14390 Carlson Circle
Tampa, FL 33626
+1 (813) 855-8687

SWDLL.DLL library

About

This document describes the SpectraWiz SWDLL library. You can find Borland Delphi Pascal and C++ examples below.

Functions & procedures

SWDscan

function SWDscan(chan: integer; var buffers: aptype): longint; stdcall; external 'swdll' index 2;

Acquire spectral scan from channels 1 to 8.

Parameters

Parameter	Type
chan	***integer***
buffers	***aptype*** aptype = ^atype *(pointer to atype)* atype = array[0..2050] *(same as float)*

Return

Code	Description
0	*Success*
1	*Busy*
2	*Timeout*
3	*Fatal (nil buffer)*

SWDscanLV

function SWDscanLV(chan: integer; var buffers: atype): longint; stdcall; external 'swdll' index 8;

Acquire spectral scan similar to SWDscan for LabView and C++ programs.
Caller allocates buffer then call to SWDscanLV fills it.

Parameters

Parameter	Type
chan	***integer***
buffers	***aptype*** aptype = ^atype *(pointer to atype)* atype = array[0..2050] *(same as float)*

14390 Carlson Circle
Tampa, FL 33626
+1 (813) 855-8687

SWDinit

procedure SWDinit; stdcall; external 'swdll' index 1;

Startup (call only once)

SWDrate

procedure SWDrate(dsf: integer); stdcall; external 'swdll' index 3;

Change detector integration rate in milli-seconds (4-65500).

Parameters

Parameter	Type
dsf	***integer***

SWDclose

procedure SWDclose; stdcall; external 'swdll' index 4;

Shutdown (call only once).

SWDupdate

procedure SWDupdate(ScansToAvg,xsmooth,TempComp: integer); stdcall; external 'swdll' index 5;

Dynamic update for data processing configuration.

Parameters

Parameter	Type	Note
ScansToAvg	***1…99***	scan averaging accumulates scans then divides by *ScansToAvg*
xsmooth	***0**=none* ***1**=5_pixel* ***2**=9_pixel* ***3**=17_pixel* ***4**=33_pixel*	smoothing uses pixel boxcar averaging across spectral scan
TempComp	***0**=none* ***1**=on*	temperature compensation uses detector optical black region pixels 1-12

14390 Carlson Circle
Tampa, FL 33626
+1 (813) 855-8687

SWDxtmode

procedure SWDxtmode(xtmode: integer); stdcall; external 'swdll' index 6;

Set *xtiming* OEM mode for capable units 0=off 1=on (RESERVED for OEMs).

Parameters

Parameter	Type
xtmode	***0**=off* ***1**=on (RESERVED for OEMs)*

SWDxtrate

procedure SWDxtrate(xtrate: integer); stdcall; external 'swdll' index 7;

Set *xtiming* resolution clock rate 0-2 where 0=fastest & lowest resolution.

Parameters

Parameter	Type
xtrate	***0-2*** *Where 0=fastest & lowest resolution*

14390 Carlson Circle
Tampa, FL 33626
+1 (813) 855-8687

Example 1: Borland Delphi Pascal

```
//
// Main data access routine
// Call SWDLL.DLL to acquire spectral data scan
// this routine is repeatedly called using a timer that is set to
//      the desired display refresh rate.
//
procedure TForm1.DoScan(Sender: TObject);
var                                 // initiated by app timer
i,v: integer;
scode: longint;
uok: word;
label SEXIT,SEXIT2,NSCAN;

begin
try
if not busy then                    // force one requestor at a time
    begin                           // and turn others away
    busy:=true;

    NSCAN:
    scode:= SWDscan(xchan, buffersx);// call SWDll xchan (1..8)
    if scode>0 then begin           // SWDll uses pchan[1..8] physical chan
        if scode=1 then begin       // SWDLL busy
            if dsf>30 then UpdateCursor;
            goto SEXIT;
            end;
        if scode=2 then begin
            uok:=MessageDlg('Scan Timeout', mtError, [mbRetry,mbIgnore], 0);
            if uok=mrRetry then begin
                SWDrate(dsf);       // re-establish data sample frequency
                goto NSCAN;         // and request another scan
                end;
            if uok=mrIgnore then goto SEXIT2; // leave system busy
            end;
        MessageDlg('SWDll nil buffer '+IntToStr(scode), mtError, [mbOk], 0);
        goto SEXIT2;                        // leave system busy
        end;

    for i:=0 to 2050 do begin
        buffers[i]:=buffersx[i];    // copy to local buffer
        bufferx[i]:=buffersx[i];    // save to raw data buffer
        end;

    smodedone:=false;
    DoSmode(buffers);               // convert to ABSOR/TRANS

    if not multigraph then begin
        DoDisplay(buffers);         // render display graph
        if AutoScaleSet then begin  // new graph for autoscale
            AutoScaleSet:=false;
            DoPaint(sender);
            end;
         end

    SEXIT:
    busy:=false;
    SEXIT2:
```

14390 Carlson Circle
Tampa, FL 33626
+1 (813) 855-8687


```
    end
else
    UpdateStatus;                   // for data cursor panel

except                              // trap errors
   on E: EInvalidOp do {nothing};
        else begin     {assume fatal access violation}
        busy:=true;
        Form1.Timer1.Enabled:=false;
        MessageDlg('Hardware Interface Problem...'+#13+
                   '1. Check configuration.'+#13+
                   '2. Restart Windows.', mtError, [mbOK], 0);
        SWinstallset:=false;
        ShutDnSWD;
        Close;
        end;
    end;
end;
```

14390 Carlson Circle
Tampa, FL 33626
+1 (813) 855-8687

Example 2: Microsoft C++

```
//////////////////////////////////////////////////////////////////
// MicroSoft C++ example for SpectraWiz SWDLL spectrometer interface
// SkelletonView.cpp : implementation of the CSkelletonView class
// see below -> CSkelletonView::OnAcquireSpectrometerdata()
//
#include "stdafx.h"            // standard MS header junk
#include "Skelleton.h"         // ...
#include "SkelletonDoc.h"      // ...
#include "SkelletonView.h"     // ...
#include "SWDll.h"             // <--- see file listing at end BELOW

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

#define TRUE 1
#define FALSE 0

int running=TRUE;
int ValidData;
float dIntensity[2051];

/////////////////////////////////////////////////////////////////////////////
// CSkelletonView

IMPLEMENT_DYNCREATE(CSkelletonView, CView)

BEGIN_MESSAGE_MAP(CSkelletonView, CView)
      //{{AFX_MSG_MAP(CSkelletonView)
      ON_COMMAND(ID_ACQUIRE_SPECTROMETERDATA, OnAcquireSpectrometerdata)
      ON_WM_LBUTTONDOWN()
      //}}AFX_MSG_MAP
      // Standard printing commands
      ON_COMMAND(ID_FILE_PRINT, CView::OnFilePrint)
      ON_COMMAND(ID_FILE_PRINT_DIRECT, CView::OnFilePrint)
      ON_COMMAND(ID_FILE_PRINT_PREVIEW, CView::OnFilePrintPreview)
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CSkelletonView construction/destruction

CSkelletonView::CSkelletonView()
{
      // TODO: add construction code here

}

CSkelletonView::~CSkelletonView()
{
}

BOOL CSkelletonView::PreCreateWindow(CREATESTRUCT& cs)
{
      // TODO: Modify the Window class or styles here by modifying
      //  the CREATESTRUCT cs
```

14390 Carlson Circle
Tampa, FL 33626
+1 (813) 855-8687


```
	return CView::PreCreateWindow(cs);
}

/////////////////////////////////////////////////////////////////////////////
// CSkelletonView drawing

void CSkelletonView::OnDraw(CDC* pDC)
{
int i;

	CSkelletonDoc* pDoc = GetDocument();
	ASSERT_VALID(pDoc);


   // TODO: add draw code for native data here

   if( ValidData == TRUE ) {

      pDC->MoveTo(50, 300 - (int)(dIntensity[0] * (float).03 ));

      for( i=0 ; i<2048 ; i++ ) {
         pDC->LineTo( 50+(int)(  (float)i/(float)4  ),
                   300 - (int)(  dIntensity[i]*(float).03 ));
      }
   }

}

/////////////////////////////////////////////////////////////////////////////
// CSkelletonView printing

BOOL CSkelletonView::OnPreparePrinting(CPrintInfo* pInfo)
{
	// default preparation
	return DoPreparePrinting(pInfo);
}

void CSkelletonView::OnBeginPrinting(CDC* /*pDC*/, CPrintInfo* /*pInfo*/)
{
	// TODO: add extra initialization before printing
}

void CSkelletonView::OnEndPrinting(CDC* /*pDC*/, CPrintInfo* /*pInfo*/)
{
	// TODO: add cleanup after printing
}

/////////////////////////////////////////////////////////////////////////////
// CSkelletonView diagnostics

#ifdef _DEBUG
void CSkelletonView::AssertValid() const
{
	CView::AssertValid();
}

void CSkelletonView::Dump(CDumpContext& dc) const
{
```

14390 Carlson Circle
Tampa, FL 33626
+1 (813) 855-8687


```
	CView::Dump(dc);
}

CSkelletonDoc* CSkelletonView::GetDocument() // non-debug version is inline
{
	ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CSkelletonDoc)));
	return (CSkelletonDoc*)m_pDocument;
}
#endif //_DEBUG

/////////////////////////////////////////////////////////////////////////////
// CSkelletonView message handlers

void CSkelletonView::OnLButtonDown(UINT nFlags, CPoint point)
{
    CView::OnLButtonDown(nFlags, point);
    running = FALSE;
}


/////////////////////////////////////////////////////////////////////////////
// Main Spectrometer Data acquisition

void CSkelletonView::OnAcquireSpectrometerdata()
{

//*** Tell OnDraw to draw data.
ValidData = TRUE;

//*** Make the do loop active.
running = TRUE;

HANDLE hHandle;
DWORD Time1 = 100;
DWORD Time2 = 50;

LPSECURITY_ATTRIBUTES  lpEventAttributes;     // pointer to security attributes
BOOL  bManualReset; // flag for manual-reset event
BOOL  bInitialState;      // flag for initial state
LPCTSTR  lpName;    // pointer to event-object name

lpEventAttributes = NULL;
bManualReset = FALSE;
bInitialState = FALSE;
lpName = NULL;

int chan, scode;
int i, j, k, dsf;
int    error;
float  *buffer;
float  **ptr;

HINSTANCE hDLL;               // Handle to DLL

typedef void (_stdcall* FncPtrVoid)();
typedef void (_stdcall* FncPtrInt)(int);
typedef int  (_stdcall* FncPtrIntFloatP)(int,float**);
typedef int  (_stdcall* FncPtrIntFloatPLV)(int,float*);
```

14390 Carlson Circle
Tampa, FL 33626
+1 (813) 855-8687

```
FncPtrVoid SWDinit; // Function pointer
FncPtrInt SWDrate;
FncPtrIntFloatP SWDscan;
FncPtrIntFloatPLV SWDscanLV;
FncPtrVoid SWDclose;

dsf = 50;          // detector integration time in ms
chan = 1;          // spectrometer chanel number
ptr = &buffer;

buffer = (float *) malloc( 2051*sizeof(float) );

hDLL = LoadLibrary("C:\\SWDLL\\swdll.dll");
if (hDLL != NULL) {

    SWDinit = (FncPtrVoid)GetProcAddress(hDLL, "SWDinit");
    SWDrate = (FncPtrInt)GetProcAddress(hDLL, "SWDrate");
    SWDscan = (FncPtrIntFloatP)GetProcAddress(hDLL, "SWDscan");
    SWDscanLV = (FncPtrIntFloatPLV)GetProcAddress(hDLL, "SWDscanLV");
    SWDclose = (FncPtrVoid)GetProcAddress(hDLL, "SWDclose");

    SWDinit();    // initialize SWDLL
    SWDrate(dsf); // set detector integration
                  // optional call to override default in SW.INI set by SWDinit
  MSG msg;

    do {
         scode = SWDscanLV( chan, buffer);  // get spectral scan

        switch (scode) {
            case 0: // valid scan
                      break;
            case 1: // busy
                    Sleep(4);
                    goto noscan;
                    break;
            case 2: // timeout
                    MessageBox("Scan timeout", "SWDLL TEST", MB_OK );
                        goto noscan;
                    break;
            case 3: // fatal error
                    MessageBox("Fatal Error", "SWDLL TEST", MB_OK );
                        goto noscan;
                    break;
            }       // end case

         for (i=0; i<2050; i++) dIntensity[i] = buffer[i];  // copy spectral data

        //Draw the data
        Invalidate( TRUE );
        CView::OnPaint();

         //Get and process any waiting messages.
         while (PeekMessage(&msg, NULL, 0, 0, PM_REMOVE)) {
             if(msg.message==WM_LBUTTONDOWN) running = FALSE;
            TranslateMessage(&msg);
             DispatchMessage(&msg);
             }
noscan:;
```

14390 Carlson Circle
Tampa, FL 33626
+1 (813) 855-8687

```
      } while( running );  // end do

//*********************************************************************

   SWDclose();               // allow SWDLL to release its resources
   FreeLibrary(hDLL);      // release dynamic link library SWDLL.DLL
   }  // end if (hDLL != NULL)

free (buffer);             // release buffer acquired via malloc
}    // end of CSkelletonView::OnAcquireSpectrometerdata()


////////////////////////////////////////////////////////////////////
// from SWDLL.H file

extern void __stdcall SWDinit( void);
extern int  __stdcall SWDscan( int chan, float **buffer);
extern void __stdcall SWDrate( int dsf);
extern void __stdcall SWDclose( void);
extern void __stdcall SWDupdate( int ScansToAvg, int xsmooth, int TempComp);
extern void __stdcall SWDxtmode( int xtmode);
extern void __stdcall SWDxtrate( int xtrate);
extern void __stdcall SWDscanLV( int chan, float *buffer);

////////////////////////////////////////////////////////////////////
```