Unit 2 – Digital Circuits (Logic)

Moving from voltages to 1's and 0's...

ANALOG VS. DIGITAL

Signal Types

- Analog signal
  - Continuous time signal where each voltage level has a unique meaning
  - Most information types are inherently analog
- Digital signal
  - Continuous signal where voltage levels are mapped into 2 ranges meaning 0 or 1
  - Possible to convert a single analog signal to a set of digital signals

Signals and Meaning

Each voltage value has unique meaning

(There is a small illegal range where meaning is undefined since threshold can vary based on temperature, small variations in manufacturing, etc.)
Analog vs. Digital

USC students used to program analog computers!

Analog vs. Digital

• Analog Advantages
  – Can be easier to build simple systems
    • AM Radio

• Digital Advantages
  – Not affected by small changes (noise) in the signal
  – Repeatable
  – The above make it easier to build large, complex systems

Mechanical Computers

• Primarily gear-based
• Two prototypes designed and partially implemented by Charles Babbage
  – Used mechanical levers, gears, and ball bearings, etc.

• __________________________
  – prototype and not fully programmable

• __________________________
  – Never completed
  – To be programmed with punch cards
  – Designed to perform 4 basic arithmetic ops. (add, sub, mul, div)

A Brief History

COMPUTERS AND SWITCHING TECHNOLOGY
**Electronic Computers**

- 1945 - ENIAC was first, fully electronic computer
- Used thousands of ______ as fundamental switching (on/off) technology
- Weighed 30 tons, required 15,000 square feet, and maximum size number was 10 decimal digits (i.e. ±9,999,999,999)
- Still required some patch panels (wire plugs) to configure it

**Vacuum Tube Technology**

- Digital, electronic computers use some sort of voltage controlled switch (on/off)
- Looks like a light bulb
- Usually 3 nodes
  - 1 node serves as the switch value allowing current to flow between the other 2 nodes (on) or preventing current flow between the other 2 nodes (off)
  - Example: if the switch input voltage is 5V, then current is allowed to flow between the other nodes

**Vacuum Tube Disadvantages**

- _____________________________
  - Especially when you need _____________________________
- _____________________________
  - Can
- _____________________________

**Transistor**

- Another switching device
- Invented by Bell Labs in 1948
- Uses semiconductor materials (silicon)
- Much smaller, faster, more reliable (doesn't burn out), and dissipated less power
Moore's Law & Transistors

• Moore's Law = Number of transistors able to be fabricated on a chip will ________________

• Transistors are the fundamental building block of computer HW
  – Switching devices: Can conduct [on = 1] or not-conduct [off = 0] based on an input voltage

How Does a Transistor Work

• Transistor inner workings
  – http://www.youtube.com/watch?v=IcrBqCFLHIY

NMOS Transistor Physics

• Transistor is started by implanting two n-type silicon areas, separated by p-type

NMOS Transistor Physics

• A thin, insulator layer (silicon dioxide or just "oxide") is placed over the silicon between source and drain
**NMOS Transistor Physics**

- A thin, insulator layer (silicon dioxide or just "oxide") is placed over the silicon between source and drain.
- Conductive polysilicon material is layered over the oxide to form the gate input.

![Diagram of NMOS Transistor Physics](image)

**NMOS Transistor Physics**

- Positive voltage (charge) at the gate input repels the extra positive charges in the p-type silicon.
- Result is a negative-charge channel between the source input and drain.

![Diagram of NMOS Transistor Physics](image)

**NMOS Transistor Physics**

- Electrons can flow through the negative channel from the source input to the drain output.
- The transistor is "on".

![Diagram of NMOS Transistor Physics](image)

**NMOS Transistor Physics**

- If a low voltage (negative charge) is placed on the gate, no channel will develop and no current will flow.
- The transistor is "off".

![Diagram of NMOS Transistor Physics](image)
**View of a Transistor**

- Cross-section of transistors on an IC
- Moore's Law is founded on our ability to keep shrinking transistor sizes
  - Gate/channel width shrinks
  - Gate oxide shrinks
- Transistor feature size is referred to as the implementation "technology node"

**DIGITAL LOGIC GATES**

**Transistors and Logic**

- Transistors act as switches (on or off)
- Logic operations (AND / OR) formed by connecting them in specific patterns
  - Series Connection
  - Parallel Connection

**Digital Logic**

- Forms the basic processing circuits for digital signals (i.e. 1's and 0's)
- Digital Logic still abstracts many of the physical issues (voltage, current, parasitics, etc.) dealt with in the study of integrated circuits
  - An alarm should sound if the key is in the ignition AND your seatbelt is NOT fastened
  - If the voltage threshold sensor rises above 3 volts create a conductive channel to excite the ignition sensor...
Gates

- Each logical operation (AND, OR, NOT) can be implemented in circuit form using the corresponding logic gate.

AND Gate

OR Gate

NOT Gate

AND Gates

- An AND gate outputs a '1' (true) if ALL inputs are '1' (true).
- Gates can have several inputs.
- Behavior can be shown in a truth table (listing all possible input combinations and the corresponding output).

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</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>1</td>
</tr>
</tbody>
</table>

2-input AND

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</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>1</td>
</tr>
</tbody>
</table>

3-input AND

OR Gates

- An OR gate outputs a '1' (true) if ANY input is '1' (true).
- Gates can also have several inputs.

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</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>1</td>
</tr>
</tbody>
</table>

2-input OR

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</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>1</td>
</tr>
</tbody>
</table>

3-input OR

NOT Gate

- A NOT gate outputs a '1' (true) if the input is '0' (false).
- Also called an "Inverter".

<table>
<thead>
<tr>
<th>X</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
NAND and NOR Gates

- Inverted versions of the AND and OR gate

\[
Z = \overline{X \cdot Y}
\]

\[
Z = \overline{X + Y}
\]

**Truth Tables**

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</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>1</td>
</tr>
</tbody>
</table>

AND

True if NOT ALL inputs are true

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</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>

OR

True if NOT ANY input is true

**XOR and XNOR Gates**

- Exclusive OR gate. Outputs a '1' if either input is a '1', but not both.

\[
Z = X \oplus Y
\]

True if an **odd** # of inputs are true

True if inputs are **different**

\[
Z = X \oplus \overline{Y}
\]

True if an **even** # of inputs are true

True if inputs are **same**

**Logic Example**

```
0  A
0  B
1  C
0  D
```

```
0  0  0
```

```
0  1  1
```

```
0  1  1
```

```
1  0  0
```

```
1  1  1
```

```
0  F
```

```
0  0
```

```
0  0
```

```
0  0
```

**Logic Example**

```
0  A
1  B
0  C
0  D
```

```
0  0
```

```
0  1
```

```
0  1
```

```
0  0
```

```
0  0
```

```
0  F
```

```
0  0
```

```
0  0
```

```
0  0
```
Delay Example

Logical Operations Summary

- All digital circuits can be described using AND, OR, and NOT
  - Note: You'll learn in future courses that digital circuits can be described with any of the other sets:
    - (AND, NOT), (OR, NOT), (NAND only), or (NOR only)
  - Normal convention: 1 = true / 0 = false
- A logic circuit takes some digital inputs and transforms each possible input combination to a desired output values

Trivia-of-the-day: The Apollo Guidance Computer that controlled the lunar spacecraft in 1969 was built out of 8,400 3-input NOR gates.

Sequencial Devices (Registers)

- AND, OR, NOT, NAND, and other gates are known as ____________
  - Outputs only depend on what the inputs are ________, not one second ago
  - This implies they have no "memory" (can't remember a value)
- Sequential logic devices provide the ability to retain or ____________ a value by itself (even after the input is changed or removed)
  - Outputs can depend on the current inputs, and previous states of the circuit (stored values.)
  - Usually have a controlling signal that indicates when the device should update the value it is remembering vs. when it should simply remember that value
  - This controlling signal is usually the ________ signal

Registers

- Registers are the most common sequential device
- Registers sample the data input (D) on the edge of a clock pulse (CP) and stores that value at the output (Q)
- Analogy: Taking a picture with your ______ when you press a button (clock pulse) the camera samples the scene (input) and __________ it as a snapshot (output)

Clock pulse

The clock pulse (positive edge) here...
Flip-Flops

- Flip-flops are the building blocks of ____________
  - 1 Flip-flop PER _____ of input/output
  - There are many kinds of flip-flops but the most common is the D- (_______) Flip-flop (a.k.a. D-FF)
- D Flip-flop triggers on the clock edge and captures the D-value at that instant and causes Q to remember it until the next edge
  - ___________ Edge: instant the clock transition from low to high (0 to 1)

Registers and Flip-flops

- A register is simply a _____ of D flip-flops that all trigger on a single clock pulse

Pulses and Clocks

- Registers need an edge to trigger
- We can generate pulses at specific times (creating an ____________ pattern) when we know the data we want has arrived
- Other registers in our hardware should trigger at a ____________ interval
- For that we use a clock signal...
  - Alternating high/low voltage pulse train
  - Controls the ordering and timing of operations performed in the processor
  - 1 _________ is usually measured from rising/positive edge to rising/positive edge
- Clock frequency (F) = # of cycles per second
- Clock Period (T) = ____________

Combinational vs. Sequential

- Sequential logic (i.e. registers) is used to store values
  - Each register is analogous to a ____________ in your software program (a variable stores a number until you need it)
- Combinational logic is used to process bits (i.e. perform operations on values
  - Analogous to operators (+,-,*) in your software program