HARDWARE AND POWER MANAGEMENT

David Snowdon
daves@cse.unsw.edu.au

1. About me
2. How hardware exists
3. Power management

ABOUT ME

→ Computer Engineering 1999–2002
→ Honours: Hardware and Software Infrastructure for Sunswift II
→ PhD: Operating system directed power management
→ Solar cars: WSC (99, 01, 03, 05)

Box

→ purpose: intelligent video surveillance;
→ 624MHz Xscale (PXA270), 64MB RAM, 4MB Flash;
→ analogue video capture, overlays, MPEG 4 encoding, audio;
→ USB, CF, Ethernet;
→ designed for L4/iguana (OK inside).
The imagination driving Australia's ICT future.

**OVERVIEW**

Overview:
- Why talk about hardware design in an OS course?
- Hardware designs are difficult to change

Because:
- Operating systems interact closely with the hardware
- Hardware and system software engineers should work together
- OS developers are great embedded systems developers
- OS people think this stuff is cool

**CASE STUDY: THE iBOX**
- Requirements;
- General design;
- Schematics;
- Layout;
- Manufacture;
- Board bring-up.

**PURPOSE**
- Processing platform for digital video surveillance;
- Teaching platform for AOS.
The imagination driving Australia’s ICT future.

Requirements

Purpose:
- Processing platform for digital video surveillance;
- Teaching platform for AOS.

Video surveillance:
- Motion detection;
- Number recognition;
- Face recognition.

Non-functional:
- Cost
- Modularity
- Size
- Development time
- Connectors: standard
- Un-trusted software.

Functionality:
- Performance: five frames per second at 352 x 288;
- IO: Ethernet, audio, video, alarm, overlay;
- Video compression: MPEG4 at 768 x 576;
- System software update: by an engineer on-site;
- Power supply: up to 24V AC or DC;
- Power consumption: less than 5W.
- Software interfaces: video, audio, network stack, overlay generation, MPEG compression.

Non-functional:
- Cost
- Modularity
- Size
- Development time
- Connectors: standard
- Un-trusted software.

It should cost nothing, use no power, be infinitely small, and take no time to develop. What’s the problem?
The imagination driving Australia’s ICT future.

**Requirements**

Trade-offs:
- Cost vs. Performance
- Size vs. Performance
- Size vs. Utility
- Power vs. Performance
- Time vs. everything
- Cost vs. Quantity
- In-house vs. out-sourcing

Summary:
- MPEG 4 requires serious processing grunt;
- Motion detection requires serious processing grunt;

**Design**

Strategy:
1. Come up with lots of options;
2. Pick one that best fits the requirements;
3. Lather, rinse, repeat;
4. Work out that you’ve made a mistake as early as possible.

We’ll go through a few of the design decisions we made here

**FPGA:**
- Field Programmable Gate Array
- RAM + logic == configurable IC
- “Programmed” using VHDL, Verilog, etc.
- Can use soft cores
FPGA:
- **Field Programmable Gate Array**
- RAM + logic = configurable IC
- “programmed” using VHDL, Verilog, etc.
- can use soft cores

For:
- high performance
- low power

Against:
- difficult to implement algorithms
- some expensive soft-cores
- no L4

DSP:
- Digital signal processor
- VLIW processors optimised for signal processing
- e.g. TI TMS320C6416: 8 GMACs/s @ 1GHz

For:
- very high signal processing performance
- various implementations with video interfaces

Against:
- no MMU
- optimisation required to achieve performance
- expensive compilers
Digital signal processor + Generic processor:
→ use a general-purpose processor for control
→ use a DSP for signal processing algorithms
→ often packaged together (e.g., TI OMAP, DaVinci)

For:
✓ solves lots of problems with a DSP alone
✓ easier to handle real-time issues

Against:
✗ higher software complexity than a single-core solution
✗ need more than one DSP for our application

General-purpose processor:
→ optimised for a wide range of tasks

For:
✓ good support from compilers and operating systems
✓ MMU
✓ system-on-a-chip designs make implementation easier/smaller

Against:
✗ very high required performance == high power, large size
✗ not optimised for signal processing
✗ not capable of MPEG 4 encode
The imagination driving Australia’s ICT future.

**DESIGN**

**SMP:**

- multiple generic cores

For:

- high performance
- retains generic capabilities while providing processing power
- OS transparently handles multiprocessing

Against:

- large size
- high power
- higher parts count

What we considered:

1. PXA270
2. TI DaVinci
3. Sierra MIPS

What next?:

- rough out the designs: identify components
- don’t worry about the easy bits (PSU, connectors)
- obtain development kits and test performance
- look at how well the software fits the design

Calculate:

- cost
- parts count
- part availability
- power consumption

We chose the PXA270, combined with an external MPEG 4 codec ASIC
The imagination driving Australia's ICT future.

**Design**

**PXA270 based solution:**
- 624MHz XScale (ARM) core — fast enough for video algorithms
- quick-capture interface for video input
- tricky circuit for overlay generation
- other useful built-in peripherals (DMA, SRAM, USB, CF, LCD)
- L4 was already ported
- good Linux support
- MMX and DSP instructions
- very low power

**Video out**
- Video In
- Audio I/O
- MPEG Codec
- CPU
- LCD
- SDRAM
- Flash
- Ethernet
- Compact Flash

The imagination driving Australia's ICT future.

**Design**
The imagination driving Australia’s ICT future.

**Schematics**

Schematics:
- Schematics define how components connect logically
- Component pins are connected to form nets
- Hierarchical schematics are possible via ports

**Defining components:**
1. Schematic component
2. Footprint

**Things to think about:**
- Get it right first time
- Decoupling – reduce power supply noise
- Regulator noise
- Pin assignment – shortest paths
- DMA

Hardware should be designed to make software elegant
The imagination driving Australia’s ICT future.

**LAYOUT**

**Layout technology and terminology:**

- design a **printed circuit board (PCB)**
- layers of fibreglass and copper
- the iBox uses an 8 layer board (8 copper layers)
- **vias** are holes used to connect between layers

SEPTEMBER 15, 2006

HARDWARE 27
Component footprints:
- through-hole
- surface-mount
The imagination driving Australia's ICT future.

**Layout**

**Trade-offs:**
- number of layers vs. size
- component size vs. manufacturability
- component size vs. signal integrity

**Routing:**
- the physical pins are connected with traces of copper
- the design software only allows connections defined by the schematic
- design rules are defined to ensure manufacturability
- the artwork is used by a manufacturer to build a PCB

**Signal integrity:** Digital systems are still analogue!
- above 100MHz, edges on digital signals tend to have high-frequency components
- transmission line effects
- inductance and capacitance in the traces/packages
- noise, cross-talk, EMI problems

So:
- keep stub lengths short
- use termination
- think about return paths
- use simulation

**Things to think about:**
- Auto-routing vs. manual routing
- boxes and enclosures
- connector locations
- pin assignment
- size of the traces
- isolating noise
- signal integrity
- simulation
The imagination driving Australia’s ICT future.

LAYOUT

The imagination driving Australia’s ICT future.

LAYOUT

iBox layout:
- 624MHz PXA270 is a uBGA — 0.5mm between balls
- We need very small vias — laser drilling
- We need buried and blind vias

MANUFACTURE

iBox layout:
- 624MHz PXA270 is a uBGA — 0.5mm between balls
- We need very small vias — laser drilling
- We need buried and blind vias

MANUFACTURE

Surface-mount:
- Solder stencil
- Pick and place
- Reflow

Through-hole:
- Wave soldering

LAYOUT
The imagination driving Australia's ICT future.

**JTAG:** Joint Test Action Group – a standard for board testing
- boundary scan
- debugging
- custom commands

**JTAG can be daisy-chained – multiple chips on one port**

**PCB Testing:**
- PCBs are tested using a “bed of nails”
- this doesn’t work with BGA and PGA packages
- JTAG defines boundary scan: the device pins can be controlled
- can be used in combination with “bed of nails”

**iBox JTAG:**
- Boundary scan lets us control all of the pins on the PXA270
- so we can manually control the bus
- so we can program the flash memory
The imagination driving Australia’s ICT future.

**BRING-UP**

**Debug:**
- some processors have extra, debugging JTAG instructions
- Xscale cores can execute instructions from a mini-instruction cache
- breakpoints
- single step
- watch variables

**Trace:**
- a trace port exports which instructions are run
- analysis tools can step backward

---

**SEPTEMBER 15, 2006**

**HARDWARE**

**44**

---

**BRING-UP**

**Bootloader:**

To load software, you need software?
- prepares the system for boot
- handles device-specific tasks
- loads and jumps to the operating system
- commonly saved in read-only memory as firmware
- can be very simple or complex
- some systems use multi-stage loaders

The iBox uses U-Boot, the NSLU2 uses Redboot

---

**SEPTEMBER 15, 2006**

**HARDWARE**

**45**

---

**BRING-UP**

**Start up on the PXA270:**

1. start executing at 0x00000000 (Flash, on iBox)
2. configure the GPIO pins
3. initialise the static memory and SDRAM
4. initialise the IRQs
5. set the core clock frequency
6. enable the caches
7. copy into RAM
8. set up a stack
9. read configuration
10. set up devices and other services

---

**SEPTEMBER 15, 2006**

**HARDWARE**

**46**

---

**BRING-UP**

**Porting an OS to the new platform:**

- configure for PXA270
- configure specific devices (network, CF, USB, etc)

---

**SEPTEMBER 15, 2006**

**HARDWARE**

**47**

---
Bringing up the iBox:
- Power supplies
- Soldering the PXA270
- Testing using JTAG
- Flash
- SDRAM
- Bootloader
- Ethernet
- The rest
Why do we care about power?

- Thermal management
- Electricity consumption
- Energy limited devices

Power-conscious systems can use energy more effectively.
Where does the energy go?:
- switching circuits
- peripherals
- voltage conversion
- batteries

Switching circuits:
- short circuit power
- gate capacitance
- leakage

Gate capacitance:
- to turn on a transistor, we charge the gate beyond its threshold voltage
- to turn it off, we discharge it
- the charge disappears and is the main source of power consumed
- the charge transferred is: \( q = Cv \)
- the energy transferred is: \( E = \frac{1}{2}Cv^2 \)
### Causes of Energy Use

**Hardware Techniques:**
- clock gating

### Dynamic Voltage Scaling

**Dynamic Frequency Scaling:**
- reduce the system frequency to reduce the power
- each switch we lose $E = \frac{1}{2}CV^2$
- $C$ comes from the gate and interconnect capacitance
- frequency is the rate at which we switch
- conclusion: power scales linearly with frequency

**Software Energy Consumption:**
- more instructions = more energy
- different instructions use different energy
- different data use different energy


The energy required to complete a given amount of work is unchanged.
Dynamic voltage scaling:
- If we reduce the frequency, the transistors don’t have to switch as fast.
- Transistor speed is dependent on the driven voltage.
- If we reduce the frequency, we can reduce the voltage.
- Given $E = \frac{1}{2} CV^2$, we get quadratic energy savings.

Voltage/frequency switching overheads:
- Frequency generation is usually via a crystal and PLL.
- Clock multipliers control the PLL.
- Clock dividers generate lower frequencies.
- The PLL has to settle following a change.
- On the PXA255 (XScale) this is $\approx 500 \mu s$.
- Or, always run the PLL at full speed and change dividers.
- On the PXA255 this takes 20 cycles.

Balancing Batteries, Power and Performance:
System Issues in CPU Speed-Setting for Mobile Computing: Thomas Martin, 2001

Peukert:

$$Q = \frac{k}{T_a}$$

- Battery capacity is related to rate of discharge.
- Steady discharge may increase the battery life.
- Can DVS be used to attain this?
The imagination driving Australia’s ICT future.

**Dynamic Voltage Scaling**

**QoS Issues:**
- reducing frequency reduces performance
- missed deadlines
- reduced QoS
- real-time techniques may be employed

**Trouble in Wonderland:**
- ignoring leakage, transistors only use power when they switch
- not every part of the circuit switches on every clock cycle
- different sections run at different frequencies
- off-chip devices aren’t affected

**September 15, 2006**

**Power Management**

64
| September 15, 2006 | Power Management | 66 |

**“Power Management and Dynamic Voltage Scaling: Myths and Facts”: Snowden, et al, 2005.**

- most academic research uses simulation for evaluation
- the naive models lead to incorrect results.

**So:**

- we take real measurements of a real system
- we run an appropriate benchmark suite (MiBench)

**Results:**

- real measurements differ dramatically from models
- optimum processor setting is dependent on workload

**The naive model:**

- $P \propto f^2$

**Limitations:**

- ignores "static" power (core-frequency independent)
- ignores external components
  - affects computation time
  - energy consumption scales differently to the CPU
The imagination driving Australia's ICT future.

**DVS: MYTHS AND FACTS**

**Experimental setup:**
- Single board computer – PLEB 2
- Instrumented power supplies

**Valid configurations:**

<table>
<thead>
<tr>
<th>( V_{core} ) (V)</th>
<th>( f_{cpu} ) (MHz)</th>
<th>( f_{internalbus} ) (MHz)</th>
<th>( f_{mem} ) (MHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>100</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td>1.0</td>
<td>200</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>1.1</td>
<td>300</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>1.3</td>
<td>400</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>1.3</td>
<td>400</td>
<td>200</td>
<td>100</td>
</tr>
</tbody>
</table>

**Execution time for CPU bound:**

![Execution time vs. Frequency for cpubound](image1)

**Execution time for Memory bound:**

![Execution time vs. Frequency for membound](image2)

**Number of cycles for CPU bound:**

![Normalized cycles vs. Turbo Frequency for cpubound](image3)
Number of cycles for Memory bound:

![Diagram showing normalised cycles vs. Turbo Frequency for membound](image)

CPU energy:

![Diagram showing CPU energy for different configurations](image)

Memory energy:

![Diagram showing memory energy for different configurations](image)

Total energy:

![Diagram showing total energy for different configurations](image)

1. maintain per-thread CPU performance counters
2. use cache miss and instruction counters
3. use a lookup table to select a frequency
4. construct the table for >10% performance hit

Demonstrated energy savings of up to 37% with a 10% slowdown for gzip

Conclusions:

→ the memory/cpu frequency ratio affects the number of cycles
→ the most effective DVS policy depends on usage
→ the lowest-energy setting depends on the workload
→ increasing frequency doesn't improve membound performance
→ idling in a low-power mode may be better than running at a low frequency
The imagination driving Australia's ICT future.

Sleep modes:
- Processors and devices have low-power modes
- These modes switch off parts of the processor
- Some state or functionality will be lost
- Lower-power modes tend to lose more functionality
- Lower-power modes take longer to enter and exit

How do we choose when to sleep?
- Processor(s)
- Devices

Case study: PXA255

<table>
<thead>
<tr>
<th>Mode</th>
<th>Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>200MHz Run/Turbo</td>
<td>178mW</td>
</tr>
<tr>
<td>200MHz Idle</td>
<td>63mW</td>
</tr>
<tr>
<td>33MHz Idle</td>
<td>45mW</td>
</tr>
<tr>
<td>Sleep</td>
<td>0.175mW</td>
</tr>
</tbody>
</table>

Case study: Hard disk

<table>
<thead>
<tr>
<th>State</th>
<th>Sleep delay</th>
<th>Wake delay</th>
<th>Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>Idle</td>
<td>0</td>
<td>0</td>
<td>657mW</td>
</tr>
<tr>
<td>Standby</td>
<td>0.85s</td>
<td>1.03s</td>
<td>235mW</td>
</tr>
</tbody>
</table>
Transition management:
- some energy is saved while “asleep”
- some energy is wasted while transitioning
- introduced delay reduces quality-of-service

We need to predict the future

- an “idleness detector” should predict the start time and duration of an idle period

Idle start:
- timeout based: (static, variable, adaptive)
- rate-based: (static, adaptive thresholds)
- periodic: (maintain a DPLL)

Idle duration:
- static (infinite, none, fixed)
- moving average (filtered, unfiltered)
- backoff (geometric increase, arithmetic decrease)
- periodic

OS support:
- Cooperative I/O (Weissel, et al)
- Ghosts in the machine (Anand, et al)
- Power-aware page allocation (Lebeck, et al)
- Timer ticks (IBM Watch project)

Other possible power down areas:
- LCD backlight (user context sensing, selective lighting)
- wireless network
- sound card
**Sleep Modes**

Conclusions:
- keep it simple - DTT with break-even time does OK.
- adaptive techniques can save power
- operating system co-operation improves
- Doug et al present potential reductions of 60% beyond a 5s timeout

**Evaluation Techniques**

Evaluation techniques:
- Simulation
- State-based accounting
- Indirect measurement
- Real measurement

**Adaptation**

Adaptation:
- applications can sometimes adjust their computation best
  - MPEG/JPEG
  - audio bitrate
  - game FPS
  - distributed process migration
  - map viewer detail (Flinn)
- applications can scale their QoS to meet a goal
  - a given battery lifetime
  - a given CPU temperature
- requires operating system feedback
### Evaluation Techniques

**Powerscope (Energy profiling):**
- Designed to be similar to performance profile
- A host computer monitors a computer being profiled
- A computer-controlled multimeters samples the actual energy used
- The profiled computer records PIDs and PC for each sample
- Post-processing attributes energy to procedures/processes

### Energy Accounting

**Energy accounting:**
- ECOSystem (Zeng, et al, 2001)
  - Currentcy is used as a primitive for energy allocation
  - Currentcy is only valid for a certain time
  - Resource containers are used for resource charging
  - Model-based techniques are used to charge RCs.
  - Internal policy is used to select resource containers
- LEA
  - Charges using real power measurements

### Energy Budgeting:
- A budget is allocated to each resource container
- Resource containers in debt are not scheduled
- The budget is periodically refreshed

<table>
<thead>
<tr>
<th>Procedure</th>
<th>Elapsed Time (s)</th>
<th>CPU Power (W)</th>
<th>Memory Power (W)</th>
</tr>
</thead>
<tbody>
<tr>
<td>synth_1t1</td>
<td>21.520</td>
<td>0.676</td>
<td>0.458</td>
</tr>
<tr>
<td>dct3j</td>
<td>3.741</td>
<td>0.657</td>
<td>0.612</td>
</tr>
<tr>
<td>dll_alloc</td>
<td>2.226</td>
<td>0.652</td>
<td>0.488</td>
</tr>
<tr>
<td>dct6j</td>
<td>4.594</td>
<td>0.652</td>
<td>0.441</td>
</tr>
<tr>
<td>do_layer3</td>
<td>0.226</td>
<td>0.639</td>
<td>0.567</td>
</tr>
<tr>
<td>main</td>
<td>0.816</td>
<td>0.653</td>
<td>0.803</td>
</tr>
<tr>
<td>dll_exec</td>
<td>0.069</td>
<td>0.635</td>
<td>0.782</td>
</tr>
<tr>
<td>stream_read_frame</td>
<td>3.642</td>
<td>0.622</td>
<td>0.547</td>
</tr>
<tr>
<td>set_pointer</td>
<td>0.817</td>
<td>0.613</td>
<td>0.649</td>
</tr>
<tr>
<td>III_hybrid</td>
<td>0.523</td>
<td>0.688</td>
<td>0.724</td>
</tr>
<tr>
<td>init_layer1</td>
<td>0.632</td>
<td>0.684</td>
<td>0.419</td>
</tr>
<tr>
<td>quote_flush</td>
<td>0.087</td>
<td>0.298</td>
<td>0.427</td>
</tr>
<tr>
<td>III_get_scale_factors</td>
<td>0.164</td>
<td>0.575</td>
<td>0.783</td>
</tr>
<tr>
<td>read_frame</td>
<td>0.165</td>
<td>0.571</td>
<td>0.821</td>
</tr>
<tr>
<td>III_get_side_info</td>
<td>0.120</td>
<td>0.569</td>
<td>0.753</td>
</tr>
<tr>
<td>stream_read_frame_body</td>
<td>0.011</td>
<td>0.543</td>
<td>1.160</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Procedure</th>
<th>Elapsed Time (s)</th>
<th>CPU Power (W)</th>
<th>Memory Power (W)</th>
</tr>
</thead>
<tbody>
<tr>
<td>decode_header</td>
<td>0.037</td>
<td>0.538</td>
<td>0.742</td>
</tr>
<tr>
<td>play_frame</td>
<td>0.021</td>
<td>0.528</td>
<td>0.574</td>
</tr>
<tr>
<td>wav_write</td>
<td>0.089</td>
<td>0.523</td>
<td>0.475</td>
</tr>
<tr>
<td>stream_read_head_red</td>
<td>0.084</td>
<td>0.521</td>
<td>0.832</td>
</tr>
<tr>
<td>__div13</td>
<td>0.002</td>
<td>0.508</td>
<td>0.000</td>
</tr>
</tbody>
</table>
The imagination driving Australia's ICT future.

ENERGY ACCOUNTING


- use CPU performance counters to estimate CPU temp
- account the energy contribution of each RC
- schedule "hot" processes interspersed with "cold" processes
- can maintain a given temperature limit