The PXC200 Fast Display DLL

The PXC200 fast display DLL (PXCDD.dll) is a set of routines which work with the PXC200 software API to allow fast display of video in a window with graphics overlay under Windows 95 and Windows NT. The main features of the package are:

- automatically displays a video image from the PXC200 in a window.
- takes advantage of accelerated graphics display hardware using DirectX.
- provides support for nondestructive graphic overlays.
- works with RGB15, RGB16, RGB24 and RGB32 data types, to support graphics display on adapters running in any of those color modes.

Video Image Display

The PXCDD library displays video by creating a frame for the PXC200 to write video into, and using a worker thread to continually update a window with the contents of that frame. On systems with high performance graphics display hardware the task of updating the window is performed mostly in the display hardware, so the live display uses relatively little processor time.

When fast display is enabled, the window is updated with whatever the video frame contains. This means that the PXCDD library does not need to control the PXC200 frame grabber directly. The application program can use any of the PXC200 video capture features, such as continuous acquire, single grabs, and triggered grabs to place video images into the buffer, and these video images will automatically appear in the window.

Alternatively, if the video frame only changes occasionally or if the precise timing of display updates is important, the application can call a function to update the display immediately whenever appropriate rather than relying on the background thread to do automatic updates.

Nondestructive Overlay

Another feature of the Fast Display DLL is nondestructive graphics overlay. This feature allows the application program to supply a graphics image which will appear on top of the video frame when it is displayed in a window, but not actually overwrite the pixel data in the video frame. For example, an application program might overlay a bitmap representing alignment marks on the displayed video image so that the program's user can adjust the position of a camera. The marks appear to overwrite the video image as seen on the window, but the actual image data in the frame is not modified.

Installing the Fast Display DLL

System Requirements

The fast display DLL is a WIN32 program which runs under either Windows 95 or Windows NT. It also requires the following:

- DirectDraw version 2 or later, or DirectX.
- A PXC200 frame grabber and the PXC200 development software.
- Microsoft Visual C 4.0, Borland C 5.0, Watcom C 11.0 or later versions of any of these.
- A display adapter compatible with DirectDraw or DirectX, with at least 4 Megabytes of video memory and graphics acceleration. More video memory is helpful, particularly when working with true color images or large overlays. Video display performance can vary dramatically between display adapters, please see the section below on performance for information about selecting a display adapter.
- If using Windows NT, use version 4, service pack 3 or newer.

Installation

The fast display DLL is distributed on a disk containing the DLL, a C include file, and some sample code. The disk is organized as follows:

Directory	Contents
\pxcdd\bin	The PXCDD DLL, and sample programs
\pxcdd\samples	Source code for the sample programs
\pxcdd\include	The C include file

To install the software, copy the entire pxcdd directory to your hard drive, or alternatively copy the contents of individual subdirectories to your pxc200 installation's bin, include and samples directories. You may also need to change some of you C compiler's environment settings to find the include file.

In addition to the software provided on the disk, you will also need to install DirectDraw or DirectX.

Programming with the Fast Display DLL

Initializing the Library

Before the PXCDD library is used it must be initialized by a call to `imagenation_OpenLibrary`. This process is similar to that used by the PXC200 library and the FRAME library. First, a global structure of type PXCDD should be created, then OpenLibrary should be called. At this point the library is ready to use, and each function in the library is called as a member of the PXCDD structure. After the application is finished using the library it should call `imagenation_CloseLibrary`.

Putting Video in a Window

An application program which has created a window can display live video in it with the following steps:

1. Use `CreateSurface` to allocate a frame the size of the desired video image. This frame will always have the same pixel type as the Windows desktop; for example, if a computer's video adapter is configured to display in 24 bit True Color mode, the pixel type of the frame returned by CreateSurface will be PBITS_RGB24.
2. Use a PXC200 function such as `Grab` or `GrabContinuous` to begin capturing video to the frame created by CreateSurface.
3. Use `EnableFastDisplay` to begin updating a window with the contents of the frame. The PXCDD library will now manage updates to the window as a background task. Alternatively, the `Display` function can be called to do a single update to the window.

If GrabContinuous is used to capture video to the frame and EnableFastDisplay is used to update the window, live video will be continuously shown in the window without any additional help from the application. This arrangement frees the application program to do things unrelated to managing video display. Alternatively, if a program needs more control over the video display, it can call Grab and Display in a loop to do window updates.

When using EnableFastDisplay, it is important to disable the display before the window being displayed to is destroyed, such as when the WM_CLOSE message is processed. If the window is destroyed while display is still occurring, a thread from the PXCDD DLL may be left running after the program finishes. This thread must be killed before the program can be run again.

Graphics Overlays

Adding an overlay to video is a three step process.

1. Create the overlay using `LoadOverlay`. This function sets up an overlay which will have the same size and contents as a specified frame. The frame used by LoadOverlay in general should not be one allocated by CreateSurface, since doing this will use more of the graphics adapter's memory than using

a frame created by AllocateMemoryFrame or AllocateBuffer. LoadOverlay makes a copy of the frame's data, so the frame can be freed or reused any time after the call to LoadOverlay.

2. Attach the overlay to a PXCDD frame using `EnableOverlay`. This function causes the overlay to be placed on top of the specified frame, which must have been created by CreateSurface. The x and y parameters to EnableOverlay give the position of the upper left corner of the overlay image relative to the video frame. Each overlay can be attached to at most one video frame, and each frame can have at most one overlay.
3. Enable video display. EnableOverlay does not actually cause anything to be displayed in a window. In order to display the overlay, either EnableFastDisplay, or Display must be called to show the video frame which the overlay is attached to. EnableOverlay and EnableFastDisplay can be called in either order; the application program can either set up the overlay first and then start displaying video, or attach an overlay to a frame which is already being displayed.

Moving Frames and Overlays

It is possible to change the position of both video frames and overlays. `MoveSurface` can be used to reposition a video frame inside a window. This can be used, for instance, to leave room on the top or left border of the window for control buttons. Overlays can be moved relative to the frame they are attached to with `MoveOverlay`. This can be useful for making pointers or cross hairs which move under user control. Because overlays are positioned relative to a video frame, calling MoveSurface on a frame which has an attached overlay will move both the frame and the attached overlay.

Drawing Graphics into Frames

Sometimes, it is useful to draw into video frames or overlays. There are three ways to do this: manipulate the frames directly using frame library functions, use DirectDraw functions, or use GDI functions.

Any frame library function can be used to access a video frame. The frame library can be used to read and write BMP files and copy regions of the bitmap to and from arrays. The bitmap of the frame can also be accessed directly using the frame library's FrameBuffer function. The frame library can also be used to create and access frames which can be passed to LoadOverlay, but once the overlay is loaded it is *not* accessible to the frame library anymore.

Each video frame created by CreateSurface, as well as each overlay which is currently loaded, is accessible as a DirectDraw surface. The DirectDraw surface pointer for any video frame can be found using the `GetDDSurface` function, while the surface pointer for an overlay can be found using `GetOverlaySurface`. Using these surface pointers, DirectDraw operations such as Blt copy can be performed.

For operations such as drawing lines and writing text, Windows GDI functions are often helpful. In order to use GDI calls on a frame, a handle to a Device Context (HDC) must be created. For active overlays made by LoadOverlay or video frames made by CreateSurface, an HDC can be created by first finding the DirectDraw surface pointer, and then calling the DirectDraw surface interface's GetDC function. Using the GetDC function locks the surface, which can prevent other graphics operations from occurring, so it is important to release the Device Context promptly using the DirectDraw surface ReleaseDC function.

Occasionally it is necessary to create a frame which is not allocated from graphics card memory, but is accessible by GDI functions. One example of this is the OCLOCK sample program, which draws an image on a frame and loads the image as an overlay. Frames of this sort can be made using calls to CreateDIBSection, but this technique is fairly complicated. The sample source file GDIFRAME.C uses this technique and isolates most of the complexity inside the AllocateGDI and CreateFrameDC functions.

Library Reference

CloseLibrary

Returns to the system any resources which were allocated by OpenLibrary. This function should be the last PXCDD function called by a program.

Declaration: void imagenation_CloseLibrary(PXCDD *pxcdd);
Parameters: none
Returns: none
Errors: none

CreateSurface

Creates a frame in VGA memory using the current display resolution bit depth. The PXC200 library can grab images to this frame as if it were allocated by AllocateBuffer. EnableFastDisplay, or Display can be used to show the contents of this frame in a window. Use the FreeFrame function from either the PXC200 library or the Frame library to release the resources allocated by this function when the frame is no longer needed.

Declaration: FRAME* CreateSurface (short dx, short dy);
Parameters:
- dx: width of surface
- dy: height of surface

Returns:
- 0 for failure
- a frame pointer if successful

Errors:
- ERR_INVALID_PARAMETER
- ERR_VIDEO_MEM
- ERR_SYSTEM_MEM
- ERR_NO_DIRECTDRAW

Display

Immediately copies the contents of a frame to a window. The frame must have been created by CreateSurface. Any overlays which have been attached to this frame with EnableOverlay will also be displayed.

Declaration: void Display(HWND hwnd, FRAME *f);
Parameters:
- hwnd: handle to the window
- f: frame to be displayed

Returns: None
Errors: None

EnableFastDisplay

Enables or disables live display of a specific frame. When display is enabled, window will be continuously updated with the contents of the specified frame. The frame must have been created using CreateSurface. Any overlays which are attached to the frame will also be displayed.

Declaration: short EnableFastDisplay (HWND destWindow, FRAME ddFrameHandle, long timingFgh, boolean enable);
Parameters:
- destWindow: a handle to the destination window
- ddFrameHandle: the frame to be displayed
- timingFgh: a handle to the frame grabber that will be updating this frame.
  This information is used to synchronize the live display to the frame grabber. If this value is 0, the display will be updated as quickly as possible without synchronizing to the frame grabber. Not synchronizing to the frame grabber can cause lower system performance and degraded image quality.
- enable: if TRUE, this enables the display; if FALSE; this disables the display

Returns:
0 for failure
non-zero for success
Errors:
ERR_INVALID_PARAMETER
ERR_WINDOW_ATTACHED
ERR_DISPLAY_INACTIVE
ERR_NOT_DD_FRAME

EnableOverlay

Attaches an overlay created by LoadOverlay to a frame, or unattaches it. When enabled, the overlay will be displayed on top of the specified frame whenever the frame is displayed. The overlay may be smaller in size than the frame it is attached to.

Declaration: short EnableOverlay (FRAME ddFrameHandle, long overlayHandle, short x, short y, boolean enable);
Parameters:
ddFrameHandle: the target surface to apply the overlay on.
overlayHandle: the overlay to be enabled.
x,y: the position inside the target surface where the overlay is displayed. Ignored if enable is FALSE.
enable: if TRUE, this enables the overlay. If FALSE, this disables the overlay.
Returns:
0 for failure
non-zero for success
Errors
ERR_INVALID_PARAMETER
ERR_NOT_DD_FRAME
ERR_OVERLAY_ATTACHED
ERR_FRAME_ATTACHED

GetBitDepth

Provides the display bitdepth that will be used by all DD library functions. This is in the format used by the FRAME library.

declaration: short GetBitDepth (void);
Parameters: none
Returns:
PBITS_Y8, PBITS_RGB15, PBITS_RGB16, PBITS_RGB24, or PBITS_RGB32, depending on the current Windows desktop's color depth.

Errors: none

GetDDSurface

Returns a DirectDraw surface pointer to a frame's image buffer. This allows the user to perform their own DirectDraw operations on the frame. The frame must have been created with CreateSurface.

Declaration: LPDIRECTDRAWSURFACE GetDDSurface (FRAME* ddFrameHandle);
Parameters:
ddFrameHandle: a handle to the frame.
Returns:

0 for failure
A pointer to the DirectDraw surface if successful

Errors:
ERR_INVALID_PARAMETER
ERR_NOT_DD_FRAME

GetError

Returns an error code for the last PXCDD function called.

declaration: short GetError (void);
Parameters: none
Returns: some combination of these error codes (combined by bitwise OR)

FUNCTION_SUCCESS	0	No error occurred
ERR_NO_DIRECTDRAW	1	A DirectDraw call failed
ERR_VIDEO_MEM	2	Out of Video RAM; system memory was used instead (the function may still complete successfully)
ERR_SYSTEM_MEM	4	Out of system memory
ERR_INVALID_PARAMETER	8	Function parameters are incorrect
ERR_NOT_DD_FRAME	16	frame handle wasn't created by CreateSurface
ERR_WRONG_BIT_DEPTH	32	frame pixel type doesn't match desktop type.
ERR_ACTIVE_FRAME	64	the frame is being used by some other function
ERR_WINDOW_ATTACHED	128	The window already has a frame attached to it
ERR_FRAME_ATTACHED	256	The frame is already attached to something else
ERR_OVERLAY_ATTACHED	512	The overlay is already attached to another frame
ERR_DISPLAY_INACTIVE	1024	Display cannot be disabled because it was not enabled

GetOverlaySurface

Returns a DirectDraw surface pointer to an overlay's image buffer. This allows the user to perform their own DirectDraw operations on the overlay. The overlay must have been created with LoadOverlay.

Declaration: LPDIRECTDRAWSURFACE GetOverlaySurface (long overlayhandle);
Parameters:
overlayhandle: a handle to the overlay
Returns:
0 for failure
A pointer to the DirectDraw surface if successful
Errors: none

GetRenderCount

Returns the number of frames which have been drawn by the EnableFastDisplay mechanism since the library was started. Subtracting two render counts can be used to compute the frame rate.

Declaration: int GetRenderCount(void);
Parameters: none
Returns: the number of frames drawn
Errors: none

LoadOverlay

creates a overlay using user-provided information. This overlay is extracted from a frame that contains the desired overlay image. This information is copied from the frame into internal buffers allocated by the library. If the user wishes to use a saved .bmp file, they can use ReadBMP to load the file,

then pass that handle into this function. Use UnloadOverlay to free the resources allocated by this function as soon as the overlay is no longer needed.

Declaration: long LoadOverlay (FRAME* overlay, long transBit)
Parameters
overlay: a frame handle to a frame containing overlay information.
transBit: the value that should be interpreted as transparent when loading the overlay. All pixels of this value will be ignored while creating the overlay.

Returns:
0 for failure
a handle to the overlay if successful

Errors:
ERR_INVALID_PARAMETER
ERR_NO_DIRECTDRAW
ERR_WRONG_BIT_DEPTH
ERR_VIDEO_MEM
ERR_SYSTEM_MEM

MoveOverlay

Specifies the position of an overlay relative to the upper left corner of the frame it is attached to.

Declaration: short MoveOverlay(long overlay, short x, short y);
Parameters:
overlay: a handle to the overlay to be positioned
x: the new horizontal position of the overlay
y: the new vertical position of the overlay
Returns:
0 on failure
non-zero for success
Errors:
ERR_INVALID_PARAMETER

MoveSurface

Specifies the position of a frame relative to the upper left corner of the window it is displayed in. The frame must have been created with CreateSurface.

Declaration: short MoveSurface(FRAME *ddframe, short x, short y);
Parameters:
ddframe: a handle to the frame to be positioned
x: the new horizontal position of the frame
y: the new vertical position of the frame
Returns:
0 on failure
non-zero for success
Errors:
ERR_INVALID_PARAMETER

OpenLibrary

Initializes some data structures used by the library. This function must be called before any other library functions are used. Use CloseLibrary to deallocate resources claimed by this function before the program exits.

Declaration: short int imagenation_OpenLibrary("pxcdd.dll", PXCDD *pxcdd, sizeof(PXCDD));
Parameters: none
Returns:
0 for failure
non-zero for success
Errors:
Fails if the DLL can't be loaded, or if DirectDraw is not available. If this function fails GetError cannot be called, so no error codes are returned.

UnloadOverlay

Removes an overlay created by LoadOverlay. The overlay handle will be invalid as soon as this function completes.

Declaration: short UnloadOverlay (long overlayHandle);
Parameters:
overlayHandle: a handle to an overlay currently in memory.
Returns:
0 for failure
non-zero if successful.
Errors:
ERR_INVALID_PARAMETER
ERR_ACTIVE_FRAME

Video Display Performance

How fast the PXCDD DLL can update the display depends primarily on three factors:

- The type of graphics adapter installed
- The amount of video memory on the adapter, and
- The graphics drivers which are installed.

The speed of the system CPU can sometimes be important, but on a system with enough video memory and a good graphics adapter increasing CPU speed will usually not improve performance significantly.

The Graphics Adapter

The DLL is designed to work with PCI graphics cards which use a flat memory addressing mode. Although almost all currently available graphics adapters support flat addressing, sometimes old graphics drivers or generic SVGA drivers will not enable this mode. Using one of these drivers will prevent video from being displayed properly.

Two graphics accelerator features which are necessary in order to get good performance are hardware blitting and hardware clipping. Again, most modern graphics cards have at least some support for these features. Hardware support for transparent overlays will not improve performance; the DLL does not take advantage of this feature because DirectDraw support for hardware overlays is not always well implemented.

Video Memory

The feature which has the largest effect on video performance is the amount of RAM on the graphics adapter. The DLL will try to allocate all of the image buffers it needs from video RAM. If there is not enough video RAM then some buffers will be allocated in system memory, which substantially decreases performance. System memory is worse than video memory for three reasons.

- Transfers from system memory to the graphics card are usually slower than data movement inside the graphics card.
- The graphics cards blitter usually cannot copy data from system memory, so the system CPU must be used to copy data which steals processor time from other applications.

- Moving data from system memory to the graphics card uses PCI bandwidth, which reduces the amount of video data the frame grabber can reliably capture. In some cases this can result in image capture errors.

The amount of video RAM needed to optimally run a given program is the total of the size of the Windows desktop, the size of all video frames, the size of all active overlays, and some extra space for a scratch buffer used internally by the DLL. The scratch buffer is always 768 by 576 pixels.

For example, a program which is running on a 1024 by 768 pixel 24 bit per pixel desktop and uses one 640 by 480 video frame with a 100 by 100 pixel overlay would need the following amount of memory:

Desktop:	1024*768 pixels	* 3 bytes	= 2 359 296 bytes
Scratch Buffer:	768*576 pixels	* 3 bytes	= 1 327 104 bytes
Video Frame:	560*480 pixels	* 3 bytes	= 921 600 bytes
Overlay:	100*100 pixels	* 3 bytes	= 30 000 bytes

Total: 4 638 000 bytes, or about 4.4 megabytes

The same program running on an 800 by 600 pixel 24 bit desktop uses about 3.6 megabytes of video RAM.

Other programs and the operating system can also use some video memory for things like icons, mouse cursors, and other DirectDraw surfaces. These things can make some video memory unavailable to the video display DLL.

Graphics Drivers

The performance of a given video card is often limited by the graphics drivers which are being used to run it. This is particularly true under Windows NT, because highly optimized graphics drivers for Windows 95 are generally easier to find. In some cases, upgrading to an improved version of the drivers for a graphics card can increase performance by a factor of two or more. Manufacturers of graphics cards and graphics accelerator chips will often make drivers available on the internet which can have better performance than those that come with Windows.

Another software component which is related to the graphics driver is DirectDraw. The PXCDD DLL uses DirectDraw to control video display, so it is important to have DirectDraw or DirectX properly installed. The DLL will work with DirectDraw version 2 or later, but newer versions are likely to give better performance.