FFADO: firewire audio for Linux

http://www.ffado.org

Linux Plumbers Conference 2008

Jonathan Woithe

1 Talk outline

- The FFADO project
- The firewire bus
- Why audio over firewire?
- Audio-specific firewire issues
- Manufacturer support
- Protocol analysis techniques
- Using FFADO
- Usability/integration issues
- Current status
- Future plans
- Helping FFADO
- Acknowledgements

2 The FFADO project

- FFADO: Free Firewire Audio Drivers for Linux
- Originally known as “FreeBob”, in reference to the “BeBob” platform used for some devices
- Renamed FFADO to be more vendor neutral after Freebob 1.0 release to reflect wider variety of devices being supported

3 The firewire bus

3.1 Properties

- High speed bidirectional serial bus
- Speeds of 400 Mbps and 800 Mbps in use today
- Up to 63 devices per bus
- The bus implements a global address space
- Each device is assigned a unique address range on the bus
- Often abstracted as a register model – each device has a collection of individually addressable registers

3.2 Packet types

- Asynchronous (“async”): addressed to a specific bus address – hense a particular address/register on a given device. No timing guarantees. Used for device configuration and control.
- Isosynchronous (“iso”): broadcast packets, sent on one of 63 “channels”. Guaranteed bandwidth and delivery timing. Used for audio data streaming.

3.3 Iso packet management

- Each is second divided into 3072 “iso cycles”
- Timing within an iso cycle specified by a “cycle offset” (0-7999)
- Iso clock runs at 24.576 MHz (3072 $\times$ 8000). This clock is synchronous across the bus.
- Each node can transmit one iso packet per iso cycle
- A node must request its iso bandwidth requirements. Can be denied if bus’s bandwidth already fully allocated.

4 Why audio over firewire?

- High bandwidth (400 / 800 Mbps, faster variants in development) – good for 20+ channels of audio at 24 bit / 48 kHz.
- Bus bandwidth almost entirely accessible to devices
- Devices have guaranteed bandwidth – suits streaming applications
- Slow devices can't block faster ones

5 Audio-specific firewire issues

- Need sample accurate synchronisation – audio data to/from device is time-stamped
- No direct access to device's hardware audio sampling clock (compare PCI devices)
- Only synchronous clock available is iso clock – devices use this to timestamp audio packets
- Devices don't lock audio clock to iso clock
- Must infer audio clock relative to iso clock (and/or wall time) based on incoming audio data timestamps. This inferred clock is then used to generate outgoing audio data timestamps.
- Also must compensate for buffer and transport delays through firewire stack and the device electronics
- FFADO uses a delay-locked loop (DLL) to do this
- Some devices can be very picky about the accuracy of the inferred audio clock

6 Manufacturer support

6.1 Ideal situation

- All firewire audio devices are class-compliant and adhere to the published standards for AV devices
- One common driver interfaces to all such devices

6.2 The reality

- There are published standards. But ...
- Most manufacturers use vendor-specific extensions for key aspects of device control
- Some manufacturers think they can do better than the standard and use their own

6.3 Why we need vendor support

- Only the vendors know the details of their own vendor-specific extensions
- Only the vendors know the details of their own protocols
- FFADO is receiving support from some device vendors: Echo, ESI, Focusrite, Terratec (now Musonic), Mackie
- Platform vendor support from BridgeCo and TCAT

6.4 Uncooperative vendors: our options

- Forget them. Not good PR – alienates people moving to Linux with significant prior investment in devices from these manufactures
- Support as best we can (protocol analysis) but discourage new purchases of such hardware

7 Protocol analysis techniques

7.1 Sniffing firewire traffic

- Put second PC on firewire bus. Can see iso packets, so audio data format can be deduced. But ...
- "Standard" OHCI firewire cards can't see async packets addressed to other nodes (by-design hardware limitation).
- Can use purpose-designed firewire analysis hardware. Expensive.
- Alternatives:
  - use card based on TI's PCILynx chip and "nosy" Linux software. These cards are hard to find. IOI technology still sells a PCI version (IOI-1394TT) at "reasonable" cost (approx US$50).
  - use older Apple Power Macs – blue-and-white G3 or original "Yikes" G4 (with PCI graphics adapter) – which used PCILynx-based firewire solution. Use with Apple's Firebug software.
- Latest nosy release (0.3) is old and doesn't work with recent kernels. Use recent git snapshot instead.

7.2 Sniffing – another way

- Use bushound (http://www.perisoft.net/bushound/) on OSs supported by hardware vendor.
- Sniffs bus traffic via a driver sitting very low in the driver stack.
- Rather expensive (US$700).
- Free binary version does exist and can be used in our context, but it runs on only one OS version.

7.3 Protocol analysis

- Capture async packets sent in response to configuration changes.
- Deduce protocol based on packet contents.
- Some protocols are easier than others. For example: some devices change configuration on register reads.

7.4 Indirect analysis

- Use if no traffic capture is possible
- Functions by comparing device registers before and after an operation
- Works well for devices whose registers are closely aligned to configuration
- Not so good on devices which rely on write sequences or side-effects
- Advantage: no “special” hardware required, more accessible
- FFADO trunk includes a tool to facilitate this method (scan-devreg)

8 Using FFADO

8.1 Interfacing to audio systems

- FFADO provides a JACK backend driver
- Userspace ALSA driver is planned

8.2 Prerequisites

- A modern JACK (0.112 seems to work well)
- User access to realtime (RT) scheduling (PAM, set_rlimits)
- libraw1394 (v1.2.1 or later), dbus (mixer data interface), Qt/Python (mixer GUI)
- libxml2 (perhaps removed in future)
- OPTIONAL: RT-patched kernel. Only needed for very low latency setups with some hardware devices.

8.3 Running (via JACK)

- OPTIONAL: boost priority of firewire IRQ handler:
  ```
  chrt -f -p 72 ‘pidof "IRQ 21"‘
  ```
  Not as effective as it could be due to scheduling prioritisation issues within the kernel firewire stack (eg: each traffic type has own softirq tasklet but all run at same priority).
- Run JACK “normally” but use “firewire” driver:
  ```
  set_rlimits jackd -R -P60 -dfirewire -r 44100 -p1024 -n4
  ```
- Audio I/Os now accessible via JACK in the usual way

9 Usability/integration issues

- RT scheduling configuration: could be easier
- Managing transistion from current firewire kernel stack (“ieee1394”) to the new stack (“firewire”, aka “juju”)
- sensible udev rules for firewire device nodes
  - One device node per bus model of “current” firewire stack
  - One device node per device model of “new” firewire stack
- Hotplug behaviour: what can be done, what makes sense for users?
- prioritisation issues in kernel firewire stack need addressing
  - Allow different softirq tasklets on system to have different priorities?
  - Handle iso traffic within irq handler?
  - Prioritise iso handlers depending on channel?

10 Current status

- Currently in beta testing for “initial release” (will be FFADO 2.0)
- Release date: late 2008 or early 2009
- FFADO 2.0 will support:
  - Interfaces based on DM1000 chip: Focusrite Sapphire, Edirol FA-101 & FA-66
  - Some Echo Audiofire devices
  - Some MOTU devices (Traveler, 828Mk2, 896HD)

11 Future plans

- Support more devices
- Encourage more vendor support
- Refine device mixer interfaces (port to Qt4)
- Lower CPU consumption
- Improve device synchronisation stability and recovery

12 Helping FFADO

- Purchase devices from FFADO-friendly vendors and tell them their support of FFADO/Linux is why you're purchasing their interface
- Avoid devices from hostile vendors and tell them why you're not buying their devices
- Download beta releases or subversion snapshots and test them
- Donate/lend devices to FFADO developers

13 Acknowledgements

- The Linux Foundation, for travel assistance to enable this paper to be delivered.
- Fellow primary developers (Daniel Wagner, Pieter Palmers)
- Companies actively supporting FFADO: Echo, ESI, Focusrite, Terratec (now Musonic), Mackie
- My employer (ATRAD), for time off to attend this conference
- Our users, for continued testing and bug reports
- The trademarks of companies referred to thoughout this presentation are acknowledged

14 Links

- FFADO project: http://www.ffado.org
- JACK: http://www.jackaudio.org
- Set_rlimits: http://www.physics.adelaide.edu.au/∼jwoithe/set_rlimits-1.3.0.tgz
- Nosy: http://bitplanet.net/nosy/
- IOI technology: http://www.ioi.com.tw/
- Bushound: http://www.perisoft.net/bushound/

Contacting me:

- jwoithe@physics.adelaide.edu.au