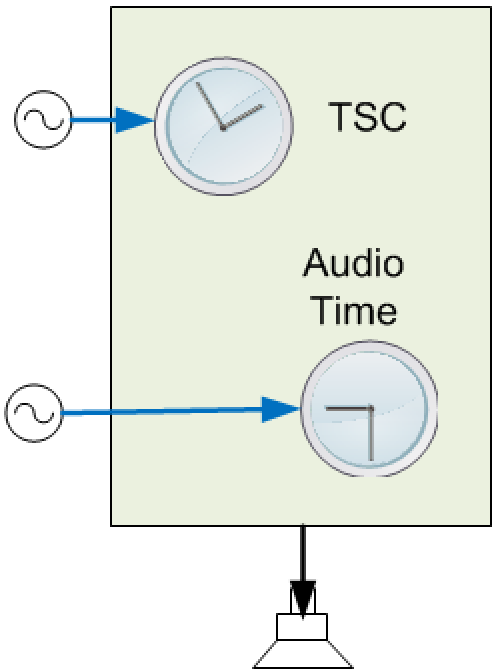
Audio/system time alignment

Pierre Bossart

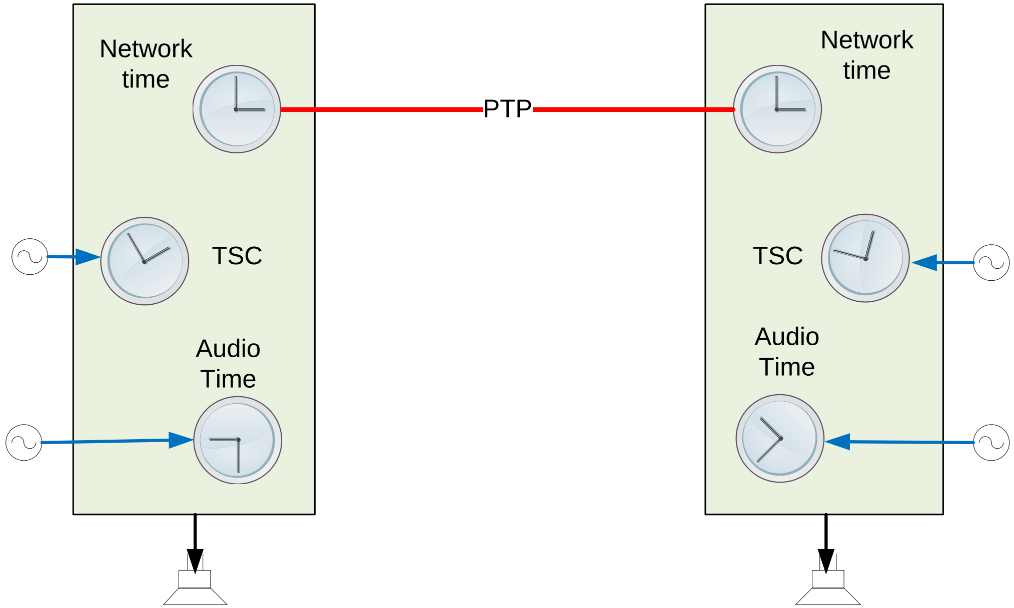
08/29/2012

TSC/Audio time alignment

- Audio time and TSC will drift by construction
- Recent evolution in Linux/PulseAudio and Android/HAL
  - Use system timer to refill ALSA buffers
  - Allows for dynamic latency changes
    - Lower power consumption
  - Need correlation between system time and audio time

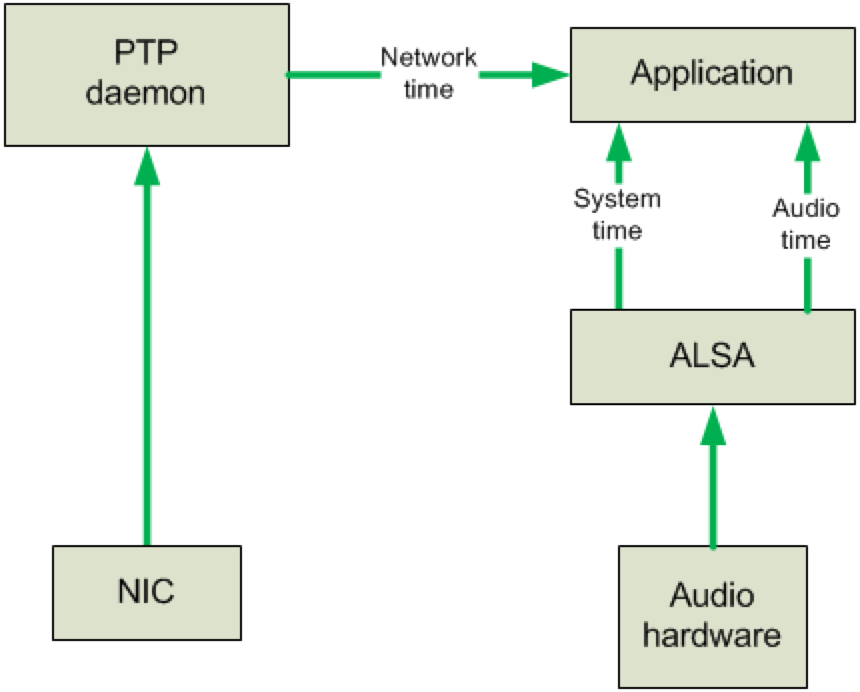


Ethernet AVB time alignment



- PTP relies on hardware timestamping to align different machines on a network
- **O(ns) accuracy! Yes NANOSECONDS!**
- Opens door to synchronized playback/record on multiple devices
  - Car entertainment
  - Pro-Audio

30,000-ft Linux architecture



Two steps:

- Measurement: system, audio, network time
- Alignment: sample-rate conversion, PLL adjustment, etc

System/audio time observability

- System time
  - ALSA can be configured to take timestamp of monotonic time
  - Available to app by calling snd_pcm_status()
- Notion of trigger
  - Timestamp also taken after .trigger is completed
    - some delay between actual start and timestamp, no hardware atomic operation
- Audio time
  - Not a concept directly available to user-space
  - Only <u>indirect</u> calculation: samples written – samples queued (snd_pcm_delay)
  - HDAudio:
    - LPIB or Wall Clock not directly accessible by user-space
    - Work-arounds for ‘Position Fix’ impact precision of audio time
  - Non-HDAudio
    - Hardware-pointer not available to user-space either
- Experiment: can we make audio time directly available to user-space?

Experiments with HDAudio

- LPIB only provides frame-level accuracy, 20us at best
- Order of magnitude higher than network synchronization
- Audio time measured with 24 MHz audio wallclock (47ns resolution)
  - Converted to ns for compatibility with system timestamps
- Practical issues
  - Wrap-around after 179s
  - Careful with overflow when converting cycles to ns (125/3 ratio)
  - Force use of SSYNC bits to take trigger_timestamp when transfers actually started
- Audio timestamp exposed along with system timestamps
- No ABI change, same STATUS ioctl
  - New field in pcm_runtime (taken from reserved bytes, no ABI change)
- New routine in ALSA-lib modified to extract audio timestamp from pcm_status structure
- Experiments with PulseAudio to align system time and audio timestamps
  - Audio time initialized to zero on start trigger
  - Relative drift = audio time / (system time – trigger_timestamp)

Results

- More precision with wall clock than with sample counters
- Difference with system time is jitter-free and quasi-monotonic
- <20ppm difference estimated in less than a second
  - PulseAudio smoother works with 10s history smoother
- Offset between sample counts and system time due to ‘PositionFix’
  - Up to 1ms offset between sample time and system time
  - 8us offset when forcing use of LPIB
  - Do we still need ‘PositionFix’ in no-interrupt mode?

Observations

- No global detection of audio/system time drift
  - Wall clock not available in D3
  - Wall clock only tracked when transfers active
- Audio timestamps exposed independently for each stream
  - If streams started at same time, application can estimate drift on a reference stream
  - If streams started at different time, need independent drift estimate on each stream
    - Not super elegant!
    - Drift should be same on all streams using same BCLK.
- Still open
  - Behavior in underflow conditions
  - Behavior with suspend
  - Currently audiotime only available for HDAudio
  - Might be useful to expose hw_pointer directly to userspace for all existing hardware
  - Actual time alignment
    - Need control on mixing granularity and ASRC group delay