Open Source Broadcast Encoding

Kieran Kunhya

LSM 2012

kierank@ob-encoder.com

Why Open Source broadcast encoding?

- Almost entirely based on open standards DVB/ATSC/ETSI/MPEG/SMPTE etc.
- Relatively little communication with other devices (usually SDI/IP in and ASI/IP out) – so few (if any) proprietary protocols to use.
- World class encoder(s) available and CPUs now very powerful

Current marketplace

- Top-end broadcast encoders are FPGA based – almost homogeneous feature set.
    - Sometimes use of CPU to offload parts of work (e.g. Rate control)
    - Modern encoders usually running Linux
- Main features are encoding from SDI or transcoding from IP/ASI.

Current use of OSS for Broadcast Encoding

- Notable users are Avail-TVN (USA) and Free (France).
- Usually in-house or at least some customisation
  - Usually have staff used to working at a relatively low level with OSS
- No turnkey system for broadcasters less familiar with OSS to deploy
  - Important to maintain realtime and handle things like teletext, captions etc

x264

- World class MPEG-4/AVC encoder in an number of areas
  - Used heavily on Web (Youtube/Facebook etc), Blu-Ray (Warner) and in other areas (e.g. Cloud gaming).
  - Technical strengths and weaknesses compared to hardware encoder.
  - Supports professional features like 10-bit 4:2:2.
- Historical links with Ateme – similar paths taken (never a formal relationship)
- LPB (and recently Avail-TVN) working on x262, MPEG-2 encoder using x264 toolkit

History of OBE

- LPB and one Norwegian IPTV provider interested in software broadcast encoding.
- Led to creation of Open Broadcast Encoder
  - First tool was VoD creation – useful to test STB support
    - No further plans with VoD
  - Later on went to realtime encoding
- Other broadcasters/operators got involved at different stages
  - Frequent communication - loosely could be called an “advisory board”

Development

- Development done on production broadcast chains
  - Ubiquitous Blackmagic SDI cards made this quite simple
- Tested using analysers on the broadcast chain
  - Often streams sent to the home
- Speedcontrol in x264 (originally developed by Avail-TVN) to manage encoder speeds to maintain realtime encoding – adds latency
- T-STD compliant MPEG-TS multiplexing
- **Features not claimed to be stable unless in production**

Goals of project

- **Match or beat mainstream broadcast encoder features/quality on commodity hardware:**
- Low-latency contribution (interviews ~200-300ms end to end) – OBE users exist
- Mid-latency contribution (~1s encoder latency) – **No OBE users exist yet!**
- Distribution (1-2s encoder latency) – OBE users exist

Psychological Goals

- Small-to-mid size broadcasters more likely to deploy– fewer people to convince (almost always no financial incentive like other areas of OSS).
- Easier to picture for many operators with a "setup encoder and forget about it" view – "replace hardware encoder with server". Compare this to integration of FFmpeg into transcoding workflow (bash scripts? metadata? etc.)
- Make current OSS users appreciate at this stage there will be loss of flexibility
- Users should feel that the encoder will behave like proprietary counterparts

OBE deployments – Louisiana Public Broadcasting (LPB)

- Statewide PBS network operated by the Louisiana Educational Television Authority (Very small EBU member sized)
- Use the state academic IP network for inter-site feeds and for six ATSC transmitters with a DVB-S2 backup.
- Use OBE to distribute coverage of legislature and nightly state lottery results
  - Enabled them to divide their satellite space segment and sell capacity for occasionals
  - Wide range of receivers, from consumer receivers to most expensive IRDs

Future LPB work

- OBE in their OB truck.
- OBE for multiple inter-site feeds.
- MPEG-2 for other uses (inter-broadcaster feeds)

OBE deployments – Norwegian IPTV

- Encode an HD and SD service of a national channel.
  - Though sent to homes only viewable by engineers for reasons unrelated to OBE.
  - Monitored using Agama probes.
- Demonstrated problems with lack of frame-synced sources

OBE deployments – USA interviews

- US company uses OBE for low-latency news interviews
- OBE provides around 300ms end-to-end latency using P-frame only stream

OBE deployments – Frikanalen

- Community television station in Norway
  - Run in part by NRK engineers
- On the national DVB-T network
  - Useful introduction to working with DVB-T – worked closely with the control centre
  - Very likely the first permanent DVB-T service encoded with OSS.
- Uses experimental ASI output
- Frikanalen also uses OSS playout

OBE deployments – Far East

- Broadcaster in the Far East using OBE on DVB-S2/DVB-T2
  - Independently deployed
  - Future plans involve a lot more OBE channels – **Watch this space!**

Current requirements

- OBE clock is locked to SDI - frame synchronisation strongly recommended.
    - Rare problems with frame drops on unsync'd sources. Cards don't provide much information about drops.
    - Use frame arrivals as clock ticks and interpolate between ticks
- CPU requirements involve latest Intel CPU for HD
    - Haswell CPU next year should provide major speed increase
- Ubuntu 64-bit mandatory – CentOS planned
    - CentOS has older components so requires testing

Why not OTT or other web streaming?

- OBE must follow market segmentation of encoders. Market is awash with OTT encoders but small number of world class television encoders.
- Over half of all enquiries about streaming.
  - Perception that OSS isn't good enough for professional encoding.
  - Need to prove to masses that OSS is capable of more than streaming! **Deploy as much as possible.**
- OTT devices don't care about hard real-time
  - Huge buffers, no respect for MPEG-TS buffering models

Why not VLC, Gstreamer, FFmpeg etc?

- VLC good for streaming
- Difficult to merge consumer hacks and broadcast hacks - Work duplication is regrettable
- OBE tries to return as much code where it's a good fit to upstreams (e.g. Swscale and v210 assembly).
    - Notably a number OBE development machines shared with FFmpeg/Libav/x264.

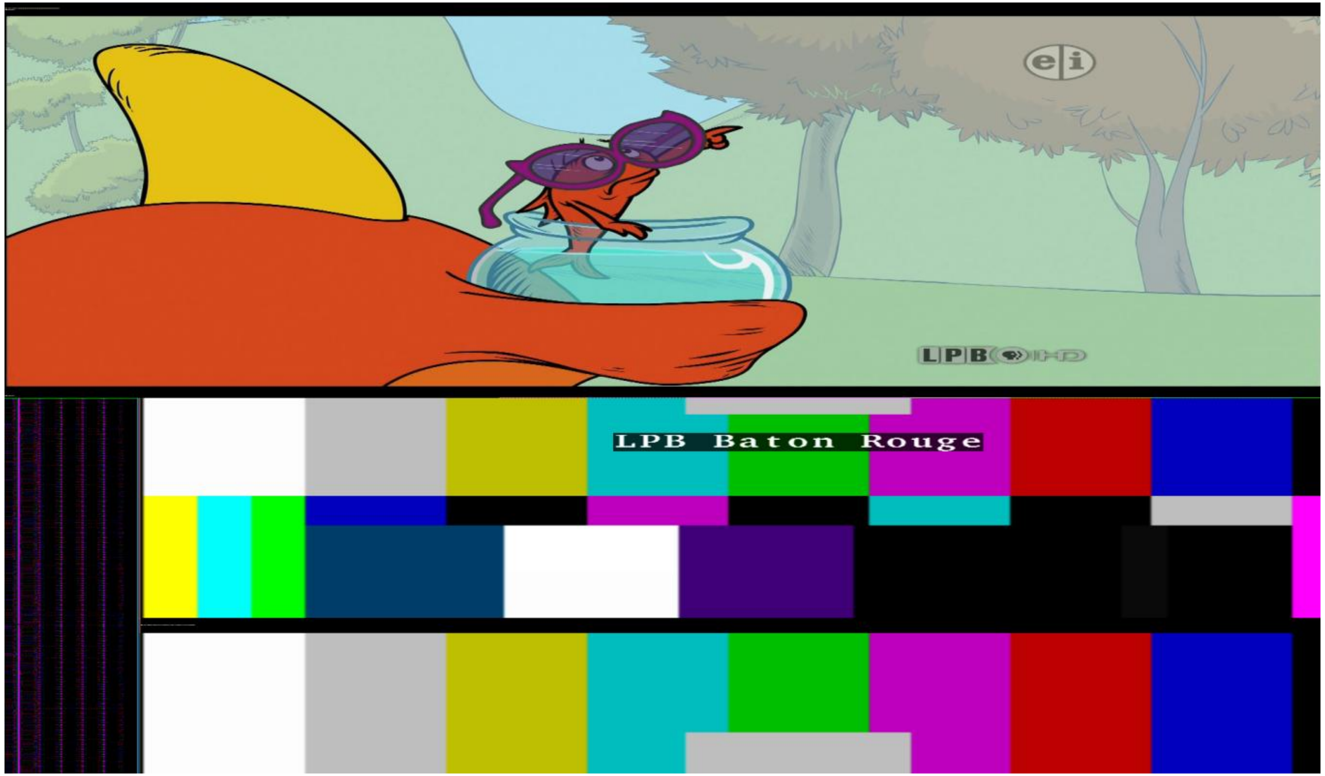
Audio

- Historically no good OSS (HE-)AAC encoder
  - Changing with Google release of Fraunhofer AAC encoder as part of Android 4.1
- Historically OSS didn’t bother with metadata, channel reconfiguration
- Channel map changes are not 100% defined
- Cards don’t provide audio control packets

Dependence on SDI/ASI hardware

- OBE is dependent on closed hardware
  - ALL usable SDI and ASI cards are buggy or have issues
    - Current ASI card (DVEO/Linsys) have increasing latency
    - DVEO/Linsys HD-SDI cards have inherent lipsync problem
    - Closed Blackmagic drivers crash sometimes upon load
    - No proper access to AES frames
    - etc...
  - Cards requiring NDAs are of course unsuitable
- Open Hardware is **very important**
  - Bugs can be fixed like software.

Example DVEO (Linsys) HD-SDI bug



Splitting out OBE components

- SDI-related code worth splitting into library for other applications to use.
  - How low-level do you go? So few cards with OSS drivers...
  - VANC code is definitely reusable but too small to form a library.

GPUs

- Lots of people ask about GPU encoding
- GPUs are NOT designed for the serial nature of video encoding
- x264's lookahead has been ported to GPU but CPU lookahead currently beats it
- (Personal opinion) "GPU encoding" is largely a marketing scam.
- HOWEVER, GPUs could be useful for filtering such as very high quality deinterlacing algorithms (think Alchemist quality...)

Patents

- Lots of OSS "commentators" make outlandish claims like GPL can't be used with a patent licence.
  - Gone are the days of robot-email patent threats - world has moved on.
  - Outlandish claims damage use
- As with proprietary software there are clear limitations and limitations which are in a grey area. I will talk about what you can do.
- MPEG-LA have said source code distribution is not a product which requires royalties. Public statements could be asked for from them and others.
  - Modifying and shipping source keeps you in the clear.

Commercial Model

- There has to be a sustainable commercial model to pay for R&D
    - Days where broadcasters/operators can fund improvements are coming to an end
- Proprietary management tools
- Support contracts down the line

How can broadcasters get involved?

- Explain your use-cases for OBE. Be aware that niche features that affect OBE heavily may not be followed up.
  - Equally talk to us before heavily modifying OBE because your code may not be accepted.
- If you'd like to deploy but need to convince higher-ups, tell us what you need in OBE (technically or commercially) to make it happen.
- Tell us about UX problems between OBE and others.

Deployments and R&D

- OBE will have a recommended set of system requirements for current SD and HD formats.
- Anything else is R&D territory
    - 1080p50/60 and 4K have been tested with file input but would require code changes to speedcontrol presets.

Future work

- Statmux, through Open Source mux (unlikely to be used at early state) or bitrate allocator (more likely).
- Dithering improvements (on the GPU?)
- Lowlatency uses MBAFF, PAFF on its way. (No fully-working cards available to receive only one field)
  - Leading to MBAFF/PAFF adaptivity
- Contribution feed Interoperability

Future work (2)

- Transcoding from ASI/IP – probably needs a new demux
  - FFmpeg demux outputs packets in non-standard order amongst other things and confuses speedcontrol
- Teletext to DVB-Subtitle conversion
- AVC-Intra over TS/MXF – Interoperability nightmare…
- 3D encoding using Multi-view Coding (MVC)?
- HEVC. x265 being written but not linked to current x264 – different licence, written in C++…

Closing remarks

- Not trying to make broadcast encoding IT-centric
- A step on the road to flexible broadcast encoding?
- Talk to other users: #obe on Freenode IRC
  - Ning is a more modern way – ob-encoder.ning.com

Thank you for listening