Implementing Genivi IVI Layer Management with DirectFB

Takanari Hayama†, ‡‡
Denis Oliver Kropp‡‡

†IGEL Co., Ltd.
‡‡DirectFB integrated media GmbH
Agenda

1. Genivi IVI Layer Management
2. DirectFB
3. DirectFB Renderer Plugin for Genivi IVI Layer Manager Service
GENIVI IVI LAYER MANAGEMENT
What is GENIVI IVI Layer Management?

- Defines a set of common API for managing composition of screen outputs of various standalone apps.
- APIs are designed to be hardware and window system independent.
Layer Management Service Overview

Layer Manager Client

- ilmClient

Layer Management Service

- LayerManagerService
- Communicator
- Renderer

Communicator Plugin(s)

Renderer Plugin

IPC

Window System

Dynamic Link

Dynamic Link
Existing Plugins in Genivi IVI LM

- **Communicator Plugins**
  - D-BUS
  - TCP/IP

- **Renderer Plugins**
  - X11/GLX
  - X11/GLES
  - Wayland DRM/GLES
  - Wayland FBDEV/GLES
  - Wayland X11/GLES
Renderer Plugin Architecture

- BaseRenderer
- Platform
- Graphic
  - Window System
  - Graphic System
  - Texture Bindings

Common Module
Platform Specific
Renderer Details

• Base Renderer
  – Base class of the Renderer plugin.

• Platform
  – Platform, e.g. X11 and Wayland, specific Renderer implementation.

• Window Systems
  – Platform specific implementation of window system.
    • Composition and input event handling.
  – Uses Graphic Systems to render client surfaces.

• Texture Bindings
  – Platform specific implementation of texture binding for GLES.

• Graphic Systems
  – Renders client surfaces onto layers.
Renderer code in LM 1.0

- **Platform**
  - TextRenderer (for testing)
  - Wayland{Drm|Fbdev|X11}GLESRenderer
  - GLXRenderer
  - X11GLESRenderer

- **Window Systems**
  - Wayland{Drm|Fbdev|X11}WindowSystem
  - X11WindowSystem

- **Texture Bindings**
  - WaylandGLESTexture
  - X11{CopyGLX|CopyGLES|EglImage|TextureFromPixmap}

- **Graphic Systems**
  - DrmGLESGraphicSystem
  - GLESGraphicsSystem
  - GLXGraphicSystem
DIRECTFB
What is DirectFB?

• Lightweight graphics middleware.
  – Small footprint (< 1.5MB core for ARM in version 1.7.0)
  – Licensed under LGPL.

• Supported features:
  – Graphics rendering with hardware acceleration.
    • All rendering primitives have software fallbacks.
  – Optimized software rendering, e.g. use of MMX.
  – OpenGL/ES2 support.
  – Multi-process, multi-layer, input handling, windowing etc.
  – Qt integration available.
Key Components

• DirectFB Core
  – Graphic renderings
  – Input managements
  – Frame buffer managements
  – Graphics hardware managements

• Fusion, One, FusionDale
  – IPC mechanism

• SaWMan
  – Application Management Framework
DirectFB Architecture

- gfxdrivers (H/W Renderer)
- input drivers (key, touch, etc.)
- GeneFx (S/W Renderer)
- video/image providers (Decoders)
- systems (H/W Manager)
- surface pools (Resource Manager)
Screen, Layer, Window, and Surface

App → Window → Surface

Windows Composition by a Window Manager

WM → Surface → Layer 0

Layer Composition by Hardware

Screen (LCD etc)

App → Window → Surface

Surface → Blit

Application may get exclusive access to a layer.

Layer 1 → Surface

Layer N → Surface
Rendering in DirectFB

Application

DirectFB Core

gfxdriver (h/w specific)

GeneFx (s/w gfxdriver)

request rendering (e.g. blit)

query if the hardware supports it

check

request rendering if supported

request rendering if gfxdriver doesn't support
GENIVI LAYER MANAGEMENT MEETS DIRECTFB
Where does DirectFB fit?

Layer Manager Client

ilmClient

Layer Management Service

LayerManagerService

Communicator

Renderer

Communicator Plugin(s)

DFBRenderer Plugin

IPC

DirectFB

Dynamic Link

Dynamic Link
DFBRenderer Architecture

Renderer Plugin

BaseRenderer

DFBRenderer

Graphic

DFBWindowSystem

DFBGraphicSystem

DFBTextureBindings

OR

GLESGraphicSystem
Before Going to Details...

**Application**
- Output
  - IDirectFBSurface

**DFBRenderer**
- App Surface
  - IDirectFBSurface
- Blit() 
  - IDirectFBSurface
- Output
- Denotes API call.

**DirectFB Core**
- Object
  - Surface
- Surface

Denotes an object to be manipulated with the interface.

Interface to manipulate objects
DirectFB Graphic System

Accelerated 2D compositing including scaling, blending, chroma key, ...

**Application**
- Output
  - IDirectFBSurface

**DirectFB Core**
- Surface
- Surface
- Surface

**DFBRenderer**
- App Surface
  - IDirectFBSurface
  - Blit()
  - Output
    - IDirectFBSurface

**Application**
- Output
  - IDirectFBSurface
  - Blit()
    - IDirectFBSurface
DirectFB Graphic System

Real-time composition in hardware layers
- fixed set of planes with same or different capabilities
- pixel format, opacity, source, dest, blend, key, orientation
DirectFB Window System

Maximum performance clients rendering directly to the hardware layer, no need for composition by the Layer Manager itself.
DirectFB Window System

Internal frame management, synchronization with clients.

Application

IDirectFBSurface

Flip()
- send update event with frame counter and time stamp (PTS)
- wait for a free buffer if required

DirectFB Core

Surface

DFBRenderer

Event Loop

App Surface

IDirectFBSurface

FrameAck()
- notify client interface about frame counter rendering with next

27,28 May 2013
Automotive Linux Summit 2013
DirectFB Advantages for GENIVI IVI LM

- Portable across platforms/APIs (no diversity needed in Layer Manager)
  - Abstract Hardware Display Layer API for Layer Manager's HW Layers
- Accelerated 2D Rendering on various hardware and software
  - DFBGraphicSystem does not require EGL or GLES
  - DirectFB 2D Clients also accelerated via GLES2 if available
- Crash safe Layer Manager by running as a Client (can restart/attach)
- Optimised performance for shared memory based Clients (non-EGL, e.g. SW)
  - DirectFB's Surface Core manages transfers between buffers, textures... with automatic updates on demand using glTexSubImage2D
- Flexible Layer Manager with Clients using different Window APIs
  - e.g. native DirectFB Clients, X11 Clients and Wayland Clients (planned)
DirectFB for Various Apps

- DirectFB Apps
- X11 Apps
  - XDirectFB
- Wayland Apps
  - libWayland-DirectFB
- Layer Manager Service
  - DFBRenderer Plugin

DirectFB Core

- 2D GPU
- 3D GPU

- IVI Apps
- Existing OSS
- Under Development†
- Hardware

† libWaylandDirectFB and DFBRenderer are planned to be released in H2 2013.
## Comparison in Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Wayland</th>
<th>X11</th>
<th>DirectFB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract 2D acceleration across platform/driver models</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Abstract Hardware Display Layer API</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Client can render directly to HW layer without composition</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Compositor as client (with restart ability)</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Compositor as server</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Compositor hierarchy</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Composition of one application in multiple compositors</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Triple buffered, frame sync</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Surface pools, bridges, wrap existing native handles</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Network transparency (clients on other VMs / components)</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Non-IPC build mode (single process, direct calls)</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Support for non-POSIX / embedded / real-time OS</td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>