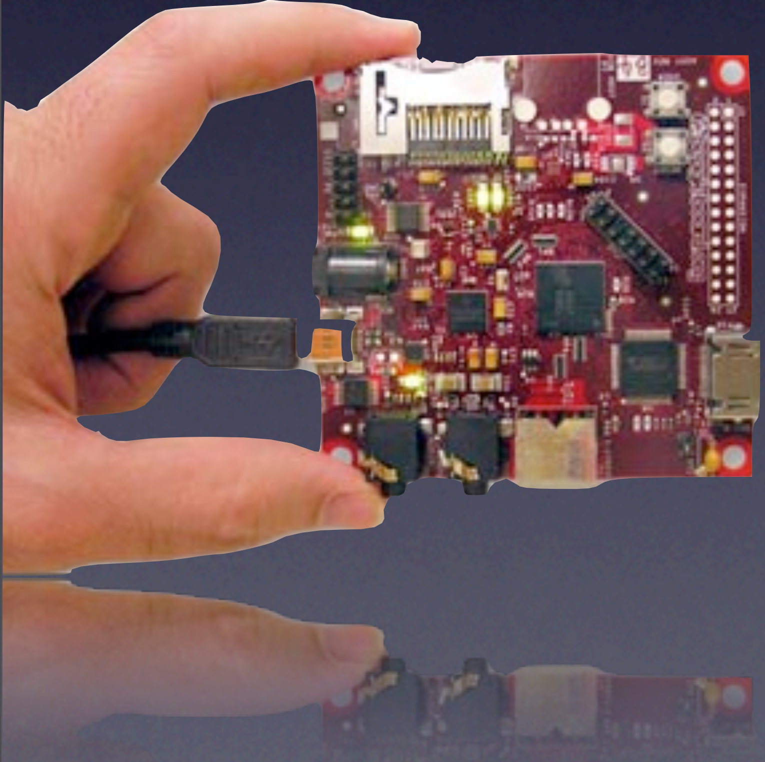
Introduction to creating 3D UI with BeagleBoard



ESC-341

Presented by Diego Dompe
diego.dompe@ridgerun.com

Agenda

- Introduction to BeagleBoard multimedia architecture features.
- Review of Linux graphic stacks
- Introduction to clutter programming
- Hands on and exercises
- Review some example applications
- Questions section (also allowed during the class)

Requirements

- BeagleBoard booting with ESC SD image + Patches.
- Install the SGX Drivers (accepting the SGX drivers license):

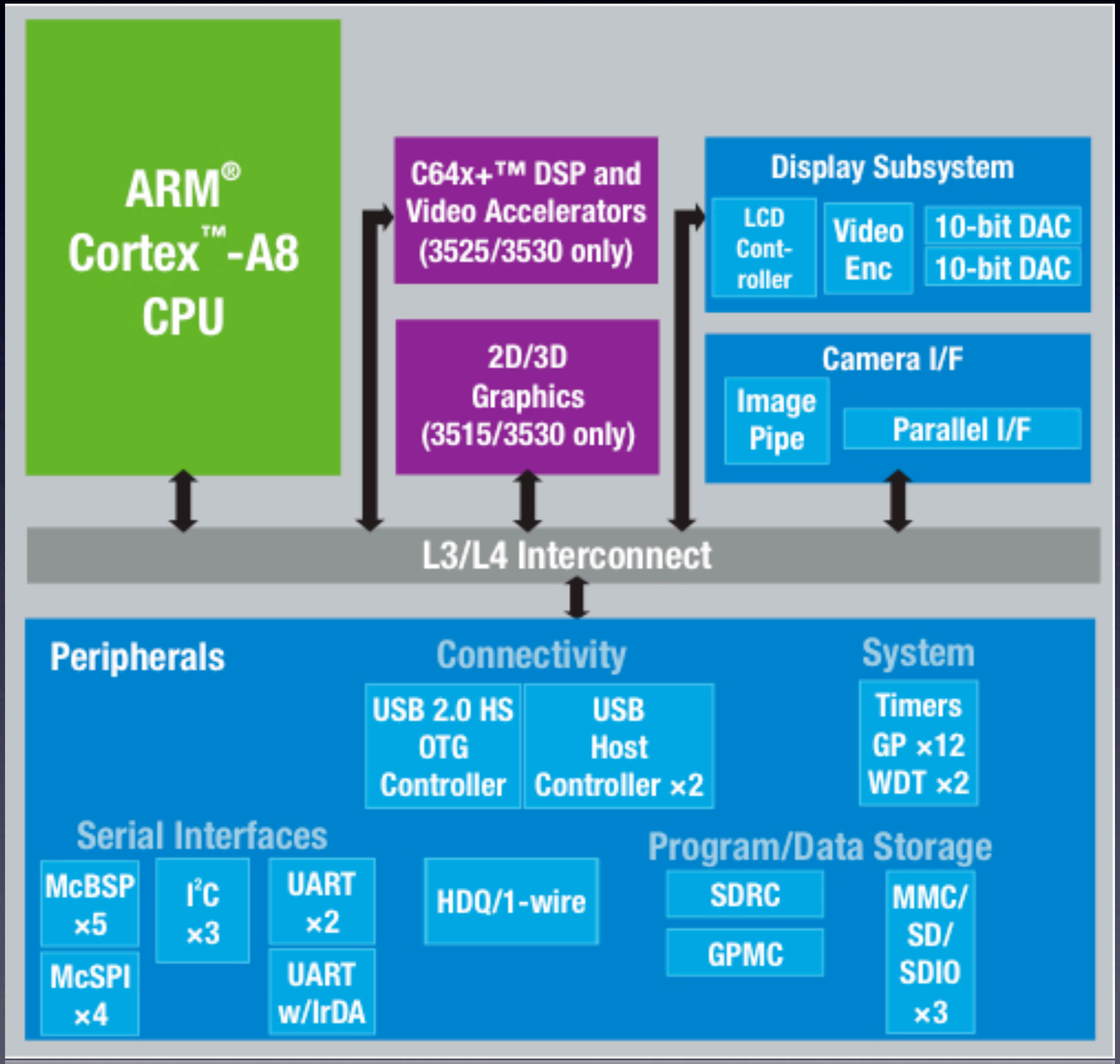
```
$ ./gfx_rel_ddk.sh
  (type ‘q’, then ‘yes’ (if you agree...)
$ cd gfx_rel
$ ./install.sh
$ reboot
(restart your board)
```

Introduction

- Primary objective for this class is to introduce audience to the main concepts and technologies available to start developing 3D UIs with BeagleBoard.
- This doesn’t class wont cover extensively the APIs available and assumes basic knowledge on C programming, Linux and computer graphics.

BeagleBoard and OMAP3 architecture

- Provides architecture with several multimedia features:
  - C64x DSP
  - Cortex A8 with Neon
  - SGX graphics chip
- How to maximize the hardware features with the software stack?



SGX Graphic Accelerator

- Standards supported: OpenGL ES 1.1, 2.0 and OpenVG.
- SDKs available for Linux and Windows from Imgtech[0]. Enables development and training on host machine.
- Drivers and libraries for BeagleBoard available early next year.

OpenGL ES for the UI?

- OpenGL ES is designed for generic propose and doesn’t provide complete solution for UI creation:
  - Input or event (picking) handling.
  - Text rendering.
- EGL layer provides standard API for OGLES implementation.

UI integration with the hardware features

- How to create user interface that integrates the best of the platform without having to write custom software stacks.
  - DSP Codecs
  - 2D / 3D hardware acceleration
  - Input / Output
- Using standard open source technologies.

Understanding Linux graphics stack

- Graphics stacks has several components:
  - Graphic environment (windowing system)
  - Support libraries
  - Widget toolkits

Popular embedded Linux graphic libraries

- X Server
- DirectFB
- Qtopia

- Cairo
- freetype / fontconfig
- pango
- gstreamer

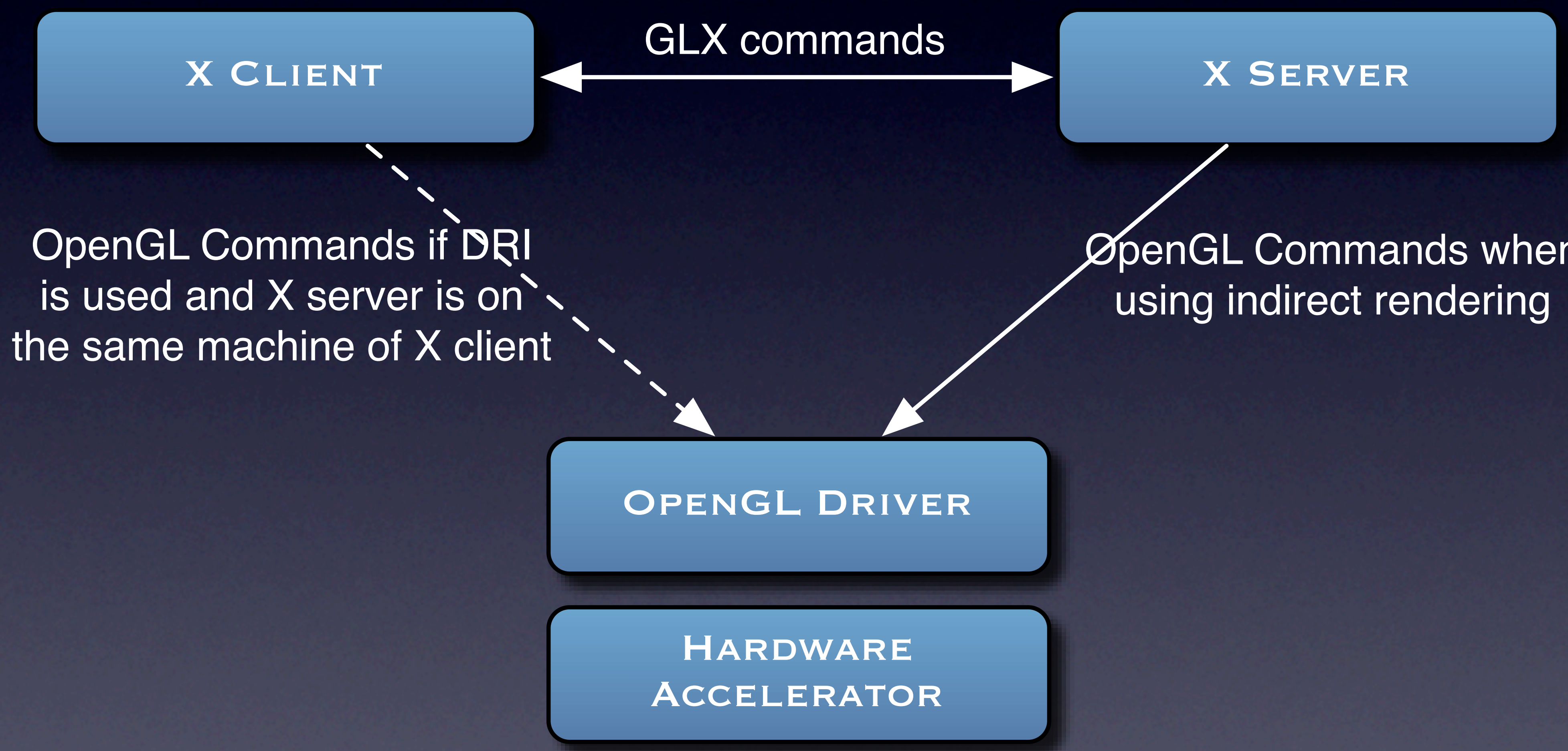
3D on Linux: X acceleration

- Several approaches:
  - AIGLX is the main one.
  - Xgl is abandonated
- Existing acceleration for Linux uses the GLX protocol.

3D on Linux: GLX

- GLX is the protocol to bing X with OpenGL:
  - Handle contexts
  - Send OpenGL commands over network
  - Can do direct or indirect rendering

3D on Linux: GLX


X Client
GLX commands
X Server
OpenGL Commands if DRI is used and X server is on the same machine of X client
OpenGL Commands when using indirect rendering
OpenGL Driver
Hardware Accelerator

GLX and OpenGL ES

- GLX doesn’t support OpenGL ES
- Alternatives are software OpenGL implementation with OpenGL ES backend: experimental projects doing this with Mesa backends for OGLES
- Existing projects doesn’t support all the OpenGL functionality
- Gallium 3D project is doing OGLES backend for the new DRI interface. Not available yet.

EGL

- EGL is an standard API for OpenGL ES and OpenVG to interact with the native windowing system.
- Depends from the vendor what the native windowing system is.
- Current SGX drivers on Beagle supports either the framebuffer or X window connection (but the former is not working well yet)

3D without X: clutter project

- Library to create fast, rich, animated user interfaces.
- Based on standard open source technologies used in Gtk+
- Support different backends: OpenGL and OpenGL ES. Uses an abstraction layer: COGL.

Clutter features

- Provides font rendering with pango.
    - i18n
- Event handling and picking.
- Provides animation support.
- Simple basic widgets: labels, images. Complex widgets are not difficult to develop.

Clutter features

- Features integration with:
  - Cairo: advance image rendering, door open for hw acceleration.
  - Gstreamer: enables integration with hw accelerated codecs.
  - Webkit
  - box2d: 2D physics engine
  - Support for several language bindings.

Clutter features

- Supports an EGL backend: work with OMAP3 SGX driver.
- EGL backend supports tslib for input, generic kernel event interface in the future.
- Gtk+ integration allows to run a clutter scene inside a standard Gtk+ windows over X or DFB.

Clutter disadvantages

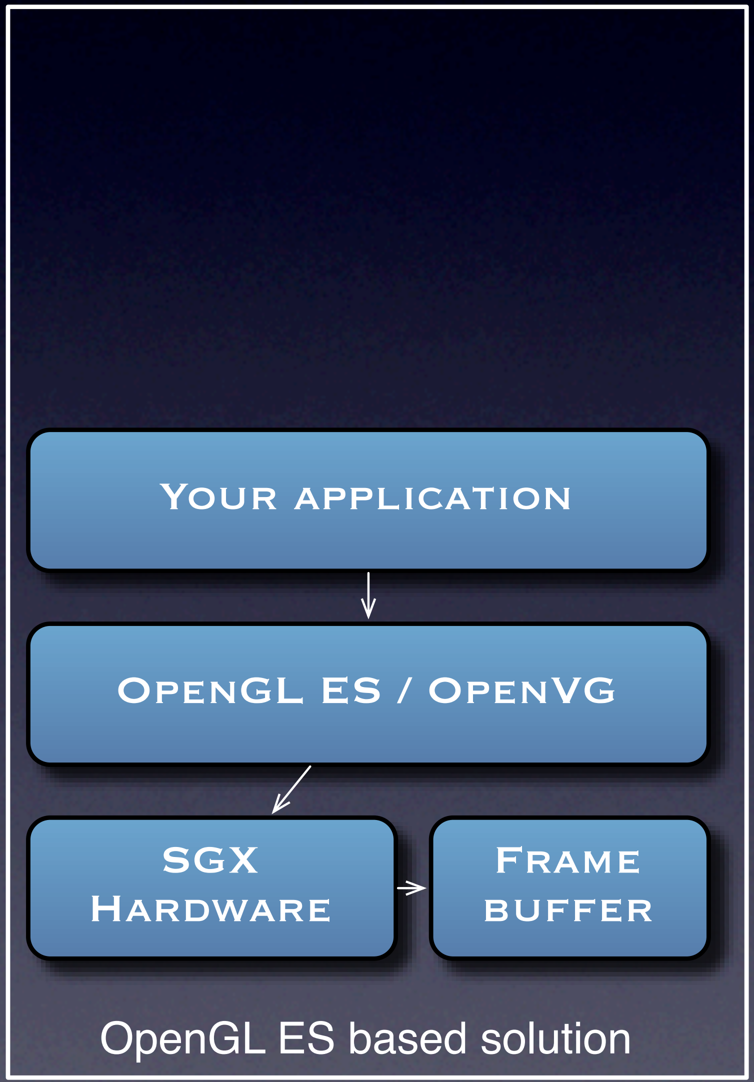
- Needs more widgets.
- Lack on screen keyboard.
- Young project, need more documentation and examples. Getting there.

Clutter usage scenarios

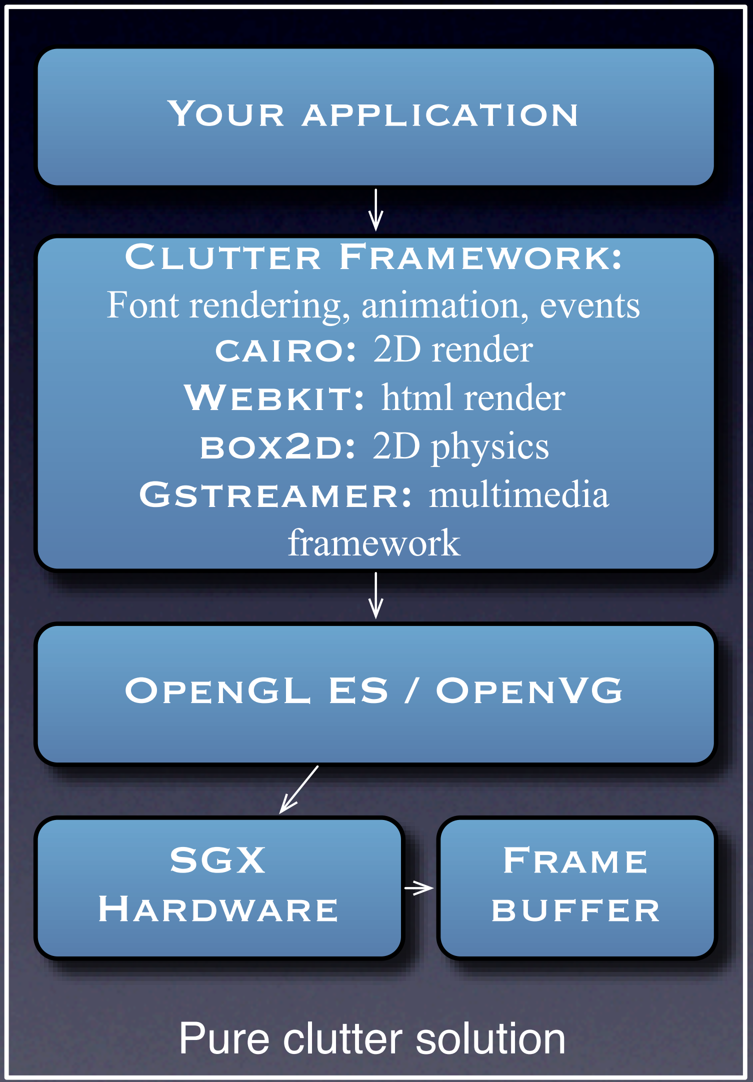
- Pure clutter solution.

- Clutter over GTK+ solution.

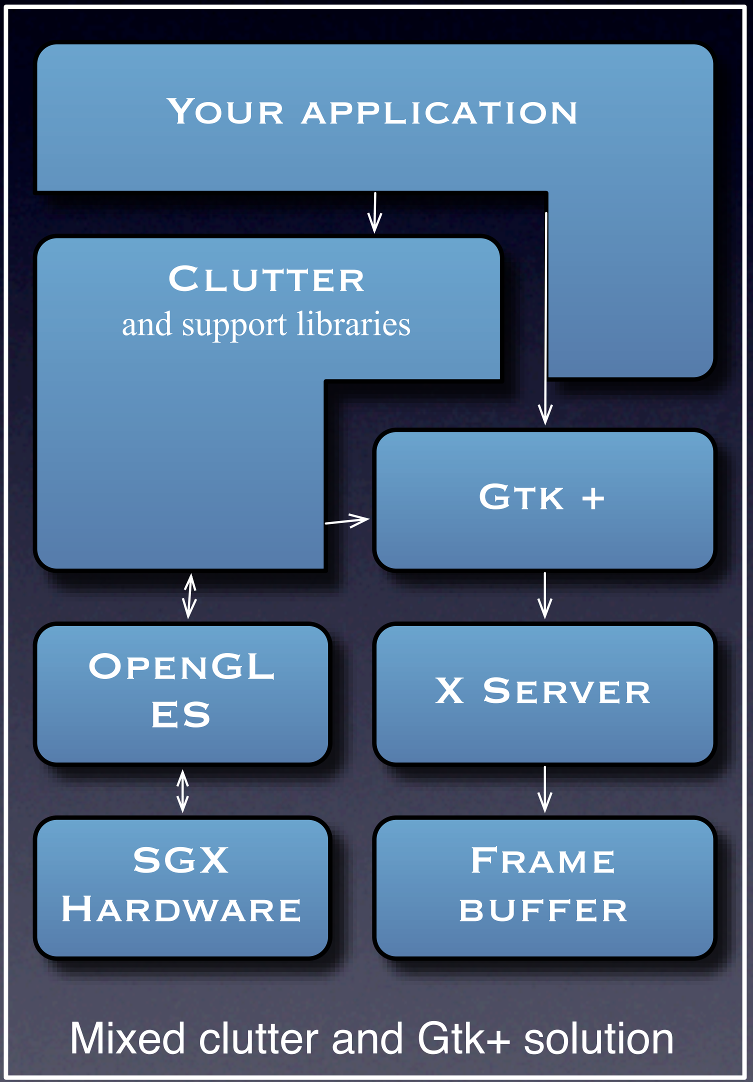
Visualizing the possible software stacks



OpenGL ES based solution



Pure clutter solution



Mixed clutter and Gtk+ solution

Hands on...

- Hello world
- Events
- Animation
- Images: textures and cairo
- 2D physics
- Gstreamer: video and audio
- Putting all together

Examples:

```
$ cd ESC341
$ touch * # Prevents some timestamp warnings
$ make
$ ./hello
$ ./hello2
$ ./events
$ ./bubbles
$ ./clock
$ ./videotest
$ ./video /root/gst/bbb_320x180.mp4
```

Gstreamer and OMAP 3

- Gstreamer video sink available (see ESC presentation on Gstreamer)
- Gstreamer can use NEON optimized Codecs on ARM side, or plugins for TI's CodecEngine for DSP Codecs.

Q&A

References

[0] http://www.imgtec.com/downloads.asp

[1] http://www.clutter-project.org/docs/clutter/0.8/

[2] http://www.openismus.com/documents/clutter_tutorial/0.8/docs/tutorial/html/

[3] http://focus.ti.com.cn/cn/lit/wp/spry110/spry110.pdf