Xglamo Graphics

Overview

- Which code path does Xglamo take for graphics?
- What operations are accelerated?
- How does Evas benefit?

Code paths

- Software implementation: fb/, render/
  - Nothing to do
- XAA: hw/xfree86/xaa/
  - Only available for xfree86
  - Cannot accelerate render/ properly
- EXA: exa/
  - Yeah!

Software implementation

- fb/fbgc.c

```
const GCOps   fbGCOps = {
    fbFillSpans,
    fbSetSpans,
    fbPutImage,
    fbCopyArea,
    fbCopyPlane,
    fbPolyPoint,
    fbPolyLine,
    fbPolySegment,
    fbPolyRectangle,
    fbPolyArc,
    miFillPolygon,
    fbPolyFillRect,
    fbPolyFillArc,
    miPolyText8,
    miPolyText16,
    miImageText8,
    miImageText16,
    fbImageGlyphBlt,
    fbPolyGlyphBlt,
    fbPushPixels
};
```

Software implementation

- render/mipict.c

```
ps->Composite       = 0;/* requires DDX support */
ps->Glyphs          = miGlyphs;
ps->CompositeRects  = miCompositeRects;
ps->Trapezoids      = miTrapezoids;
ps->Triangles       = miTriangles;
ps->TriStrip        = miTriStrip;
ps->TriFan          = miTriFan;
```

```
   Composite(CARD8  op,
             PicturePtr pSrc,
             PicturePtr pMask,
             PicturePtr pDst,
             INT16   xSrc,
             INT16   ySrc,
             INT16   xMask,
             INT16   yMask,
             INT16   xDst,
             INT16   yDst,
             CARD16  width,
             CARD16  height)
```

XAA

- hw/xfree86/xaa/XAA.HOWTO

```
2)  The Primitives
  2.0  Generic Flags
  2.1  Screen to Screen Copies
  2.2  Solid Fills
  2.3  Solid Lines
  2.4  Dashed Lines
  2.5  Color Expand Fills
    2.5.1 Screen to Screen Color Expansion
    2.5.2 CPU to Screen Color Expansion
      2.5.2.1 The Direct Method
      2.5.2.2 The Indirect Method
  2.6  8x8 Mono Pattern Fills
  2.7  8x8 Color Pattern Fills
  2.8  Image Writes
    2.8.1 The Direct Method
    2.8.2 The Indirect Method
  2.9 Clipping
```

EXA

- exa/exa.h: (quoted)

```
Required:

Bool (*PrepareSolid)(PixmapPtr pPixmap, int alu, Pixel planemask, Pixel fg);
void (*Solid)       (PixmapPtr pPixmap, int x1, int y1, int x2, int y2);
void (*DoneSolid)   (PixmapPtr pPixmap);
Bool (*PrepareCopy) (PixmapPtr pSrcPixmap, PixmapPtr pDstPixmap, int dx, int dy,
                                                  int alu, Pixel planemask);
void (*Copy)        (PixmapPtr pDstPixmap, int srcX, int srcY, int dstX, int dstY,
                                                  int width, int height);
void (*DoneCopy)    (PixmapPtr pDstPixmap);
void (*WaitMarker)  (ScreenPtr pScreen, int marker);

Optional:

Composite
UploadToScreen
DownloadToScreen
```

What ops are accelerated?

- hw/kdrive/glamo/glamo-draw.c:
  - Solid and Copy

Xglamo & Render

- Composite is accelerated only when it is actually Copy
  - op == PictOpSrc
  - pMask == NULL
  - pSrc->format == pDst->format

```
Composite(CARD8  op,
          PicturePtr pSrc,
          PicturePtr pMask,
          PicturePtr pDst,
          INT16   xSrc,
          INT16   ySrc,
          INT16   xMask,
          INT16   yMask,
          INT16   xDst,
          INT16   yDst,
          CARD16  width,
          CARD16  height)
```

Xglamo & GC Ops

-

```
const GCOps   fbGCOps = {
    fbFillSpans,
    fbSetSpans,
    fbPutImage,
    fbCopyArea,
    fbCopyPlane,
    fbPolyPoint,
    fbPolyLine,
    fbPolySegment,
    fbPolyRectangle,
    fbPolyArc,
    miFillPolygon,
    fbPolyFillRect,
    fbPolyFillArc,
    miPolyText8,
    miPolyText16,
    miImageText8,
    miImageText16,
    fbImageGlyphBlt,
    fbPolyGlyphBlt,
    fbPushPixels
};
```

Xglamo & Evas

- NOTHING!
  - GC ops draw while Evas is built upon objects
  - Evas objects have alpha
  - Evas draws to a buffer, purely in software, and upload it to X server (src/modules/engines/software_16_x11/evas_x_buffer.c)
- ... until Composite is accelerated