EE241 - Spring 2005
Advanced Digital Integrated Circuits

Lecture 11: Voltage Optimization

Power/Energy Optimization Space

<table>
<thead>
<tr>
<th></th>
<th>Constant Throughput/Latency</th>
<th>Variable Throughput/Latency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Energy</td>
<td>Design Time</td>
<td>Sleep Mode</td>
</tr>
<tr>
<td>Leakage</td>
<td>Stack effects + Multi-(V_T)</td>
<td>Sleep T's Multi-(V_{DD}) Variable (V_T) + Input control</td>
</tr>
</tbody>
</table>

Run Time

Variable Throughput/Latency

Constant Throughput/Latency

Energy

Sleep Mode
Multiple Supply Voltages

- Block-level supply assignment
  - Higher throughput/lower latency functions are implemented in higher $V_{DD}$
  - Slower functions are implemented with lower $V_{DD}$
  - Called “Voltage islands”
  - Separate supply grids, level conversion performed at block boundaries
- Multiple supplies inside a block
  - Non-critical paths moved to lower supply voltage
  - Level conversion within the block
  - Physical design challenging

Multiple Supplies in a Block

Conventional Design

- Critical Path

CVS Structure

- Critical Path
- Lower $V_{DD}$ portion is shaded

“Clustered voltage scaling”

M. Takahashi, ISSCC’98.
Pulsed LCFF

- M/S and pulsed half-latch LCFFs (MSHL, PHL)
  - Smaller # of MOSFETs / clock loading
  - Faster level conversion using half-latch structure
  - Shorter D-Q path from pulsed circuit

![Pulsed LCFF Diagram](image)

Ishihara, ISLPED'03

---

Pulsed LCFF

- Pulsed precharge LCFF (PPR)
  - Fast level conversion by precharge structure
  - Suppressed charge/discharge toggle by conditional capture
  - Short D-Q path

![Pulsed Precharge LCFF Diagram](image)

Ishihara, ISLPED'03
Multiple Supply Voltages

- Two supply voltages per block are optimal
- Optimal ratio between the supply voltages is 0.7
- Level conversion is performed on the voltage boundary, using a level-converting flip-flop (LCFF)
- An option is to use an asynchronous level converter
  - More sensitive to coupling and supply noise

Three $V_{DD}$’s

From Kuroda

$V_1 = 1.5V$, $V_{TH} = 0.3V$, $\rho(t):\lambda$
The more \( V_{DD} \)'s, the less power, but the effect saturates.
Power reduction effect will be decreased as \( V_{DD} \)'s are scaled.
Optimum \( V_2/V_1 \) is around 0.7.

Optimum Numbers of Supplies

ALU Block Diagram
Low Swing Bus & Level Converter

- Delay of INV1 does not increase
- INV2 is placed near 9:1 MUX to increase noise immunity
- Level conversion is done by a domino 9:1 MUX

Measured Results: Energy & Delay

Room temp.

- Energy: -25.3% (VDDL=1.4V) 
- Delay: +2.8% (VDDL=1.4V)
- Energy: -33.3% (VDDL=1.2V) 
- Delay: +8.3% (VDDL=1.2V)

The dual-supply technique expands the power-delay optimization space
Distributing Multiple Supply Voltages

Conventional

- **V\textsubscript{DDH} circuit**
- **V\textsubscript{DDL} circuit**

Shared N-well

- **V\textsubscript{DDH} circuit**
- **V\textsubscript{DDL} circuit**

Conventional

- **N-well isolation**

(a) Dedicated row

(b) Dedicated region
Reducing the Supply Voltage: Concurrency versus Clock Speed

Example: reference datapath

- Critical path delay: \( T_{\text{adder}} + T_{\text{comparator}} = 25\text{ns} \)
  \( \Rightarrow f_{\text{ref}} = 40\text{Mhz} \)
- Total capacitance being switched = \( C_{\text{ref}} \)
- \( V_{\text{dd}} = V_{\text{ref}} = 5\text{V} \)
- Power for reference datapath = \( P_{\text{ref}} = C_{\text{ref}} \cdot V_{\text{ref}}^2 \cdot f_{\text{ref}} \)

(from Chandrakasan92) (IEEE JSSC)
Parallel Data Path

- The clock rate can be reduced by half with the same throughput ⇒ \( f_{\text{par}} = \frac{f_{\text{ref}}}{2} \)
- \( V_{\text{par}} = \frac{V_{\text{ref}}}{1.7} \), \( C_{\text{par}} = 2.15C_{\text{ref}} \)
- \( P_{\text{par}} = (2.15C_{\text{ref}})(\frac{V_{\text{ref}}}{1.7})^2 \left( \frac{f_{\text{ref}}}{2} \right) \approx 0.36 P_{\text{ref}} \)

Pipelined Data Path

- Critical path delay is less ⇒ max [\( T_{\text{adder}} \), \( T_{\text{comparator}} \)]
- Keeping clock rate constant: \( f_{\text{pipe}} = f_{\text{ref}} \)
  Voltage can be dropped ⇒ \( V_{\text{pipe}} = \frac{V_{\text{ref}}}{1.7} \)
- Capacitance slightly higher: \( C_{\text{pipe}} = 1.15C_{\text{ref}} \)
- \( P_{\text{pipe}} = (1.15C_{\text{ref}})(\frac{V_{\text{ref}}}{1.7})^2 f_{\text{ref}} \approx 0.39 P_{\text{ref}} \)
## A Simple Data Path: Summary

<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>

## Architecture Choices

- **Flexibility**
  - 100-1000 MOPS/mW
  - Direct Mapped Hardware

- **Energy**
  - 0.5-5 MIPS/mW
  - Embedded FPGA
  - Reconfigurable Processors (Maia)
  - Embedded Processor (lpArm)

Factor of 100-1000

Brodersen & Rabaey
Two Types of Processing

- Fixed-rate processing (e.g. signal processing for multimedia or communications)
  - Stream-based computation
  - No advantage in obtaining throughput in excess of the real-time constraint

- Variable-rate or burst-mode computation (e.g. general purpose computation)
  - mostly idle (or low-load) with bursts of computation
  - Faster is better

variable-rate processing
Voltage as a design variable

Adapting voltage to workload yields cubic reduction!
Common Design Approach: Fixed $V_{DD}$

- **Compute ASAP:**
  - Always high throughput
  - Excess throughput

- **Clock Frequency Reduction:**
  - Energy/operation remains unchanged...
  - while throughput scaled down with $f_{CLK}$

Dynamic Voltage Scaling (DVS)

1. Vary $f_{CLK}, V_{DD}$
2. Dynamically adapt

- Dynamically scale energy/operation with throughput.
- Always minimize speed $\rightarrow$ minimize average energy/operation.
- Extend battery life up to 10x with the exact same hardware!
Adaptive Supply Voltages

Exploit Data Dependent Computation Times To Vary the Supply

from [Nielsen94]
(IEEE Transactions on VLSI Systems)

Variable Algorithmic Workload

Exploit Time Varying Computation Requirements to Vary the Power Supply Voltage
Typical MPEG IDCT Histogram

Required speed $\propto f$

Normalized power $P \propto fV^2$

Dynamic Power Reduction Through Software-Hardware Cooperation

If you don’t need to hustle, relax and save power.

S. Lee et al, DAC, June 2000
Feedback loop sets $V_{DD}$ so that $F_{ERR} \rightarrow 0$.

Ring oscillator delay-matched to CPU critical paths.

Custom loop implementation can optimize $C_{DD}$. 

Dynamic operation can increase energy efficiency > 10x.
Zero is implemented heuristic algorithm. Difficult to optimize compute-intensive code (MPEG). Big drop in energy when less speed required (3.3-4.5x).

### Recent DVS-Enabled Microprocessors

- **Xscale:** 180nm 1.8V bulk-CMOS [Intel00]
  - 0.7-1.75V, 200-1000MHz, 55-1500mW (typ)
  - Max. Energy Efficiency: ~23 MIPS/mW

- **PowerPC:** 180nm 1.8V bulk-CMOS [Nowka02]
  - 0.9-1.95V, 11-380MHz, 53-500mW (typ)
  - Max. Energy Efficiency: ~11 MIPS/mW

- **Crusoe:** 130nm 1.5V bulk-CMOS [Transmeta03]
  - 0.8-1.3V, 300-1000MHz, 0.85-7.5W (peak)

- **Pentium M:** 130nm 1.5V bulk-CMOS [Intel03]
  - 0.95-1.5V, 600-1600MHz, 4.2-31W (peak)
V_{DD}-Hopping

Application slicing and software feedback guarantee real-time operation.

Two hopping levels are sufficient.

Challenge: Design over Wide Range of Voltages

- Circuit design constraints. (Functional verification)
- Circuit delay variation. (Timing verification)
- Noise margin reduction. (Power grid, coupling)
- Delay sensitivity. (Local power distribution)

Design verification complexity similar to high-performance processor design @ fixed V_{DD}
Relative Delay Variation

Four extreme cases of critical paths:

- Timing verification only needed at min. & max. $V_{DD}$.
- Should also consider $V_{dd}$ variations.

 Delay Sensitivity

$$\frac{\partial \text{Delay}}{\partial \text{Delay}} \approx \frac{\partial \text{Delay}}{\partial V_{DD}} \cdot \frac{\Delta V_{DD}}{\text{Delay}(V_{DD})}, \quad \Delta V_{DD} = I(V_{DD}) \cdot R$$

Design of local power grid (for timing constraints) only need to consider $V_{DD} = 2V_T$. 

Burd ISSCC’00
Design for Dynamically Varying VDD

- Static CMOS logic.
- Ring oscillator.
- Dynamic logic (& tri-state busses).
- Sense amp (& memory cell).

Max. allowed $|dV_{DD}/dt| \rightarrow$ Min. $C_{DD} = 100\text{nF}$ (0.6$\mu$m)
Circuits continue to properly operate as $V_{DD}$ changes

Static CMOS Logic

$V_{in} = 0 \quad V_{out} = V_{DD}$

$r_{ds}^{PMOS}$

$C_L$

max. $\tau = 4\text{ns}$

0.6$\mu$m CMOS: $|dV_{DD}/dt| < 200\text{V}/\mu\text{s}$

- Static CMOS robustly operates with varying $V_{DD}$.
**Ring Oscillator**

*Simulated with \( \frac{dV_{DD}}{dt} = 20\, \text{V/\mu s} \)*

- Output \( f_{CLK} \) instantaneously adapts to new \( V_{DD} \).

---

**Dynamic Logic**

- False logic low: \( \Delta V_{DD} > V_{TP} \)
- Latch-up: \( \Delta V_{DD} > V_{be} \)

0.6\mu m CMOS: \( |dV_{DD}/dt| < 20\, \text{V/\mu s} \)

- Cannot gate clock in evaluation state.
- Tri-state busses fail similarly → Use hold circuit.
System-Level Issues: Reducing Waste

Avoid switching any capacitance unneededly
Sharing increases capacitance

- Application-specific processing
- Preservation of data correlations
- Locality of reference
- Distributed processing
- Demand-driven / Data-driven computation

Clock gating

Requires careful skew control ...
Fortunately well handled in today's EDA tools
Clock-gating efficiently reduces power, NOW

Without clock gating

With clock gating

90% of F/F's were clock-gated.

70% power reduction by clock-gating alone.

Courtesy M. Ohashi, Matsushita, ISSCC 2002, Paper #22.1

Pre-computation

Inputs $x_1$, ..., $x_n$ are not applied if pre-computing holds

Other options:
- guarded evaluation
- set output directly

Courtesy M. Ohashi, Matsushita, ISSCC 2002, Paper #22.1
Circuit-level Activity Reduction

from [Alidina94]
(1994 International Workshop on Low-power Design)

Circuit-Level Activity Encoding

from [Stan94]
(1994 International Workshop on Low-power Design)
Eliminating Redundant Computations

- Dynamically vary the number of operations per sample.
- Trade power consumption and filter quality

from [Ludwig95]
(CICC95)

Eliminating Redundant Computations

Switched Capacitance Reduction = \frac{\text{Peak Number of Operations}}{\text{Average Number of Operations}}
= 2
Strong Function of Signal Statistics
Number Representation

Two’s Complement

- Sign-extension activity significantly reduced using sign-magnitude representation

Number Representation - Accumulator Example

Two’s Complement

- Sign magnitude datapath switches 30% less capacitance for uniformly distributed inputs
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

Number of Bus Transitions Per Cycle

\[ N = 2 \left( \frac{1}{1} + \frac{1}{2} + \frac{1}{4} + \cdots + \frac{1}{128} \right) = 4 \]

Application-Specific Processing Reduces “Implementation Overhead”
### The Architectural Trade-off

<table>
<thead>
<tr>
<th></th>
<th>Energy</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>64-point FFT</td>
<td>16-State Viterbi</td>
</tr>
<tr>
<td></td>
<td>Energy per</td>
<td>Decoder Energy per</td>
</tr>
<tr>
<td></td>
<td>Transform (nJ)</td>
<td>Decoded bit (nJ)</td>
</tr>
<tr>
<td>Direct-Mapped Hardware</td>
<td>1.78</td>
<td>0.022</td>
</tr>
<tr>
<td>FPGA</td>
<td>683</td>
<td>5.5</td>
</tr>
<tr>
<td>Low-Power DSP</td>
<td>436</td>
<td>19.6</td>
</tr>
<tr>
<td>High-Performance DSP</td>
<td>1700</td>
<td>108</td>
</tr>
<tr>
<td></td>
<td>2,200</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>200,000</td>
<td>150</td>
</tr>
</tbody>
</table>

*Numbers taken from vendor-published benchmarks*

Orders of magnitude lower efficiency even for an optimized processor architecture

### Towards Heterogeneous Architectures for SOC

- **Janus Chip - ST Micro and Parades**
- **Xilinx Vertex Pro**
- **Berkeley Pleiades**
• Voltage as a Design Variable
  Match voltage and frequency to required performance
• Minimize waste (or reduce switching capacitance)
  Match computation and architecture
  Preserve locality inherent in algorithm
  Exploit signal statistics
  Energy (performance) on demand

More easily accomplished in application-specific than programmable devices