3. Implementing Logic in CMOS

Jacob Abraham

Department of Electrical and Computer Engineering
The University of Texas at Austin

VLSI Design
Fall 2020

September 3, 2020
N- and P-channel **Networks**

- N- and P-channel networks implement logic functions
  - Each network connected between **Output** and $V_{DD}$ or $V_{SS}$
  - Function defines path between the terminals

Parallel network: "OR" function
Series network: "AND" function
Straightforward way of constructing static CMOS circuits is to implement dual N- and P- networks

- N- and P- networks must implement *complementary* functions
- Duality *sufficient* for correct operation (but not necessary)
Example: \[ F = (A \cdot B) + (C \cdot D) \]

1. Take uninverted function \[ F = (A \cdot B) + (C \cdot D) \] and derive N-network
2. Identify \textit{AND}, \textit{OR} components: \( F \) is \textit{OR} of \( AB, CD \)
3. Make connections of transistors
4. Construct P-network by taking complement of N-expression \((AB + CD)\), which gives the expression, 
\((\overline{A} + \overline{B}) \cdot (\overline{C} + \overline{D})\)

5. Combine P and N circuits
Layout of Complex Gate

AND-OR-INVERT (AOI) gate

Note: Arbitrary shapes are not allowed in some nanoscale design rules
Example of Compound Gate

\[ F = (A + B + C) \cdot D \]

Note:

N- and P- graphs are duals of each other

In this case, the function is the complement of the switching function between \( F \) and GND

Question: Does it make any difference to the function if the transistor with input \( D \) is connected between the parallel \( A, B, C \), transistors and GND?

What about the electrical behavior?
Example of More Complex Gate

\[
\text{OUT} = (A + B) \cdot (C + D) \cdot (E + F + (G \cdot H))
\]
Example of More Complex Gate

\[
\text{OUT} = (A + B) \cdot (C + D) \cdot (E + F + (G \cdot H))
\]
Exclusive-NOR Gate in CMOS

Note: designs such as these should be checked very carefully for correct behavior using circuit simulation
Pseudo nMOS Logic

Based on the old NMOS technology where a “depletion” transistor was used as a pullup resistor

What happens when there is no path from Z to ground (i.e., $Z = 1$)?

What happens when there is a path from Z to ground (i.e., $Z = 0$)?
Functions realized by N and P networks must be **complementary**, and one of them must conduct for every input combination.

\[ F = a \cdot b + \overline{a} \cdot \overline{b} + a \cdot c + c \cdot d + \overline{c} \cdot \overline{d} \]

The N and P networks are **NOT** duals, but the **switching functions** they implement are complementary.
Example of Another Complex CMOS Gate

This circuit does not have a pMOS network – just one transistor for each function; it will work only if F and G are complements of each other. Why?

Can evaluate the voltages at F and G (\{0,V_{DD}\}) for each value of \(x, y, \) and \(z\)

\[
F = x \cdot \overline{y} \cdot z + \overline{x} \cdot y \cdot z + x \cdot y \cdot \overline{z} + x \cdot y \cdot z
\]

\[
G = \overline{x} \cdot y \cdot \overline{z} + x \cdot \overline{y} \cdot \overline{z} + \overline{x} \cdot \overline{y} \cdot z + \overline{x} \cdot \overline{y} \cdot \overline{z}
\]
Example of Another Complex CMOS Gate

This circuit does not have a pMOS network – just one transistor for each function; it will work only if $F$ and $G$ are complements of each other. Why?

Can evaluate the voltages at $F$ and $G$ ($\{0, V_{DD}\}$) for each value of $x, y,$ and $z$

\[
F = x \cdot \overline{y} \cdot z + \overline{x} \cdot y \cdot z + x \cdot y \cdot \overline{z} + x \cdot y \cdot z
\]

\[
G = \overline{x} \cdot y \cdot \overline{z} + x \cdot \overline{y} \cdot \overline{z} + \overline{x} \cdot y \cdot z + \overline{x} \cdot \overline{y} \cdot \overline{z}
\]
Can also follow every path from F and G to GND and identify values of \(x, y,\) and \(z\) which will enable the path to be enabled.

\[
\overline{F} = x \cdot \overline{y} + x \cdot \overline{y} \cdot \overline{z} + \overline{x} \cdot y \cdot \overline{z}
\]
\[
F = (x + y) \cdot (\overline{x} + y + z) \cdot (x + \overline{y} + z)
\]
\[
= (y + x \cdot z) \cdot (x + \overline{y} + z)
\]
\[
= x \cdot y + y \cdot z + x \cdot z
\]

\[
G = x \cdot \overline{y} \cdot \overline{z} + x \cdot \overline{y} \cdot \overline{z} + \overline{x} \cdot \overline{y}
\]
\[
= \overline{x} \cdot \overline{y} + \overline{x} \cdot \overline{z} + \overline{y} \cdot \overline{z}
\]

Can you describe the functions in simple terms?
(Hint: look at the number of input variables which are true (or false) when the output is 1.)
Can also follow every path from F and G to GND and identify values of x, y, and z which will enable the path to be enabled.

\[
\overline{F} = \overline{x} \cdot \overline{y} + x \cdot \overline{y} \cdot \overline{z} + \overline{x} \cdot y \cdot \overline{z}
\]
\[
F = (x + y) \cdot (\overline{x} + y + z) \cdot (x + \overline{y} + z)
\]
\[
= (y + x \cdot z) \cdot (x + \overline{y} + z)
\]
\[
= x \cdot y + y \cdot z + x \cdot z
\]
\[
G = \overline{x} \cdot y \cdot \overline{z} + x \cdot \overline{y} \cdot \overline{z} + \overline{x} \cdot \overline{y}
\]
\[
= \overline{x} \cdot \overline{y} + \overline{x} \cdot \overline{z} + \overline{y} \cdot \overline{z}
\]

Can you describe the functions in simple terms? (Hint: look at the number of input variables which are true (or false) when the output is 1.)
Can also follow every path from F and G to GND and identify values of \(x, y,\) and \(z\) which will enable the path to be enabled.

\[
\begin{align*}
\overline{F} &= \overline{x} \cdot \overline{y} + x \cdot \overline{y} \cdot \overline{z} + \overline{x} \cdot y \cdot \overline{z} \\
F &= (x + y) \cdot (\overline{x} + y + z) \cdot (x + \overline{y} + z) \\
&= (y + x \cdot z) \cdot (x + \overline{y} + z) \\
&= x \cdot y + y \cdot z + x \cdot z \\
G &= \overline{x} \cdot y \cdot \overline{z} + x \cdot \overline{y} \cdot \overline{z} + \overline{x} \cdot \overline{y} \\
&= \overline{x} \cdot \overline{y} + \overline{x} \cdot \overline{z} + \overline{y} \cdot \overline{z}
\end{align*}
\]

Can you describe the functions in simple terms? (Hint: look at the number of input variables which are true (or false) when the output is 1.)
Voltages represent digital logic values

- **Strength** of signal:
  - How close it approximates ideal voltage
- $V_{DD}$ and $GND$ rails are strongest 1 and 0
- nMOS transistors pass a strong 0
  - But degraded or weak 1
- pMOS transistors pass a strong 1
  - But degraded or weak 0

Therefore, nMOS transistors are best for the “pull-down” network
Pass Transistors and Transmission Gates

Transistors can be used as switches; however, they could produce degraded outputs

![Diagram of transistor operation]

Transmission gates pass both 0 and 1 well

![Diagram of transmission gate operation]
Pass Transistor Logic

“Pull-Up” Circuit

- Used to restore degraded logic 1 from output of nMOS pass transistor
Group similar transistors, so they can be in the same well
Tristate Buffer produces Z (high impedance) when not enabled

<table>
<thead>
<tr>
<th>EN</th>
<th>A</th>
<th>Y</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>

Non-Restoring Tristate
- Transmission gate acts as a tristate buffer
- Only two transistors, but nonrestoring
- Noise on A is passed to Y
Tristate inverter produces restored output, but complements signal

\[ Y = \begin{cases} 'Z' & \text{if } EN = 0 \\
\overline{A} & \text{if } EN = 1 \end{cases} \]
How many transistors are needed? (The better design uses 3 NAND gates and 1 inverter)
Transmission Gate MUX

Nonrestoring MUX
- Uses two transmission gates ⇒ only 4 transistors

Inverting MUX – adds an inverter
- Uses compound gate AOI22
- Alternatively, a pair of tristate inverters (same thing)
A 4:1 MUX chooses one of 4 inputs using two selects
- Two levels of 2:1 MUXes
- Alternatively, four tristates
D Latch

Basic Memory Element

- When CLK = 1, latch is transparent
  - D flows through to Q like a buffer
- When CLK = 0, the latch is opaque
  - Q holds its old value independent of D

a.k.a., transparent latch or level-sensitive latch
D Latch Design: MUX chooses between D and old Q

D Latch Operation
Another common storage element

- When CLK rises, D is copied to Q
- At all other times, Q holds its value
- **positive edge-triggered flip-flop** or **master-slave flip-flop**
- Built from “master” and “slave” D latches
D Flip-Flop Operation

CLK = 0

CLK = 1

CLK

D

Q
Back-to-back flops can malfunction from clock skew

- Second flip-flop fires late
- Sees first flip-flop change and captures its result
- Called **hold-time failure** or **race condition**
Non-Overlapping Clocks

A simple way to prevent races

- This works as long as non-overlap exceeds clock skew
- Used in safe (conservative) designs
- Industry does not generally use this approach – managing skew more carefully instead
Gate Layout

Building a library of standard cells

- Layout can be time consuming
- One solution is to have layouts of commonly used functions (Inverter, NAND, OR, MUX, etc.), designed to fit together very well

Standard cell design methodology

- $V_{DD}$ and $GND$ should abut (standard height)
- Adjacent gates should satisfy design rules
- nMOS at bottom and pMOS at top
- All gates include well and substrate contacts
- One of the large industry suppliers is ARM, others include TSMC and other foundries
Examples of Standard Cell Layout

Inverter

Horizontal N-diffusion and P-diffusion strips
Vertical Polysilicon gates
Metal1 $V_{DD}$ rail at top, Metal1 $GND$ rail at bottom
32$\lambda$ by 40$\lambda$

NAND3
**Wiring Track** is the space required for a wire. Example, $4\lambda$ width, $4\lambda$ spacing from neighbor $= 8\lambda$ pitch. Transistors also consume one wiring track.

Example, **well spacing**: wells must surround transistors by $6\lambda$. Implies $12\lambda$ between opposite transistor flavors. Leaves room for one wire track.
Example of Area Estimation

Estimate area by counting wiring tracks
Multiply by 8 to express in $\lambda$

Estimating area of O3AI
Sketch a stick diagram and estimate area
Example Circuit 1

Fill in the Karnaugh map to represent the Boolean function implemented by the pass-transistor circuit.
Example Circuit 1

F(a,b,c,d,e):

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>5</th>
<th>4</th>
<th>20</th>
<th>21</th>
<th>17</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>18</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>7</td>
<td>6</td>
<td>22</td>
<td>23</td>
<td>19</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>11</td>
<td>14</td>
<td>30</td>
<td>31</td>
<td>27</td>
<td>26</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>15</td>
<td>12</td>
<td>28</td>
<td>29</td>
<td>25</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>11</td>
<td>8</td>
<td>9</td>
<td>13</td>
<td>12</td>
<td>28</td>
<td>24</td>
</tr>
</tbody>
</table>
Find the function, $F$, implemented by the following circuit

$$
\bar{A} + BC + \bar{B} \bar{C}
$$
Example Circuit 2

Find the function, F, implemented by the following circuit

\[ \overline{A} + BC + \overline{B} \overline{C} \]
Find the functions X and Y implemented by the following circuit.
Example Circuit 3

Find the functions $X$ and $Y$ implemented by the following circuit

\[ Y = A \oplus B \]
\[ X = \overline{A \oplus B} \]
Label the circuit so that it implements the function:

\[ F = a \cdot (b \cdot c + \bar{b} \cdot \bar{c}) \]