18-322  Lecture 18

Arithmetic Blocks
Low-Power Design

Textbook: Chapter 7, Section 4.4

Outline

• Arithmetic building blocks
  » Adders
  » Multipliers

• Low-power design
  » Reducing power consumption
  » Data-path/Control circuitry
A Generic Digital Processor

The Binary Adder

\[ S = A \oplus B \oplus C_i \]
\[ = AB\overline{C}_i + \overline{A}B\overline{C}_i + \overline{A}\overline{B}C_i + ABC_i \]
\[ C_0 = AB + BC_i + AC_i \]
Complementary Static CMOS Full Adder

AB + (A+B)Ci

28 Transistors

ABCi + \overline{C}_6(A+B+C_i)

The Ripple-Carry Adder

Worst case delay linear with the number of bits

\[ t_d = O(N) \]

\[ t_{adder} = (N - 1)t_{carry} + t_{sum} \]

Goal: Make the fastest possible carry path circuit
Inversion Property

Minimize Critical Path by Reducing Inverting Stages

Exploit Inversion Property

Note: need 2 different types of cells
A better structure: the Mirror Adder

Express Sum and Carry as a function of $P, G, D$

Define 3 new variable which ONLY depend on $A, B$

- $\text{Generate} \ (G) = AB$
- $\text{Propagate} \ (P) = A \oplus B$
- $\text{Delete} = A \land B$

$$C_o(G,P) = G + PC_i$$
$$S(G,P) = P \oplus C_i$$

Can also derive expressions for $S$ and $C_o$ based on $D$ and $P$
Carry-Bypass Adder

Idea: If (P0 and P1 and P2 and P3 = 1) then \( C_{o3} = C_0 \), else “kill” or “generate”.

Carry-Bypass Adder (cont.)

Note: the topological path worst-case delay is much higher than the true critical path!
**Carry Ripple vs. Carry Bypass**

- **Ripple Adder**
- **Bypass Adder**

**LookAhead - Basic Idea**

\[ C_{0,k} = G_k + P_k C_{0,k-1} \]
\[ C_{0,k} = G_k + P_k (G_{k-1} + P_{k-1}(\ldots + P_1 (G_0 + P_0 C_{i,0}))) \]
The Binary Multiplication

\[
\begin{array}{c}
101010 \\
\times 101111 \\
\hline
101010 \\
000000 \\
101010 \\
+ 101010 \\
\hline
111001110
\end{array}
\]

AND operation

Partial Products

The Array Multiplier

Digital Integrated Circuits
The MxN Array Multiplier — Critical Path

Optimization is very difficult (several critical paths)!

Optimize this!

$\text{Critical Path 1}$

$\text{Critical Path 2}$

$\text{Critical Path 1 & 2}$

$\text{Carry-Save Multiplier}$

Optimization easier (unique critical path)!

$t_{\text{mult}} = (N-1)t_{\text{carry}} + (N-1)t_{\text{and}} + t_{\text{merge}}$
Wallace-Tree Multiplier

Multipliers —Summary

- Optimization Goals Different vs. Binary Adder
- Once Again: Identify the Critical Path
- Other possible techniques
  - Logarithmic versus Linear (Wallace Tree Mult)
  - Data encoding (Booth)
  - Pipelining
Design as a Trade-Off

Outline

✓ Arithmetic building blocks
  » Adders
  » Multipliers

● Low-power design
  » Reducing power consumption
  » Data-path/Control circuitry
How about POWER?
Reducing power consumption

- Load capacitance ($C_L$)
  - Roughly proportional to the chip area
- Switching activity (avg. number of transitions/cycle)
  - Very data dependent
  - A big portion due to glitches (real-delay)
- Clock frequency ($f$)
  - Lowering only $f$ decreases average power, but total energy is the same and throughput is worse

- Voltage supply ($V_{DD}$)
  - Biggest impact: 50% reduction in $V_{DD}$, 75% reduction in power

Using parallelism (1)

$$P_{ref} = C_{ref} V_{DD}^2 f_{ref}$$

Assume: $t_p = 25$ns (worst-case, all modules) at $V_{DD} = 5V$
Using parallelism (2)

- $C_{par} = 2.15C$ (extra-routing needed)
- $f_{par} = f/2$ ($t_{p,new} = 50\text{ns} \Rightarrow V_{DD} \sim 2.9\text{V}$; $V_{DD,par} = 0.58 V_{DD}$)
- $P_{par} = C_{par} V_{DD}^2 f_{par} = 0.36 P_{ref}$

Using pipelining

- $C_{pipe} = 1.15C$
- Delay decreases 2 times ($V_{DD,pipe} = 0.58 V_{DD}$)
- $P_{pipe} = 0.39 P$
Low energy gates – gate sizing

- Use the smallest transistors that satisfy the delay constraints
  - Slack time - difference between required time and arrival time of a signal at a gate output
  - Positive slack - size down
  - Negative slack - size up
- Make gates that toggle more frequently smaller
- Slope engineering to reduce short circuit currents

Low energy gate netlists – pin ordering

- Better to postpone the introduction of signals with a high transition rate (signals with signal probability close to 0.5)
Question for you (5 min):

» Which of the two designs is more energy efficient?
  
  – Assume:
    • Zero-delay model
    • All inputs have a signal probability of 0.5
  
  – Hint: Calculate $p_{0 \rightarrow 1}$ for $W$, $X$ and $F$

For zero-delay model

» Chain design is better

» But ignores glitching
  
  – Depending on the gate delays, the chain design may be worse
Control circuits

- State encoding has a big impact on the power efficiency
- Energy driven -> try to minimize number of bit transitions in the state register
  - Fewer transitions in state register
  - Fewer transitions propagated to combinational logic

Dual supply voltage

- Use two $V_{DD}$s (e.g., 2.5V and 1.5V)
  - Use the higher supply for gates on the critical path
  - Use the lower supply for gates off the critical path
- Pro
  - Reduces energy without a performance loss
- Cons
  - Slight area penalty
  - Increased design time
  - Need level converters to interconnect gates on different supplies (to avoid static currents)
Clock gating

- Clock gating logic gates off the clock so that there's no switching power in the downstream logic.

Bus encoding

- Reduces number of bit toggles on the bus
- Different flavors
  - Bus-invert coding
    - Uses an extra bus line invert:
      - if the number of transitions is $< \frac{K}{2}$, invert = 0 and the symbol is transmitted as is
      - if the number of transitions is $> \frac{K}{2}$, invert = 1 and the symbol is transmitted in a complemented form
  - Low-weight coding
    - Uses transition signaling instead of level signaling
Bus invert coding

Under uniform random signal conditions (non-correlated data), 25% upper bound on toggle reduction

Source: M. Stan et al., 1994