Linux *Kernel Hacking* Free Course 2003

UNIVERSITA' DEGLI STUDI DI ROMA TorVergata Università

D. P. Bovet, M. Cesati

Driver for a PCI sound card, part I

Collecting information about the card (1)

In order to write a device driver for a sound card (or any hardware device), you need a good source of technical information about the peripheral

Usually, the best reference is the technical manual provided by the vendor

If you can't get the technical manual, you're in trouble:

- you might reverse engineer an existing driver, but this is a long, tedious task (also illegal in many countries)
- You might assume that your card is very similar to another card already supported, and try to customize the existing device driver

Collecting information about the card (2)

Let's assume that a Avance Logic's ALS4000 sound card is installed in a PCI bus of our system, and its system identifier is `02:0a.0`

The `/sbin/lspci` command shows the information stored in the configuration registers of the card:

```
# /sbin/lspci -s 02:0a.0 -n
02:0a.0 Class 0401: 4005:4000
# /sbin/lspci -s 02:0a.0 -v
02:0a.0 Multimedia audio controller:
            Avance Logic Inc. ALS4000 Audio Chipset
Subsystem: Avance Logic Inc. ALS4000 Audio Chipset
Flags: bus master, medium devsel, latency 32, IRQ 11
I/O ports at b400 [size=128]
Capabilities: [dc] Power Management version 2
```

Collecting information about the card (3)

We can thus learn that:

- The PCI identifiers of the card are:

```
0401 : 4005 : 4000
```

- The card can raise interrupts, and the assigned IRQ vector is 11
- The card does not have local memory
- The card uses I/O ports mapped to I/O addresses from `0xb400` to `0xb47f` (128 bytes)
- The card supports PCI DMA (*bus mastering*)

ALS4000 technical manual

In the past, the vendor refused to release technical information on the ALS4000 card

A few years ago, however, a technical manual from Avance Logic popped out on the Internet

See the ALSA web site at:

```
http://www.alsa-project.org/
        alsa/ftp/datasheets/avance_logic/
```

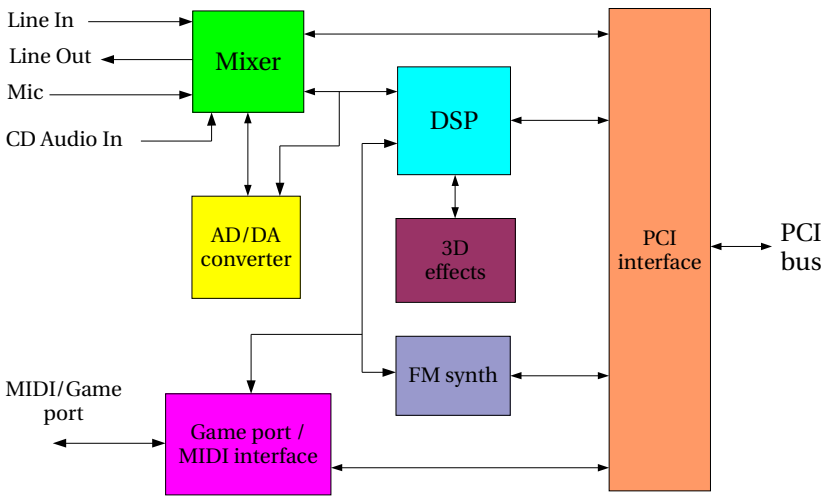
A glimpse at the ALS4000 sound card (1)

The ALS4000 sound card is a cheap device made by Avance Logic (but today Avance Logic's IC business is owned by Realtek)

The core of the peripheral is a 100-pin ALS4000 chip, which includes:

- a Digital Signal Processor (DSP)
- an ADC/DAC converter
- a Mixer
- a FM synthesizer
- a Game port / MIDI interface
- a PCI interface

A glimpse at the ALS4000 sound card (2)



Pulse Code Modulation

A sound card captures and plays sounds by using Pulse Code Modulation (PCM)

PCM is the process of converting an electrical impulse into a streams of numbers, and vice versa

- The ADC measures the voltage of the analog input line at regulars period of time (the sample rate), and produces a stream of discrete values
- The DAC converts a stream of discrete values into electrical impulses

A WAV or AIFF file just holds PCM discrete values

Mixer

The *mixer* is a component that handles the lines carrying analog signals. It allows to:

- adjust the voltage level of the lines
- change the route of the signals inside the device
- filter the signals

Using the mixer, you may adjust the master volume level, trim the balance between left and right stereo channels, select the CD player as input source, fiddle with treeble and bass response, and so on

Digital Signal Processor

The *Digital Signal Processor* is a component that handles the stream of PCM discrete values. It is used to:

- get a PCM stream from the ADC converter; the discrete values are stored in a memory buffer
- send a PCM stream stored in a memory buffer to the DAC converter
- transform a PCM stream applying to it special effects

In the ALS4000 technical manual, the DSP is also called *ESP* (*Enhanced Sound Processor*), since it is coupled with a coprocessor for 3D (spatial) sound effects like chorus and reverb

The FM synthesizer

A synthesizer is a software or hardware component that provides predefined artificial sounds (like the sound of a musical instrument)

FM synthesis controls the timbre of sounds by modulating the frequency of a sine wave by the frequency of another sine wave

The de-facto standard for on-board FM synthesizers is the Yamaha YMF262 (OPL3); it is a digital synthesizer, because it outputs a PCM stream of discrete values

The ALS4000 chip includes an ALS120 FM synthesizer, which is compatible with the Yamaha OPL3

The Game / MIDI interface

The ALS4000 sound card includes a MPU401-compatible game / MIDI interface

It does not handle sounds at all: it is a port for external devices (like the serial and the parallel port), which is used to connect:

- peripherals like joysticks and game pads
- MIDI instruments like musical keyboards, external synthesizers, drum machines, ...

MIDI is a simple protocol designed to pass messages between different devices

The sound card can receive messages from the port, synthesize the corresponding sounds and play them

Device driver duties

A *complete* device driver for the ALS4000 sound card must:

- drive each component of the sound card (DSP, mixer, synthesizer, ...)
- offer suitable, standard interfaces (ALSA, OSS/Free, or OSS/Linux) between each component and User Mode applications
- allow sound capture, playback, synthesis, ...

For lack of time we won't be able to write a *complete* device driver

Good drivers already exist for the ALS4000 sound card

Device driver design choices

- The device driver will support all ALS4000 cards present in the system
  - >> lists instead of arrays
  - >> dynamic data structures instead of static ones
- The device driver will work safely in a multiprocessor system
  - >> semaphores and spin locks
- The device driver will make use of interrupts and DMA
- The device driver will be architecture independent

Writing a PCI driver in 5+3 steps

#1: initialize a table of PCI identifiers

#2: initialize a PCI device driver descriptor

#3: initialize the PCI peripheral and the device driver

#4: release the device driver

#5: load and unload the module

#6: create an interface for the User Mode programs

#7: handle the interrupts

#8: handle the DMA