The Digital Abstraction

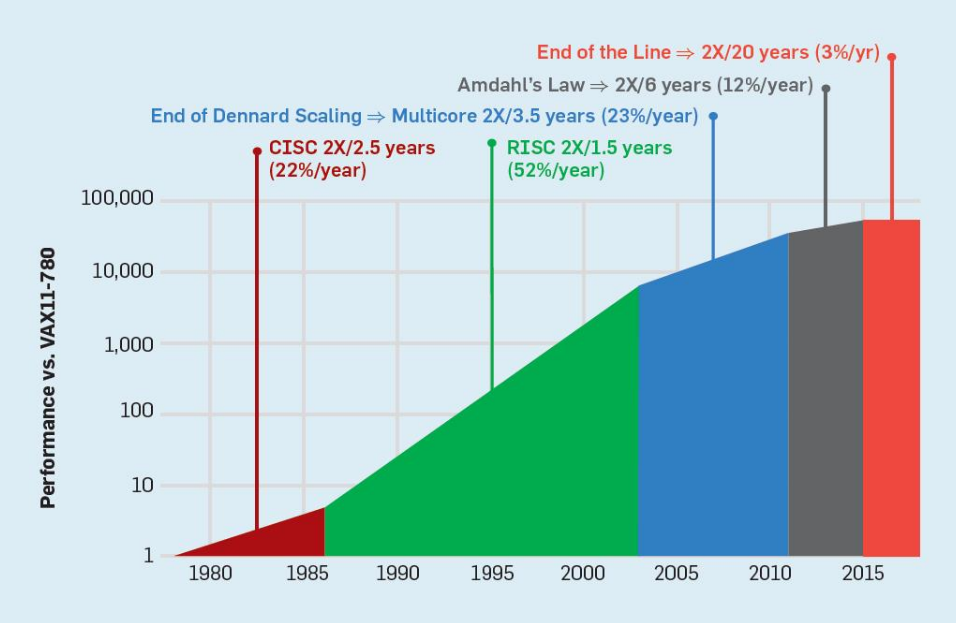
Reminders:
- No recitation tomorrow
- Lab 1 due today
- Sign up for Lab 1 checkoff meeting

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

Why Learn Hardware Design?

- Designing cutting-edge systems requires intimate knowledge of hardware
- Chip fabrication technology is reaching its limits → more companies are building custom hardware



Hennessy and Patterson, Turing Award Lecture CACM 2019, https://dl.acm.org/citation.cfm?id=3282307

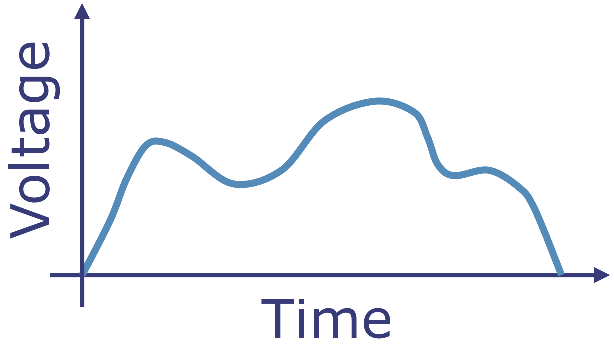
- You are likely to use and design specialized hardware, not just general-purpose processors!

The Digital Abstraction

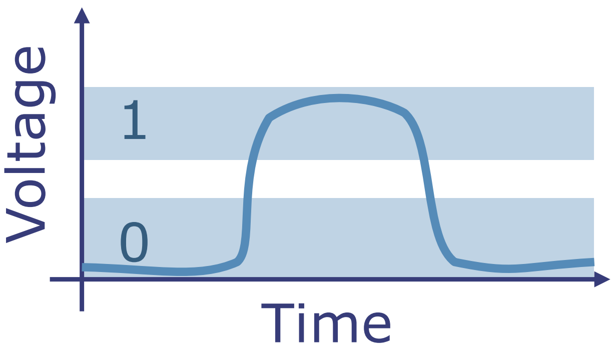
Building Digital Systems in an Analog World

Analog vs. Digital Systems

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



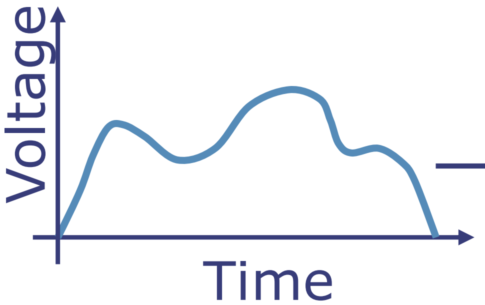
- Digital systems represent and process information using discrete symbols
  - Typically binary symbols (bits)
  - Encoded using ranges of a physical quantity (e.g., voltage)



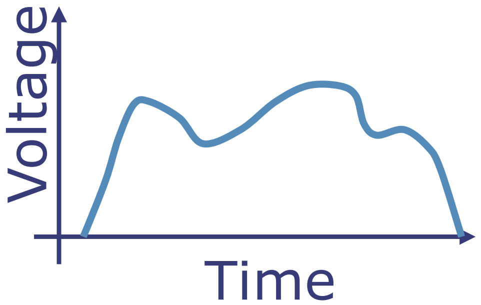
Digital systems tolerate noise

Example: Analog Audio Equalizer

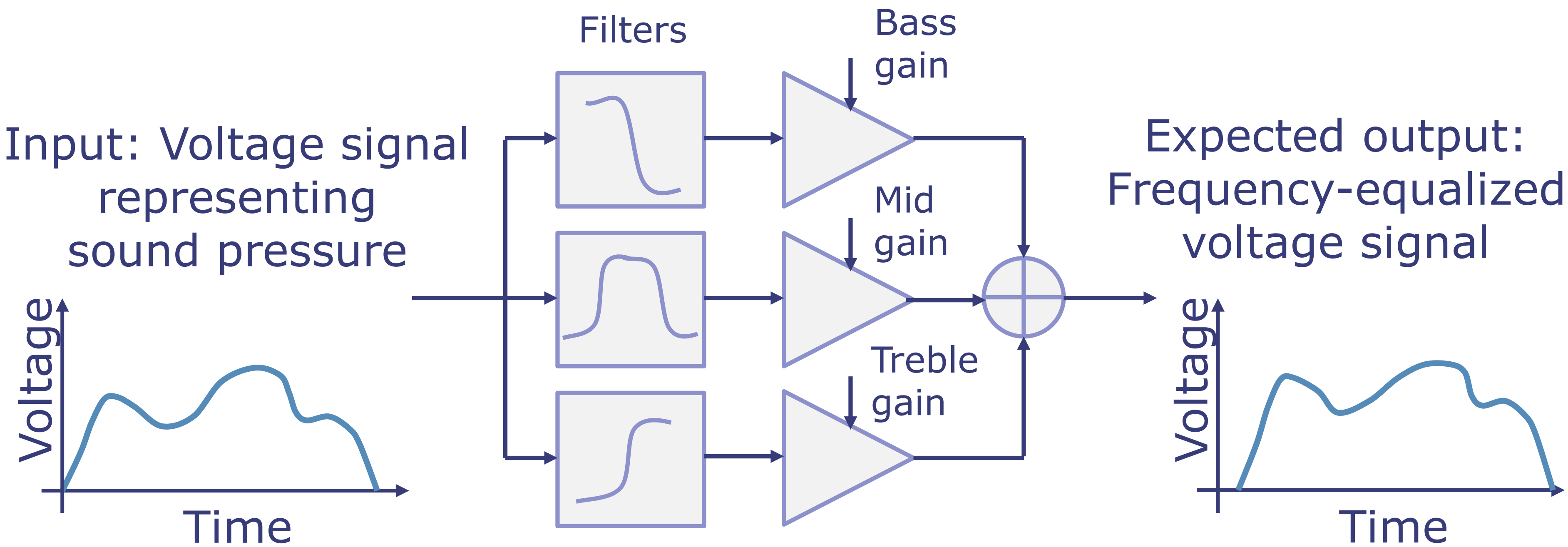
Input: Voltage signal representing sound pressure



Expected output: Frequency-equalized voltage signal



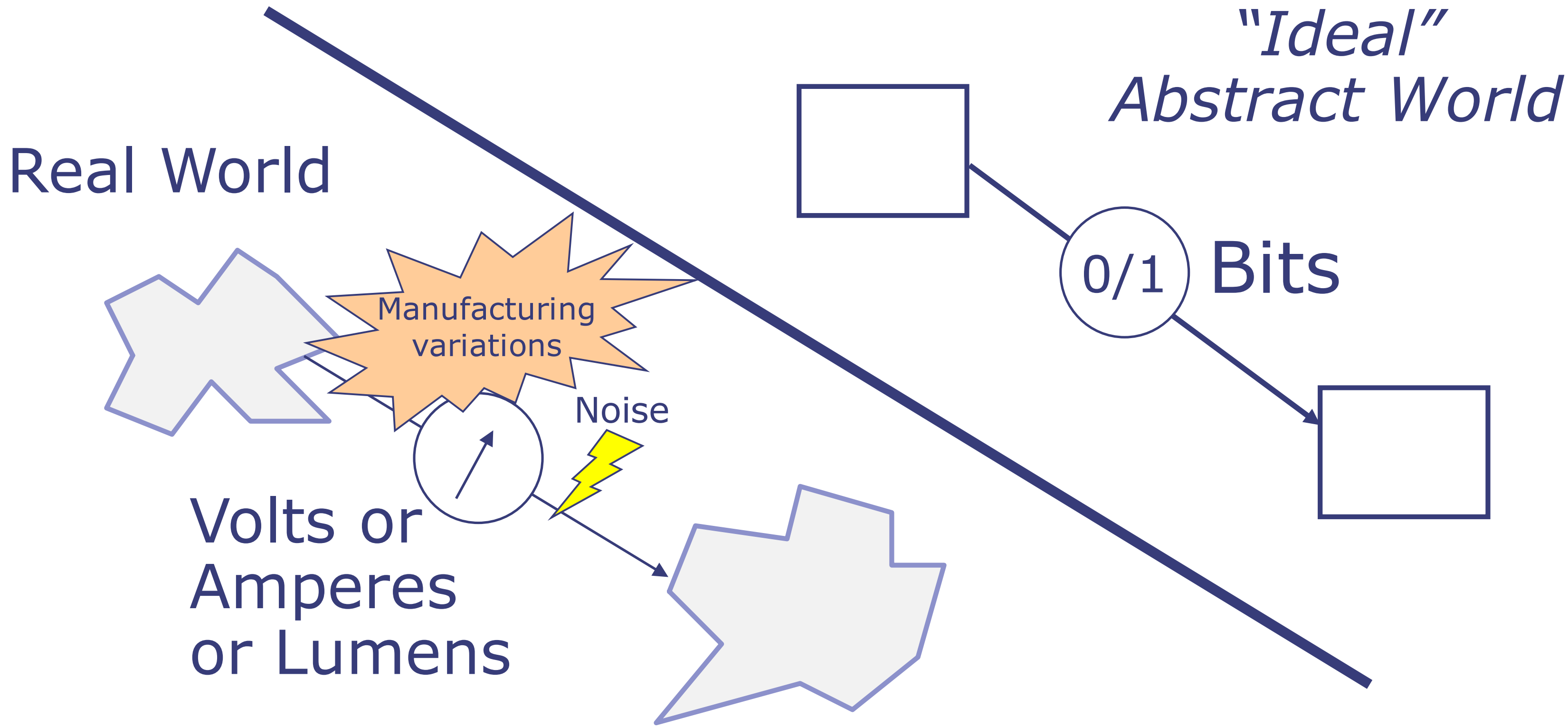
Example: Analog Audio Equalizer



Does output match expected output? *Not quite!*

Why or why not? *Noise*
Manufacturing variations
Components degrade over time
…

The Digital Abstraction

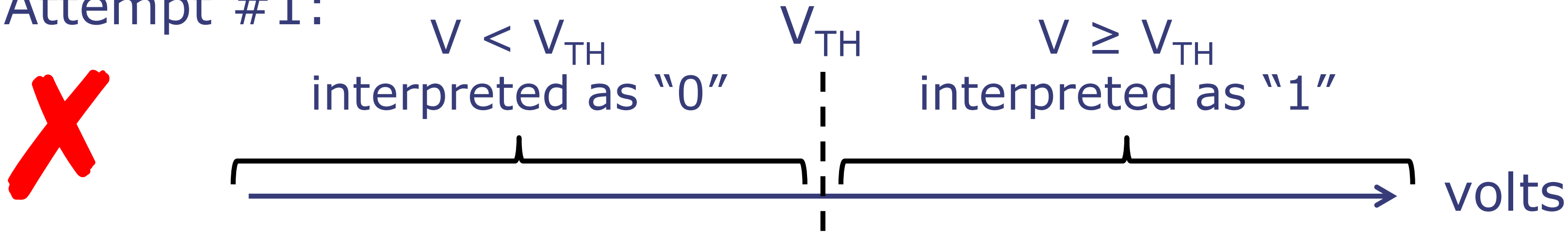


Keep in mind that the world is not digital, we would simply like to engineer it to behave that way. In the end we must use real physical phenomena to implement digital designs!

Using Voltages “Digitally”

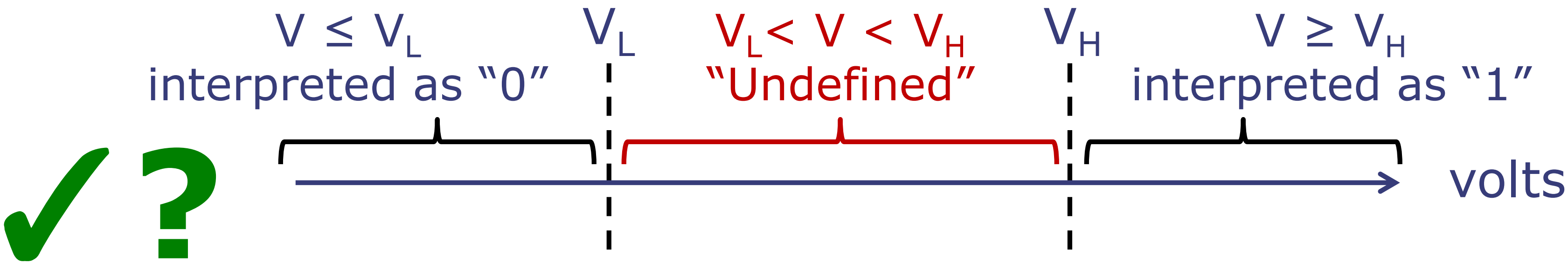
- Key idea: Encode two symbols, “0” and “1” (1 bit)
- Use the same convention for *every* component and wire in our digital system

Attempt #1:

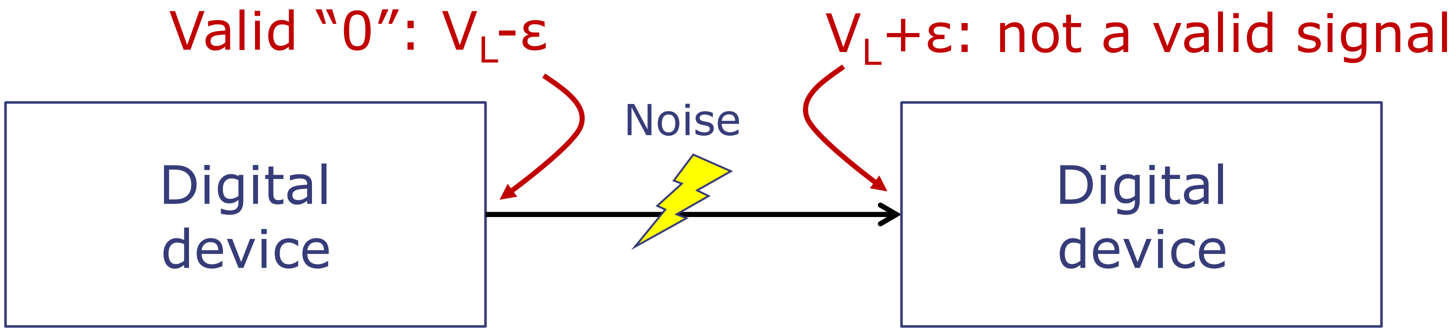


Not quite correct. Why? *Hard to distinguish* $V_{TH}-\varepsilon$ *from* $V_{TH}+\varepsilon$

Attempt #2:



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.

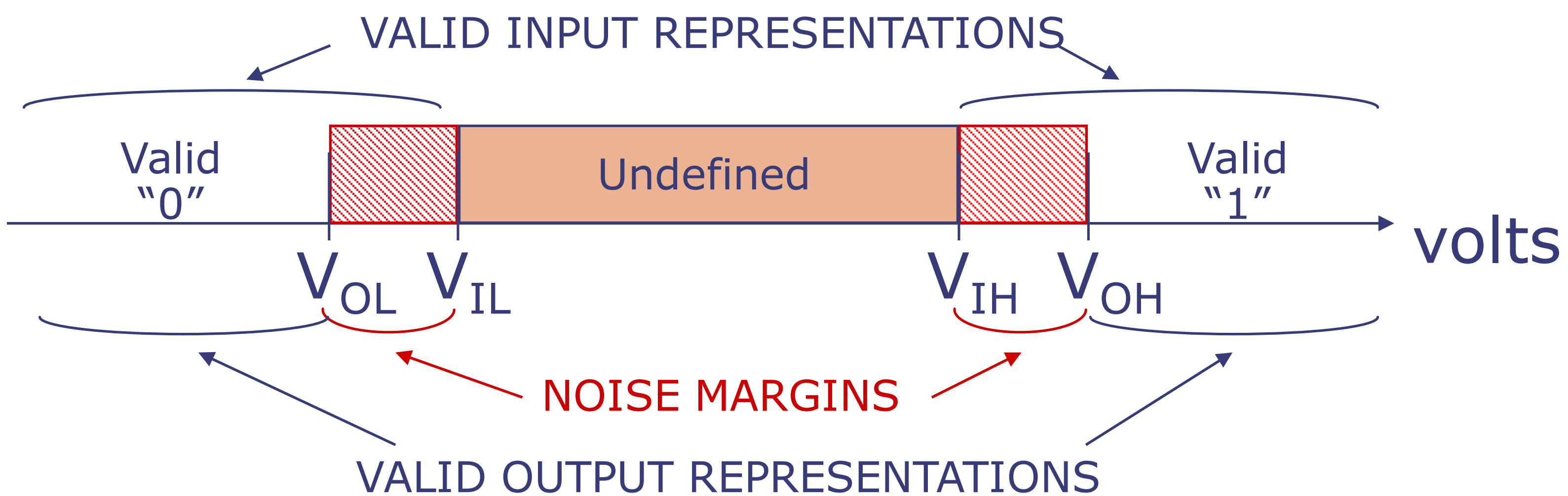
How can we address this?

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

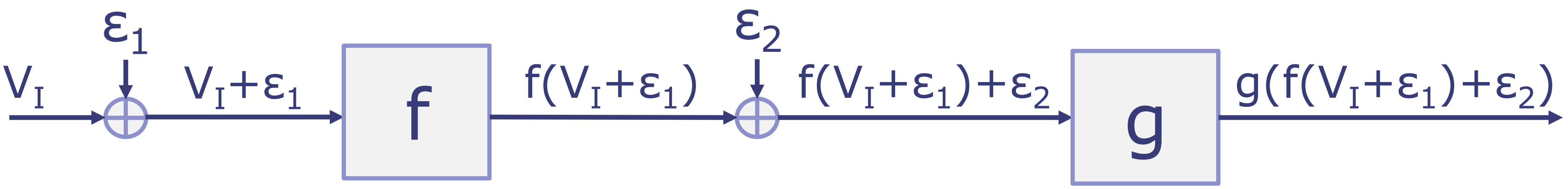
- Digital output: "0" ≤ V_{OL}, "1" ≥ V_{OH}
- Digital input: "0" ≤ V_{IL}, "1" ≥ V_{IH}
- $\mathbf{V_{OL} < V_{IL} < V_{IH} < V_{OH}}$



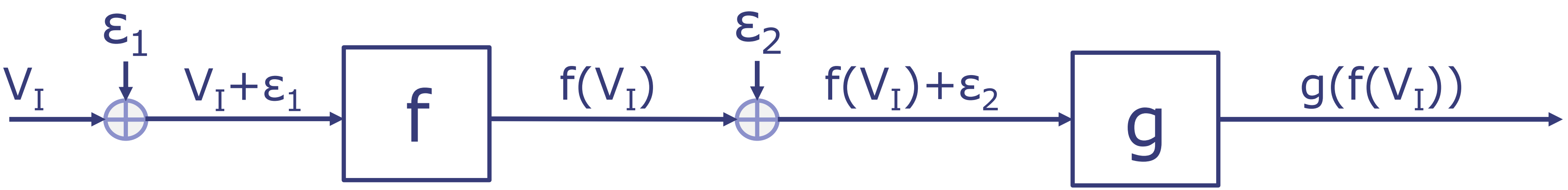
A digital device accepts marginal inputs and provides unquestionable outputs (to leave room for noise).

Digital Systems are Restorative

Analog systems: Noise accumulates



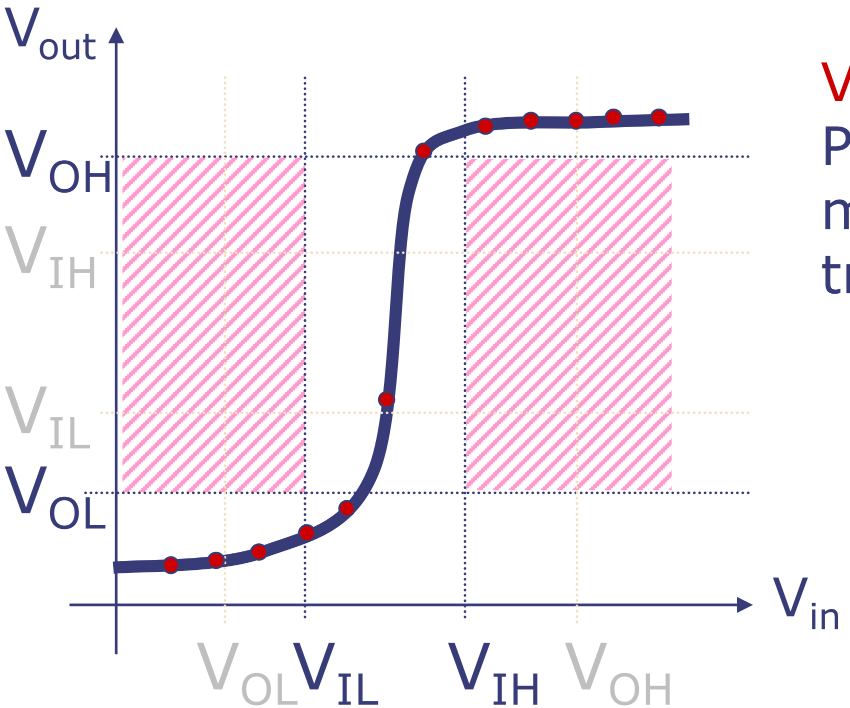
Digital systems: Noise is canceled at each stage



Intuitively, canceling noise requires *active components*, i.e., components that inject energy into the system

Voltage Transfer Characteristic

Buffer: A simple digital device that copies its input value to its output

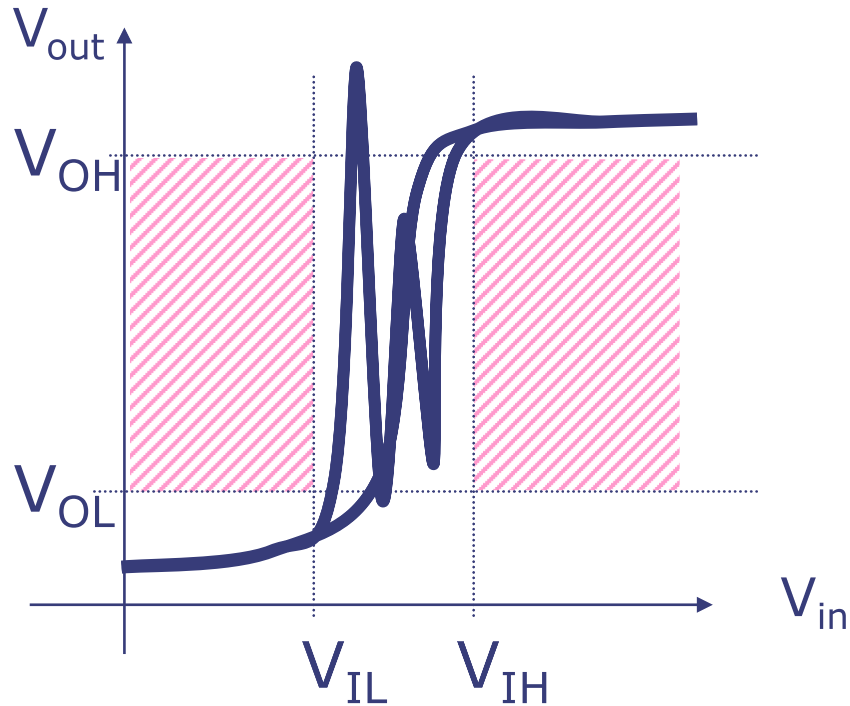


Voltage Transfer Characteristic (VTC): Plot of V_{out} vs. V_{in} where each measurement is taken after any transients have died out.

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

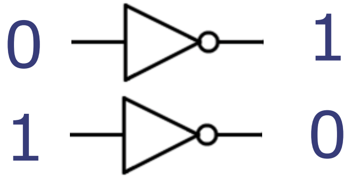


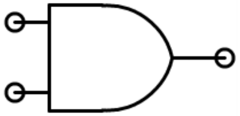
1) Note the center white region is taller than it is wide (V_{OH}-V_{OL} > V_{IH}-V_{IL}). Net result: device must have GAIN > 1 and thus be ACTIVE

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: Inverter AND





Output is 1 if both inputs are 1, 0 otherwise

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

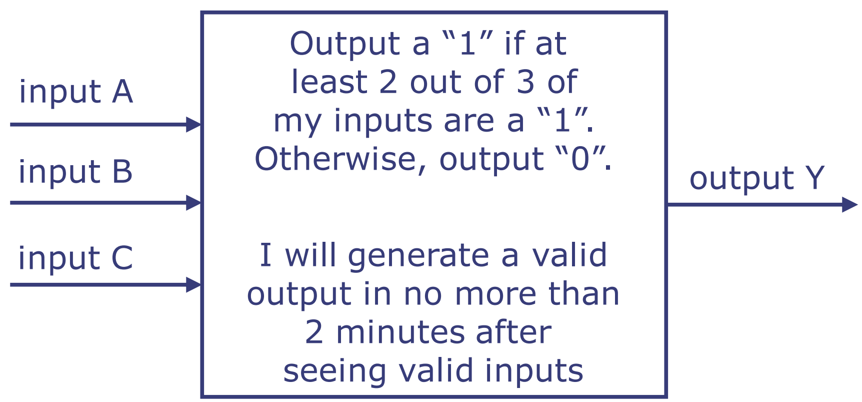
Introduction to Combinational Circuits

Combinational Devices

A <u>combinational device</u> is a circuit element that has

Static discipline

- 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



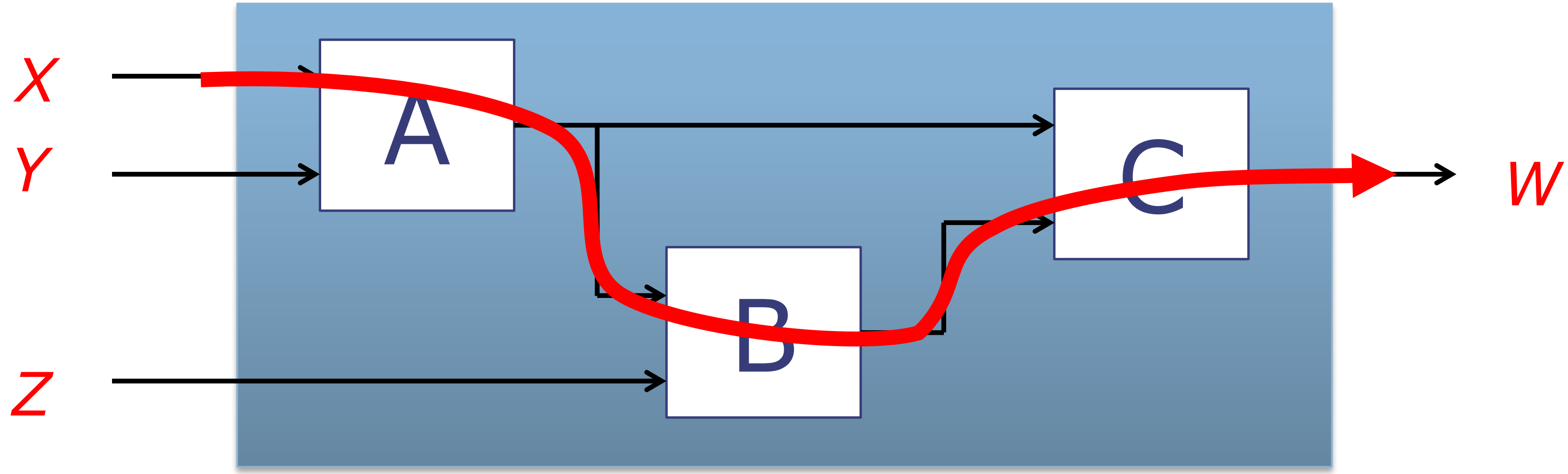
Composing Combinational Devices

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?



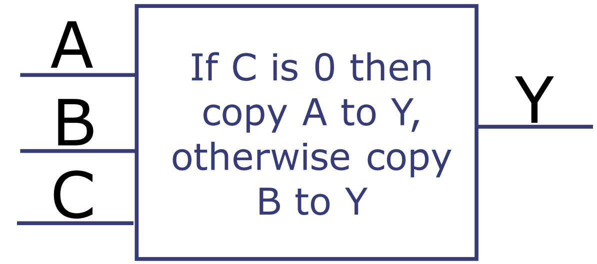
Does it have digital inputs? Yes
Does it have digital outputs? Yes
Can you derive a functional description?

$$W = f_C(f_A(X, Y), f_B(f_A(X, Y), Z))$$

Can you derive a t_{PD}? $t_{PD} = t_{PD,A} + t_{PD,B} + t_{PD,C}$

Functional Specifications

- 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 (·), OR (+), and NOT (overbar)

$$Y = \bar{C} \cdot A + C \cdot B$$

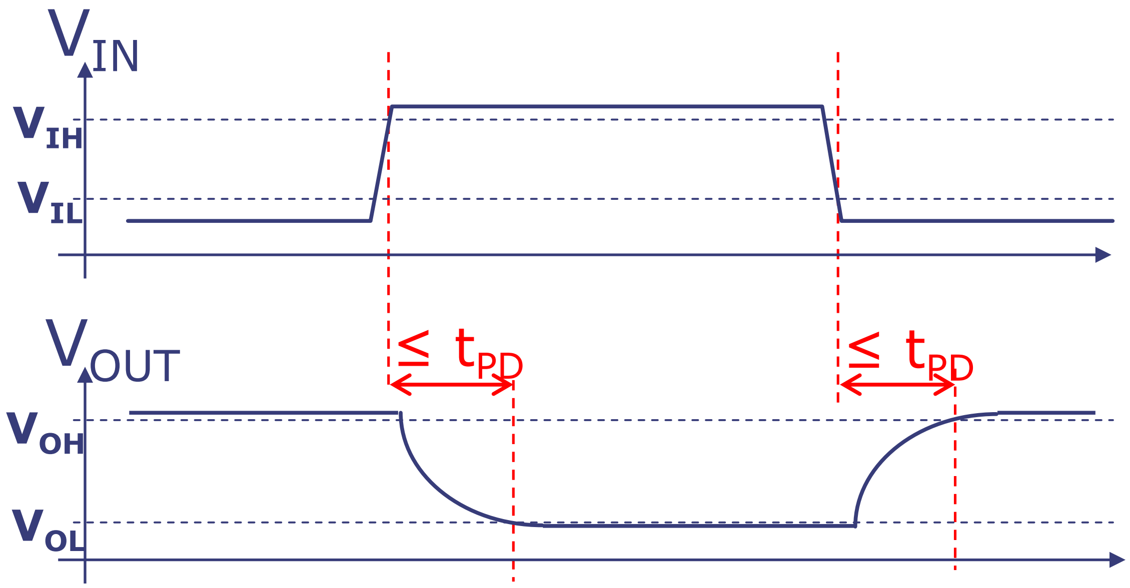
Truth Table

C	**B**	**A**	**Y**
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

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

Timing Specifications

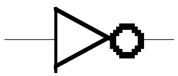
- Propagation delay (t_{PD}): An upper bound on the delay from valid inputs to valid outputs



Goal: Minimize t_{PD} !

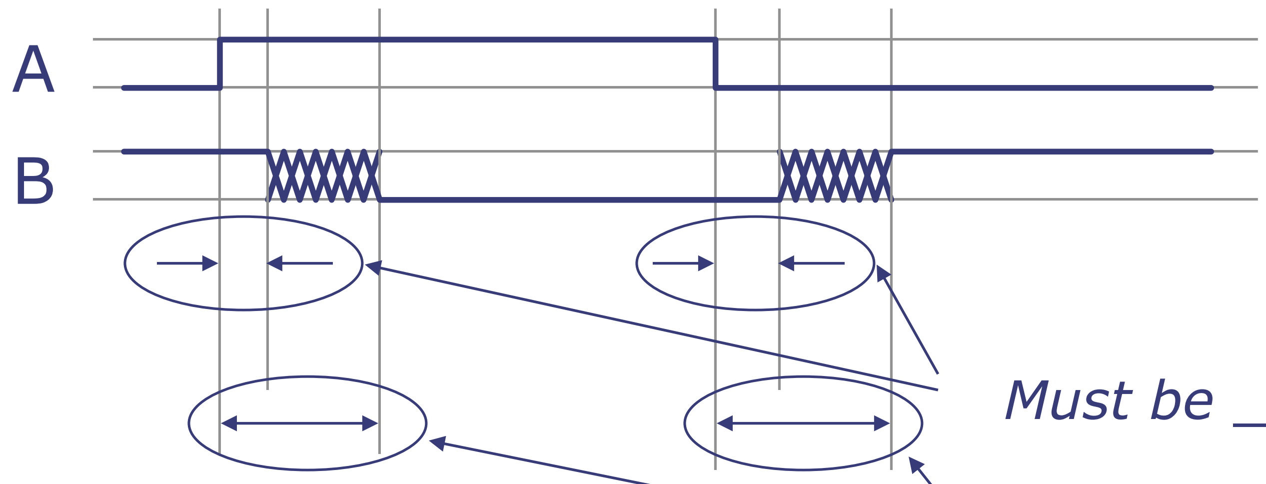
- Contamination delay (t_{CD}): A lower bound on the delay from invalid inputs to invalid outputs
  - Used later (for sequential logic), can ignore for now

The Combinational Contract

A  B

A	B
0	1
1	0

t_{PD} propagation delay
t_{CD} contamination delay



Must be $\geq t_{CD}$

Must be $\leq t_{PD}$

No promises during 

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