The Digital Abstraction

Reminders:
- Lab 1 due today
- Lab 1 checkoff due 9/29
- Lab 2 due 9/30
6.004 Course Outline

- Module 1: Assembly language
  - From high-level programming languages to the language of the computer

- Module 2: Digital design
  - Combinational and sequential circuits

- Module 3: Computer architecture
  - Simple and pipelined processors
  - Caches and the memory hierarchy

- Module 4: Computer systems
  - Operating system and virtual memory
  - Parallelism and synchronization
The Digital Abstraction

Building Digital Systems in an Analog World
Analog vs. Digital Systems

- Analog systems represent and process information using **continuous signals**
  - e.g., voltage, current, temperature, pressure, ...

- Digital systems represent and process information using **discrete symbols**
  - Typically binary symbols (bits)
  - Encoded using ranges of a physical quantity (e.g., voltage)

Digital systems tolerate noise
Example: Analog Audio Equalizer

Input: Voltage signal representing sound pressure

Expected output: Frequency-equalized voltage signal
Example: Analog Audio Equalizer

Input: Voltage signal representing sound pressure

Filters

- Bass gain
- Mid gain
- Treble gain

Bass
Mid
Treble

Expected output: Frequency-equalized voltage signal

Does output match expected output?  Not quite!

Why or why not?

- Noise
- Manufacturing variations
- Components degrade over time

...
The Digital Abstraction

Keep in mind that the world is not digital, we would simply like to engineer it to behave that way. In the end we must use real physical phenomena to implement digital designs!
Using Voltages “Digitally”

- Key idea: Encode two symbols, “0” and “1” (1 bit)
- Use the same convention for every component and wire in our digital system

Attempt #1:

![Diagram showing incorrect voltage levels](image)

Not quite correct. Why? Hard to distinguish \( V_{TH} - \epsilon \) from \( V_{TH} + \epsilon \)

Attempt #2:

![Diagram showing correct voltage levels](image)

✓ ?
Will This System Work?

Upstream device transmits a signal at $V_L - \varepsilon$, a valid “0”. Noise on the wire causes the downstream device to receive $V_L + \varepsilon$, which is undefined.

**How can we address this?**

**Output voltages should use narrower ranges, so that signal will still be valid when it reaches an input even if there is noise.**
Noise Margins

Proposed fix: Different specifications for inputs and outputs

- Digital output: “0” ≤ \( V_{OL} \), “1” ≥ \( V_{OH} \)
- Digital input: “0” ≤ \( V_{IL} \), “1” ≥ \( V_{IH} \)
- \( V_{OL} < V_{IL} < V_{IH} < V_{OH} \)

A digital device accepts marginal inputs and provides unquestionable outputs (to leave room for noise).
Digital Systems are Restorative

Analog systems: Noise accumulates

\[ V_I \xrightarrow{\varepsilon_1} V_I + \varepsilon_1 \xrightarrow{f} f(V_I + \varepsilon_1) \xrightarrow{\varepsilon_2} f(V_I + \varepsilon_1) + \varepsilon_2 \xrightarrow{g} g(f(V_I + \varepsilon_1) + \varepsilon_2) \]

Digital systems: Noise is canceled at each stage

\[ V_I \xrightarrow{\varepsilon_1} V_I + \varepsilon_1 \xrightarrow{f} f(V_I) \xrightarrow{\varepsilon_2} f(V_I) + \varepsilon_2 \xrightarrow{g} g(f(V_I)) \]

Intuitively, canceling noise requires *active components*, i.e., components that inject energy into the system.
Buffer: A simple digital device that copies its input value to its output

Voltage Transfer Characteristic (VTC): Plot of $V_{out}$ vs. $V_{in}$ where each measurement is taken after any transients have died out.

Note: VTC does not tell you anything about how fast a device is — it measures static behavior, not dynamic behavior.

VTC must avoid the shaded regions (aka “forbidden zones”), which correspond to valid inputs but invalid outputs.
Voltage Transfer Characteristic

1) Note the center white region is taller than it is wide ($V_{OH} - V_{OL} > V_{IH} - V_{IL}$). Net result: device must have $\text{GAIN} > 1$ and thus be $\text{ACTIVE}$

2) Note the VTC can do anything when $V_{IL} < V_{IN} < V_{IH}$
Types of Digital Circuits

- **Combinational circuits**
  - Do not have memory
  - Each output is a function of current input values
  - Examples:

- **Sequential circuits**
  - Have memory, i.e., state
  - Each output depends on current state + current inputs

Digital inputs → Combinational Circuits → Digital outputs

Inverter:

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

AND:

Output is 1 if both inputs are 1, 0 otherwise.
Introduction to Combinational Circuits
A combinational device is a circuit element that has

- one or more digital inputs
- one or more digital outputs
- a functional specification that details the value of each output for every possible combination of valid input values
- a timing specification consisting (at a minimum) of a propagation delay ($t_{PD}$): an upper bound on the required time to produce valid, stable output values from an arbitrary set of valid, stable input values

### Static discipline

Input $A$

Input $B$

Output $Y$

Input $C$

Output a "1" if at least 2 out of 3 of my inputs are a "1". Otherwise, output "0".

I will generate a valid output in no more than 2 nanoseconds after seeing valid inputs.
A set of interconnected elements is a combinational device if

– each circuit element is combinational
– every input is connected to exactly one output or to a constant (0 or 1)
– the circuit contains no directed cycles
Example: Is This a Combinational Device?

A, B, and C are combinational devices. Is the following circuit a combinational device?

Does it have digital inputs? Yes
Does it have digital outputs? Yes
Can you derive a functional description?

\[ W = f_C(f_A(X, Y), f_B(f_A(X, Y), Z)) \]

Can you derive a \( t_{PD} \)?

\[ t_{PD} = t_{PD,A} + t_{PD,B} + t_{PD,C} \]
There are many ways to specify the function of a combinational device.

We will use two systematic approaches:

- **Truth tables** enumerate the output values for all possible combinations of input values.
- **Boolean expressions** are equations containing binary (0/1) variables and three operations: AND (\(\cdot\)), OR (\(+\)), and NOT (overbar).

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

Any combinational function can be specified as a truth table or Boolean expression (next lecture).
Timing Specifications

- **Propagation delay ($t_{PD}$):** An upper bound on the delay from valid inputs to valid outputs

![Graph showing propagation delay]

Goal: Minimize $t_{PD}$!

- **Contamination delay ($t_{CD}$):** A lower bound on the delay from invalid inputs to invalid outputs
  - Used later (for sequential logic), can ignore for now

September 23, 2021
The Combinational Contract

A → B

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

t\text{PD} \quad \text{propagation delay}

t\text{CD} \quad \text{contamination delay}

Must be \[ \geq t_{\text{CD}} \]

Must be \[ \leq t_{\text{PD}} \]

No promises during \[ \text{xxxxx} \]
Summary

- Digital systems tolerate noise
- Digital encoding
  - Valid voltage levels for representing “0” and “1”
  - Undefined range avoids mistaking “0” for “1” and vice versa
  - Noise margins require tougher standards for outputs than for inputs
- Combinational devices
  - Have Tinkertoy-set simplicity, modularity
  - Predictable composition: “parts work → whole thing works”
  - Must obey static discipline
    - Digital inputs & outputs: restores marginal input voltages
    - Complete functional specification
    - Valid inputs lead to valid outputs in bounded time
Thank you!

Next lecture:
Boolean Algebra