Lecture 2 - 225 C

Architecture and System Level Optimization of Power Consumption

Two Kinds of Computation

- Signal Processing (e.g. for multimedia and wireless communications)
  - Stream based computation
  - No advantage in obtaining throughput in excess of the realtime constraint
- General purpose processing (for downloaded code)
  - Bursty - mostly idle with bursts of computation
  - Faster is better

Potential of computation specific energy optimization

- Conventional general purpose processors
  - Clock rate is everything ... somehow we’ll get the power in and out
  - 10-100 watts, 100-1000 Mops = .01Mops/mW
- Energy optimized but general purpose
  - Keep the generality, but reduce the energy as much as possible - e.g. StrongArm
  - .5 Watts, 130 Mops = .3 Mops/mW
- Energy optimized and dedicated
  - 100 Mops/mW

Switching Energy

\[ \text{Energy/transition} = C_L \times V_{dd}^2 \]

\[ \text{Power} = \text{Energy/transition} \times f = C_L \times V_{dd}^2 \times f \]

Power-Delay Product

- Strong function of voltage (V^2 dependence).
- Relatively independent of logic function and style.

Normalized Delay vs. Supply Voltage

Lowering V_{dd} reduces energy but increases delays
Architecture Trade-offs - Reference Datapath

- Critical path delay \( \Rightarrow T_{adder} + T_{comparator} \) (= 25ns)
  \( \Rightarrow f_{ref} = 40\text{MHz} 

- Total capacitance being switched = \( C_{ref} \)

- Power for reference datapath = \( P_{ref} = C_{ref}V_{ref}^2f_{ref} \)

天花(Chandrakasan92) (IEEE SSC) 7

Architecture and System Level Optimization of Power Consumption

Parallel Datapath

- The clock rate can be reduced by half with the same throughput \( \Rightarrow f_{par} = f_{ref}/2 \)

- \( V_{par} = V_{ref}/1.7, C_{par} = 2.15C_{ref} \)

- \( P_{par} = (2.15C_{ref})(V_{ref}/1.7)^2(f_{ref}/2) = 0.36P_{ref} \)

天花(Chandrakasan95) (IEEE TCAD) 8

Architecture and System Level Optimization of Power Consumption

The More Parallel the Better??

Capacitance overhead starts to dominate at “high” levels of parallelism and results in an optimum voltage

天花(Chandrakasan92) (IEEE SSC) 9

Architecture and System Level Optimization of Power Consumption

Pipelined Datapath

- Critical path delay is less \( \Rightarrow \max [T_{adder}, T_{comparator}] \)

- Keeping clock rate constant: \( f_{pipe} = f_{ref} \)

- Voltage can be dropped \( \Rightarrow V_{pipe} = V_{ref}/1.7 \)

- Capacitance slightly higher: \( C_{pipe} = 1.15C_{ref} \)

- \( P_{pipe} = (1.15C_{ref})(V_{ref}/1.7)^2f_{ref} = 0.39P_{ref} \)

天花(Chandrakasan95) (IEEE TCAD) 10

Architecture and System Level Optimization of Power Consumption

Architecture Summary for a Simple Datapath

<table>
<thead>
<tr>
<th>Architecture type</th>
<th>Voltage</th>
<th>Area</th>
<th>Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple datapath (no pipelining or parallelism)</td>
<td>5V</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Pipelined datapath</td>
<td>2.9V</td>
<td>1.3</td>
<td>0.39</td>
</tr>
<tr>
<td>Parallel datapath</td>
<td>2.9V</td>
<td>3.4</td>
<td>0.36</td>
</tr>
<tr>
<td>Pipeline-Parallel</td>
<td>2.0V</td>
<td>3.7</td>
<td>0.2</td>
</tr>
</tbody>
</table>

天花(Chandrakasan95) (IEEE TCAD) 11

Architecture and System Level Optimization of Power Consumption

Algorithmic Transformations

Loop-unrolling does not reduce power consumption

天花(Chandrakasan95) (IEEE TCAD) 12

Architecture and System Level Optimization of Power Consumption
Loop Unrolling Enables Other Transformations

After Algebraic Transformations, & Constant Propagation
$C_{\text{eff}} = 3$
Voltage = 2.9
Throughput = 2
Power = 12.5 (x2 reduction)

Pipelining

Speed vs. Power Optimization

Area can be traded for higher throughput or lower power

Multiple Supply Voltage Systems: Filter Example

Power (5V) / Power (5V,3V, 2.4V) ≈ 1.5

from [Raje95]
Similar approach to logic design proposed in [Usami95]

Optimizing Multiplications

A = IN * 0 0 1 1
B = IN * 0 1 1 1
A = (IN >>4 + IN >>3)
B = (IN >>4 + IN >>3 + IN >>2)
A = (IN >>4 + IN >>3)
B = (A + IN >>2)

Only Scaling
Scaling & Common Sub-expression

Time-multiplexed Architectures

Parallel busses for I,Q

Time-shared bus for I,Q

Can destroy signal correlations and increase the switching activity

Number Representation

Two’s Complement
Sign Magnitude

● Sign-extension activity significantly reduced using sign-magnitude representation
Two’s Complement vs. Sign-Magnitude

- Two’s complement datapath has a significantly higher glitching activity

Reducing Activity by Reordering Inputs

- 30% reduction in switching energy

Resource Sharing Can Increase Activity

Memory Architecture

- Serial Access
- Parallel Access

General Purpose computing - Do we just optimize power?

- NO!

What is important?

Operations per Battery Life:
Minimize Energy Consumed per Operation

Operations per Second:
Maximize Throughput = Operations/second

The complete subsystem should be optimized

Power dissipation is distributed
Proposed Design Methodology - (Tom Burd, Anthony Stratakos and Trevor Pering)

- Instruction Set Architecture
- Energy efficient system organization
- Dynamically adjust throughput to user's needs
- Apply energy efficient circuit and architecture design techniques

Energy Efficient Processor System

Demonstration Vehicle

Redesign the InfoPad processor subsystem

- 45 mW Clock Oscillator
- 120 mW ARM60
- 400 mW PLD
- 45 mW Processor Bus
- 40 mW I/O Interface
- 600 mW SRAM 128k x 8

Current System: 10 MIPS @ 1.2W

Processor Usage Model

- Desired Throughput
- Compute-intensive and low-latency computation
- Ceiling: Set by top speed of the processor
- Not always computing
- Background and high-latency computation

Simplest Approach: Compute ASAP

- Delivered Throughput
- 80 MIPS
- Excess throughput
- Wake up → Compute ASAP → Go to idle/sleep mode
- Always high throughput
- Always high energy

Another Approach: Reduce Clock Frequency

- Delivered Throughput
- 80 MIPS
- Frequency set by user
- Reduced
- PowerBook Control Panel

- Energy remains unchanged... while throughput & power scale down with $f_{CLK}$
- Reducing power dissipation not always equivalent to reducing energy consumption

Clock rate reduction doesn’t help energy consumption

- Energy is independent of clock rate
  - Number of operations = $N_{ops}$
  - Energy/operation = $CV^2$
  - Total energy = $CV^2 \times N_{ops}$

- Reducing the clock rate only degrades throughput, but no savings in battery life - unless the voltage is changed
Dynamic Voltage Scaling

**Delivered Throughput**

Reduce throughput & $f_{CLK}$
Reduce energy/operation

Dynamically scale energy with clock rate

Extend battery life by up to 10x with the same hardware

Key: Process scheduler determines operating point.

**Scale Energy with Throughput, $f_{CLK}$**

Constant supply voltage.

~10x Energy Reduction

Reduced supply voltage, circuit speed tracks $f_{CLK}$.

Normalized data (simulated, 0.6um process)

---

**Minimal Hardware Implementation**

Modify existing DC-DC converter [Stratakos] feedback loop

- 10 m/sec per frequency transition
- Clock tracks over process and temp.

Add Register to ISA

**DVS in Practice**

**Fixed Throughput, Energy/operation**

Throughput = 10 MIPS
Energy/op. = 1 nJ/inst.

Throughput = 80 MIPS
Energy/op. = 9 nJ/inst.

(Peak throughput 11% of the time... average energy/op = 2 nJ/inst)

---

**Main Memory: IC Design**

Use existing low-power memory block [Burstein]

3.2 mm$^2$, 0.6 um

4 kByte Block
Access time = 22 ns
Energy/access = 120 pJ

Design 64 kByte IC:
Access time ~ 40 ns
Energy/access ~ 300 pJ

5-10x better than commercial

Key: SRAM must be DVS Compatible.

**Main Memory: Architecture**

Standard memory architecture design

Proposed memory architecture design

Only activate one SRAM → power reduced by 4x

Micro-power bus driver makes extra load negligible power
Self-timed Approach for Eliminating Glitching

- Enable tri-state drivers after sense-amp outputs are valid to eliminate glitching on the data-bus.

Achievable energy levels

10 MIPS, 1 nJ/inst. ⇔ 80 MIPS, 9 nJ/inst. (10 mW)

DC-DC Converter

LP-ARM CPU

Processor Bus

DC-DC

CPU

100 pJ

500 pJ

I/O Interface

0.5 MB SRAM (8 ICs)

100 pJ

300 pJ

Improves energy efficiency by an order of magnitude

Critical circuit - High efficiency DC-DC conversion using a switching regulator

- Arbitrary $V_{dd} (< V_{in})$ generated using the Buck converter

$V_{dd} = V_{in}^2$ Duty Cycle at Node X

- Chief sources of inefficiencies:
  - Conduction loss ($I^2R$)
  - Switching loss ($C_x V_{in}^2 f_s$ and $L_s I^2 f_s$)
  - Gate-drive loss ($C_g V_{in}^2 f_s$)

from [Stratakos94] (IEEE PESC)

Soft-Switching Eliminates $C_x V^2 f$ Loss

- Dead-time when neither FET conducts
- Current reverses
- $L_f$ charges and discharges $C_x$

FETS ARE SWITCHED WITH $V_{DS} = 0$

What happens when $I_{out}$ changes?

- $I_{out} \downarrow \Rightarrow C_x$ discharges slowly
- $I_{out} \uparrow \Rightarrow C_x$ discharges quickly

Rectifier Discharges $C_x$

Body Diode Conduction

- Inverter node transition times depends on $I_{out}$
- Typical schemes use fixed dead time set by gate delays

Adaptive Dead-time Control Needed for varying $I_{out}$
Switcher Design: Power Transistor Sizing

\[ P_{\text{total}} = P_{gd} + P_{cl} \]

Minimize \( P_{\text{total}} = P_{\text{gate-drive}} + P_{\text{conduction loss}} \)

\[ W_{\text{opt}} = \frac{b}{a f_s} \]

Low Voltage Support Circuitry: Level Converter

- Compatibility with 3.3V/5V standard components
- \((V_{OH})_{\text{IN}} = 1.1\text{V to } 5\text{V and } (V_{OH})_{\text{OUT}} = 1.1\text{ to } 5\text{V}\)

Other uses of adaptive DC-DC converters

- Adaptive supplies
- Self-timed circuits
- Adaption to varying algorithmic workloads

Adaptive Power Supply Voltages

Exploit Data Dependent Computation Times To Vary the Supply
from [Nielsen94] 
(IEEE Transactions on VLSI Systems)

But Self-timed Circuits are Expensive...

Guaranteed transition for every operation

\[ \alpha_{0 \rightarrow 1} = 1 \]

Use Synchronous DSP instead

Critical path based voltage optimization

- Feedback adjusts the regulated voltage to the point where the equivalent critical path is about to fail

Exploit Data Dependent Computation Times To Vary the Supply
Case Study: A Portable Multimedia I/O Terminal

Protocol, ECC, Buffering, Video Decompression, and I/O (InfoPad Terminal Developed at U.C. Berkeley) from [Chandrakasan94]

Chipset Summary (1.2-\(\mu\)m, \(V_t = 0.7-0.9V\))

<table>
<thead>
<tr>
<th>Chip Description</th>
<th>Area (mm(\times)mm)</th>
<th>Minimum Supply Voltage</th>
<th>Power at 1.5V</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protocol</td>
<td>9.4 x 9.1</td>
<td>1.1V</td>
<td>1.9mW</td>
</tr>
<tr>
<td>Frame-buffer SRAM</td>
<td>7.8 x 6.5</td>
<td>1.1V</td>
<td>1mW</td>
</tr>
<tr>
<td>Video Controller</td>
<td>6.7 x 6.4</td>
<td>1.1V</td>
<td>150(\mu)W</td>
</tr>
<tr>
<td>Luminance Decompression</td>
<td>8.5 x 6.7</td>
<td>1.1V</td>
<td>115(\mu)W</td>
</tr>
<tr>
<td>Chrominance Decompression</td>
<td>8.5 x 9.0</td>
<td>1.1V</td>
<td>100(\mu)W</td>
</tr>
<tr>
<td>Color Space Conversion and Triple DAC</td>
<td>4.1 x 4.7</td>
<td>1.3V</td>
<td>1.1mW</td>
</tr>
</tbody>
</table>

Video Decompression Module

- Luminance Decompression
  - Ping-pong frame-buffer
  - Lookup Table
- Chrominance Decompression
  - Ping-pong frame-buffer
  - Lookup Table

Digital YIQ -> Digital RGB

\[
\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} D_{11} & D_{12} & D_{13} \\ D_{21} & D_{22} & D_{23} \\ D_{31} & D_{32} & D_{33} \end{bmatrix} \begin{bmatrix} Y \\ I \\ Q \end{bmatrix}
\]

- Optimized matrix multiplication (6mults -> 8 adds)
  - Hardwired shift-add operations
  - Coefficient scaling to minimize shift-add operations
  - Exploit multiple coefficients multiplied with the same input
- 100 \(\mu\)Watts compared to commercial 1 Watt - Why??

Color Space Translator and Triple DAC

Key Features:
- Digital YIQ -> Analog RGB
- Optimized Multiplications
- Number Representation
- Optimized Time-sharing
- Integrated low-voltage DAC’s

Power @ 1.3V: 0.93mW

Clock Rate: 2.5MHz

Size: 4.1mm x 4.7mm

1.2\(\mu\)m technology

Power reduction approaches which make up the factor of 10,000 improvement

<table>
<thead>
<tr>
<th>Design Consideration</th>
<th>Approach</th>
<th>Power Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>14MHz-&gt;2.5MHz</td>
<td>5.6</td>
</tr>
<tr>
<td>Supply Voltage</td>
<td>5V-1.5V</td>
<td>11</td>
</tr>
<tr>
<td>Library Optimization</td>
<td>Minimum Sized Devices Single Phase Clocking</td>
<td>2-3</td>
</tr>
<tr>
<td>Matrix Multiplication</td>
<td>Hardwired Shift-add Coefficient Optimization</td>
<td>7</td>
</tr>
<tr>
<td>Resource Allocation</td>
<td>Fully Parallel Implementation</td>
<td>1.5-2</td>
</tr>
<tr>
<td>Number Representation</td>
<td>Sign-Magnitude</td>
<td>1.2</td>
</tr>
<tr>
<td>Off Chip Drivers</td>
<td>Integrate Processing and DAC</td>
<td>1.4</td>
</tr>
<tr>
<td>Bitwidth</td>
<td>8bits-&gt;16bits</td>
<td>1.3</td>
</tr>
</tbody>
</table>
## Summary

- Signal statistics can be exploited to minimize the number of transitions required to perform a given function.
- Architectural voltage scaling is a key technique for low-voltage operation.
- Variable power supply reduces power and buffering trades latency for power.
- **Orders of magnitude of power reduction are possible.**