Outline

- Power-performance tradeoffs
  - Sensitivities to sizing, supplies, threshold
  - Supply voltage optimization
Power-Performance Tradeoffs

Sensitivity to Sizing and Supply

- Gate sizing ($W_i$)
  \[
  \frac{\partial E_{sw}}{\partial W_i} = \frac{e_j}{\tau_{nom} (f_j - f_{j-1})}
  \]
  \[
  \frac{\partial D}{\partial W_i} = \frac{e_j}{\tau_{nom} (f_j - f_{j-1})}
  \]

- Supply voltage ($V_{dd}$)
  \[
  \frac{\partial E_{sw}}{\partial V_{dd}} = \frac{E_{sw}}{D} \frac{1 - x_v}{\alpha - 1 + x_v}
  \]
  \[
  \frac{\partial D}{\partial V_{dd}} = \frac{E_{sw}}{D} \frac{1 - x_v}{\alpha - 1 + x_v}
  \]

\[x_v = (V_{th} + \Delta V_{th})/V_{dd}\]
Sensitivity to $V_{th}$

- Threshold voltage ($V_{th}$)

\[ \frac{\partial E}{\partial \Delta V_{th}} = P_{lk} \left( \frac{V_{DD} - V_{th} - \Delta V_{th}}{\alpha n V_i} - 1 \right) \]

Low initial leakage

$\Rightarrow$ speedup comes for “free”
Reducing active power

- Downsizing transistors ($C_L$)
  - Slows down logic
- Lowering the supply voltage ($V_{DD}$)
  - Slows down logic
  - Reducing swing slows down the succeeding stage
- Reducing frequency ($f$)
  - Does not reduce energy
- Reducing switching activity ($\alpha$)
  - Logic restructuring, clock gating
- Reducing glitching
  - Balancing logic

$$P_{dyn} \sim \alpha \cdot C_L \cdot V_{swing} \cdot V_{DD} \cdot f$$
$$E \sim \alpha \cdot C_L \cdot V_{swing} \cdot V_{DD}$$

---

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>Active</td>
<td>Logic design</td>
<td>Clock gating</td>
</tr>
<tr>
<td></td>
<td>Scaled $V_{DD}$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Trans. sizing</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Multi-$V_{DD}$</td>
<td></td>
</tr>
<tr>
<td>Leakage</td>
<td>Stack effects</td>
<td>Sleep $T'$s</td>
</tr>
<tr>
<td></td>
<td>Trans sizing</td>
<td>Multi-$V_{DD}$ Variable $V_{Th}$</td>
</tr>
<tr>
<td></td>
<td>Scaling $V_{DD}$</td>
<td>+ Input control</td>
</tr>
<tr>
<td></td>
<td>+ Multi-$V_{Th}$</td>
<td></td>
</tr>
</tbody>
</table>
## Energy-Performance Tradeoffs

<table>
<thead>
<tr>
<th>Enable Time/Perf. Impact</th>
<th>Design Time</th>
<th>Run Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Near-zero perf. penalty</td>
<td>Clock gating, Architectural switching reduction, Multi-V&lt;sub&gt;Th&lt;/sub&gt;</td>
<td>Dynamic V&lt;sub&gt;DD&lt;/sub&gt;, Dynamic V&lt;sub&gt;Th&lt;/sub&gt;</td>
</tr>
<tr>
<td>True tradeoffs</td>
<td>Fine-granularity clock gating, V&lt;sub&gt;DD&lt;/sub&gt;, V&lt;sub&gt;Th&lt;/sub&gt; adjustments, Multi-V&lt;sub&gt;DD&lt;/sub&gt;, Sizing, logic styles, Stack forcing</td>
<td>Sleep T's</td>
</tr>
</tbody>
</table>

## Scaling Trends

Trends (from IEDM) in $I_{DSAT}$, $V_{DD}$, $V_{Th}$
## Power /Energy Optimization Space

<table>
<thead>
<tr>
<th>Energy</th>
<th>Constant Throughput/Latency</th>
<th>Variable Throughput/Latency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active</td>
<td>Design Time</td>
<td>Sleep Mode</td>
</tr>
<tr>
<td>Logic design</td>
<td>Scaled $V_{DD}$</td>
<td>Clock gating</td>
</tr>
<tr>
<td>Trans. sizing</td>
<td>Multi-$V_{DD}$</td>
<td></td>
</tr>
<tr>
<td>Leakage</td>
<td>Stack effects</td>
<td>Sleep T's</td>
</tr>
<tr>
<td>Trans sizing</td>
<td>Multi-$V_{DD}$ Variable $V_{Th}$ + Input control</td>
<td></td>
</tr>
<tr>
<td>Scaling $V_{DD}$</td>
<td>+ Multi-$V_{Th}$</td>
<td></td>
</tr>
</tbody>
</table>

## Supply Voltage Adjustment

- **How to maintain throughput under reduced supply?**
  - Introducing more parallelism/pipelining
    - Area increase – cost up
    - Cost/power tradeoff
  - **Multiple voltage domains**
    - Separate supply voltages for different blocks
    - Lower VDD for slower blocks
    - Cost of DC-DC converters
  - **Dynamic voltage scaling – with variable throughput**
  - **Reducing $V_{Th}$ to improve speed**
    - Leakage issues
Reducing $V_{dd}$

- Strong function of voltage ($V^2$ dependence).
- Relatively independent of logic function and style.
- Power Delay Product Improves with lowering $V_{DD}$.

Chandrakasan, JSSC'92

Reducing $V_{DD}$

Chandrakasan, JSSC'92
Lower $V_{DD}$ Increases Delay

\[ T_d = \frac{C_l \cdot V_{dd}}{I} \]

\[ I \sim (V_{dd} - V_t)^2 \]

\[ T_d(V_{dd}=5) = \frac{(2) \cdot (5 - 0.7)^2}{(5) \cdot (2 - 0.7)^2} \approx 4 \]

- Relatively independent of logic function and style.

Trade-off Between Power and Delay

\[ \text{Power} = a \cdot f \cdot C \cdot V_{DD}^2 + I_0 \cdot 10^{\frac{V_{TH}}{s}} \cdot V_{DD} \]

\[ \text{Delay} \propto \frac{C \cdot V_{DD}}{(V_{DD} - V_{TH})^{1.3}} \]

Equi-delay

50nm node, FO3 INV
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

Architecture Trade-off for Fixed-rate Processing

Reference Datapath

- Critical path delay $\Rightarrow T_{adder} + T_{comparator} (= 25\text{ns})$
  $\Rightarrow f_{ref} = 40\text{Mhz}$
- Total capacitance being switched $= C_{ref}$
- $V_{dd} = V_{ref} = 5\text{V}$
- Power for reference datapath $= P_{ref} = C_{ref} V_{ref}^2 f_{ref}$
  from [Chandrakasan92] (IEEE JSSC)
Parallel Datapath

- The clock rate can be reduced by half with the same throughput \( \Rightarrow f_{par} = f_{ref} / 2 \)
- \( V_{par} = V_{ref} / 1.7 \), \( C_{par} = 2.15C_{ref} \)
- \( P_{par} = (2.15C_{ref})(V_{ref}/1.7)^2 (f_{ref}/2) = 0.36 P_{ref} \)

Area = 1476 x 1219 \( \mu \)m²

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

Area = 640 x 1081 \( \mu \)m²
### A Simple Datapath: 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>

### Power /Energy Optimization Space

<table>
<thead>
<tr>
<th>Energy</th>
<th>Constant Throughput/Latency</th>
<th>Variable Throughput/Latency</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Design Time</td>
<td>Sleep Mode</td>
</tr>
<tr>
<td>Active</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Logic design</td>
<td>Scaled $V_{dd}$</td>
<td>Clock gating</td>
</tr>
<tr>
<td>Scaled $V_{dd}$</td>
<td>Trans. sizing</td>
<td></td>
</tr>
<tr>
<td>Multi-$V_{dd}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Leakage</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stack effects</td>
<td>Scaled $V_{dd}$</td>
<td>Sleep T's</td>
</tr>
<tr>
<td>Trans sizing</td>
<td>+ Multi-$V_{Th}$</td>
<td>Multi-$V_{dd}$ Variable $V_{Th}$</td>
</tr>
<tr>
<td>Scaling $V_{dd}$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Reducing Active Power

- Downsizing, lowering the supply on the critical path will lower the operating frequency
- Downsize (lowering supply) non-critical paths
  - Narrows down the path delay distribution
  - Increases impact of variations

Multiple Supply Voltages

- Block-level supply assignment
  - Higher throughput/lower latency functions are implemented in higher $V_{DD}$
  - Slower functions are implemented with lower $V_{DD}$
  - “Voltage islands” as called by IBM
  - 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
Leakage Issue

- Driving from $V_{DDL}$ to $V_{DDH}$
- Level converter

Multiple Supplies in a Block

Conventional Design

CVS Structure

M. Takahashi, ISSCC’98.

Lower $V_{DD}$ portion is shaded

“Clustered voltage scaling”
Multiple Supplies in a Block

CVS

Layout:

Three $V_{DD}$'s

From Kuroda

$V_1 = 1.5V$, $V_{TH} = 0.3V$, $\rho(t):\lambda$
Optimum Numbers of Supplies

- The more $V_{DD}$’s, the less power, but the effect will be saturated.
- Power reduction effect will be decreased as $V_{DD}$’s are scaled.
- Optimum $V_2/V_1$ is around 0.7.

Hamada, CICC’01

---

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 (combinatorial) level converter
  - More sensitive to coupling and supply noise
Level-Converting Flip-Flop

Dual-Supply-Datapath: Layout Issue

(a) Dedicated row (Conventional)

(b) Possible layout reduction (Conventional)

(c) Shared-well layout

A shared-well technique is appropriate for random placement of cells
**Standard-Cell Dual-Supply-Voltage**

- A $V_{DDH}$ circuit is assigned only to a critical path
- A $V_{DDL}$ circuit is used in a non-critical path and for driving a large capacitive load

**Shared-Well Dual-Supply-Voltage**

- Both circuits can be placed in the same N-well
- Cell layout becomes complex
- An intrinsic negative back-biasing of PMOS degrades speed

Shimazaki, ISSCC'03
The dual-supply technique expands the power-delay optimization space.

Next Lecture

- Continue power-performance tradeoffs