Lecture # 12  Physical Limits of Logic
• Midterm Results
• Timing Diagrams
• Capacitance Loading

http://inst.EECS.Berkeley.EDU/~ee42/

Midterm Exam #1 Results

<table>
<thead>
<tr>
<th>Ave</th>
<th>18.2</th>
<th>17.9</th>
<th>16.3</th>
<th>21.6</th>
<th>73.9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ave/Max</td>
<td>0.73</td>
<td>0.72</td>
<td>0.65</td>
<td>0.86</td>
<td>0.74</td>
</tr>
<tr>
<td>Stdev</td>
<td>8.4</td>
<td>7.9</td>
<td>7.0</td>
<td>5.6</td>
<td>22.4</td>
</tr>
<tr>
<td>Stdev/Max</td>
<td>0.34</td>
<td>0.32</td>
<td>0.28</td>
<td>0.22</td>
<td>0.22</td>
</tr>
</tbody>
</table>

These are circuits that accomplish a given logic function such as "OR". We will shortly see how such circuits are constructed. Each of the basic logic gates has a unique symbol, and there are several additional logic gates that are regarded as important enough to have their own symbol. The set is: AND, OR, NOT, NAND, NOR, and EXCLUSIVE OR.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C=A·B</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>NAND</td>
</tr>
<tr>
<td>A</td>
<td>B</td>
<td>NOR</td>
</tr>
<tr>
<td>A</td>
<td>B</td>
<td>EXCLUSIVE OR</td>
</tr>
</tbody>
</table>

With a combination of logic gates we can construct any logic function. In these two examples we will find the truth table for the circuit.

DeMorgan’s Theorem:

\[ A + B + C = \overline{A \cdot B \cdot C} \] or \[ A + B + C = \overline{A} \cdot \overline{B} \cdot \overline{C} \]

Example of Using DeMorgan’s Theorem:

Build this circuit

Each is a NAND gate

Thus any sum of products expression can be immediately synthesized from NAND gates alone.
What Are Some Limitations of Digital?

It takes a lot of bits to represent even simple audio signals and if we convert a real-time audio signal to digital, we need to transmit a lot of bits every second. Example: An ordinary audio signal is sampled every 50 µs (to achieve 10KHz frequency performance) and evaluated (converted to digital form) to an accuracy of 1 part in 10,000. Every sample requires how many bits?

\[2^{13} = 8,192 \text{ and } 2^{14} = 16,384\]

so we need 14 bits for each sample. Now we need to transmit these bits 20,000 times per second. The bit rate is 280,000 baud.

The transmission of digital signals as pulses through circuits and over transmission media leads to pulse degradation, just as analog signals are degraded. As a consequence, we must (a) detect and regenerate the signal before it gets “buried in the noise,” and (b) accept that propagation delays are intrinsic to signal flow, and take these delays into account in our design (e.g., to avoid making a control decision based on a piece of information that has not yet arrived!). The RC transient lectures treated pulse degradation in real circuits quantitatively.

A 128M MPG gives only 1 hour playback!

PHYSICAL LIMITATIONS OF LOGIC GATES

Every node in any circuit (such as the internal circuit of a NAND gate) has capacitance and all interconnects have resistance. Thus it takes time to charge these capacitances.

Thus, output of all circuits, including logic gates is delayed from input.

LOGIC GATE DELAY \(\tau_D\)

Time delay \(\tau_D\) occurs between input and output: “computation” is not instantaneous.

Value of input at \(t = 0^+\) determines value of output at later time \(t = \tau_D\)

Logical Synthesis of XOR

Inputs have different delays, but we ascribe a single worst-case delay \(\tau_D\) to every gate.

A Valve is a Transistor

Current flows when \(V_{IN}\) is high

Can be modeled by a 10kΩ resistor

Valves in Series \(\Rightarrow\) NAND

Valves in Parallel \(\Rightarrow\) NOR

Since an “On” transistor is like a resistor the delay to 50% is given by

\[\tau_D = 0.69RC\]
Timing Diagram for Delays in Logic

Action of each gate takes one gate delay

Logic level

A
B
\overline{B}
X
Y
F

EFFECT OF GATE DELAY

Cascade of Logic Gates

Inputs have different delays, but we ascribe a single worst-case delay $\tau$ to every gate

How many “gate delays for shortest path? ANSWER: 2

How many gate delays for longest path? ANSWER: 3

TIMING DIAGRAMS

Glitching: temporary switching to an incorrect value

Note $\overline{\overline{C}}$ becomes valid one gate delay after $B$ switches

Note that $\overline{B \cdot \overline{C}}$ becomes valid two gate delays after $B$ & $C$ switch, because the invert function takes one delay and the NAND function a second.

No change at $t = 3\tau$