

The cover art consists of a light gray background with a faint, stylized globe showing continents. Overlaid on the left side of the globe is a vertical column of squares in various shades of gray, creating a pixelated or mosaic effect. The title text is positioned in the center-left, overlapping both the globe and the pixelated column.

Yuan's QCAP SDK SC580 Easy Programming Guide

1.1.0.127.8, 2014.04.18

1. Introduction

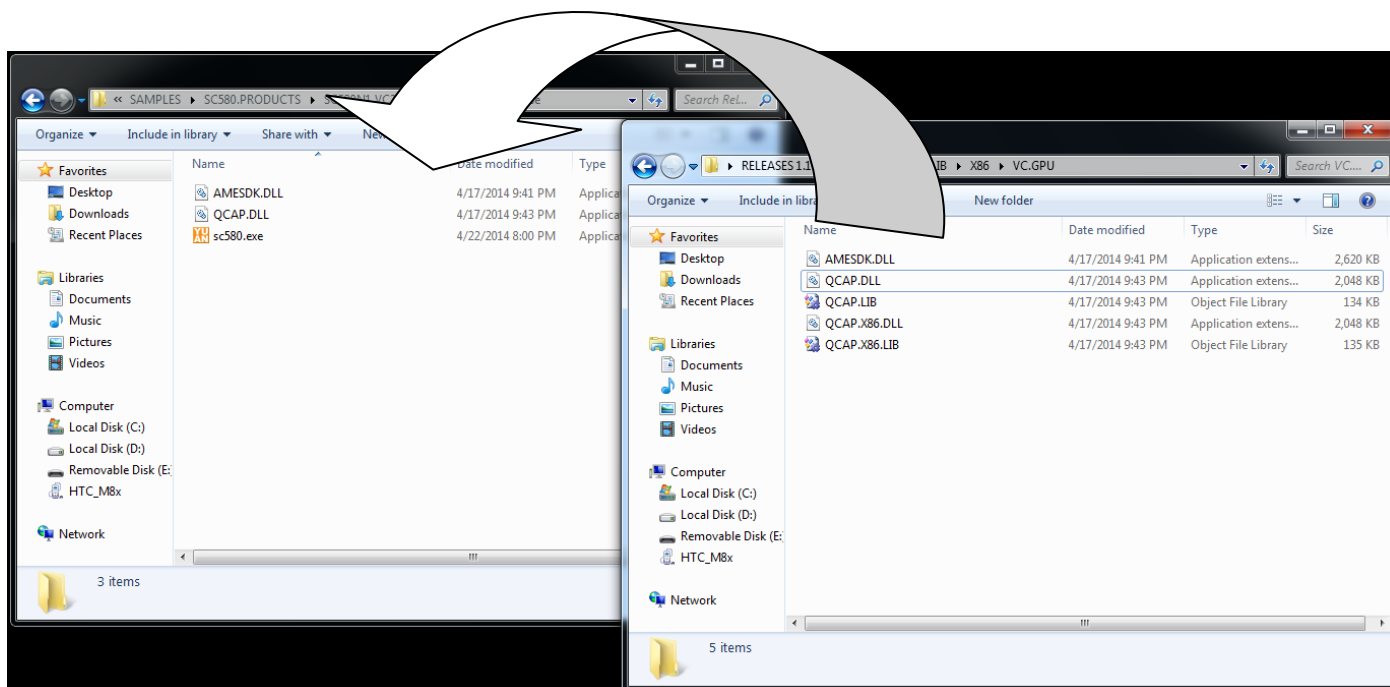
Overview

In this document, we will guide and teach users how step by step to use QCAP SDK functions that implement SC580 capture card.

Setting DLL File

VC, LabView	X86	QCAP.DLL, AMESDK.DLL
	X64	QCAP.X64.DLL, AMESDK.X64.DLL
C#, VB, Delphi	X86	QCAP.DLL, QCAP.NET.DLL, AMESDK.DLL
	X64	QCAP.X64.DLL, QCAP.NET.X64.DLL, AMESDK.X64.DLL

Developer should copy QCAP.DLL and AMESDK.DLL with .EXE file as is shown in below figure.





2. Initialize Device APIs

```
QCAP_SET_SYSTEM_CONFIGURATION( ... );

QCAP_CREATE( "FH8735 PCI", 0, ... );

QCAP_REGISTER_FORMAT_CHANGED_CALLBACK( pDevice, on_format_changed_callback, ... );

QCAP_REGISTER_NO_SIGNAL_DETECTED_CALLBACK( m_pDevice, on_no_signal_detected_callback, ... );

QCAP_REGISTER_SIGNAL_REMOVED_CALLBACK( m_pDevice, on_no_signal_removed_callback, ... );

QCAP_REGISTER_VIDEO_PREVIEW_CALLBACK( m_pDevice, on_video_preview_callback, ... );

QCAP_REGISTER_AUDIO_PREVIEW_CALLBACK( m_pDevice, on_audio_preview_callback, ... );

QCAP_REGISTER_VIDEO_HARDWARE_ENCODER_CALLBACK( m_pDevice, 0, on_video_main_encoder_callback, ... );

QCAP_REGISTER_VIDEO_HARDWARE_ENCODER_CALLBACK( m_pDevice, 1, on_video_sub_encoder_callback, ... );

QCAP_SET_VIDEO_DEINTERLACE_TYPE( m_pDevice, QCAP_SOFTWARE_DEINTERLACE_TYPE_BLENDING );

QCAP_SET_VIDEO_DEINTERLACE( m_pDevice, TRUE );

QCAP_SET_VIDEO_INPUT( m_pDevice, QCAP_INPUT_TYPE_AUTO );

QCAP_SET_AUDIO_INPUT( m_pDevice, QCAP_INPUT_TYPE_EMBEDDED_AUDIO );

QCAP_SET_AUDIO_VOLUME( m_pDevice, 100 );

QCAP_SET_VIDEO_HARDWARE_ENCODER_PROPERTY_EX( m_pDevice, 0, ... );

QCAP_SET_VIDEO_HARDWARE_ENCODER_PROPERTY_EX( m_pDevice, 1, ... );

QCAP_RUN( m_pDevice );
```

3. Uninitialize Device APIs

```
QCAP_STOP( m_pDevice );

QCAP_DESTROY( m_pDevice );
```



4. Start Channel Record APIs

4.1 Channel Record from Main Hardware Encoder

```
QCAP_SET_VIDEO_RECORD_PROPERTY( m_pDevice, 0, QCAP_ENCODER_TYPE_HARDWARE, ... );  
QCAP_SET_AUDIO_RECORD_PROPERTY( m_pDevice, 0, QCAP_ENCODER_TYPE_SOFTWARE, ... );  
QCAP_START_RECORD( m_pDevice, 0, "CHANNEL01.MP4" );
```

4.2 Channel Record from Sub Hardware Encoder

```
QCAP_SET_VIDEO_RECORD_PROPERTY( m_pDevice, 1, QCAP_ENCODER_TYPE_HARDWARE, ... );  
QCAP_SET_AUDIO_RECORD_PROPERTY( m_pDevice, 1, QCAP_ENCODER_TYPE_SOFTWARE, ... );  
QCAP_START_RECORD( m_pDevice, 1, "CHANNEL01.MP4" );
```

4.3 Channel Record from General Software Encoder

```
QCAP_SET_VIDEO_RECORD_PROPERTY( m_pDevice, 2 or 3, QCAP_ENCODER_TYPE_SOFTWARE, ... );  
QCAP_SET_AUDIO_RECORD_PROPERTY( m_pDevice, 2 or 3, QCAP_ENCODER_TYPE_SOFTWARE, ... );  
QCAP_START_RECORD( m_pDevice, 2 or 3, "CHANNEL01.MP4" );
```

Note!! The iRecNum, 0, is used by main hardware encoder and the iRecNum, 1, is used by sub hardware encoder.

5. Stop Channel Record APIs

```
QCAP_STOP_RECORD( m_pDevice, ... );
```



6. Start Share Record APIs

```
QCAP_SET_VIDEO_SHARE_RECORD_PROPERTY( 0, ... );  
QCAP_SET_AUDIO_SHARE_RECORD_PROPERTY( 0, ... );  
QCAP_START_SHARE_RECORD( 0, "SHARE01.MP4", dwFlags );
```

Note!! For compression data user, the QCAP_RECORD_FLAG_ENCODE flag should be cleared from the parameters, dwFlags, in QCAP_START_SHARE_RECORD API to disable the software encoder's resource.

7. Set Share Record Data APIs

7.1 Set Share Record Data from Uncompression Buffer

```
QRETURN on_video_preview_callback( ..., BYTE * pFrameBuffer, ULONG nFrameBufferLen, ... )  
{  
    ...  
    if( g_n_share_record_state > 0 ) {  
        QCAP_SET_VIDEO_SHARE_RECORD_UNCOMPRESSION_BUFFER( ..., pFrameBuffer, nFrameBufferLen, ... );  
    }  
    ...  
}  
  
QRETURN on_audio_preview_callback( ..., BYTE * pFrameBuffer, ULONG nFrameBufferLen, ... )  
{  
    ...  
    if( g_n_share_record_state > 0 ) {  
        QCAP_SET_AUDIO_SHARE_RECORD_UNCOMPRESSION_BUFFER( ..., pFrameBuffer, nFrameBufferLen, ... );  
    }  
    ...  
}
```



7.2 Set Share Record Data from Compression Buffer

```
QRETURN on_video_main_encoder_callback( ..., BYTE * pStreamBuffer, ULONG nStreamBufferLen, ... )
{
    ...

    if( g_n_share_record_state > 0 ) {

        QCAP_SET_VIDEO_SHARE_RECORD_COMPRESSION_BUFFER( ..., pStreamBuffer, nStreamBufferLen, ... );
    }
    ...
}

QRETURN on_audio_preview_callback( ..., BYTE * pFrameBuffer, ULONG nFrameBufferLen, ... )
{
    ...

    if( g_n_share_record_state > 0 ) {

        QCAP_SET_AUDIO_SHARE_RECORD_UNCOMPRESSION_BUFFER( ..., pFrameBuffer, nFrameBufferLen, ... );
    }
    ...
}
```

8. Stop Share Record APIs

```
QCAP_STOP_SHARE_RECORD( 0 );
```



9. Share Record with Multi-Threading

```
DWORD WINAPI on_video_preview_callback_ex( LPVOID params )
{
    ...

    HANDLE events[ 2 ] = { g_h_share_record_thread_stop_events[ 0 ],
                          g_h_share_record_buffer_ready_events[ 0 ] };

    while( TRUE ) {

        DWORD returns = WaitForMultipleObjects( 2, events, FALSE, INFINITE );

        if( returns == (WAIT_OBJECT_0) ) { break ; }

        if( returns == (WAIT_OBJECT_0 + 1) ) {

            EnterCriticalSection( g_h_share_record_access_critical_sections[ 0 ] );

            BYTE * po = g_p_share_record_buffers[ 0 ];

            ULONG sz = g_n_share_record_buffer_lengths[ 0 ];

            if( g_n_share_record_state > 0 ) {

                LeaveCriticalSection( g_h_share_record_access_critical_sections[ 0 ] );

                QCAP_SET_VIDEO_SHARE_RECORD_UNCOMPRESSION_BUFFER( ..., po, sz, ... );

            }
            else {

                LeaveCriticalSection( g_h_share_record_access_critical_sections[ 0 ] );

            }

        }

    }

    ...
}

QRETURN on_video_preview_callback( ..., BYTE * pFrameBuffer, ULONG nFrameBufferLen, ... )
{
    ...

    EnterCriticalSection( g_h_share_record_access_critical_sections[ 0 ] );

    if( g_n_share_record_state > 0 ) {


        g_p_share_record_buffers[ 0 ] = pFrameBuffer;

        g_n_share_record_buffer_lengths[ 0 ] = nFrameBufferLen;

        SetEvent( g_h_share_record_buffer_ready_events[ 0 ] );

    }
    LeaveCriticalSection( g_h_share_record_access_critical_sections[ 0 ] );

    ...
}
```



```

DWORD WINAPI on_audio_preview_callback_ex( LPVOID params )
{
    ...

    HANDLE events[ 2 ] = { g_h_share_record_thread_stop_events[ 1 ],
                           g_h_share_record_buffer_ready_events[ 1 ] };

    while( TRUE ) {

        DWORD returns = WaitForMultipleObjects( 2, events, FALSE, INFINITE );

        if( returns == (WAIT_OBJECT_0) ) { break ; }

        if( returns == (WAIT_OBJECT_0 + 1) ) {

            EnterCriticalSection( g_h_share_record_access_critical_sections[ 1 ] );

            BYTE * po = g_p_share_record_buffers[ 1 ];

            ULONG sz = g_n_share_record_buffer_lengths[ 1 ];

            if( g_n_share_record_state > 0 ) {

                LeaveCriticalSection( g_h_share_record_access_critical_sections[ 1 ] );

                QCAP_SET_AUDIO_SHARE_RECORD_UNCOMPRESSION_BUFFER( ..., po, sz, ... );

            }
            else {

                LeaveCriticalSection( g_h_share_record_access_critical_sections[ 1 ] );

            }

        }

    }

    ...
}

QRETURN on_audio_preview_callback( ..., BYTE * pFrameBuffer, ULONG nFrameBufferLen, ... )
{
    ...

    EnterCriticalSection( g_h_share_record_access_critical_sections[ 1 ] );

    if( g_n_share_record_state > 0 ) {

        g_p_share_record_buffers[ 1 ] = pFrameBuffer;

        g_n_share_record_buffer_lengths[ 1 ] = nFrameBufferLen;

        SetEvent( g_h_share_record_buffer_ready_events[ 1 ] );

    }
    LeaveCriticalSection( g_h_share_record_access_critical_sections[ 1 ] );

    ...
}

```




10. Start Broadcast APIs

```
QCAP_CREATE_BROADCAST_RTSP_SERVER( 0, ..., &pServer, ... );  
QCAP_SET_VIDEO_BROADCAST_SERVER_PROPERTY( pServer, ..., dwFlags );  
QCAP_SET_AUDIO_BROADCAST_SERVER_PROPERTY( pServer, ... );  
QCAP_START_BROADCAST_SERVER( pServer );
```

Note!! For compression data user, the QCAP_BROADCAST_FLAG_ENCODE flag should be cleared from the parameter, dwFlags, in QCAP_START_SHARE_RECORD API to disable the software encoder's resource.

11. Set Broadcast Data APIs

11.1 Set Broadcast Data from Uncompression Buffer

```
QRETURN on_video_preview_callback( ..., BYTE * pFrameBuffer, ULONG nFrameBufferLen, ... )  
{  
    ...  
    if( g_n_broadcast_server_state > 0 ) {  
        QCAP_SET_VIDEO_BROADCAST_SERVER_UNCOMPRESSION_BUFFER( ..., pFrameBuffer, nFrameBufferLen, ... );  
    }  
    ...  
}  
  
QRETURN on_audio_preview_callback( ..., BYTE * pFrameBuffer, ULONG nFrameBufferLen, ... )  
{  
    ...  
    if( g_n_broadcast_server_state > 0 ) {  
        QCAP_SET_AUDIO_BROADCAST_SERVER_UNCOMPRESSION_BUFFER( ..., pFrameBuffer, nFrameBufferLen, ... );  
    }  
    ...  
}
```



11.2 Set Broadcast Data from Compression Buffer

```
QRETURN on_video_main_encoder_callback( ..., BYTE * pStreamBuffer, ULONG nStreamBufferLen, ... )
{
    ...

    if( g_n_broadcast_server_state > 0 ) {

        QCAP_SET_VIDEO_BROADCAST_SERVER_COMPRESSION_BUFFER( ..., pStreamBuffer, nStreamBufferLen, ... );
    }
    ...
}

QRETURN on_audio_preview_callback( ..., BYTE * pFrameBuffer, ULONG nFrameBufferLen, ... )
{
    ...

    if( g_n_broadcast_server_state > 0 ) {

        QCAP_SET_AUDIO_BROADCAST_SERVER_UNCOMPRESSION_BUFFER( ..., pFrameBuffer, nFrameBufferLen, ... );
    }
    ...
}
```

12. Stop Broadcast APIs

```
QCAP_STOP_BROADCAST_SERVER( pServer );
```



13. Broadcast with Multi-Threading

```
DWORD WINAPI on_video_preview_callback_ex( LPVOID params )
{
    ...

    HANDLE events[ 2 ] = { g_h_broadcast_server_thread_stop_events[ 0 ],
                          g_h_broadcast_server_buffer_ready_events[ 0 ] };

    while( TRUE ) {

        DWORD returns = WaitForMultipleObjects( 2, events, FALSE, INFINITE );

        if( returns == (WAIT_OBJECT_0) ) { break ; }

        if( returns == (WAIT_OBJECT_0 + 1) ) {

            EnterCriticalSection( g_h_broadcast_server_access_critical_sections[ 0 ] );

            BYTE * po = g_p_broadcast_server_buffers[ 0 ];

            ULONG sz = g_n_broadcast_server_buffer_lengths[ 0 ];

            if( g_n_broadcast_server_state > 0 ) {

                LeaveCriticalSection( g_h_broadcast_server_access_critical_sections[ 0 ] );

                QCAP_SET_VIDEO_BROADCAST_SERVER_UNCOMPRESSION_BUFFER( ..., po, sz, ... );

            }
            else {

                LeaveCriticalSection( g_h_broadcast_server_access_critical_sections[ 0 ] );

            }

        }

    }

    ...
}

QRETURN on_video_preview_callback( ..., BYTE * pFrameBuffer, ULONG nFrameBufferLen, ... )
{
    ...

    EnterCriticalSection( g_h_broadcast_server_access_critical_sections[ 0 ] );

    if( g_n_broadcast_server_state > 0 ) {


        g_p_broadcast_server_buffers[ 0 ] = pFrameBuffer;

        g_n_broadcast_server_buffer_lengths[ 0 ] = nFrameBufferLen;

        SetEvent( g_h_broadcast_server_buffer_ready_events[ 0 ] );

    }
    LeaveCriticalSection( g_h_broadcast_server_access_critical_sections[ 0 ] );

    ...
}
```



```

DWORD WINAPI on_audio_preview_callback_ex( LPVOID params )
{
    ...

    HANDLE events[ 2 ] = { g_h_broadcast_server_thread_stop_events[ 1 ],
                           g_h_broadcast_server_buffer_ready_events[ 1 ] };

    while( TRUE ) {

        DWORD returns = WaitForMultipleObjects( 2, events, FALSE, INFINITE );

        if( returns == (WAIT_OBJECT_0) ) { break ; }

        if( returns == (WAIT_OBJECT_0 + 1) ) {

            EnterCriticalSection( g_h_broadcast_server_access_critical_sections[ 1 ] );

            BYTE * po = g_p_broadcast_server_buffers[ 1 ];

            ULONG sz = g_n_broadcast_server_buffer_lengths[ 1 ];

            if( g_n_broadcast_server_state > 0 ) {

                LeaveCriticalSection( g_h_broadcast_server_access_critical_sections[ 1 ] );

                QCAP_SET_AUDIO_BROADCAST_SERVER_UNCOMPRESSION_BUFFER( ..., po, sz, ... );

            }
            else {

                LeaveCriticalSection( g_h_broadcast_server_access_critical_sections[ 1 ] );

            }

        }

    }

    ...
}

QRETURN on_audio_preview_callback( ..., BYTE * pFrameBuffer, ULONG nFrameBufferLen, ... )
{
    ...

    EnterCriticalSection( g_h_broadcast_server_access_critical_sections[ 1 ] );

    if( g_n_broadcast_server_state > 0 ) {

        g_p_broadcast_server_buffers[ 1 ] = pFrameBuffer;

        g_n_broadcast_server_buffer_lengths[ 1 ] = nFrameBufferLen;

        SetEvent( g_h_broadcast_server_buffer_ready_events[ 1 ] );

    }

    LeaveCriticalSection( g_h_broadcast_server_access_critical_sections[ 1 ] );

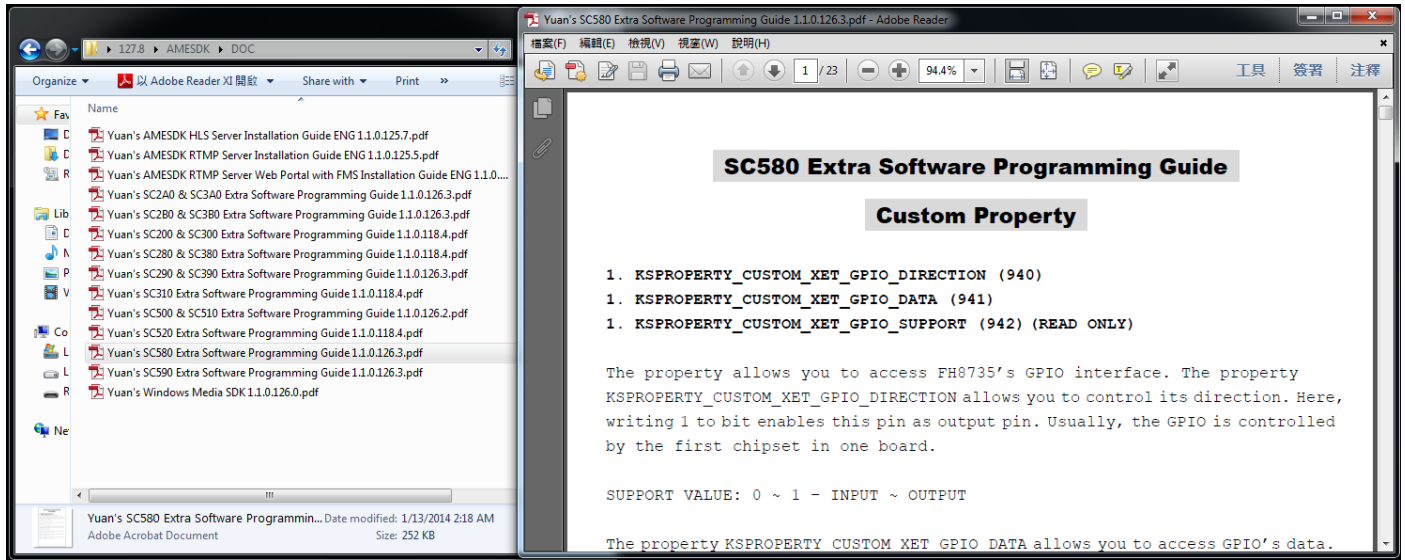
    ...

}

```

14. Custom Preoperty for SC580

For more custom device property programming, please reference product's Extra Prgoramming Guide in SDK packet. User can find Extra Prgoramming Guide as shown below.

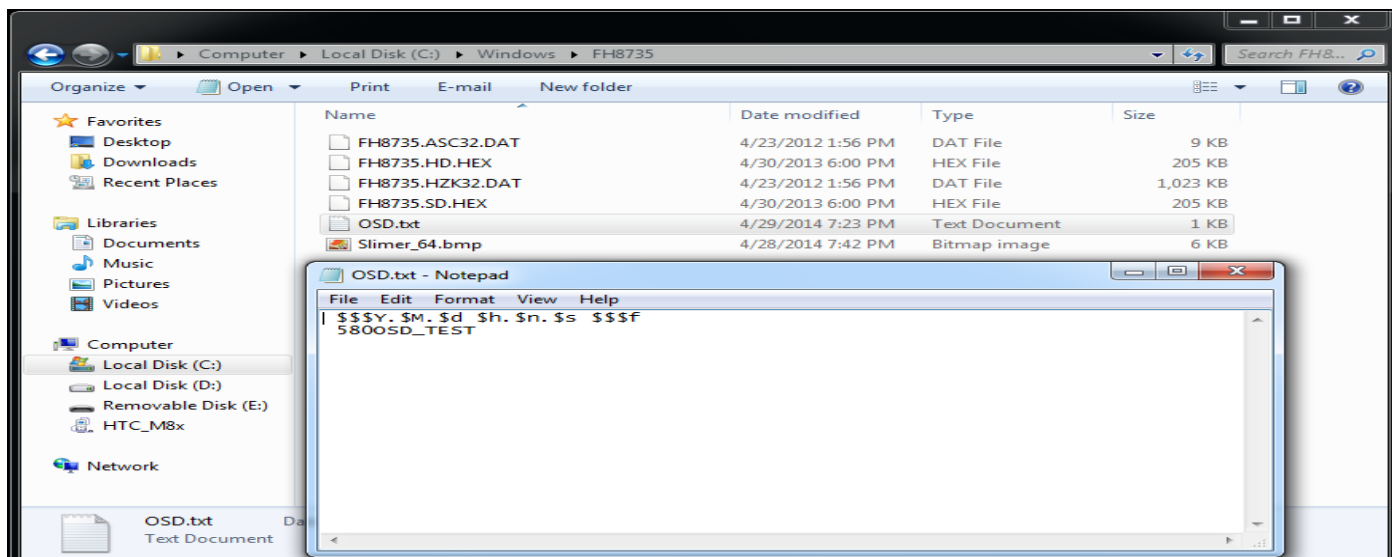




In there we give example for 580 OSD text and picture.

Example for 580 OSD text in hardware encoder.

STEP 1. Set OSD format



STEP 2. Set OSD device custom property

```
QCAP_SET_DEVICE_CUSTOM_PROPERTY( pDevice, 929, 0x00000001 );  
QCAP_SET_DEVICE_CUSTOM_PROPERTY( pDevice, 920, 0x00000000 );  
CHAR path[ ] = "C:\\WINDOWS\\FH8735\\OSD.TXT";  
QCAP_SET_DEVICE_CUSTOM_PROPERTY_EX( pDevice, 921, ( BYTE * )( path ), strlen( path ) );
```



Example for 580 OSD picture in hardware encoder.

Note!! 580 allow software to load one 8 bits (256 colors) BMP file into its board memory.

```
ULONG params[ 4 ] = { 0,    /*Picture index is 0 or 1 */
                      1,    /*Picture start left position*/
                      1,    /*Picture start top position*/
                      255   /*Picture transparent is from 0~255*/
                      };

CHAR path[ ] = "C:\\WINDOWS\\FH8735\\Slimer_64.bmp";

QCAP_SET_DEVICE_CUSTOM_PROPERTY_EX( pDevice, 970, ( BYTE * )params, sizeof( params ) );

QCAP_SET_DEVICE_CUSTOM_PROPERTY_EX( pDevice, 971, ( BYTE * )(path), strlen( path) );
```