Low-Power CMOS VLSI Design

Lan-Da Van (范倫達), Ph. D.
Department of Computer Science,
National Chiao Tung University,
Taiwan, R.O.C.
Fall, 2017

ldvan@cs.nctu.edu.tw
http://www.cs.nctu.tw/~ldvan/
Outline

**Introduction**

- Low-Power Process-Level Design (Ignore here)
- Low-Power Logic/Circuit-Level Design
- Low-Power Algorithm/Architecture-Level Design
- Low-Power System-Level Design

**Conclusion**

**References**
Low Power Design– An Ongoing and Important Discipline

Historical figure of merit for VLSI design
- Performance (circuit speed and system quality)
- Chip area (circuit cost). But now,…

Power dissipation is now an important metric in VLSI design.
- No single major source for power savings across all design levels – Required a new way of THINKING!!!
- Companies lack the basic power-conscious culture and designers need to be educated in this respect.

Overall Goal - To reduce power dissipations but maintaining adequate throughput rate.
Motivation - Microprocessor

<table>
<thead>
<tr>
<th>Processor</th>
<th>Source</th>
<th>Date (ship)</th>
<th>Bits</th>
<th>Clock</th>
<th>SPEC-92</th>
<th>SPEC-95</th>
<th>Units / Issue</th>
<th>Pipe Stages int/issue/fp</th>
<th>Cache (i/d)</th>
<th>Vdd (V)</th>
<th>Tech (um)</th>
<th>Mem (Mi)</th>
<th>Power (W)</th>
<th>Size (mm²)</th>
<th>Xisistor (10⁶)</th>
</tr>
</thead>
<tbody>
<tr>
<td>i386SX</td>
<td>[41]</td>
<td>88</td>
<td>w/32</td>
<td>33</td>
<td>6.2</td>
<td>3.3</td>
<td>1/1</td>
<td>4/na/na</td>
<td>NA</td>
<td>NA</td>
<td>5.0</td>
<td>1.0</td>
<td>-2</td>
<td>3</td>
<td>0.28</td>
</tr>
<tr>
<td>i486</td>
<td>[45]</td>
<td>89</td>
<td>w/32</td>
<td>50</td>
<td>27.9</td>
<td>13.1</td>
<td>2/1</td>
<td>5/na/5?</td>
<td>8 u</td>
<td>4</td>
<td>5.0</td>
<td>1.0</td>
<td>2</td>
<td>1.2</td>
<td></td>
</tr>
<tr>
<td>i486DX</td>
<td>[10]</td>
<td>91</td>
<td>w/32</td>
<td>32.2</td>
<td>16.0</td>
<td></td>
<td>2/1</td>
<td>5/na/5?</td>
<td>8 u</td>
<td>4</td>
<td>5.0</td>
<td>1.0</td>
<td>8</td>
<td>1.2</td>
<td></td>
</tr>
<tr>
<td>i486DX2</td>
<td>[20]</td>
<td>91</td>
<td>w/32</td>
<td>66</td>
<td></td>
<td></td>
<td>2/1</td>
<td>5/na/5?</td>
<td>8 u</td>
<td>4</td>
<td>5.0</td>
<td>1.0</td>
<td>8</td>
<td>1.2</td>
<td></td>
</tr>
<tr>
<td>i486DX4</td>
<td>[42]</td>
<td>93</td>
<td>w/32</td>
<td>51</td>
<td>27</td>
<td></td>
<td>2/1</td>
<td>5/na/5?</td>
<td>16 u</td>
<td>4?</td>
<td>3.3</td>
<td>0.6</td>
<td>4</td>
<td>1.6</td>
<td></td>
</tr>
<tr>
<td>P5</td>
<td>[2,10]</td>
<td>93</td>
<td>w/32</td>
<td>66</td>
<td>78</td>
<td>63.6</td>
<td>3/2</td>
<td>5/na/8</td>
<td>8/8</td>
<td>2/2</td>
<td>5.0</td>
<td>0.6</td>
<td>3</td>
<td>16.0</td>
<td>290</td>
</tr>
<tr>
<td>P54VRT</td>
<td>[vi,sp]</td>
<td>94</td>
<td>w/32</td>
<td>90</td>
<td>110</td>
<td>84.4</td>
<td>3/2</td>
<td>5/na/8</td>
<td>8/8</td>
<td>2/2</td>
<td>2.9</td>
<td>0.6</td>
<td>4</td>
<td>6.5</td>
<td>3.0</td>
</tr>
<tr>
<td>P54C</td>
<td>[2,11,vi]</td>
<td>94</td>
<td>w/32</td>
<td>100</td>
<td>122</td>
<td>93.2</td>
<td>3/2</td>
<td>5/na/8</td>
<td>8/8</td>
<td>2/2</td>
<td>3.3</td>
<td>0.6</td>
<td>4</td>
<td>5.0</td>
<td>148</td>
</tr>
<tr>
<td>P54CQS</td>
<td>[47,vi]</td>
<td>95</td>
<td>w/32</td>
<td>120</td>
<td>37</td>
<td></td>
<td>3/2</td>
<td>5/na/8</td>
<td>8/8</td>
<td>2/2</td>
<td>3.3</td>
<td>0.35</td>
<td>4</td>
<td>10.0</td>
<td>90</td>
</tr>
<tr>
<td>P54CS</td>
<td>[53,vi]</td>
<td>95</td>
<td>w/32</td>
<td>133</td>
<td>174</td>
<td>121</td>
<td>3/2</td>
<td>5/na/8</td>
<td>8/8</td>
<td>2/2</td>
<td>3.3</td>
<td>0.35</td>
<td>4</td>
<td>90</td>
<td>3.1</td>
</tr>
<tr>
<td>P55C</td>
<td>[bw]</td>
<td>96</td>
<td>w/32</td>
<td>166</td>
<td>71</td>
<td>62</td>
<td>3/2</td>
<td>5/na/8</td>
<td>16/16</td>
<td></td>
<td>2.5</td>
<td>0.28</td>
<td>2</td>
<td>29.2</td>
<td>30</td>
</tr>
<tr>
<td>P6</td>
<td>[vi]</td>
<td>95</td>
<td>w/32</td>
<td>200</td>
<td>320</td>
<td>283</td>
<td>7/3</td>
<td>5/na/8</td>
<td>8/8</td>
<td>2/2</td>
<td>3.3</td>
<td>0.35</td>
<td>4</td>
<td>35</td>
<td>28.1</td>
</tr>
<tr>
<td>Klamath (P6)</td>
<td>[vi]</td>
<td>97</td>
<td>w/32</td>
<td>166</td>
<td>293</td>
<td>261</td>
<td>7/3</td>
<td>5/na/8</td>
<td>8/8</td>
<td>2/2</td>
<td>3.3</td>
<td>0.35</td>
<td>4</td>
<td>29.4</td>
<td>23.4</td>
</tr>
<tr>
<td>Deschutes (P6)</td>
<td>[vi]</td>
<td>97</td>
<td>w/32</td>
<td>166</td>
<td>293</td>
<td>261</td>
<td>7/3</td>
<td>5/na/8</td>
<td>8/8</td>
<td>2/2</td>
<td>3.3</td>
<td>0.35</td>
<td>4</td>
<td>29.4</td>
<td>23.4</td>
</tr>
<tr>
<td>Merced (P7)</td>
<td>[vi]</td>
<td>977</td>
<td>w/64</td>
<td>166</td>
<td>293</td>
<td>261</td>
<td>7/3</td>
<td>5/na/8</td>
<td>8/8</td>
<td>2/2</td>
<td>3.3</td>
<td>0.35</td>
<td>4</td>
<td>29.4</td>
<td>23.4</td>
</tr>
</tbody>
</table>
Low Power Competitive Reasons

- **Battery Powered Systems**
  - Extend *battery life*
  - Reduce weight and size

- **High-Performance Systems**
  - **Cost**
    - Package (chip carrier, heat sink, card slots, …)
    - Power Systems (supplies, distribution, regulators, …)
    - Fans (noise, power, reliability, area, …)
    - Operating cost to customer – Re-start issue.
  - **Reliability**
    - Failure rate increases by 4X for $T @ 110\text{C}$ vs $70\text{C}$
  - **Size and Weight**
The Power Crisis: Portability

Expected Battery Lifetime increase
Over next 5 years: 30-40%

PDA, Cellular Phone, Notebook Computer, etc.
A Multimedia Terminal: The Infopad

Present day battery technology (year 1990) – 20 lbs for 10hrs
VLSI Signal Processing System Design Space

- System Level
- Algorithm Level
- Architecture Level
- Logic Level
- Circuit Level
- Process Level

- Cost
- Performance
- Area
- Test
- Power
Low Power System Design Space

- **System**
  - Power budgeting, S/H partitioning, power management, core selection

- **Algorithm**
  - Algorithmic reduction, data transformation, CSE, low-complexity operation

- **Architecture**
  - Parallelism, pipelining, re-timing, unfolding, signal ordering, glitch minimization, data representation, resource allocation, multi-clock

- **Logic/Circuit**
  - Logic style, arithmetic, glitch/noise minimization, re-sizing, adaptive voltage scaling, multi-Vdd, multi-Vth, multi-clock, layout, power-driven P&R

- **Process**
  - Low-power device, alternative technology, multi-Vth
Outline

- Introduction
- Low-Power Process-Level Design (Ignore here)
- **Low-Power Logic/Circuit-Level Design**
- Low-Power Algorithm/Architecture-Level Design
- Low-Power System-Level Design
- Conclusion
- References
Where Does Power Go in CMOS?

Source of power dissipation

- $P = P_{\text{dynamic}} + P_{\text{short-circuit}} + P_{\text{leakage}} + P_{\text{static}}$

Definitions:

- Dynamic/switching power: $P = \alpha CV^2f$
  - Charging and discharging parasitic capacitors
  - $\alpha$: switching activity factor
- Short circuit power $P = I_{\text{sc}}V$
  - Direct path between supply rail during switching
- Leakage power $P = I_{\text{leakage}}V$
  - Reverse bias diode leakage
  - Sub-threshold conduction
- Static power $P = I_{\text{static}}V$
  - Each input node is connected to fixed stable voltage
Dynamic Power Consumption (1/2)

\[ \text{Power} = \frac{\text{Energy}}{\text{transition}} \times \text{transition rate} \]

\[ = C_L \times V_{dd}^2 \times f_{0\rightarrow1} \]

\[ = C_L \times V_{dd}^2 \times Pb_{0\rightarrow1} \times f \]

\[ = C_{EFF} \times V_{dd}^2 \times f \]

\[ = Pb_{0\rightarrow1} \times C_L \times V_{dd}^2 \times f \]

- \( C_{EFF} = \text{Effective Capacitance} = C_L \times Pb_{0\rightarrow1} \)
Dynamic Power Consumption (2/2)

- Need to reduce $Pb_{0 \rightarrow 1}$, $C_L$, $V_{dd}$, and $f$ for low power design
  - Reduce the probability, $P_{0 \rightarrow 1}$
  - Minimize the geometry and remove the redundancy
  - Reduce the power supply level
  - Use lowest clock frequency

- Power dissipation is data dependent function of switching activity. $\Rightarrow$ Pattern Dependent!
Choice of Logic Style

CONVENTIONAL CMOS Adder

OPTIMIZED static Adder

DCVSL Adder

CPI Adder
Choice of Logic Style

- Power-delay product improves as voltage decreases
- The “best” logic style minimizes power-delay (i.e., energy) for a given delay constraint.

![Diagram showing power-delay product vs. delay for different logic styles and decreasing Vdd](image-url)
Type of Logic Function: NOR

Example: Static-style 2-input NOR gate

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Truth Table of 2-Input NOR Gate

Assume:

\[ P(A=1) = \frac{1}{2} \]
\[ P(B=1) = \frac{1}{2} \]

Then:

\[ P(\text{Out}=1) = \frac{1}{4} \]

\[ P(0 \rightarrow 1) = P(\text{Out}=0) \times P(\text{Out}=1) \]
\[ = \frac{3}{4} \times \frac{1}{4} = \frac{3}{16} \]

\[ \alpha_{0 \rightarrow 1} = \frac{3}{16} \]
2-Input NOR Gate Transition Probability

\[
P_0 \rightarrow 1 = P_0 P_1 = (1 - (1 - P_A)(1 - P_B))(1 - P_A)(1 - P_B)
\]

\[
P_1 = (1 - P_A)(1 - P_B)
\]
Type of Logic Function: XOR

Example: Static-style 2-input XOR gate

Assume:
\[ P(A=1) = 1/2 \]
\[ P(B=1) = 1/2 \]

Then:
\[ P(\text{Out}=1) = 1/2 \]
\[ P(0 \rightarrow 1) = P(\text{Out}=0)*P(\text{Out}=1) = 1/2 * 1/2 = 1/4 \]

\[ a_{0 \rightarrow 1} = 1/4 \]

Truth Table of 2-Input XOR Gate

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
\[ P_1 = P_A (1-P_B) + P_B (1-P_A) = P_A + P_B - 2P_A P_B \]

\[ P_{0 \rightarrow 1} = P_0 P_1 = (1 - (P_A + P_B - 2P_A P_B)) (P_A + P_B - 2P_A P_B) \]
Which One is Your Choice?

WHICH ONE IS FOR LOW-POWER DESIGN?
Glitching Activity in CMOS Network

also called: dynamic hazards

\[ \alpha_{0->1} \] can be greater than 1 due to glitching!
Glitching in a Carry Ripple Adder

Transitions due to carry propagation
Chain vs Tree Datapath (1/2)

<table>
<thead>
<tr>
<th></th>
<th>O1</th>
<th>O2</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>P₁ (Chain)</td>
<td>1/4</td>
<td>1/8</td>
<td>1/16</td>
</tr>
<tr>
<td>P₀=1-P₁ (Chain)</td>
<td>3/4</td>
<td>7/8</td>
<td>15/16</td>
</tr>
<tr>
<td>P₀-&gt;₁ (Chain)</td>
<td>3/16</td>
<td>7/64</td>
<td>15/256</td>
</tr>
<tr>
<td>P₁ (Tree)</td>
<td>1/4</td>
<td>1/4</td>
<td>1/16</td>
</tr>
<tr>
<td>P₀=1-P₁ (Tree)</td>
<td>3/4</td>
<td>3/4</td>
<td>15/16</td>
</tr>
<tr>
<td>P₀-&gt;₁ (Tree)</td>
<td>3/16</td>
<td>3/16</td>
<td>15/256</td>
</tr>
</tbody>
</table>
Chain vs Tree Datapath (2/2)

<table>
<thead>
<tr>
<th></th>
<th>O1</th>
<th>O2</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_{0 \rightarrow 1}$ (Chain)/$P_{0 \rightarrow 1}$ (Tree)</td>
<td>1</td>
<td>0.58</td>
<td>1</td>
</tr>
<tr>
<td>$\alpha_{0 \rightarrow 1}$ (Chain)/$\alpha_{0 \rightarrow 1}$ (Tree)</td>
<td>1</td>
<td>0.83</td>
<td>1.47</td>
</tr>
</tbody>
</table>

Ideal w/t delay
Practical with delay

Which one is for low-power design?
**Glitching at the Datapath Level**

**Irregular**

![Irregular Diagram](image)

\[(A + B) + (C + D)\]

**Regular**

![Regular Diagram](image)

\[(A + B) + C + D\]

**Two Glitches!**

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Normalized # of Transitions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Tree</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
</tr>
</tbody>
</table>

- Can be reduced by reducing the logic depth and balancing signal paths.
How to Minimize Glitching?

Equalize Length of Timing Paths through Design!
Data Representation (1/2)

Two’s Complement Representation

- Transition Probability
- Bit Number
- $B = \log_2 (3\sigma)$
- $\rho$ values: $0.99$, $0.75$, $0.50$, $0.25$, $0.00$, $0.25$, $0.50$, $0.75$, $0.99$
- Very anti-correlated
- Uncorrelated

Sign-magnitude Representation

- Transition Probability
- Bit Number
- $B = \log_2 (3\sigma)$
- $\rho$ values: $0.99$, $0.75$, $0.50$, $0.25$, $0.00$, $0.25$, $0.50$, $0.75$, $0.99$
- Very correlated

Sign Bit

Bit Position
## Data Representation (2/2)

### (Binary v.s. Gray Encoding)

#### Data coding representation

<table>
<thead>
<tr>
<th>Decimal Value</th>
<th>Binary Code</th>
<th>Grey Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td>1</td>
<td>0001</td>
<td>0001</td>
</tr>
<tr>
<td>2</td>
<td>0010</td>
<td>0011</td>
</tr>
<tr>
<td>3</td>
<td>0011</td>
<td>0010</td>
</tr>
<tr>
<td>4</td>
<td>0100</td>
<td>0110</td>
</tr>
<tr>
<td>5</td>
<td>0101</td>
<td>0111</td>
</tr>
<tr>
<td>6</td>
<td>0110</td>
<td>0101</td>
</tr>
<tr>
<td>7</td>
<td>0111</td>
<td>0100</td>
</tr>
<tr>
<td>8</td>
<td>1000</td>
<td>1100</td>
</tr>
<tr>
<td>9</td>
<td>1001</td>
<td>1101</td>
</tr>
<tr>
<td>10</td>
<td>1010</td>
<td>1111</td>
</tr>
<tr>
<td>11</td>
<td>1011</td>
<td>1110</td>
</tr>
<tr>
<td>12</td>
<td>1100</td>
<td>1010</td>
</tr>
<tr>
<td>13</td>
<td>1101</td>
<td>1011</td>
</tr>
<tr>
<td>14</td>
<td>1110</td>
<td>1001</td>
</tr>
<tr>
<td>15</td>
<td>1111</td>
<td>1000</td>
</tr>
</tbody>
</table>

#### Binary Vs. Grey coding

- Binary Coded
- Grey Coded

![Graph showing binary and grey coded values for various words.](image-url)
Outline

- Introduction
- Low-Power Process-Level Design (Ignore here)
- Low-Power Logic/Circuit-Level Design
- Low-Power Algorithm/Architecture-Level Design
- Low-Power System-Level Design
- Conclusion
- References
Signal Reordering Operation

Ex1. \( Y = AB + AC = A(B + C) \)

Ex2. \( Y = 3X = X + (X << 1) \)
Resource Sharing Can Increase Activity (1/2)

Separate Bus Structure

# of Bus Transitions Per Cycle = 2(1+1/2+1/4+...)=4,
Where 2 means 2 separate buses, 1 denotes the transition probability of LSB, ½ denotes the transition probability of 2nd LSB, and etc.
Resource Sharing Can Increase Activity (2/2)

Adder Activity

Bit Position
Lowering $V_{dd}$ Increases Delay

- Relatively independent of logic function and style.
Reducing $V_{dd}$

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

\[ P \times t_d = E_t = C_L \times V_{dd}^2 \]

\[ \frac{E(V_{dd=2})}{E(V_{dd=5})} = \frac{(C_L) \times (2)^2}{(C_L) \times (5)^2} \]

\[ E(V_{dd=2}) \approx 0.16 \times E(V_{dd=5}) \]
Architecture Trade-offs: Reference Datapath

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

Area = 636 x 833 \( \mu \text{mm}^2 \)
Parallel Datapath

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

Area = 1476 x 1219 \( \mu^2 \)
Pipelined Datapath

- $f_{\text{pipe}} = f_{\text{ref}}$
- $C_{\text{pipe}} = 1.1C_{\text{ref}}$
- $V_{\text{pipe}} = V_{\text{ref}}/1.7$

- Voltage can be dropped while maintaining the original throughput.

- $P_{\text{pipe}} = C_{\text{pipe}} V_{\text{pipe}}^2 f_{\text{pipe}} = (1.1C_{\text{ref}}) (V_{\text{ref}}/1.7)^2 f_{\text{ref}} = 0.37 P_{\text{ref}}$

Area = 640 x 1081 $\mu^2$
**Summary: A Low-Power Data Path**

<table>
<thead>
<tr>
<th>Architecture type</th>
<th>Voltage</th>
<th>Area</th>
<th>Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference Datapath (no pip/par)</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>0.37</td>
</tr>
<tr>
<td>Parallel datapath</td>
<td>2.9V</td>
<td>3.4</td>
<td>0.34</td>
</tr>
<tr>
<td>Pipeline-parallel datapath</td>
<td>2.0V</td>
<td>3.7</td>
<td>0.18</td>
</tr>
</tbody>
</table>

- Desire to operate at lowest possible speeds (using low supply voltages)
- Use architecture optimization to compensate for slower operation
### Computational Complexity of DCT Algorithms

<table>
<thead>
<tr>
<th>DCT Algorithm</th>
<th>Multiplies (8x8)</th>
<th>Additions (8x8)</th>
<th>Implemented by</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brute Force</td>
<td>4096</td>
<td>4096</td>
<td>-</td>
</tr>
<tr>
<td>Row-Col DCT</td>
<td>1024</td>
<td>1024</td>
<td>Bell core (16x16)</td>
</tr>
<tr>
<td>Chen’s Algorithm</td>
<td>256</td>
<td>416</td>
<td>Telettra</td>
</tr>
<tr>
<td>Lee’s Algorithm</td>
<td>192</td>
<td>464</td>
<td>SGS - Thompson</td>
</tr>
<tr>
<td>Feig’s Algorithm (scaled DCT)</td>
<td>54</td>
<td>462</td>
<td>IBM (GP computer)</td>
</tr>
</tbody>
</table>

- Reducing # of operations (switching events) is important in reducing the power.
- Routing and layout issues for irregular structures vs. regular structures.
Low-Power Cache and Register Configuration

- **Application profiling**
  - Trade-off between performance, power and size

- **Rule of thumb**
  - Access and storage the most frequently used instructions
  - Avoid accessing larger cache/register
  - Partition cache and register
  - Aware of partitioning
Outline

- Introduction
- Low-Power Process-Level Design (Ignore here)
- Low-Power Logic/Circuit-Level Design
- Low-Power Algorithm/Architecture-Level Design
- **Low-Power System-Level Design**
  - Low Power System Perspective
  - Low Power Applications
- Conclusion
- References
Power Down Techniques

Operating States

- **ACTIVE OR FULL-ON (FASTEST CLOCK)**
- **STANDBY (SLOW CLOCK)**
- **SUSPEND OR SLEEP (SLOWEST CLOCK or SHUT DOWN)**

Activity Monitor

μ - processor
# Software versus Hardware

<table>
<thead>
<tr>
<th></th>
<th>Advantage</th>
<th>Disadvantage</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Software</strong></td>
<td>• Free but not always</td>
<td>• High power consumption</td>
</tr>
<tr>
<td></td>
<td>• High flexibility</td>
<td>• Slow in execution</td>
</tr>
<tr>
<td></td>
<td>• Ease of compatibility</td>
<td>• Inefficient</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Larger staff</td>
</tr>
<tr>
<td><strong>Hardware</strong></td>
<td>• High speed</td>
<td>• High die cost</td>
</tr>
<tr>
<td></td>
<td>• Low power</td>
<td>• Low flexibility</td>
</tr>
<tr>
<td></td>
<td>• High efficiency</td>
<td>• Low compatibility</td>
</tr>
<tr>
<td></td>
<td>• Less staff</td>
<td></td>
</tr>
</tbody>
</table>
Energy-Efficient Software Coding

- Potential for power reduction via software modification is relatively unexploited.
- Code size and algorithmic efficiency can significantly affect energy dissipation.
- Pipelining at software level - VLIW coding style

References:

Power Hunger – Clock Network

- H-Tree – design deficiencies based on Elmore delay model.
- PLL – every designer (digital or analog) should have the knowledge of PLL.
  - Multiple frequencies in chips/systems – by PLL
  - Low main frequency, But
  - Jitter and noise, gain and bandwidth, pull-in and lock time, stability …
- Asynchronous => Use gated clocks, sleep mode
Power Analysis in the Design Flow

- Too Little
  - Z = X^4Y
  - if (Z < 0) then Z = 0
- Fast & Accurate
- too Late
- Algorithm
- Power
- Architecture
- Exploration
- Time
- Area
- Mem
- Ctrl

Lan-Da Van
Applications I: Wireless Computing/Communication

Fiber Optic Backbone > 10 Gbits/sec

- COMPUTE SERVERS
- WIRELESS BASE STATION
- VIDEO DATABASE
  - Compressed Video
- SPEECH RECOGNITION
  - LARGE COMMERCIAL DATABASE
  - Airline schedule, Newspaper, ...

PERSONAL COMMUNICATORS

InfoPad (A Portable Multimedia Terminal)
- Speech I/O and Pen Input
- X-terminal
- Full-motion Video
Applications II: A Portable Multimedia Terminal
Applications III: System on Chip (SOC)

- Entire system function
  - Logic + Memory
  - More than two types of devices

- Allow more freedoms in architecture

- Hardware and software partition

Diagram:

- Idea
- Alg./Arc. Exploration
- HW+SW
- HW
- Virtual SoC Platform
- SOC

Notes:

- Entire system function
  - Logic + Memory
  - More than two types of devices

- Allow more freedoms in architecture

- Hardware and software partition

- Virtual SoC Platform
- SOC

- ASIC Approach
- BUT Hard to SOC Design!
Conclusions

- Low-Power and high-speed tradeoff design is an essential requirement for many applications.
- Low power impacts on the cost, size, weight, performance, and reliability.

- **Reduce** $P_{0\rightarrow1}$, $C_L$, $V_{dd}$, and $f$ for low power design across each level!!
Reference


Self-Test Exercises

STE1: Calculate the switching activity EQUATION EXPRESSION of 2-input AND gate and simulate the histogram of transition probability ($P_{0->1}$) vs $P_A$ and $P_B$.

STE2: Calculate the switching activity EQUATION EXPRESSION of 3-input NAND gate.