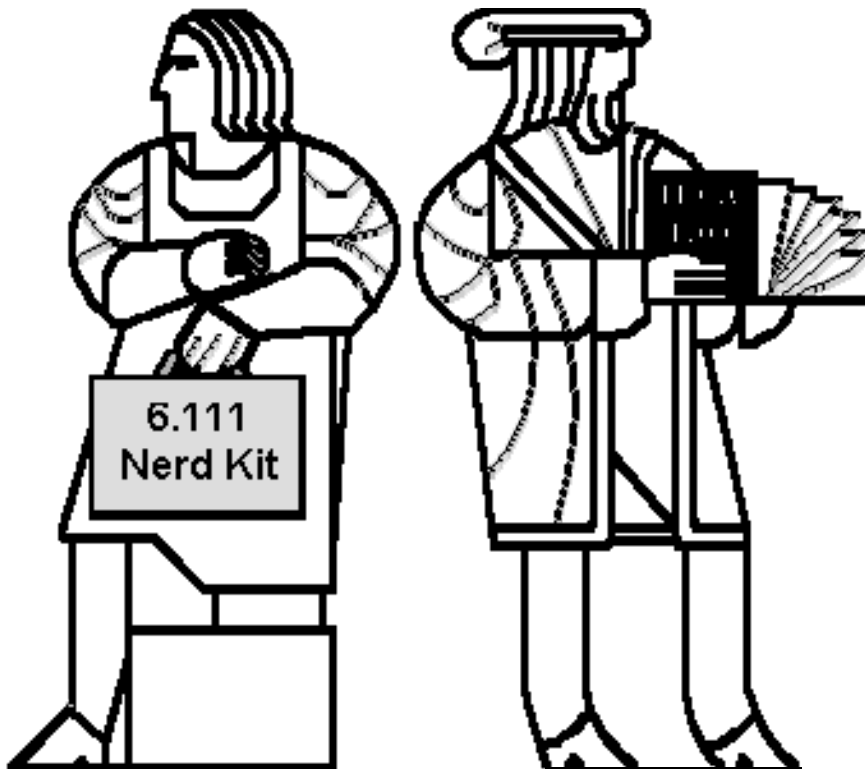# Welcome to 6.111!
Introductory Digital Systems Laboratory

6.111
Nerd Kit

Handouts:

Info form (yellow)
Course Calendar
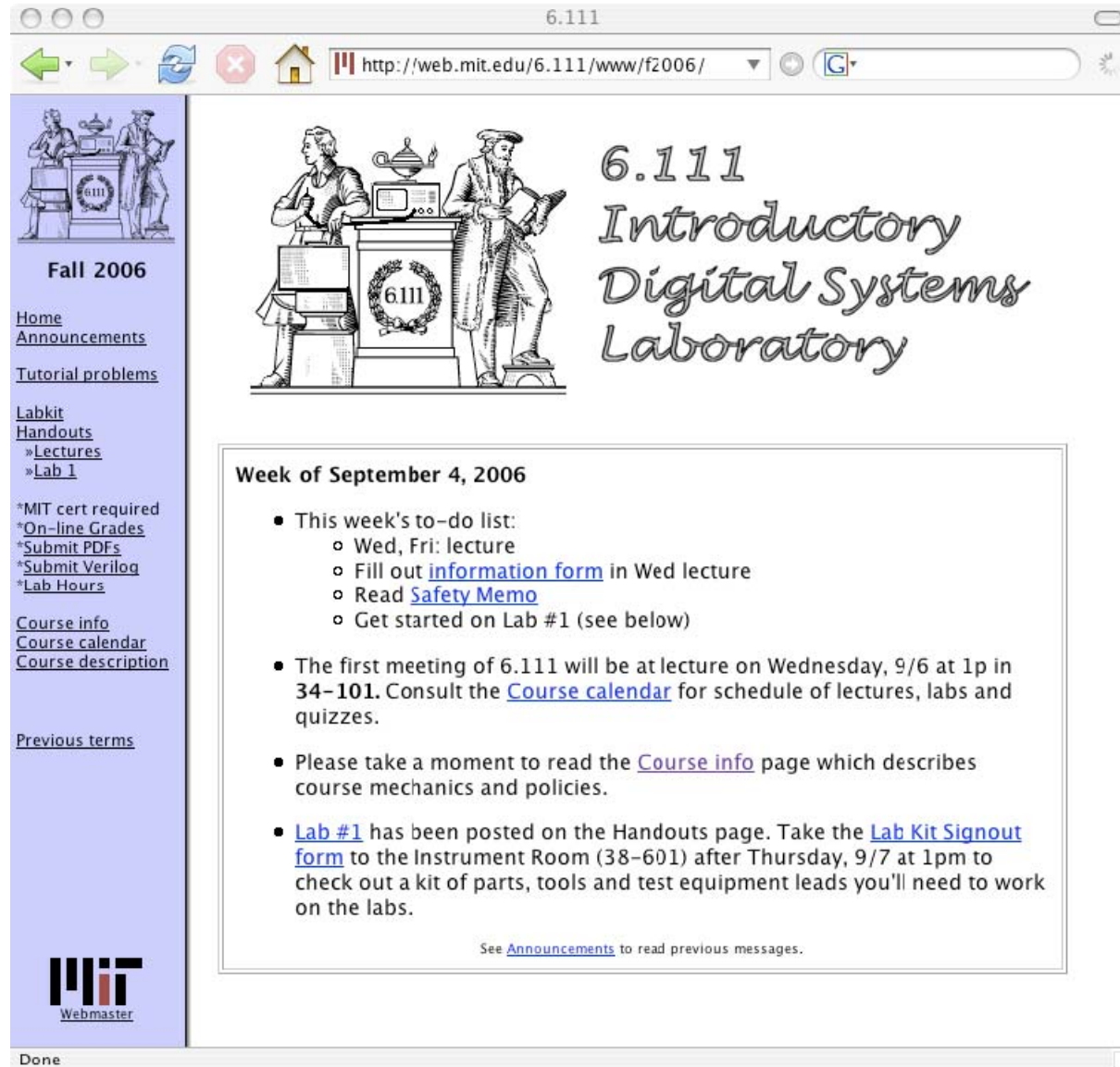Lecture slides

Lectures:
Chris Terman
TAs:
Javier Castro
Cassie Huang

Course Website (http://mit.edu/6.111)

6.111 Goals

- **Fundamentals of logic design**
  - combinational and sequential blocks
- **System integration with multiple components**
  - FPGAs, memories, discrete components, etc.
- **Learn a Hardware Description Language (Verilog)**
- **Interfacing issues with analog components**
  - ADC, DAC, sensors, etc.
- **Understand different design methodologies**
- **Understand different design metrics**
  - component/gate count and implementation area, switching speed, energy dissipation and power
- **Design & implement a substantial digital system**
- **Have fun!**

Labs: learning the ropes

- **Lab 1**
  - Experiment with gates, design & implement some logic
  - Learn about lab equipment in the Digital Lab (38-600): oscilloscopes and logic analyzers
- **Lab 2**
  - Introduction to Verilog & the labkit
- **Lab 3**
  - Design and implement a Finite State Machine (FSM)
  - Use Verilog to program an FPGA
  - Report and its revision will be evaluated for CI-M
- **Lab 4**
  - Design a complicated system with multiple FSMs (Major/Minor FSM)
  - Voice recorder using AC97 codec and SRAMs
- **Lab 5**
  - Video circuits: a simple Pong game

Final Project

- Done in groups of two (or sometimes three)
- Open ended
- You and the staff negotiate a project proposal
  - Must emphasize digital concepts, but inclusion of analog interfaces (e.g., data converters, sensors or motors) common and often desirable
  - Proposal Conference, several Design Reviews
- Design presentation to staff
- Staff will provide help with project definition and scope, design, debugging, and testing
- It is extremely difficult for a student to receive an A without completing the final project

Evaluation

- **Midterm** (11/1): 20%
- **Labs**: 30%
  - Labs 1 & 2: 3%, Lab 3: 6%, Labs 4 & 5: 9%
- **CI-M paper**: 10%
- **Final Project**: 40%
  - Deadlines and participation: 5%
  - Quality and organization of presentation and report: 5%
  - Complexity, innovation and risk: 10%
  - Problem definition: 2%
  - Architecture: 3%
  - Design (modularity, Verilog): 5%
  - Functionality: 10%
- A large number of students do "A" level work and are, indeed, rewarded with a grade of "A". The corollary to this is that, since average performance levels are so high, punting any part of the subject can lead to a disappointing grade.

Why Digital? A Thought Experiment
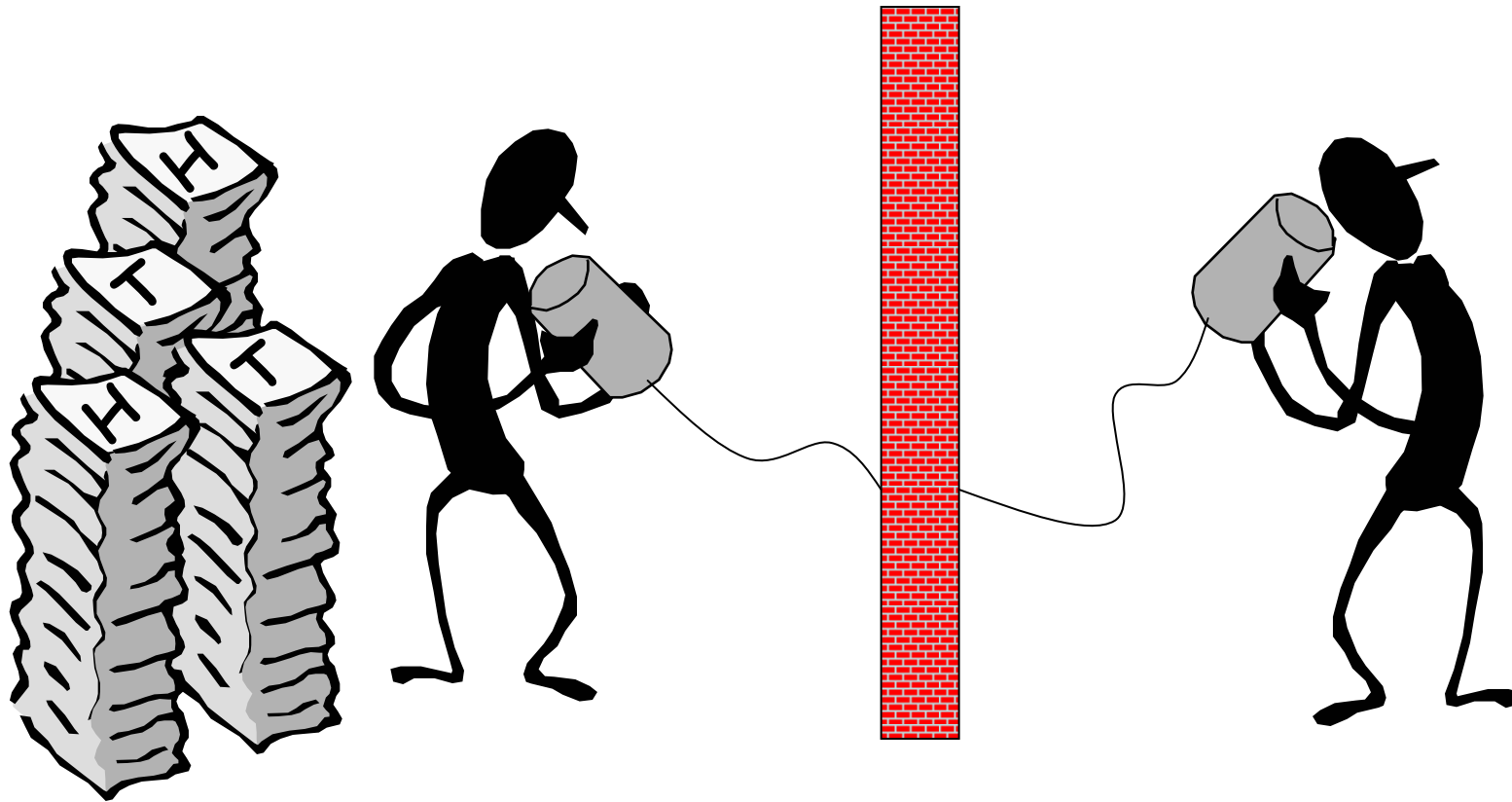
Goal: transmit results of 100 coin flips

Experiment #1: Analog Encoding
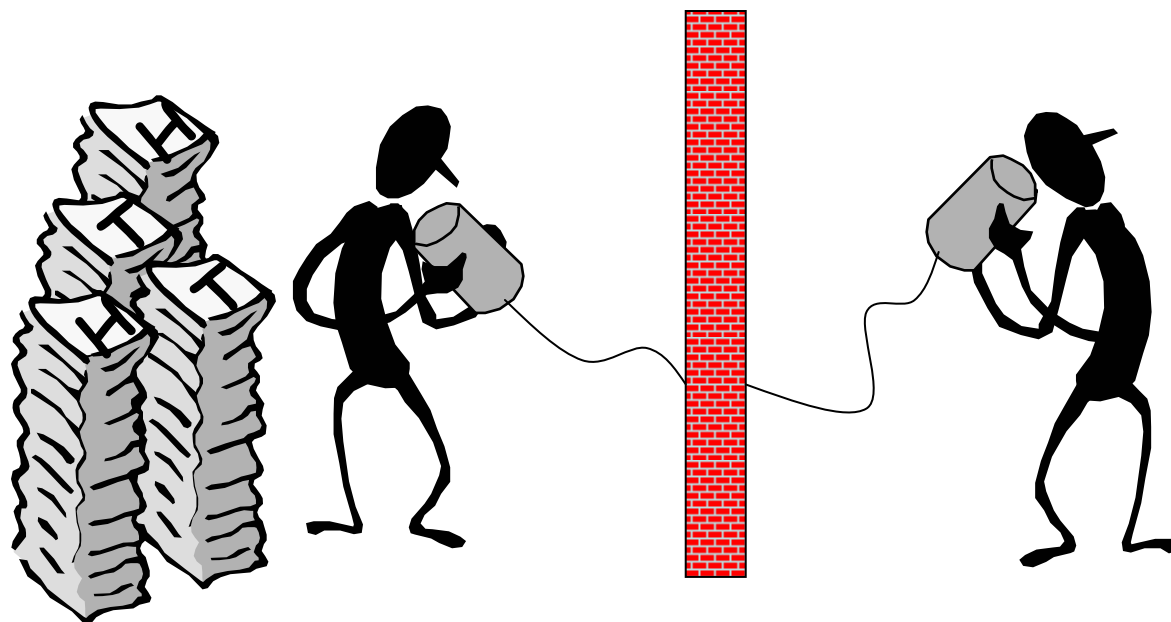
100 coin flips $\rightarrow 2^{100}$ possibilities
Transmit voltage $N/2^{100}$ for possibility #N
Required voltage resolution = $1/2^{100}$ = ~8e-31 volts

🚫 **impossible to reliably transmit/receive voltages with that resolution**

Rethink basic system architecture

- Noise and inaccuracy are inevitable; we can't reliably transmit/receive/manipulate "infinite" information-- we must design our system to tolerate some amount of error if it is to process information reliably.

- A system is a structure that is guaranteed to exhibit a specified behavior, assuming all of its components obey their specified behaviors.

How is this achieved?   CONTRACTS!

Every system component will have clear obligations and responsibilities. If contracts are violated all bets are off.

Going Digital

- Digital representation = information encoded as a sequence of symbols chosen from a (small) set.

- Keep in mind that the world is not digital, we will simply engineer it to behave that way. Furthermore, we must use real physical (analog, continuous) phenomena to implement digital designs!

- Common choices
  - Binary symbols (0, 1)
  - If we have DC connectivity (wired): encode using voltages/currents
  - If we don't have DC connectivity (wireless): encode using frequency/phase

We'll work with these

- Going digital keeps the contracts simple – limit quantum of information we process in exchange for reliability

Using Voltages Digitally

- Key idea: don't allow "0" to be mistaken for a "1" or vice versa

- Use the same "uniform representation convention" for *every* component and wire in our digital system

- To implement devices with high reliability, we outlaw "close calls" via a representation convention which forbids a range of voltages between "0" and "1".
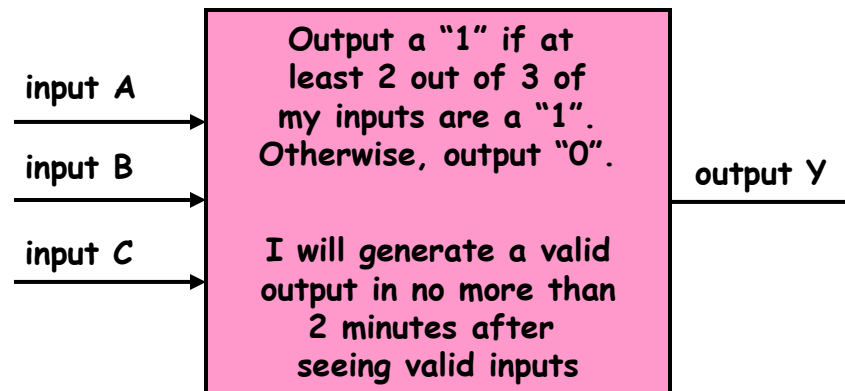
Invalid

| Valid "0" | Forbidden Zone | Valid "1" |

volts
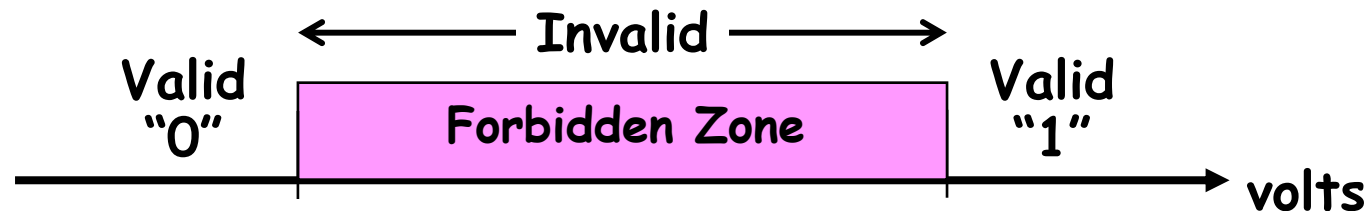
Consequence: notion of valid and invalid signals

A Digital Processing Element

- 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 minimum) of an upper bound t_{pd} on the required time for the device to compute the specified output values from an arbitrary set of stable, valid input values

Static discipline

input A →

input B →

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 minutes after seeing valid inputs
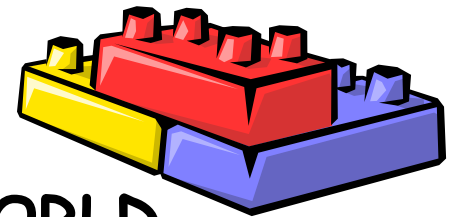
→ output Y

Why have processing blocks?

- The goal of modular design:

ABSTRACTION

- What does that mean anyway:
  - Rules simple enough for a 6-3 to follow...
  - Understanding BEHAVIOR without knowing IMPLEMENTATION
  - Predictable composition of functions
  - Tinker-toy assembly
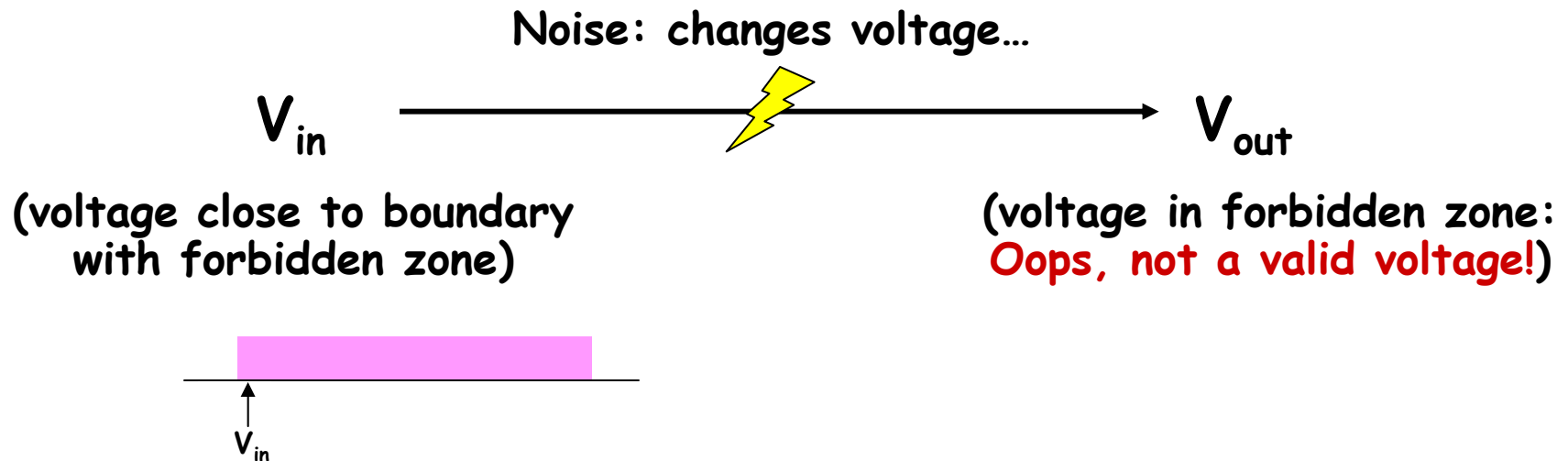 - Guaranteed behavior under REAL WORLD circumstances

A Combinational Digital System

- A set of interconnected elements is a *combinational device* if
  - each circuit element is a combinational device
  - every input is connected to exactly one output or a constant (eg, some vast supply of 0's and 1's)
  - the circuit contains no directed cycles

- Why is this true?
  - Given an acyclic circuit meeting the above constraints, we can derive functional and timing specs for the input/output behavior from the specs of its components!
  - We'll see lots of examples soon. But first, we need to build some combinational devices to work with…
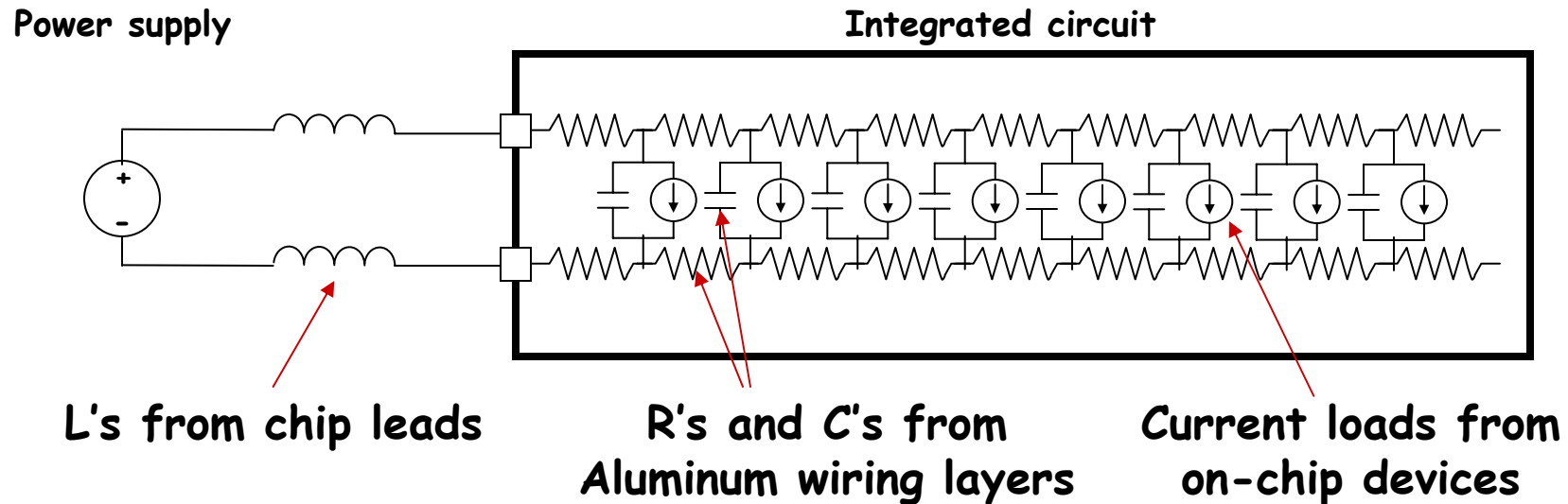
Wires: theory vs. practice

Does a wire obey the static discipline?

Noise: changes voltage…

V_{in} ⚡ V_{out}

(voltage close to boundary
with forbidden zone)

(voltage in forbidden zone:
Oops, not a valid voltage!)

V_{in}

Questions to ask ourselves:

In digital systems, where does noise come from?
How big an effect are we talking about?

Power Supply Noise

Power supply   **Integrated circuit**

L's from chip leads   **R's and C's from**   **Current loads from**
                                Aluminum wiring layers   **on-chip devices**

ΔV from:
- **IR drop**

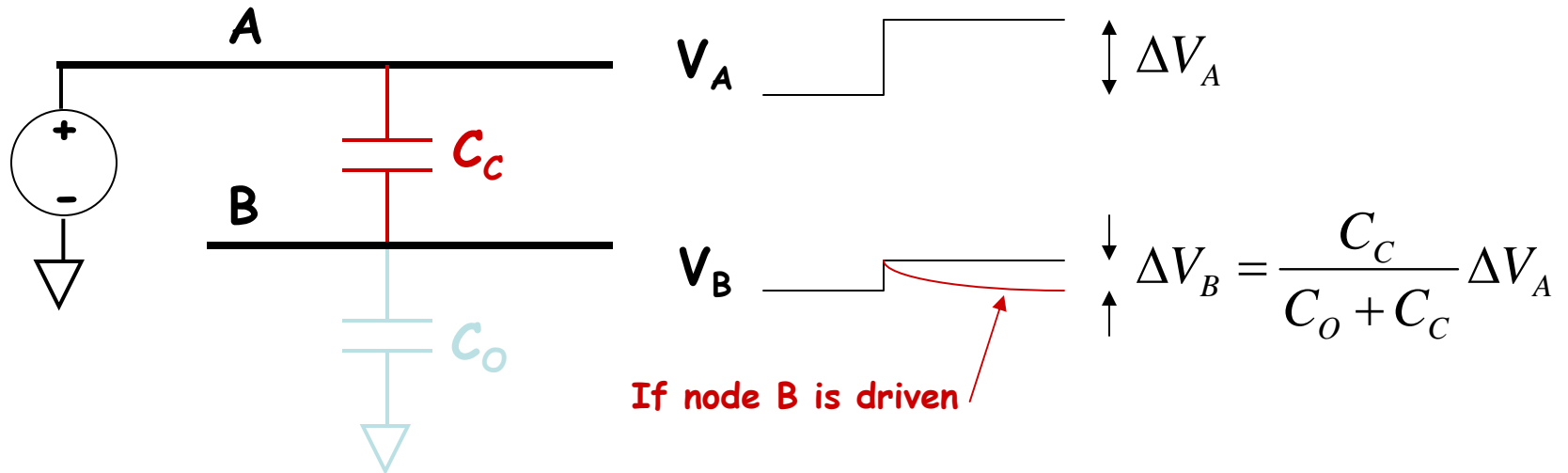 (between gates: 30mV, within module: 50mV, across chip: 350mV)
- **L(dI/dt) drop**

  (use extra pins and bypass caps to keep within 250mV)
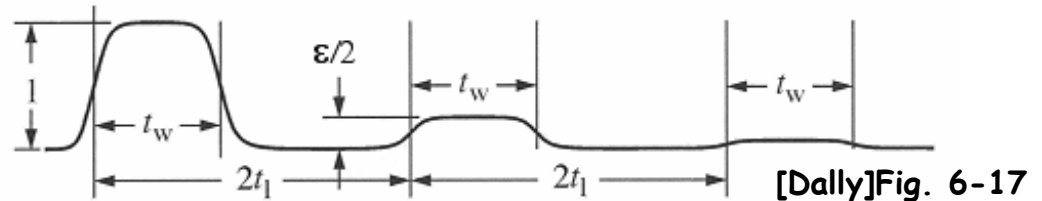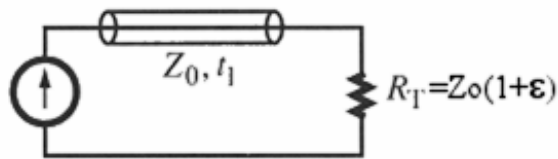- **LC ringing** triggered by current "steps"

Crosstalk

$$\Delta V_B = \frac{C_C}{C_O + C_C} \Delta V_A$$

If node B is driven

This situation frequently happens on integrated circuits where there are many overlapping wiring layers. In a modern integrated circuit ΔV_A might be 2.5V, C_O = 20fF and C_C = 10fF → ΔV_B = 0.83V! Designers often try to avoid these really bad cases by careful routing of signals, but some crosstalk is unavoidable.
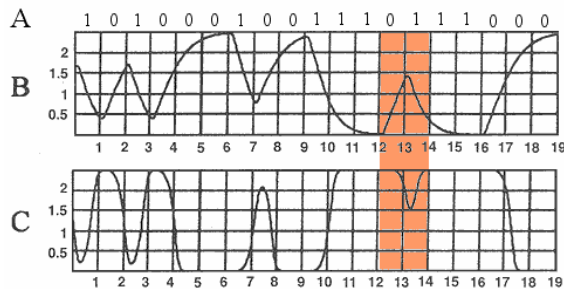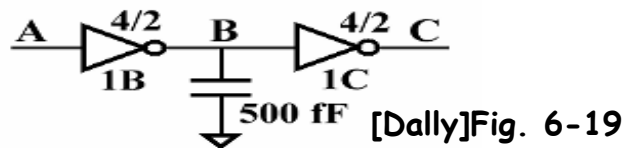
Intersymbol Interference

ΔV from energy storage left over from earlier signaling on the wire:

- **transmission line discontinuities**
  (reflections off of impedance mismatches and terminations)

Z_0, t_1 $R_T = Z_o(1+\varepsilon)$
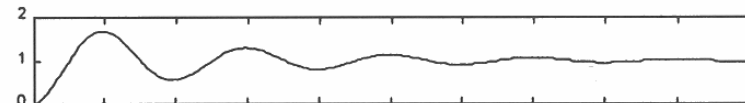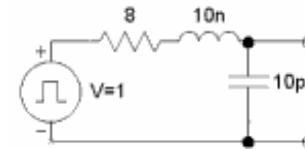
[Dally]Fig. 6-17

- **charge storage in RC circuit**
(narrow pulses are lost due to
incomplete transitions)

[Dally]Fig. 6-19

[Dally]Fig. 6-20

- **RLC ringing**
(triggered by voltage "steps")

**Fix: slower operation, limiting
voltage swings and slew rates**

Needed: Noise Margins!

Does a wire obey the static discipline?

$$V_{in}$$
(marginally valid)

Noise

$$V_{out}$$
(invalid!)
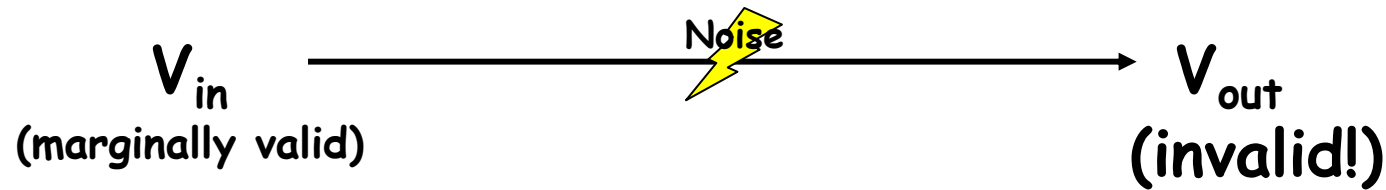
No! A combinational device must restore marginally valid signals. It must accept marginal inputs and provide unquestionable outputs (i.e., to leave room for noise).

VALID INPUT REPRESENTATIONS

| Valid "0" | | Forbidden Zone | | Valid "1" |

$$V_{OL} \quad V_{IL} \qquad\qquad V_{IH} \quad V_{OH}$$

volts

NOISE MARGINS

VALID OUTPUT REPRESENTATIONS
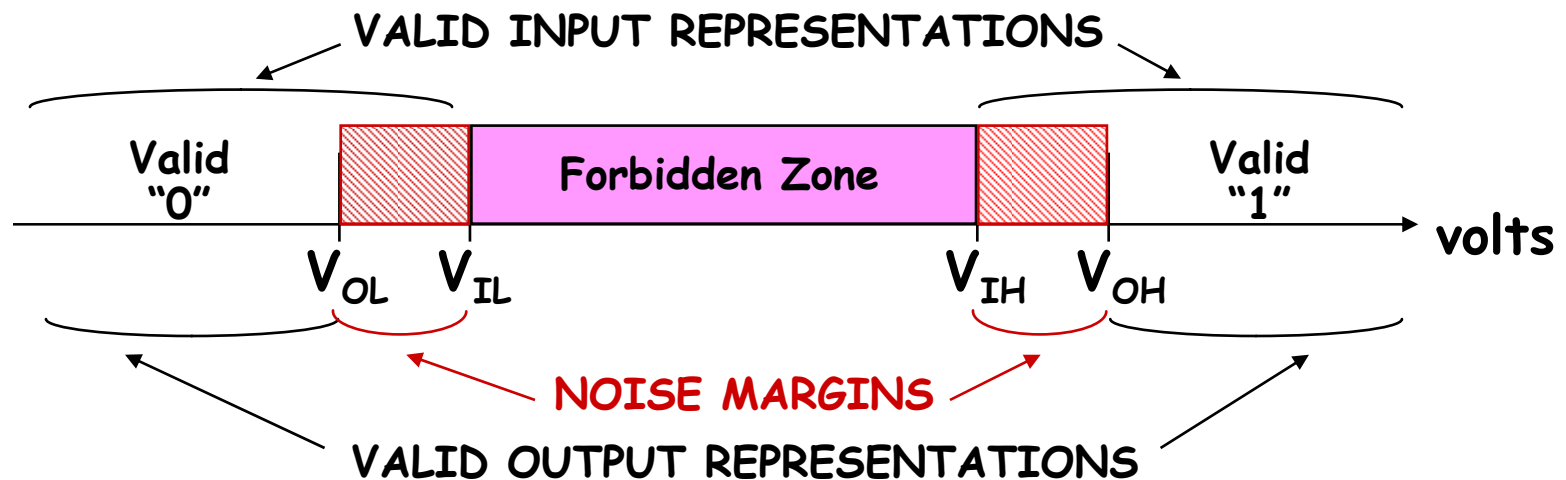
Sample DC (signaling) Specification

HCMOS family characteristics   FAMILY SPECIFICATIONS

DC CHARACTERISTICS FOR 74HCT
Voltages are referenced to GND (ground = 0 V)

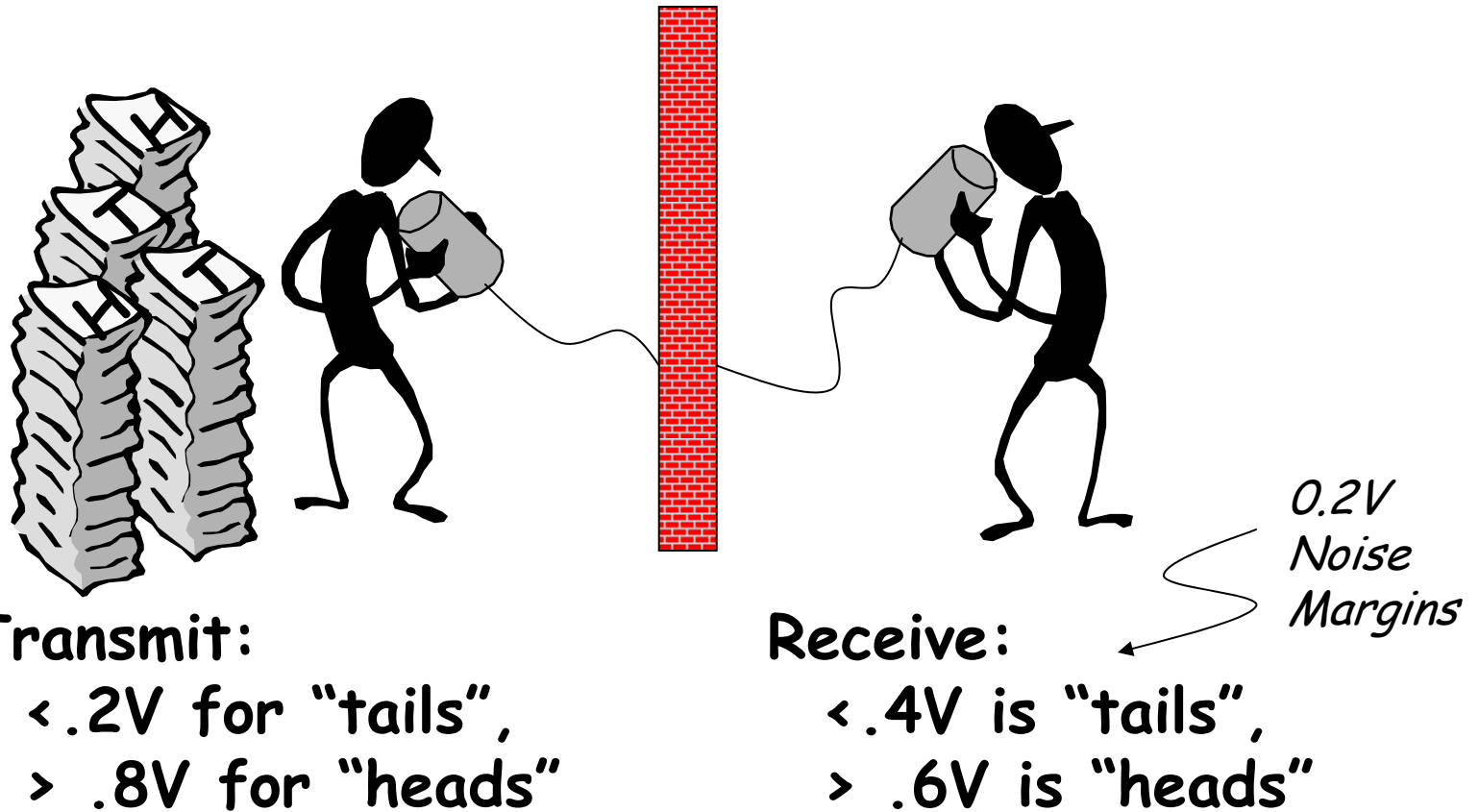| SYMBOL | PARAMETER | T_{amb} (°C) 74HCT | | | | | | | UNIT | TEST CONDITIONS | | |
		+25			−40 to +85		−40 to +125			V_{CC} (V)	V_I	OTHER
		min.	typ.	max.	min.	max.	min.	max.				
V_{IH}	HIGH level input voltage	2.0	1.6		2.0		2.0		V	4.5 to 5.5		
V_{IL}	LOW level input voltage		1.2	0.8		0.8		0.8	V	4.5 to 5.5		
V_{OH}	HIGH level output voltage all outputs	4.4	4.5		4.4		4.4		V	4.5	V_{IH} or V_{IL}	$-I_O = 20\ \mu A$
V_{OH}	HIGH level output voltage standard outputs	3.98	4.32		3.84		3.7		V	4.5	V_{IH} or V_{IL}	$-I_O = 4.0$ mA
V_{OH}	HIGH level output voltage bus driver outputs	3.98	4.32		3.84		3.7		V	4.5	V_{IH} or V_{IL}	$-I_O = 6.0$ mA
V_{OL}	LOW level output voltage all outputs		0	0.1		0.1		0.1	V	4.5	V_{IH} or V_{IL}	$I_O = 20\ \mu A$
V_{OL}	LOW level output voltage standard outputs		0.15	0.26		0.33		0.4	V	4.5	V_{IH} or V_{IL}	$I_O = 4.0$ mA
V	LOW level output		0.16	0.26		0.33		0.4	V	4.5	V	$I_O = 6.0$ mA

Experiment #2: Digital Encoding

0.2V
Noise
Margins

Transmit:
    <.2V for "tails",
    > .8V for "heads"

Receive:
    <.4V is "tails",
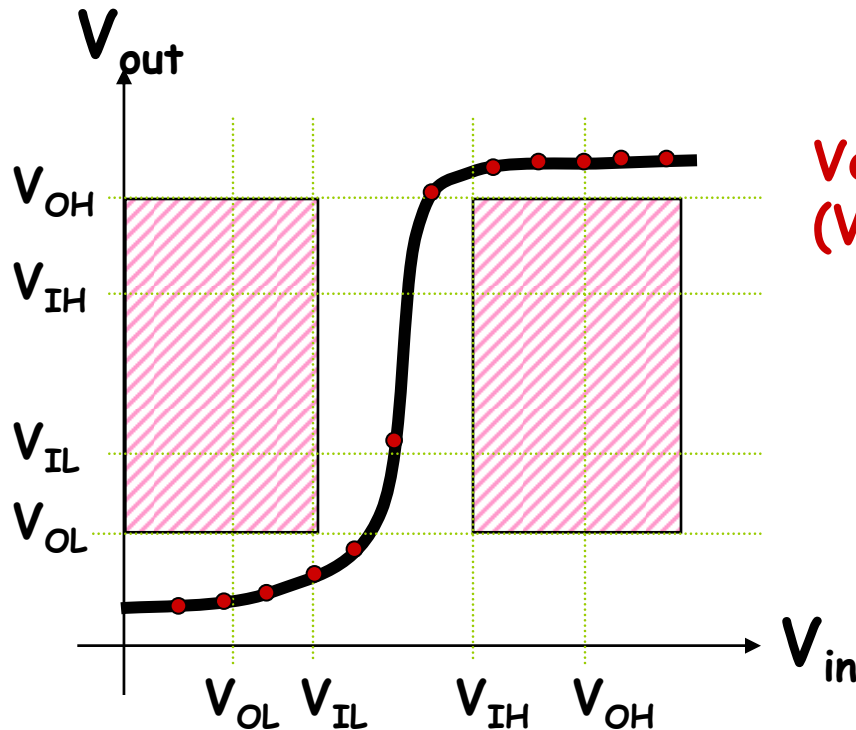   > .6V is "heads"

100 coin flips → one transmission for each flip

✓ **But when does receiver make measurements?**
Is � "HT" or "HHHTTT"?

Example device: A Buffer

$0 \dashrightarrow 0$ $1 \dashrightarrow 1$

**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

Static Discipline requires that we avoid the shaded regions aka "forbidden zones"), which correspond to *valid* inputs but *invalid* outputs. Net result: combinational devices must have **GAIN > 1** and be **NONLINEAR**.

Can this be a combinational device?

Suppose that you measured the voltage transfer curve of the device shown below. Could we build a logic family using it as a single-input combinational device?
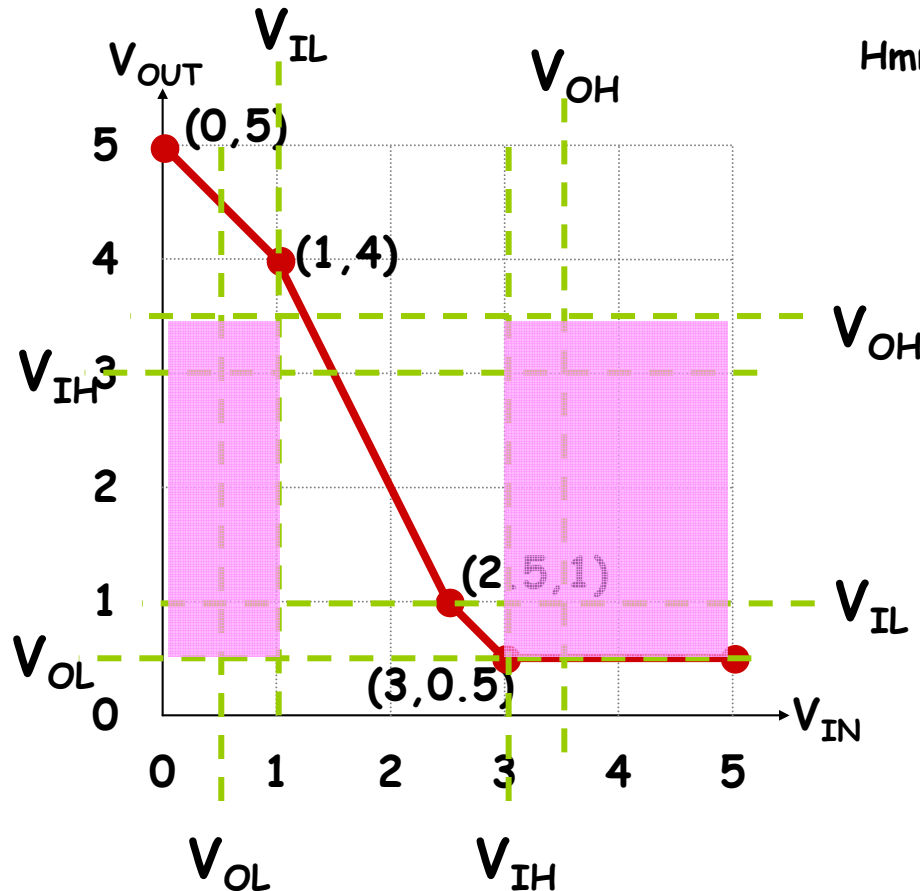
Hmmm, it had better be an *INVERTER...*

The device must be able to actually produce the desired output level. Thus, V_{OL} can be no lower than 0.5 V.

Try V_{OL} = 0.5 V

V_{IH} must be high enough to produce V_{OL}

Try V_{IH} = 3 V

Now, choose noise margins – find an N and set
$V_{OH} = V_{IH} + N$
$V_{IL} = V_{OL} + N$
Such that
V_{IH} IN generates V_{OL} or less out; AND
V_{IL} IN generates V_{OH} or more out.

Try N = 0.5 V

Summary

- Use voltages to encode information
- "Digital" encoding
  - valid voltage levels for representing "0" and "1"
  - forbidden zone avoids mistaking "0" for "1" and vice versa

- Noise
  - Want to tolerate real-world conditions: NOISE.
  - Key: tougher standards for output than for input
  - devices must have gain and have a non-linear VTC

- Combinational devices
  - Each logic family has Tinkertoy-set simplicity, modularity
  - predictable composition: "parts work $\rightarrow$ whole thing works"
  - static discipline
    - digital inputs, outputs; restore marginal input voltages
    - complete functional spec
    - valid inputs lead to valid outputs in bounded time