VA (Video Acceleration) API

Jonathan Bian

2009 Linux Plumbers Conference

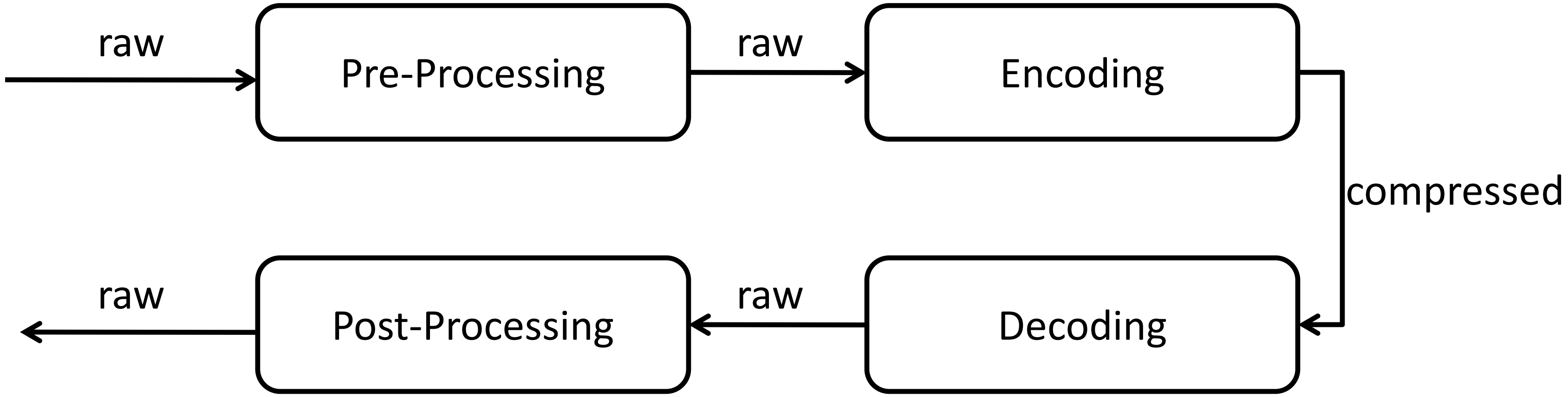
Motivation for creating a new API

- Lack of a video decode acceleration API for Unix-like OS that fully exposes fixed function video decode accelerators in modern GPUs
  - XvMC was designed to accelerate MPEG-2 motion compensation and somewhat difficult to extend to the newer entry-points (e.g. VLD) and formats (e.g. H.264 and VC-1)
  - OpenMAX DL interface targets optimizing fine-grained primitives on programmable cores and is not suitable for exposing fixed function hardware accelerators

Design goals

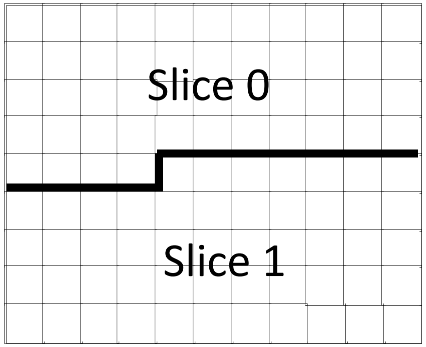
- The API should accommodate hardware video accelerators from many different IHVs
- The decode/encode API should be independent of the windowing system to make codec implementations portable
- The interface should be lightweight and extensible, and enable efficient transfer of parameters and coded data between the app and the hardware accelerator
- An open API development environment for wider industry and community adoption

Scope of video coding



VA API supports both decoding and encoding

Slice level decode



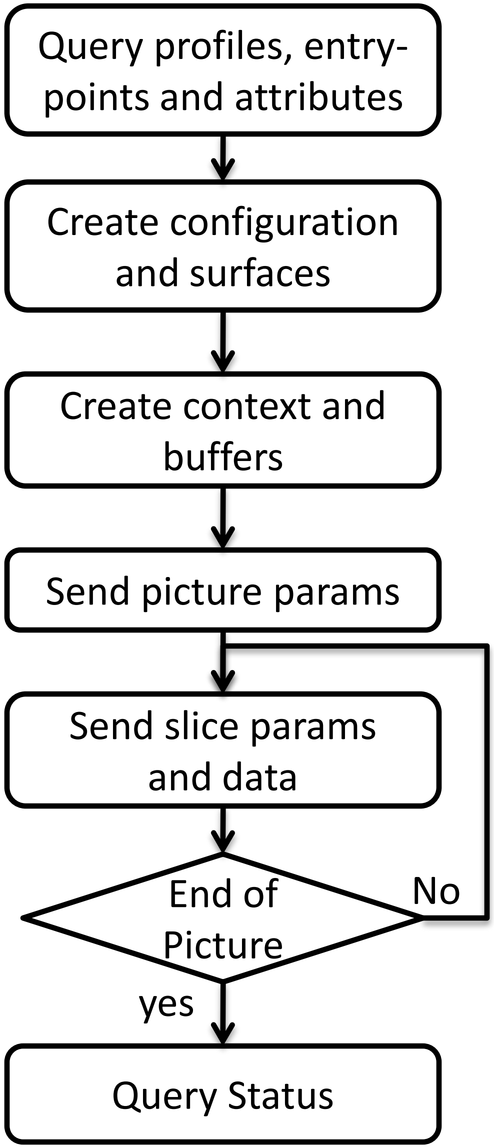
- A coded picture is divided into slices which are typically made of consecutive macroblocks in raster scan
- Slices are self-contained and form the basic units for error recovery

- Slice level decode (a.k.a. VLD entry-point) is supported by nearly all current video hardware
- App sets up picture and slice parameters, and sends compressed slice data to the HW

API objects

- *Context*: represents a virtual decode or encode pipeline
- *Configuration*: negotiates and configures pipeline profiles, entry-points and attributes
- *Buffer*: stores coding parameters (e.g. picture and slice parameters) and compressed video data
- *Surface*: stores uncompressed video frames (e.g. reference frames) operated on by the hardware accelerator and typically in hardware native formats (opaque to the app)
- *Image*: stores graphical and video pixels in well-defined formats, can be used to retrieve surface data or get client image data into a surface, or compositing video with graphics (e.g. menus and subtitles)

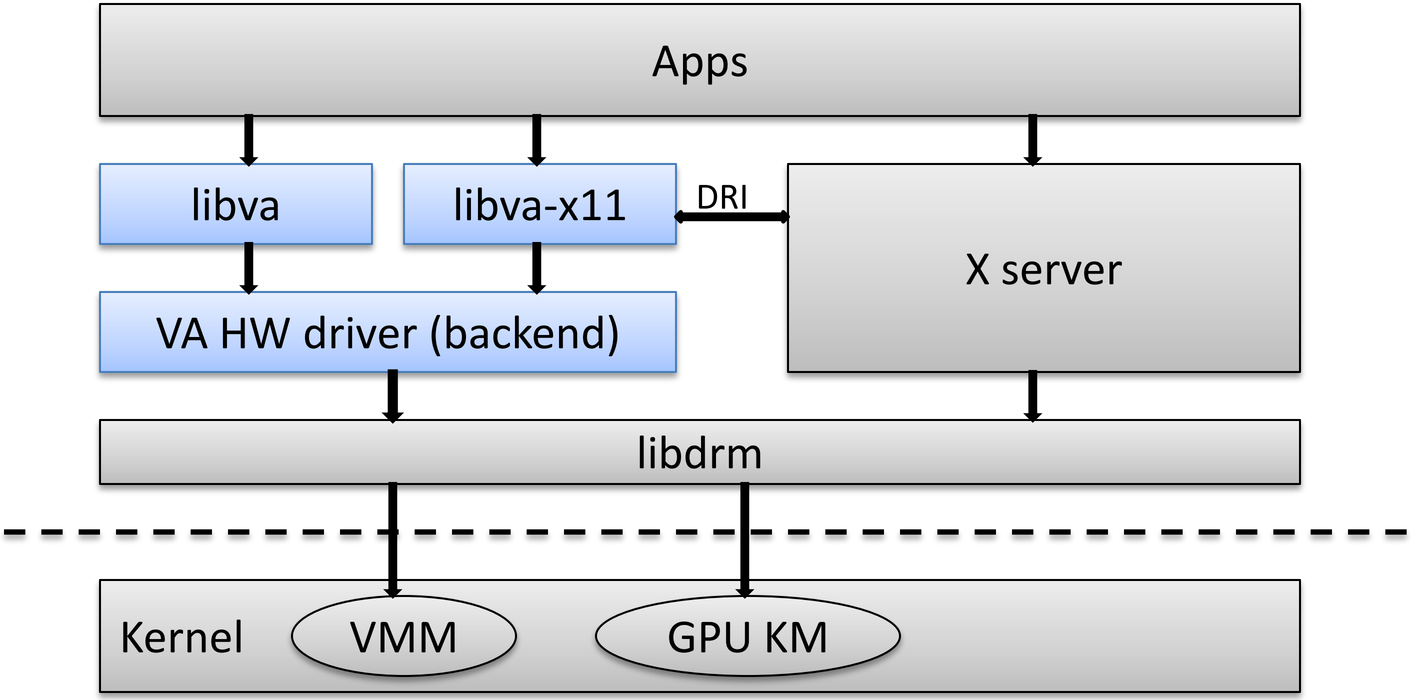
Basic schema for decoding one picture


Query profiles, entry-points and attributes
Create configuration and surfaces
Create context and buffers
Send picture params
Send slice params and data
End of Picture
No
yes
Query Status

Composition and presentation

- Composition is done by blending an image with an uncompressed video surface
  - Supports chroma keying, global alpha, per-pixel alpha
- Presentation is performed as a single function call, with a X drawable as the render target
  - Currently supported operations include de-interlacing, HSBC, scaling, color space conversion, and clipping rects
  - If the target drawable is a pixmap, then the video frame can be turned into a texture through the texture_from_pixmap extension

Implementation under X



Major milestones

- March 2007: First draft of the API spec announced on the x.org mailing list
- Sept 2007: libva git repository created on freedesktop.org
- Dec 2008: ffmpeg/mplayer support by Gwenole Beauchesne
- June 2009: API version 0.30 released with support for encode
- July 2009: Intel integrated graphics driver released, libva development mailing list created

Apps and drivers support

- Open-source apps: ffmpeg, mplayer, vlc
- Commercial codecs: GStreamer (Fluendo), Helix (Real Networks)
- Drivers:
  - Native HW: Intel G45, GMA500; S3 Chrome
  - Backends: VDPAU, XvBA

Future directions/thoughts

- As GPUs get increasingly capable, more advanced post-processing features (e.g. STDC, LACE) become viable
  - Separate post-processing operations from presentation and enable a modularized post (pre) processing pipeline with pluggable components
- Elementary stream entry-points to target consumer electronics and low power platforms
- Integration with OpenCL
  - Ability to download OpenCL kernels into the post (pre) processing pipeline
  - Extend OpenCL to support fixed function decode and encode

For more information

- Browse the VA API project page:
  - http://www.freedesktop.org/wiki/Software/vaapi
  - Links to the source repository, drivers and backends, sample code and apps
- Join the libva mailing list
  - http://lists.freedesktop.org/mailman/listinfo/libva
  - Discuss issues with libva and drivers, as well as enhancements and proposals, submit patches