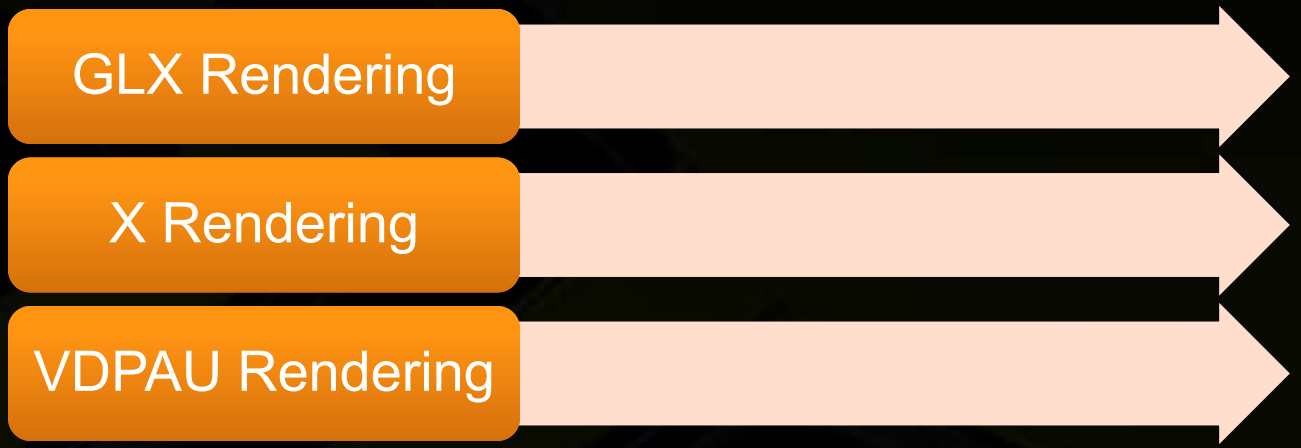
X Presentation and Synchronization

Synchronization Problems

There is no defined ordering between X rendering and direct rendering; it is left up to the application



This synchronization becomes unrealistic between applications when applications aren't aware of each other

Synchronization Solutions

- A new type of operation is needed to synchronize multiple rendering streams: X Sync object
- Inspired by GL sync objects
- Contains nothing but binary state: triggered, not triggered
- Rendering streams can be stalled until sync object reaches the triggered state

Basic Sync Object Example


```
/* Init some objects */
Sync sync = XCreateSyncObject();
Window win1 = XCreateWindow();
Window win2 = XCreateWindow()
fork();
```

```
/* Parent process */

XWaitSync(sync);


/*
 * X rendering from here on
 * will be deferred until
 * the sync is triggered.
 */

CopyWinContents(win2, win1);
```

```
/* Child process */

DrawToWin(win1);

/*
 * Set sync state to
 * triggered
 */
XTriggerSync(sync);
```

Sync Object Ordering

- This is the most important property of sync objects
- Sync object operations (Wait and Trigger) happen in-band with the rendering stream they are executed in
- This is important for cross-API usage when each API has its own rendering stream

More Complex Uses

- Sync objects can be exported to other APIs
  - GLX/GL: X sync object <-> GLsyncARB
  - VDPAU
- Extended sync objects can be triggered by various events
  - VSYNC on monitor X
  - Frame number N on monitor X
  - Timer interrupts
  - Operating System events, e.g., File I/O completion
  - others

One Important Problem Solved

- Compiz compositing before X or other GL application rendering completes can be safely and efficiently avoided
- X can expose sync objects that trigger only after rendering related to a given damage event has completed
- Compiz creates X sync objects of this type, imports them to GLsyncARB objects, and prefaces its compositing with waits on them

Presentation Problems

Linux desktop graphical complexity has grown exponentially in recent years, but presentation mechanisms have not kept up

- X has no real presentation control mechanism
- GLX presentation mechanisms all assume windows are onscreen
- GLX auxiliary buffers aren't accessible in other X extensions

Presentation Problems (Cont.)

Advanced presentation mechanisms need to provide the following:

- Precise control over when presentation occurs, relative to system and hardware events
- Feedback on where and when presentation occurred
- Feedback on when buffers are in use by presentation

All these still need to work even when presentation is controlled by a composite manager rather than the X server or a direct rendering client

Presentation Solution (Part 1)

How do we ensure X operations happen at particular times relative to other operations?

- Easy; use sync objects
- Note that sync objects can be stacked:

```
// Wait for a timer, then for the next vblank
// before compositing

XWaitSync(minTimeSync);
XWaitSync(vblankSync);
XComposite();
```

Presentation Solution (Part 2)

How can explicit presentation be added to X?

- Give X explicit multi-buffering:
  - Build on the composite framework
  - Allow application to explicitly allocate as many backing pixmaps as it wants
  - Each window may now have MULTIPLE backing pixmaps
  - In this situation, X will redirect the window when first backing pixmap is allocated, just as it would if a composite manager redirected it
  - The application may then present its contents simply by setting one of its backing pixmaps to the “front” pixmap

What about GLX?

New GLX extensions are needed

- New way to create GLX drawable from an X window with N X-managed back buffers
- Porting existing applications is easy:
  - `glXSwapBuffers(win)` -> `glXPresentBuffer(win, buf)`
  - `glXSwapInterval()` -> Use GL/X sync objects
  - `glDrawBuffers()` -> ???

What about Composite Managers?

- Presentation requests are forwarded to the current composite manager, if any
- If not, automatic compositing is performed

When did everything happen?

- The presentation command can be preceded by a sync wait
- Presentation commands can also optionally take a sync object as an argument
- The sync object would be triggered when the presentation was visible, either by the composite manager or the X server
- Add new state to sync objects: Triggered timestamp

Questions?