位為 ps，填入零代表不指定寬度。

i64Threshold_uV[in]:

Type: __int64

觸發電壓，單位為 uV。需要注意的是，當使用的探棒的設定倍率非 x1 時，傳入的 i64Threshold_uV 必須除以探棒的倍率。例如探棒設定倍率是 x10，設定臨界電壓 1.6V 時，則傳入的 i64Threshold_uV 應該是 1600000/10。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKSetPatternTrig(int iSrcA, int iSrcB, int iNotA, int iNotB, int iAndOr, __int64 i64Width_ps, __int64 i64ThresholdA_uV, __int64 i64ThresholdB_uV)

功能:

設定觸發模式為碼型觸發。***(僅適用於TravelScope系列)***

參數:

iSrcA[in]:

Type: int

第一組觸發來源通道。

參數		說明
TRIG_SOURCE_CH1	0	通道 1
TRIG_SOURCE_CH2	1	通道 2
TRIG_SOURCE_EXT	2	外部觸發

iSrcB[in]:

Type: int

第二組觸發來源通道。

參數		說明
TRIG_SOURCE_CH1	0	通道 1

TRIG_SOURCE_CH2	1	通道 2
TRIG_SOURCE_EXT	2	外部觸發

iNotA[in]:

Type: int

第一組觸發訊號反向。

iNotB[in]:

Type: int

第二組觸發訊號反向。

iAndOr[in]:

Type: int

請參考以下表格填入所需參數。

參數		說明
TRIG_PATTERN_ANDOR_AND	0	邏輯AND
TRIG_PATTERN_ANDOR_OR	1	邏輯OR

i64Width_ps[in]:

Type: __int64

觸發指定寬度，單位為 ps。

i64ThresholdA_uV[in]:

Type: __int64

第一組觸發電壓，單位為 uV。需要注意的是，當使用的探棒的設定倍率非 x1 時，傳入的 i64ThresholdA_uV 必須除以探棒的倍率。例如探棒設定倍率是 x10，設定臨界電壓 1.6V 時，則傳入的 i64ThresholdA_uV 應該是 1600000/10。

i64ThresholdB_uV[in]:

Type: __int64

第二組觸發電壓，單位為 uV。同上，當使用的探棒的設定倍率非 x1 時，傳入的 i64ThresholdB_uV 必須除以探棒的倍率。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKSetStateTrig(int iSrcA, int iSrcB, int iNotA, int iNotB, int iAndOr, __int64 i64ThresholdA_uV, __int64 i64ThresholdB_uV)

功能:

設定觸發模式為碼型觸發。***(僅適用於TravelScope系列)***

參數:

iSrcA[in]:

Type: int

第一組觸發來源通道。

參數		說明
TRIG_SOURCE_CH1	0	通道 1
TRIG_SOURCE_CH2	1	通道 2
TRIG_SOURCE_EXT	2	外部觸發

iSrcB[in]:

Type: int

第二組觸發來源通道。

參數		說明
TRIG_SOURCE_CH1	0	通道 1
TRIG_SOURCE_CH2	1	通道 2
TRIG_SOURCE_EXT	2	外部觸發

iNotA[in]:

Type: int

第一組觸發訊號反向。

iNotB[in]:

Type: int

第二組觸發訊號反向。

iAndOr[in]:

Type: int

請參考以下表格填入所需參數。

參數		說明
TRIG_PATTERN_ANDOR_AND	0	邏輯AND
TRIG_PATTERN_ANDOR_OR	1	邏輯OR

i64ThresholdA_uV[in]:

Type: __int64

第一組觸發電壓，單位為 uV。需要注意的是，當使用的探棒的設定倍率非 x1 時，傳入的 i64ThresholdA_uV 必須除以探棒的倍率。例如探棒設定倍率是

x10，設定臨界電壓 1.6V 時，則傳入的 i64ThresholdA_uV 應該是 1600000/10。

i64ThresholdB_uV[in]:

Type: __int64

第二組觸發電壓，單位為 uV。同上，當使用的探棒的設定倍率非 x1 時，傳入的 i64ThresholdB_uV 必須除以探棒的倍率。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKSetTrigCouple(int iTrigCoupleFlag)
BOOL uDsoSDKGetTrigCouple(int & iTrigCoupleFlag)

功能:

設定/讀取通道擷取模式。預設值為REJECT_NONE。***(僅適用於TravelScope系列)***

參數:

iTrigCoupleFlag[in]:

Type: int

請參考以下表格填入所需參數。

參數		說明
REJECT_NONE	0	None
REJECT_HF	1	高頻拒斥 (High Frequency Reject)
REJECT_LF	2	低頻拒斥 (Low Frequency Reject)
REJECT_NOISE	3	雜訊拒斥 (Noise Reject)

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKCaptureEx( )

功能:

DSO 根據觸發設定開始擷取訊號。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

備註:

所有擷取參數及觸發參數設定必須在此函式之前執行。

BOOL uDsoSDKStop()

功能:

停止DSO資料擷取。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKForceTrig()

功能:

強迫DSO進入觸發狀態。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

備註:

當擷取模式為一般或單擊模式，且DSO擷取的訊號一直無法觸發成功時，可以使用該函式使DSO進入觸發狀態。

BOOL uDsoSDKReadIniFile(LPCSTR szFilePath)

功能:

透過Ini檔來設定DSO的各項參數。

參數:

szFilePath[in]:

Type: LPCSTR

Ini檔位置，可傳入相對路徑 Ex: ../DSOIni.ini，或是絕對路徑Ex: D:\Dso.ini

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

備註:

格式設定請參考Dso.ini及Dso_7Dev.ini內的格式及說明。

讀取 DSO 擷取狀態

int uDsoSDKGetErrorCodeEx()

功能:

當呼叫其他函式得到回傳的錯誤訊息時，可呼叫此函式來取得DSO的錯誤訊息。

回傳值:

參數		說明
DSO_ERROR_BADPOINTER	1	傳入空指標或是指標指向的記憶體空間不足
DSO_UNDEFINED_PARAMETER	2	傳入未定義之參數
DSO_CHANNELINDEX_OUT_RANGE	3	傳入通道編號大於啟動的示波器總通道數
DSO_DEVICEINDEX_OUT_RANGE	4	傳入示波器編號大於目前啟動的示波器數量
DSO_UNSUPPORT_VOLTDIV	5	傳入未支援的電壓刻度
DSO_VOLTPOS_OUT_RANGE	6	傳入電壓位置超出範圍
DSO_TRIGPOS_OUT_RANGE	7	觸發位置設定超出可允許範圍
DSO_TREASHOLD_OUT_RANGE	8	觸發電壓設定超出可允許範圍
DSO_WAIT_TIME_OUT_RANGE	9	觸發等待時間設定超出可允許範圍
DSO_TRIGGER_NOT_SUPPORT	10	使用的機種不支援該種觸發模式
DSO_VIDEO_TRIG_ERROR	11	在等效模式不能使用視訊觸發
DSO_TRIG_WAIT_ERROR	12	在等效模式不能使用等候模式
DSO_RECORD_LENGTH_ERROR	13	取樣點數超出範圍或取樣點數不是 8 的倍數
DSO_NO_HARDWARE	14	找不到 DSO 的硬體
DSO_UNKNOWN_ERROR	15	未知原因的錯誤
DSO_SAMPLE_RATE_ERROR	16	使用了不支援的取樣率
DSO_MEMORY_NOT_ENOUGH	17	記憶體不足
DSO_MEASURE_TYPE_ERROR	18	指定量測種類方式錯誤
DSO_MEASURE_RANGE_ERROR	19	指定量測功能起始/結束點錯誤
DSO_MEASURE_DATA_ERROR	20	量測時檢測到資料錯誤，無法產生計算結果
DSO_GROUP_INDEX_OUT_RANGE	21	DSO 群組範圍選擇超出範圍
DSO_MAIN_DLL_NOT_FOUND	22	DsoRun.dll 不存在
DSO_MAIN_DLL_FUNCTION_ERR	23	SDK 函式無法找到 DsoRun.dll 中特定函式的位置
DSO_MAIN_DLL_LOAD_ERR	24	SDK 函式無法加載 DsoRun.dll
DSO_GET_TEMP_NAME_ERR	25	SDK 函式無法取得暫存檔案名稱
DSO_GET_TEMP_DIR_ERR	26	SDK 函式無法找到 Windows 暫存資料夾
DSO_FILE_COPY_ERR	27	SDK 複製 DsoRun.dll 到暫存資料夾時發生錯誤
DSO_NOT_INITIAL	28	DSO 硬體尚未初始化
DSO_GROUP_PARAMETER_ERR	29	DSO Group 參數錯誤

DSO_FILE_NOT_EXIST	30	指定路徑檔案不存在
DSO_REINITIAL	31	DSO 已經初始完成，請先呼叫關閉函式後再重新進行初始化
DSO_USB_DRIVER_ERROR	32	DSO USB driver 出錯，請重新插拔 DSO 來排除此問題
DSO_USB_UNPLUG	33	DSO USB 線已被拔除
DSO_STACK_ID_ERROR	34	傳入的 DSO 序號格式不符合規定，請參考使用手冊中相關描述
DSO_MUST_NOT_INITIAL	35	DSO 環境設定必須在尚未初始前執行
DSO_READ_EMPTY	36	示波器資料讀取錯誤，必須先擷取過後才能讀取資料
DSO_SEQUENCE_ERROR	37	函式的前置處理函式尚未完成，請先呼叫並完成前置處理函式
DSO_DLL_VER_TOO_OLD	38	DSORun.dll 版本過舊，最低需求版本為 1.4.0.2
DSO_VOLTOFS_ERROR	39	Voltage Offset 設定超出規格範圍
DSO_DELAYTRIG_ERROR	40	觸發延遲時間超出可設定範圍 (0~50s)
DSO_TRIG_HOLDOFF_ERROR	41	觸發延滯時間超出可設定範圍 (0~10s)
DSO_PASSFAIL_VALUE_ERROR	48	Pass/Fail 容許範圍超出目前使用的 VoltDiv/TimeDiv 設定

int uDsoSDKGetStatus(int iDev)

功能:

取得各裝置的擷取狀態。

參數:

iDev[in]:

Type: int

指定DSO裝置的編號，從0為開始為第一台。

回傳值:

回傳值	說明
9	正在擷取 Pre-Trigger 的資料內容
5	正在等待觸發訊號
3	正在擷取 Post-Trigger 的資料內容
1	正在讀取資料

0, 2	資料擷取完成
32	USB Driver 錯誤
64	硬體裝置已拔除

BOOL uDsoSDKDataReady()

功能:

檢查該群組內所有裝置的擷取狀態是否完成。

回傳值:

如果擷取的動作結束，則回傳值為 **(TRUE)**。

而如果擷取動作未結束，則回傳值為 **(FALSE)**。

讀取 DSO 波形及量測功能

DSO 讀取資料、量測功能以 RAW Data 格式為主，FFT 功能則以 Double (mV) 格式為主，此 SDK DLL 中有提供函式可將 RAW Data 格式轉為 Double(mV)或是 Double(uV)格式。

BOOL uDsoSDKReadExRaw(int iDev, int* piFlag, short* lpsData, double pdbYofsA[2], double pdbYofsB[2], double pdbYMul[2] , int *piTrigOfs)

功能:

讀取DSO所擷取的波形資料及校正參數。

參數:

iDev [in]:

Type: int

填入欲讀取資料的裝置。

piFlag [out]:

Type: int *

回傳擷取狀態旗標。

旗標		說明
DSDF_ETSMODE	0x00002000	此次擷取模式為 ETS 模式。
DSDF_DBLMODE	0x00004000	此次擷取模式使用單一通道擷取模式，取樣率提高至 200MS/s。(適用於 DS1000 系列)
DSDF_TIMEOUT	0x00008000	表示該次觸發在觸發等待時間內，並沒有收到有效

		的觸發訊號。
DSDF_TRIGSRC_CH1	0x00010000	表示觸發來源為 Channel 1
DSDF_TRIGSRC_CH2	0x00020000	表示觸發來源為 Channel 2

lpsData [out]:

Type: short *

放波形資料的緩衝區，每次讀取波形資料時，不管該通道是被設定為 On 或 Off ， 都會同時讀取 2 個通道的資料。緩衝區的前半段為 CH1 資料，後半段為 CH2 資料。配置波形資料緩衝區大小的計算式如下

Buffer Size = Record Length * 2 Channel * Sizeof (short)

Record Length：

根據 Capture 時所設定之記錄長度，以取樣點為為單位。

pdbYofsA, pdbYofsB, pdbYMul [out]:

Type: double *

校正參數。

piTrigOfs [out]:

Type: int *

用以讀取 Trigger 偏差值，單位為 ps。

若填入 NULL 則由 DLL 協助計算觸發偏移。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

範例:

如何取出波形資料及資料換算:

```
short       *pWaveData;
short       *pCH1, *pCH2;
int         iData_CH1, iData_CH2;
pWaveData = new short[2 * iRecordLength];      // Prepare the waveform buffer
pCH1 = m_pWaveData;                            // Set CH1 waveform pointer
pCH2 = m_pWaveData + iRecordLength;            // Set CH2 waveform pointer

// 換算第一個點的數值，單位是 V
iData_CH1 = ((pCH1[0]– dYOfsA[0] ) * dYMul[0] + dYOfsB[0] – VOffset) * Probe;
iData_CH2 = ((pCH2[0]– dYOfsA[1] ) * dYMul[1] + dYOfsB[1] – VOffset) * Probe;
```

注意:

請依照上述步驟存取波形，即使該通道顯示是關閉(OFF)的狀態。

BOOL uDsoSDKRawToDbl_mv(short* lpsSrc, double * lpdbDst, int iLength, double dbYofsA, double dbYofsB, double dbYMul, double dbVOffset_uv, int iProbe)
BOOL uDsoSDKRawToDbl_uv(short* lpsSrc, double * lpdbDst, int iLength, double dbYofsA, double dbYofsB, double dbYMul, double dbVOffset_uv, int iProbe)

功能:

將DSO讀取到的波形資訊從RAW Data格式轉為Double (mV)或是Double (uV)格式。

參數:

lpsSrc[in]:

Type: short *

DSO讀取到的波形資訊，為RAW Data格式。

lpdbDst[out]:

Type: double *

轉換過後的波形資訊，為double格式。

iLength[in]:

Type: int

傳入的波形資訊長度。

dbYofsA, dbYofsB, dbYMul[in]:

Type: int

校正資訊。

dbVOffset_uv[in]:

Type: double

示波器電壓偏移值，單位(uV)。

iProbe[in]:

Type: int

目前所使用的探棒倍率。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKGetFFTData(double* lpdbSrc_mv, int iRecordLength, int iType, int iWindow, double* lpdbFFT)

功能:

快速傅利葉轉換(FFT)。

參數:

lpdbSrc_mv[in]:

Type: double *

DSO讀取到的波形資訊，需經由uDSOSDKRawToDbl_mv轉為double格式。

iRecordLength[in]:

Type: int

傳入的波形資訊長度。

iType[in]:

Type: int

該參數用來選擇FFT刻度。

參數		說明
FFT_TYPE_LINEAR_RMS	0	線性均方根
FFT_TYPE_DBV_RMS	1	dBV均方根
FFT_TYPE_DBM_RMS	2	dBm均方根

iWindow[in]:

Type: int

該參數用來選擇FFT視窗。

參數		說明
FFT_WINDOW_RECTANGULAR	0	Rectanglar
FFT_WINDOW_BLACKMAN	1	Blackman
FFT_WINDOW_HANN	2	Hann
FFT_WINDOW_HAMMING	3	Hamming
FFT_WINDOW_HARRIS	4	Harris
FFT_WINDOW_TRIANGULAR	5	Triangular
FFT_WINDOW_COSINE	6	Cosine
FFT_WINDOW_LANCZOS	7	Lanczos
FFT_WINDOW_GUASS	8	Guass

***lpdbFFT[out]:**

Type: double

儲存波形資料的緩衝區，需準備和iRecordLength大小相同的記憶體緩衝空間。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

LPDSMEAS 結構介紹

```
typedef struct _MEASUREDATA
{
    __int64     i64ScData;      // 讀回的資料是以多少Sample Clock擷取的(單位為Hz)
    int         iRecordLength;
    int         iVoltDiv[2];
    double      dYOfsA[2];
    double      dYOfsB[2];
    double      dYMul[2];       // Volt(mV) = ( Raw Data - dYOfsA ) * dYMul + dYOfsB
    double      dVOfs[2];
    int         iProbe[2];
    LPWORD      lpwWaveData[2];
}DSMEAS, FAR *LPDSMEAS;
```

此結構用以儲存 DSO 量測時正副通道之參數，其中副通道的參數只有在開啟部分計算功能，如相位差、上升延遲等需要參考其他通道數據時才會使用。

i64ScData:

擷取資料時的取樣率，單位為 S/s。

iRecordLength:

擷取資料時使用的記憶體長度，單位為點。

iVoltDiv:

擷取資料時主、副通道分別使用的電壓刻度，單位為 uV。

dYOfsA、dYOfsB、dYMul:

擷取資料時主、副通道所對應的校正參數。

dbVoltOfs:

擷取資料時主、副通道使用的電壓偏移值，單位為 uV。

iProbe:

擷取資料時主、副通道設定的探棒倍率，Ex: x1, x10, x100。

lpwWaveData:

主、副通道的波形資料，資料取得方式限定為 DDEX_RAW 才能夠做計算。

BOOL uDsoSDKMeasurement(int * piType, int iStart, int iEnd, LPDSMEAS lpDsMeas, bool * pfForceStop, bool * pfResult, double * dbValue)

功能:

設定DSO量測功能參數。

參數:

piType[in]:

Type: int *

依序傳入需要量測類別陣列，最後需以MEASURE_END結尾。

參數		說明
MEASURE_FREQ	0	頻率
MEASURE_PERIOD	1	週期
MEASURE_VMAX	2	最大值
MEASURE_VMIN	3	最小值
MEASURE_VHIGH	4	高值
MEASURE_VLOW	5	低值
MEASURE_VPP	6	峰對峰值
MEASURE_VAMP	7	振幅
MEASURE_RMS	8	均方根
MEASURE_VMEAN	9	平均值
MEASURE_HIGH_DUTY	10	正工作週期
MEASURE_LOW_DUTY	11	負工作週期
MEASURE_HIGH_PERIOD	12	正脈波寬
MEASURE_LOW_PERIOD	13	負脈波寬
MEASURE_RISETIME	14	上升時間
MEASURE_FALLTIME	15	下降時間
MEASURE_POVERSHOOT	16	正過激
MEASURE_NOVERSHOOT	17	負過激
MEASURE_VMID	18	中間值
MEASURE_CYCLE_RMS	19	週期均方根
MEASURE_CYCLE_MEAN	20	週期平均值
MEASURE_RISE_DELAY	21	兩通道間的上升延遲
MEASURE_FALL_DELAY	22	兩通道間的下降延遲
MEASURE_RPRESHOOT	23	上升前衝
MEASURE_FPRESHOOT	24	下降前衝
MEASURE_PHASE	25	相位差

MEASURE_FREQ_AVG	26	平均頻率
MEASURE_RISE_EDGE	27	上升緣數
MEASURE_FALL_EDGE	28	下降緣數
MEASURE_ALL_EDGE	29	變化緣數
MEASURE_END	-1	結尾標記

iStart[in]:

Type: int

選擇量測的起始位置，輸入0代表整段記憶體的起始點，單位：取樣點。

iEnd[in]:

Type: int

選擇量測的結束位置，輸入0代表整段記憶體的起始點，單位：取樣點。

lpDsMeas[in]:

Type: LPDSMEAS

填入量測參數結構，包含各通道的波形資料及部分擷取參數。

pfForceStop[in]:

Type: bool *

強制中斷計算函式的旗標，在多工架構下可以透過此旗標中斷量測的計算。

pfResult[out]:

Type: bool *

由外部準備一段與**piType**數量相同的記憶體，函式執行後會回傳量測的結果，若量測成功時回傳**TRUE**，無法得到量測值時回傳**FALSE**。

pdbValue[out]:

Type: bool *

由外部準備一段與**piType**數量相同的記憶體，函式執行後會回傳量測的數值。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

範例:

同時量測頻率及均方根值時，參數設定及回傳如下:

執行此函式前設定

```
int iType[3] = {
    MEASURE_FREQ,
    MEASURE_RMS,
    MEASURE_END
};
```

```
bool fResult[3] = {0};
double dbValue[3] = {0};
```

執行此函式後

fResult[0] 代表 MEASURE_FREQ 的計算成功與否。

fResult[1] 代表 MEASURE_RMS 的計算成功與否。

dbValue[0] 代表 MEASURE_FREQ 的計算結果值。

dbValue[1] 代表 MEASURE_RMS 的計算結果值。

BOOL uDsoSDKMeasExInit(int iCh, int iVoltDiv, double dbVOfs, double dbYOfsA, double dbYOfsB, double dbYMul, int iProbe, __int64 iScData, int iBufLen, LPWORD lpwWaveform)

功能:

初始化DSO量測功能參數，此Ex系列函式和標準函式功能相同，僅將參數拆開成多個函式方便C#等程式語言編寫。

參數:

iCh[in]:

Type: int

指定後方的資料是屬於哪一個通道

iVoltDiv [in]:

Type: int

設定擷取波形使用的電壓刻度，單位: uV。

dbVOfs[in]:

Type: int

設定擷取波形使用的電壓偏移，單位: uV。

dbYOfsA, dbYOfsB, dbYMul[in]:

Type: double

擷取校正參數，須從硬體讀取此參數後傳入此函式中方能得到正確的計算值。

iProbe[in]:

Type: int

設定擷取波形使用的探棒倍率，填入1, 10, 100代表x1, x10, x100倍率的探棒。

iScData[in]:

Type: __int64

設定擷取波形使用的取樣率，單位: Hz。

iBufLen[in]:

Type: int

設定擷取波形總長度，單位: 取樣點。

lpwWaveform [in]:

Type: unsigned short*

設定擷取到的波形資料陣列。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKMeasExAddType(int iType)

功能:

設定欲啟用之DSO量測功能項目，此Ex系列函式和標準函式功能相同，僅將參數拆開成多個函式方便C#等程式語言編寫。

參數:

iType[in]:

Type: int

傳入需要的量測類別。

參數		說明
MEASURE_FREQ	0	頻率
MEASURE_PERIOD	1	週期
MEASURE_VMAX	2	最大值
MEASURE_VMIN	3	最小值
MEASURE_VHIGH	4	高值
MEASURE_VLOW	5	低值
MEASURE_VPP	6	峰對峰值
MEASURE_VAMP	7	振幅
MEASURE_RMS	8	均方根
MEASURE_VMEAN	9	平均值
MEASURE_HIGH_DUTY	10	正工作週期
MEASURE_LOW_DUTY	11	負工作週期
MEASURE_HIGH_PERIOD	12	正脈波寬
MEASURE_LOW_PERIOD	13	負脈波寬
MEASURE_RISETIME	14	上升時間
MEASURE_FALLTIME	15	下降時間
MEASURE_POVERSHOOT	16	正過激

MEASURE_NOVERSHOOT	17	負過激
MEASURE_VMID	18	中間值
MEASURE_CYCLE_RMS	19	週期均方根
MEASURE_CYCLE_MEAN	20	週期平均值
MEASURE_RISE_DELAY	21	兩通道間的上升延遲
MEASURE_FALL_DELAY	22	兩通道間的下降延遲
MEASURE_RPRESHOOT	23	上升前衝
MEASURE_FPRESHOOT	24	下降前衝
MEASURE_PHASE	25	相位差
MEASURE_FREQ_AVG	26	平均頻率
MEASURE_RISE_EDGE	27	上升緣數
MEASURE_FALL_EDGE	28	下降緣數
MEASURE_ALL_EDGE	29	變化緣數
MEASURE_END	-1	結尾標記

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKMeasExStart(int iStart, int iEnd, bool * pbForceStop)

功能:

根據先前輸入的擷取參數進行資料計算，須先執行uDsoSDKMeasExInit及uDsoSDKMeasExAddType以提供擷取環境設定。

此Ex系列函式和標準函式功能相同，僅將參數拆開成多個函式方便C#等程式語言編寫。

參數:

iStart[in]:

Type: int

選擇量測的起始位置，輸入0代表整段記憶體的起始點，單位：取樣點。

iEnd[in]:

Type: int

選擇量測的結束位置，輸入0代表整段記憶體的起始點，單位：取樣點。

pfForceStop[in]:

Type: bool *

強制中斷計算函式的旗標，在多工架構下可以透過此旗標中斷量測的計算。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

BOOL uDsoSDKMeasExGetResult(int iType)

功能:

回傳資料計算的結果，此Ex系列函式和標準函式功能相同，僅將參數拆開成多個函式方便C#等程式語言編寫。

參數:

iType[in]:

Type: int

傳入量測類別。

回傳值:

若指定量測類別的計算結果成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

double uDsoSDKMeasExGetValue(int iType)

功能:

回傳資料計算的內容，此Ex系列函式和標準函式功能相同，僅將參數拆開成多個函式方便C#等程式語言編寫。

參數:

iType[in]:

Type: int

傳入量測類別。

回傳值:

回傳指定量測類別的計算內容，根據量測類別不同單位也不相同。

BOOL uDsoSDKMeasExFreeResource()

功能:

清除Ex系列量測函式所額外配置的記憶體。

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

訊號產生器控制函式

BOOL uDsoSDKFGSetting(int iDev, LPVOID lpvData)

功能:

設定FG。***(僅適用於TravelScope系列)***

參數:

iDev[in]:

Type:int

指定DSO裝置的編號，從0為開始為第一台。

lpvData[in]:

Type:LPVOID

設定FG結構。

FG輸入參數：

參數		說明
FGSDK_DATA_CH1 FGSDK_DATA_CH2	0x0000 0x0005	填入指定通道的基本參數，對應的結構為 **CarrierParameter**
FGSDK_CW_MODE_CH1 FGSDK_CW_MODE_CH2	0x0001 0x0006	將指定通道的運作模式改為連續輸出模式，參數傳入 **NULL** 即可
FGSDK_MODUL_MODE_CH1 FGSDK_MODUL_MODE_CH2	0x0002 0x0007	將指定通道的運作模式改為調變模式，對應的結構為 **ModulationParameter**
FGSDK_SWEEP_MODE_CH1 FGSDK_SWEEP_MODE_CH2	0x0003 0x0008	將指定通道的運作模式改為掃描模式，對應的結構為 **SweepParameter**
FGSDK_BURST_MODE_CH1 FGSDK_BURST_MODE_CH2	0x0004 0x0009	將指定通道的運作模式改為脈波模式，對應的結構為 **BurstParameter**
FGSDK_TRIG_DATA	0x000A	填入觸發參數，對應的結構為 **TriggerParameter**
FGSDK_SENDPTN	0x000B	控制 **FG** 將波形輸出，參數陣列傳入 **NULL** 即可
FGSDK_PHASERESET	0x000C	重置 **FG** 兩個通道的相位， 參數陣列傳入 **NULL** 即可
FGSDK_ARRAY_MODE	0x1000	不使用結構，直接傳入資料陣列，可與其他參數合併使用

FG 輸入資料結構：

typedef struct _CARRIER_PARAMETER

```
{
    double dbID;            // FG功能選擇參數
    int iWaveFunction;      // 載波的波形
    double dbFreq;          // 載波的頻率，調整範圍為0.01Hz ~ 1MHz
    double dbPhase;         // 載波的起始相位，調整範圍為 -180~180度
    double dbDuty;          // 選擇脈衝波時為工作週期，調整範圍為 0~100%
                            // 選擇鋸齒波時為對稱性，調整範圍為 0~100%
    double dbLeading;       // 脈衝波前緣時間 (μs)
    double dbTrailing;      // 脈衝波後緣時間 (μs)
    double dbVoltAmp;       // 該通道輸出電壓的振幅，調整範圍為 0~2.5V
    double dbVoltOffset;    // 該通道輸出電壓的偏移，固定為振幅的一半
                            // 載波波形為直流時，此數值的調整範圍為0~2.5V
} CarrierParameter;
```

iWaveFunction:

參數		說明
FGSDK_WAVE_DC	0x0000	直流，可藉由偏移值來控制直流位準
FGSDK_SINE	0x0001	正弦波
FGSDK_SQUARE	0x0002	方波
FGSDK_TRIANGLE	0x0003	三角波
FGSDK_RAMP	0x0004	鋸齒波，可調整對稱性來改變鋸齒波的波形
FGSDK_PULSE	0x0005	脈衝波，可調整工作週期/前緣/後緣來控制脈衝波的波形

```
typedef struct _MODULATION_PARAMETER
{
    double dbID;            // FG功能選擇參數
    int iWaveFunction;      // 調變訊號的波形
    double dbFreq;          // 調變訊號的頻率，調整範圍為 0.01Hz~1MHz
    double dbPhase;         // 調變訊號的相位，調整範圍為 -180~180度
    double dbDuty;          // 選擇脈衝波時為工作週期，調整範圍為 0~100%
                            // 選擇鋸齒波時為對稱性，調整範圍為 0~100%
    double dbLeading;       // 脈衝波前緣時間 (μs)
    double dbTrailing;      // 脈衝波後緣時間 (μs)
    int iModulType;         // 調變種類
    double dbModulScale;    // 調變參數
} ModulationParameter;
```

iWaveFunction:

參數		說明
FGSDK_WAVE_DC	0x0000	調變模式不支援直流
FGSDK_SINE	0x0001	正弦波
FGSDK_SQUARE	0x0002	方波
FGSDK_TRIANGLE	0x0003	三角波
FGSDK_RAMP	0x0004	鋸齒波，可調整對稱性來改變鋸齒波的波形
FGSDK_PULSE	0x0005	脈衝波，可調整工作週期/前緣/後緣來控制脈衝波的波形

iModulType:

參數		說明
FGSDK_MODUL_AM	0x0000	AM，調變參數的範圍為 ±100%
FGSDK_MODUL_FM	0x0001	FM，調變參數的範圍為 ±目前載波的頻率
FGSDK_MODUL_PM	0x0002	PM，調變參數的範圍為 ±180度
FGSDK_MODUL_ASK	0x0003	ASK，調變參數的範圍為 ±100%
FGSDK_MODUL_FSK	0x0004	FSK，調變參數的範圍為 0Hz~1MHz
FGSDK_MODUL_PSK	0x0005	PSK，調變參數的範圍為 ±180度

```
typedef struct _SWEEP_PARAMETER
{
    double dbID;            // FG功能選擇參數
    bool bSweepMode;        // 掃描模式
    bool bSweepType;        // 掃描種類
    double dbStartFreq;     // 起始頻率，調整範圍為 1Hz ~ 1MHz
    double dbStopFreq;      // 結束頻率，調整範圍為 1Hz ~ 1MHz
    double dbSweepTime;     // Sweep time，調整範圍為 1μs ~ 100s
    double dbHoldTime;      // Hold time，調整範圍為 1μs ~ 100s
    double dbReturnTime;    // Return time，調整範圍為 1μs ~ 100s
} SweepParameter;
```

bSweepMode:

參數		說明
FGSDK_SWEEP_MODE_TRIG	0x0000	觸發模式，每次觸發時執行一次掃描

FGSDK_SWEEP_MODE_REPEAT	0x0001	連續模式，每次掃描完後緊接著下一次掃描

bSweepType:

參數		說明
FGSDK_SWEEP_TYPE_LINEAR	0x0000	線性模式，掃描時頻率以線性的方式做增減
FGSDK_SWEEP_TYPE_LOG	0x0001	對數模式，掃描時頻率以對數的方式做增減

```
typedef struct _BURST_PARAMETER
{
    double dbID;            // FG功能選擇參數
    int iBurstCycle;        // 脈波次數, 調整範圍為 1 ~ 99999
    int iTrigDelay;         // 觸發延遲, 調整範圍為 1μs ~ 343s
} BurstParameter;

typedef struct _TRIG_PARAMETER
{
    double dbID;            // FG功能選擇參數
    bool bTrigSource;       // 觸發來源
    bool bTrigEdge;         // 觸發緣選擇: 上升緣 / 下降緣
    double dbTrigFreq;      // 內部觸發頻率, 調整範圍 0.01Hz ~ 1MHz
} TriggerParameter;
```

bTrigSource:

參數		說明
FGSDK_TRIG_TYPE_INT	0x0000	由FG內部產生的訊號作觸發，需設定觸發週期
FGSDK_TRIG_TYPE_EXT	0x0001	由外部訊號進行觸發，需設定觸發緣

bTrigEdge:

參數		說明
FGSDK_EDGE_FALLING	0x0000	下降緣

FGSDK_EDGE_RISING	0x0001	上升緣

```
typedef struct _NO_PARAMETER
{
    double dbID;            // FG功能選擇參數
}NoParameter;
```

回傳值:

若該函式執行成功，該回傳值為 **(TRUE)**。否則回傳值為 **(FALSE)**。

Ex1 Struct type:

```
CarrierParameter sCh1Data = {
    FGSDK_DATA_CH1,     // FG function index
    FGSDK_PULSE,        // Wave function
    1000,               // Frequency
    0,                  // Phase
    35,                 // Duty cycle
    100,                // Leading time
    200,                // Trailing time
    2.0,                // Voltage amplitude
    1.0,                // Voltage offset
};
CarrierParameter sCh2Data = {
    FGSDK_DATA_CH2,     // FG function index
    FGSDK_SINE,         // Wave function
    1500,               // Frequency
    20,                 // Phase
    0,                  // Duty cycle
    0,                  // Leading time
    0,                  // Trailing time
    2.5,                // Voltage amplitude
    1.25,               // Voltage offset
};
ModulationParameter sCh1ModulData = {
    FGSDK_MODUL_MODE_CH1, // FG function index
```

```
        FGSDK_SINE,                 // Wave function
        100,                        // Frequency
        0,                          // Phase
        0,                          // Duty cycle
        0,                          // Leading time
        0,                          // Trailing time
        FGSDK_MODUL_AM,             // Modulation type
        100,                        // Modulation scale
};

TriggerParameter sTrigParameter =       {
        FGSDK_TRIG_DATA,            // FG function index
        FGSDK_TRIG_TYPE_INT,        // Trigger source
        FGSDK_EDGE_RISING,          // Trigger edge
        100,                        // Internal Trigger source frequency
};
NoParameter sCh2CWMode = {
        FGSDK_CW_MODE_CH2
};
NoParameter sSendPtn =      {
        FGSDK_SENDPTN
};
//Set Ch1 carrier parameters
puDsoSetting(iDevice, DSS_FGCONTROL, & sCh1Data);
//Set Ch2 carrier parameters
puDsoSetting(iDevice, DSS_FGCONTROL, & sCh2Data);
//Set Ch1 to Modulation mode
puDsoSetting(iDevice, DSS_FGCONTROL, & sCh1ModulData);
//Set Ch2 to Continuous Wave mode
puDsoSetting(iDevice, DSS_FGCONTROL, & sCh2CWMode);
//Set Trigger data
puDsoSetting(iDevice, DSS_FGCONTROL, & sTrigParameter);
//Send FG pattern
puDsoSetting(iDevice, DSS_FGCONTROL, & sSendPtn);
```

Ex2 Array type:

```
double pdbParaCh1[9] = {
    FGSDK_DATA_CH1 | FGSDK_ARRAY_MODE, // FG function index
    FGSDK_PULSE,          // Wave function
    1500,                 // Frequency
    20,                   // Phase
    20,                   // Duty cycle
    30,                   // Leading time
    70,                   // Trailing time
    2.5,                  // Voltage amplitude
    1.25,                 // Voltage offset
};
double pdbModulCh1[9] = {
    FGSDK_MODUL_MODE_CH1| FGSDK_ARRAY_MODE, // FG function index
    FGSDK_SINE,           // Wave function
    100,                  // Frequency
    0,                    // Phase
    0,                    // Duty cycle
    0,                    // Leading time
    0,                    // Trailing time
    FGSDK_MODUL_AM,       // Modulation type
    100,                  // Modulation scale
};
double pdbParaCh2[9] = {
    FGSDK_DATA_CH2 | FGSDK_ARRAY_MODE, // FG function index
    FGSDK_SINE,           // Wave function
    5500,                 // Frequency
    0,                    // Phase
    0,                    // Duty cycle
    0,                    // Leading time
    0,                    // Trailing time
    1.5,                  // Voltage amplitude
    0.75,                 // Voltage offset
};
```

```
double sTrigParameter[4] =
{
    FGSDK_TRIG_DATA | FGSDK_ARRAY_MODE, // FG function index
    FGSDK_TRIG_TYPE_INT,      //Trigger source
    FGSDK_EDGE_RISING,        //Trigger edge
    100,                      //Internal Trigger source frequency
};

double dbCh2CwMode = FGSDK_CW_MODE_CH2;
double dbSendPtn = FGSDK_SENDPTN;

//Set Ch1 carrier parameters
puDsoSetting(iDevice, DSS_FGCONTROL, pdbParaCh1);
//Set Ch2 carrier parameters
puDsoSetting(iDevice, DSS_FGCONTROL, & pdbParaCh2);
//Set Ch1 to Modulation mode
puDsoSetting(iDevice, DSS_FGCONTROL, & pdbModulCh1);
//Set Ch2 to Continuous Wave mode
puDsoSetting(iDevice, DSS_FGCONTROL, & dbCh2CwMode);
//Set Trigger data
puDsoSetting(iDevice, DSS_FGCONTROL, & sTrigParameter);
//Send FG pattern
puDsoSetting(iDevice, DSS_FGCONTROL, & dbSendPtn);
```

皇晶科技股份有限公司
http://www.acute.com.tw/
地址：新北市三重區24159重新路五段609巷12號6樓之7 (湯城園區)
電話：(02)2999-3275
傳真：(02)2999-3276
E-mail: service@acute.com.tw