EECS 151/251A
Spring 2019
Digital Design and Integrated Circuits

Instructors: Wawrzynek

Lecture 9
CMOS abstraction
MOSFET (Metal Oxide Semiconductor Field Effect Transistor).

The gate acts like a capacitor. A high voltage on the gate attracts charge into the channel. If a voltage exists between the source and drain a current will flow. In its simplest approximation, the device acts like a switch.
CMOS Transistors – State-of-the-Art
Drain versus Source - Definition

MOS transistors are symmetrical devices (Source and drain are interchangeable)

Source is the node w/ the lowest voltage
MOS Transistor as a Resistive Switch

Let’s look beneath the abstraction:
origins of $R_{on}$ and $V_T$
Transistor “resistance”

- Actually, nonlinear I/V characteristic:

- Linearizing makes all delay and power calculations simple (usually just 1st order ODEs):
MOSFET Threshold Voltage

Transistor “turns on” when $V_{gs}$ is $> V_t$.

- $I_{ds}$
- $V_d$
- $V_s$
- $V_{g}$
- $I_{ds}$
- $V_{gs}$
- $V_{dd}$
- $I_{off} = 0$ ???
- $1.2 \text{ mA} = I_{on}$
- $0.25 = V_t$
- $0.7 = V_{dd}$

- $0.7 = V_{dd}$
ON/OFF Switch Model of MOS Transistor

\[ |V_{GS}| < |V_T| \]

\[ |V_{GS}| \geq |V_T| \]
Plot on a “Log” Scale to See “Off” Current

Process engineers can:

- Increase $I_{on}$ by lowering $V_t$ - but that raises $I_{off}$
- Decrease $I_{off}$ by raising $V_t$ - but that lowers $I_{on}$.

$0.25 \approx V_t$

$1.2 \text{ mA} = I_{on}$

$I_{off} \approx 10 \text{ nA}$

$0.7 = V_{dd}$
A More Realistic Switch

Transistors in the sub 100 nm age

\[ |V_{GS}| < |V_T| \]

\[ |V_{GS}| > |V_T| \]
NMOS Transistor

$V_{GS} > 0$

$Y = Z$ if $X = 1$
A Complementary Switch

Y = Z if X = 0

PMOS Transistor

Source is the node w/ the highest voltage!
The CMOS Inverter: A First Glance

Represents the sum of all the capacitance at the output of the inverter and everything to which it connects: (drains, interconnections gate capacitance of next gate(s))
The Switch Inverter
First-Order DC Analysis*

\[ V_{OL} = 0 \]
\[ V_{OH} = V_{DD} \]

*First-order means we will ignore Capacitance.
Switch logic
Static Logic Gate

- At every point in time (except during the switching transients) each gate output is connected to either $V_{DD}$ or $V_{GND}$ via a low resistive path.

- The output of the gate assumes at all times the value of the Boolean function implemented by the circuit (ignoring, once again, the transient effects during switching periods).

Example: CMOS Inverter
Building logic from switches

**Series**

\[ Y = X \text{ if } A \text{ AND } B \]

**Parallel**

\[ Y = X \text{ if } A \text{ OR } B \]

(output undefined if condition not true)
Logic using inverting switches

**Series**

\[ Y = X \text{ if } \overline{A} \text{ AND } \overline{B} \]
\[ = \overline{A + B} \]

**Parallel**

\[ Y = X \text{ if } \overline{A} \text{ OR } \overline{B} \]
\[ = \overline{AB} \]

(output undefined if condition not true)
Static Complementary CMOS

PUN and PDN are dual logic networks
PUN and PDN functions are complementary

Inverting switches
\[ F(In_1, In_2, \ldots, In_N) \]

Non-Inverting switches
PUN is the **dual** to PDN  
(can be shown using DeMorgan’s Theorems)

\[
\overline{A + B} = \overline{AB} \\
\overline{AB} = \overline{A + B}
\]

Static CMOS gates are always inverting

\[\text{AND} = \text{NAND} + \text{INV}\]
Example Gate: NAND

- **PDN:** $G = AB \Rightarrow$ Conduction to GND
- **PUN:** $F = \overline{A} + \overline{B} = \overline{AB} \Rightarrow$ Conduction to $V_{DD}$
- $G(\text{In}_{1}, \text{In}_{2}, \text{In}_{3}, \ldots) \equiv F(\overline{\text{In}_{1}}, \overline{\text{In}_{2}}, \overline{\text{In}_{3}}, \ldots)$

Truth Table of a 2 input NAND 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>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>
Example Gate: NOR

Truth Table of a 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>

OUT = \overline{A+B}
Complex CMOS Gate

\[ \text{OUT} = D + A \cdot (B + C) \]

\[ \text{OUT} = D \cdot A + B \cdot C \]

Diagram of the Complex CMOS Gate with inputs A, B, C, and D, and output OUT.
Non-inverting logic

Why is this a bad idea?

PUN and PDN are dual logic networks
PUN and PDN functions are complementary
Switch Limitations

![Diagram of switch limitations]

\[ V_{DD} \rightarrow V_{DD} \]

Good 1

\[ V_{DD} \rightarrow 0 \]

Good 0

\[ 0 \rightarrow V_{DD} \]

Bad 1

\[ V_{DD} \rightarrow |V_{Tp}| \]

Bad 0

Tough luck ...
Transmission Gates

- Transmission gates are the way to build “switches” in CMOS.
- In general, both transistor types are needed:
  - nFET to pass zeros.
  - pFET to pass ones.
- The transmission gate is bi-directional (unlike logic gates).

Does not directly connect to Vdd and GND, but can be combined with logic gates or buffers to simplify many logic structures.
Transmission-gate Multiplexor

2-to-1 multiplexor:
\[ c = sa + s'b \]

Switches simplify the implementation:

Compare the cost to logic gate implementation.

Care must be taken to not string together many pass-transistor stages. Occasionally, need to “rebuffer” with static gate.
4-to-1 Transmission-gate Mux

- The series connection of pass-transistors in each branch effectively forms the AND of s1 and s0 (or their complement).

- Compare cost to logic gate implementation

Any better solutions?
Alternative 4-to-1 Multiplexor

- This version has less delay from in to out.
- In both versions, care must be taken to avoid turning on multiple paths simultaneously (shorting together the inputs).
Tri-state Buffers

Tri-state Buffer:

<table>
<thead>
<tr>
<th>OE</th>
<th>IN</th>
<th>OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Z</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Z</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

“high impedance” (output disconnected)

Variations:

Inverting buffer

Inverted enable

transmission gate useful in implementation
Tri-state Buffers

Tri-state buffers are used when multiple circuits all connect to a common wire. Only one circuit at a time is allowed to drive the bus. All others “disconnect” their outputs, but can “listen”.

Tri-state buffers enable “bidirectional” connections.
Tri-state Based Multiplexor

Multiplexor

If \( s=1 \) then \( c=a \) else \( c=b \)

Transistor Circuit for inverting multiplexor:
Latches and Flip-flops

Positive Level-sensitive latch:

Positive Edge-triggered flip-flop built from two level-sensitive latches:

Latch Implementation:
Summary: Complimentary CMOS Properties

- Full rail-to-rail swing
- Besides leakage (due to $I_{off}$), no static power dissipation
- Direct path current during switching