Outline

- Pseudo-nMOS Logic
- Dynamic Logic
- Pass Transistor Logic
Introduction

• What makes a circuit fast?
  – $I = C \frac{dV}{dt} \rightarrow t_{pd} \propto (C/I) \Delta V$
  – low capacitance
  – high current
  – small swing
• Logical effort is proportional to $C/I$
• pMOS are the enemy!
  – High capacitance for a given current
• Can we take the pMOS capacitance off the input?
• Various circuit families try to do this…

Pseudo-nMOS

• In the old days, nMOS processes had no pMOS
  – Instead, use pull-up transistor that is always ON
• In CMOS, use a pMOS that is always ON
  – Ratio issue
  – Make pMOS about ¼ effective strength of pulldown network
Pseudo-nMOS Gates

- Design for unit current on output to compare with unit inverter.
- pMOS fights nMOS

Inverter  NAND2  NOR2

\[ g_u = \quad \quad g_u = \quad \quad g_u = \]
\[ g_d = \quad \quad g_d = \quad \quad g_d = \]
\[ g_{avg} = \quad \quad g_{avg} = \quad \quad g_{avg} = \]
\[ p_u = \quad \quad p_u = \quad \quad p_u = \]
\[ p_d = \quad \quad p_d = \quad \quad p_d = \]
\[ p_{avg} = \quad \quad p_{avg} = \quad \quad p_{avg} = \]
Pseudo-nMOS Gates

• Design for unit current on output to compare with unit inverter.
• pMOS fights nMOS
Pseudo-nMOS Design

Ex: Design a k-input AND gate using pseudo-nMOS. Estimate the delay driving a fanout of H

- G = 1 * 8/9 = 8/9
- F = GBH = 8H/9
- P = 1 + (4+8k)/9 = (8k+13)/9
- N = 2
- D = NF^{1/N} + P = \frac{4\sqrt{2H}}{3} + \frac{8k+13}{9}
Pseudo-nMOS Power

- Pseudo-nMOS draws power whenever $Y = 0$
  - Called static power $P = I \cdot V_{DD}$
  - A few mA / gate * 1M gates would be a problem
  - This is why nMOS went extinct!
- Use pseudo-nMOS sparingly for wide NORs
- Turn off pMOS when not in use

Dynamic Logic

- Dynamic gates uses a clocked pMOS pullup
- Two modes: precharge and evaluate

Diagram showing precharge and evaluate modes for different types of logic gates: static, pseudo-nMOS, and dynamic.
The Foot

- What if pulldown network is ON during precharge?
- Use series evaluation transistor to prevent fight.

Logical Effort

Inverter  NAND2  NOR2

unfooted

footed
Logical Effort

Inverter

NAND2

NOR2

Monotonicity

- Dynamic gates require *monotonically rising* inputs during evaluation
  - 0 -> 0
  - 0 -> 1
  - 1 -> 1
  - But not 1 -> 0

Output should rise but does not

Output violates monotonicity during evaluation
Monotonicity Woes

- But dynamic gates produce monotonically falling outputs during evaluation
- Illegal for one dynamic gate to drive another!

A = 1

Y should rise but cannot

Monotonicity Woes

- But dynamic gates produce monotonically falling outputs during evaluation
- Illegal for one dynamic gate to drive another!
Domino Gates

- Follow dynamic stage with inverting static gate
  - Dynamic / static pair is called domino gate
  - Produces monotonic outputs

![Domino Gates Diagram]

Domino Optimizations

- Each domino gate triggers next one, like a string of dominos toppling over
- Gates evaluate sequentially but precharge in parallel
- Thus evaluation is more critical than precharge
- HI-skewed static stages can perform logic

![Domino Optimizations Diagram]
Dual-Rail Domino

- Domino only performs noninverting functions:
  - AND, OR but not NAND, NOR, or XOR
- Dual-rail domino solves this problem
  - Takes true and complementary inputs
  - Produces true and complementary outputs

<table>
<thead>
<tr>
<th>sig_h</th>
<th>sig_l</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Precharg</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>$\phi$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>‘1’</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>invalid</td>
</tr>
</tbody>
</table>

Example: AND/NAND

- Given $A_h$, $A_l$, $B_h$, $B_l$
- Compute $Y_h = A \cdot B$, $Y_l = \neg(A \cdot B)$
Example: AND/NAND

- Given $A_h, A_l, B_h, B_l$
- Compute $Y_h = A \cdot B$, $Y_l = \neg(A \cdot B)$
- Pulldown networks are conduction complements

![AND/NAND Circuit Diagram]

Example: XOR/XNOR

- Sometimes possible to share transistors

![XOR/XNOR Circuit Diagram]
Leakage

- Dynamic node floats high during evaluation
  - Transistors are leaky ($I_{OFF} \neq 0$)
  - Dynamic value will leak away over time
  - Formerly milliseconds, now nanoseconds!
- Use keeper to hold dynamic node
  - Must be weak enough not to fight evaluation

Charge Sharing

- Dynamic gates suffer from charge sharing
Charge Sharing

- Dynamic gates suffer from charge sharing

\[ V_x = V_Y = \frac{C_Y}{C_x + C_Y} V_{DD} \]
Secondary Precharge

- Solution: add secondary precharge transistors
  - Typically need to precharge every other node
- Big load capacitance $C_Y$ helps as well

\[ \phi \quad A \quad B \]

\[ \text{secondary precharge transistor} \]

Noise Sensitivity

- Dynamic gates are very sensitive to noise
  - Inputs: $V_{IH} \approx V_{th}$
  - Outputs: floating output susceptible noise
- Noise sources
  - Capacitive crosstalk
  - Charge sharing
  - Power supply noise
  - Feedthrough noise
  - And more!
Domino Summary

- Domino logic is attractive for high-speed circuits
  - 1.5 – 2x faster than static CMOS
  - But many challenges:
    - Monotonicity
    - Leakage
    - Charge sharing
    - Noise
- Widely used in high-performance microprocessors

NMOS Transistors in Series/Parallel

- Primary inputs drive both gate and source/drain terminals
- NMOS switch closes when the gate input is high
- \[ X = Y \text{ if } A \text{ and } B \]
- \[ X = Y \text{ if } A \text{ or } B \]
- Remember –
  NMOS transistors pass a strong 0 but a weak 1
### PMOS Transistors in Series/Parallel

- Primary inputs drive both gate and source/drain terminals
- PMOS switch closes when the gate input is low

\[
\begin{align*}
X &= Y \text{ if } A \land B = A + B \\
X &= Y \text{ if } A \lor B = A \bullet B
\end{align*}
\]

- Remember –
  PMOS transistors pass a strong 1 but a weak 0

### Pass Transistor (PT) Logic

- Gate is static – a low-impedance path exists to both supply rails under all circumstances
- \( N \) transistors instead of \( 2N \)
- No static power consumption
- Ratioless
- Bidirectional (versus undirectional)
VTC of PT AND Gate

Pure PT logic is not regenerative - the signal gradually degrades after passing through a number of PTs (can fix with static CMOS inverter insertion)

NMOS Only PT Driving an Inverter

- \( V_x \) does not pull up to \( V_{DD} \), but \( V_{DD} - V_{Tn} \)
- Threshold voltage drop causes static power consumption (\( M_2 \) may be weakly conducting forming a path from \( V_{DD} \) to GND)
- Notice \( V_{Tn} \) increases of pass transistor due to body effect (\( V_{SB} \))
Voltage Swing of PT Driving an Inverter

- Body effect – large $V_{SB}$ at x - when pulling high (B is tied to GND and S charged up close to $V_{DD}$)
- So the voltage drop is even worse

$$V_x = V_{DD} - (V_{Tn0} + \gamma(\sqrt{|2\phi_f| + V_x} - \sqrt{|2\phi_f|}))$$

Cascaded NMOS Only PTs

- Pass transistor gates should never be cascaded as on the left
- Logic on the right suffers from static power dissipation and reduced noise margins
Solution 1: Level Restorer

- Full swing on x (due to Level Restorer) so no static power consumption by inverter
- No static backward current path through Level Restorer and PT since Restorer is only active when A is high
- For correct operation $M_r$ must be sized correctly (ratioed)

Transient Level Restorer Circuit Response

- Restorer has speed and power impacts: increases the capacitance at x, slowing down the gate; increases $t_r$ (but decreases $t_f$)
Solution 2: Multiple $V_T$ Transistors

- Technology solution: Use (near) zero $V_T$ devices for the NMOS PTs to eliminate most of the threshold drop (body effect still in force preventing full swing to $V_{DD}$)

- Impacts static power consumption due to subthreshold currents flowing through the PTs (even if $V_{GS}$ is below $V_T$)

Out

\[ \text{In}_2 = 0V \]

\[ \text{In}_1 = 2.5V \]

\[ A = 2.5V \]

\[ B = 0V \]

Sneak path

Solution 3: Transmission Gates (TGs)

- Most widely used solution

\[ A \quad \overline{C} \quad B \]

\[ \overline{C} = \text{GND} \]

\[ A = V_{DD} \]

\[ C = V_{DD} \]

\[ \overline{C} = \text{GND} \]

\[ A = \text{GND} \]

\[ C = V_{DD} \]

- Full swing bidirectional switch controlled by the gate signal $C$, $A = B$ if $C = 1$
Solution 3: Transmission Gates (TGs)

- Most widely used solution

- Full swing bidirectional switch controlled by the gate signal C, A = B if C = 1

Resistance of TG

- \( V_{out} \) vs. \( V \)
- \( R_p \), \( R_n \), \( R_{eq} \)
- \( W/L_p = 0.50/0.25 \)
- \( W/L_n = 0.50/0.25 \)
Pass Transistor Circuits

- Use pass transistors like switches to do logic
- Inputs drive diffusion terminals as well as gates

CMOS + Transmission Gates:
- 2-input multiplexer
- Gates should be restoring

TG Multiplexer

\[ F = \overline{(I_{n1} \cdot S + I_{n2} \cdot \overline{S})} \]
Transmission Gate XOR

A ⊕ B

A

B

weak 0 if !A

weak 1 if A

an inverter
TG Full Adder

Differential TG Logic (DPL)
CPL

- **Complementary Pass-transistor Logic**
  - Dual-rail form of pass transistor logic
  - Avoids need for ratioed feedback
  - Optional cross-coupling for rail-to-rail swing

Differential PT Logic (CPL)

- **AND/NAND**
  - $F = AB$

- **OR/NOR**
  - $F = A + B$

- **XOR/XNOR**
  - $F = A \oplus B$
CPL Properties

- **Differential** so complementary data inputs and outputs are always available (so don’t need extra inverters)
- Still static, since the output defining nodes are always tied to $V_{DD}$ or GND through a low resistance path
- Design is **modular**; all gates use the same topology, only the inputs are permuted.
- Simple XOR makes it attractive for structures like **adders**
- Fast (assuming number of transistors in series is small)
- Additional routing overhead for complementary signals
- Still have static power dissipation problems

CPL Full Adder

[Diagram of CPL Full Adder]
CPL Full Adder

\[ A \quad B \quad C_{in} \quad C_{in} \quad C_{in} \quad C_{in} \quad I_{Sum} \quad I_{Sum} \quad I_{Sum} \quad I_{Sum} \quad I_{Sum} \quad I_{Sum} \quad Sum \quad Sum \quad Sum \quad Sum \quad Sum \quad Sum \quad I_{C_{out}} \quad I_{C_{out}} \quad I_{C_{out}} \quad I_{C_{out}} \quad C_{out} \quad C_{out} \quad C_{out} \quad C_{out} \quad C_{out} \quad C_{out} \]

10/24/2005  VLSI Design I; A. Milenkovic  55