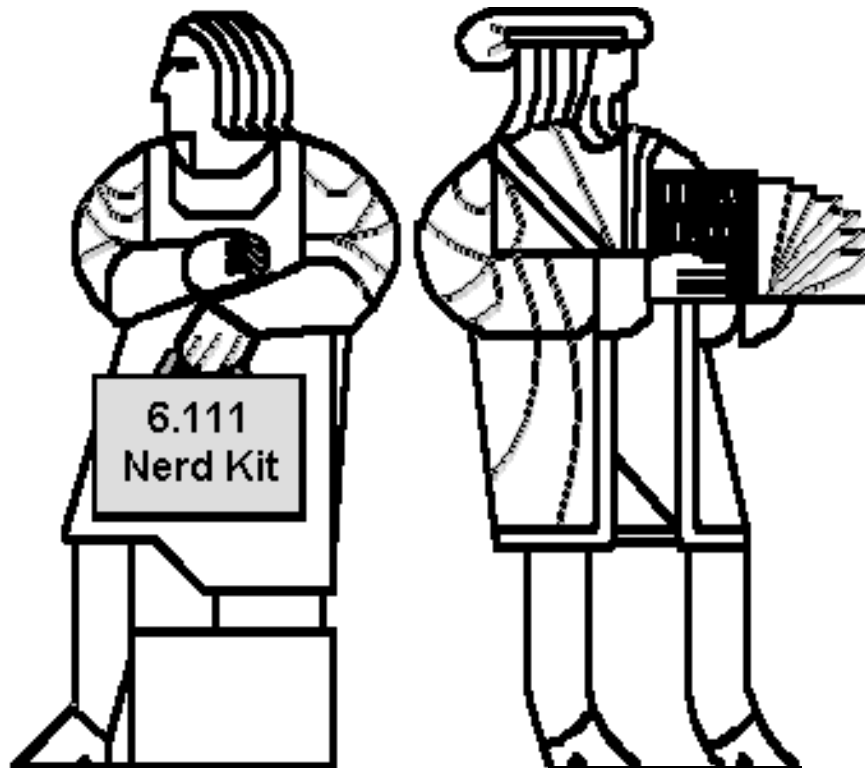


Welcome to 6.111!

Introductory Digital Systems Laboratory



Handouts:

Info form (yellow)

Course Calendar

Lecture slides

Lectures:

Ike Chuang

Chris Terman

TAs:

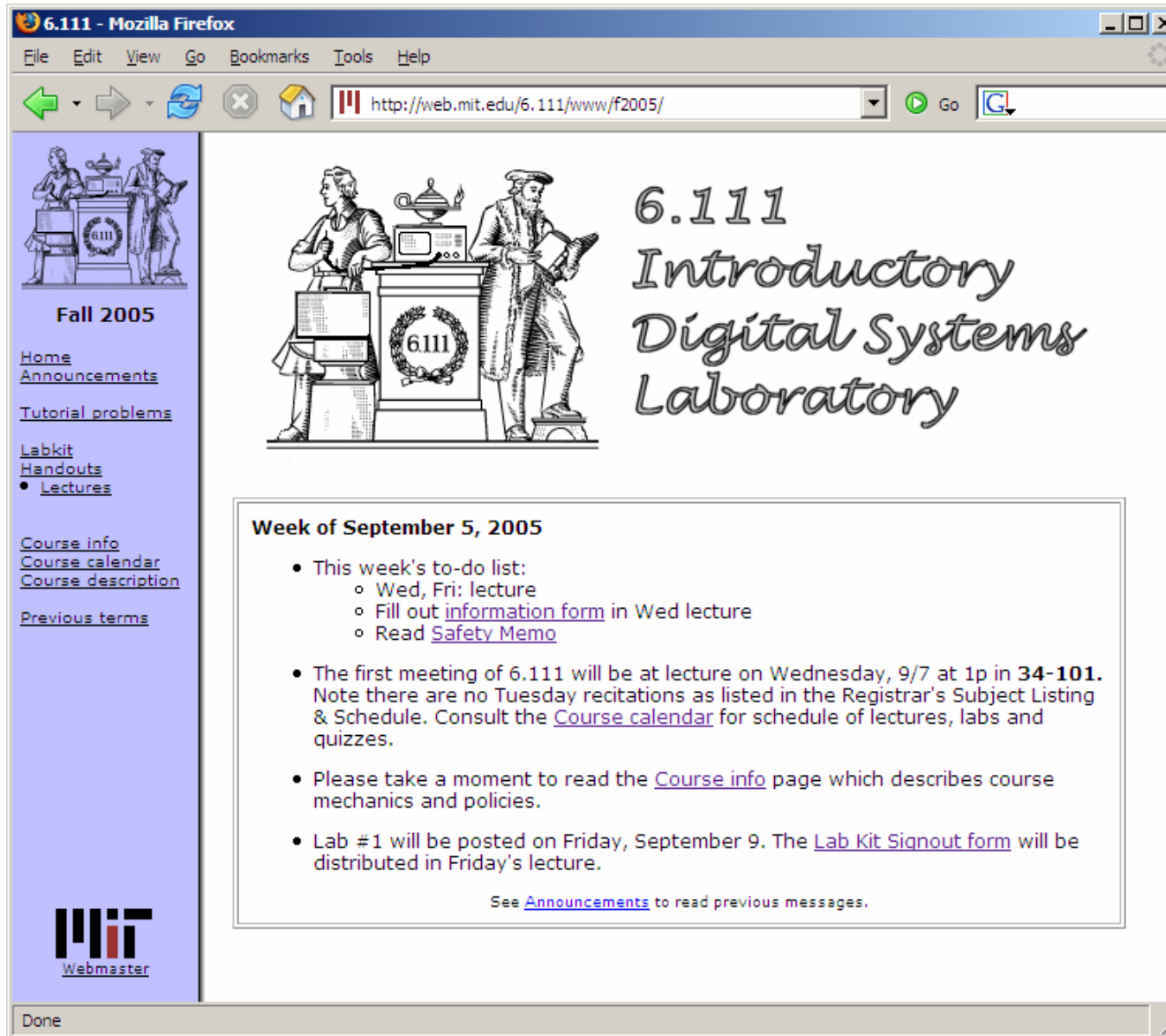
Javier Castro

Eric Fellheimer

Jae Lee

Willie Sanchez

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



The screenshot shows a Mozilla Firefox browser window displaying the course website for 6.111. The browser's address bar shows the URL <http://web.mit.edu/6.111/www/f2005/>. The website features a navigation menu on the left with links for Home, Announcements, Tutorial problems, Labkit, Handouts, Lectures, Course info, Course calendar, Course description, and Previous terms. The main content area includes a header with the course title "6.111 Introductory Digital Systems Laboratory" and a central illustration of two figures in a laboratory setting. Below the header, a section titled "Week of September 5, 2005" lists several items: a to-do list for Wednesday and Friday lectures, the first meeting date and location (Wednesday, 9/7 at 1p in 34-101), a request to read the Course info page, and information about Lab #1 being posted on Friday, September 9. A link to Announcements is provided at the bottom of the list.

6.111
Introductory
Digital Systems
Laboratory

Week of September 5, 2005

- This week's to-do list:
 - Wed, Fri: lecture
 - Fill out [information form](#) in Wed lecture
 - Read [Safety Memo](#)
- The first meeting of 6.111 will be at lecture on Wednesday, 9/7 at 1p in **34-101**. Note there are no Tuesday recitations as listed in the Registrar's Subject Listing & Schedule. Consult the [Course calendar](#) for schedule of lectures, labs and quizzes.
- Please take a moment to read the [Course info](#) page which describes course mechanics and policies.
- Lab #1 will be posted on Friday, September 9. The [Lab Kit Signout form](#) will be distributed in Friday's lecture.

See [Announcements](#) to read previous messages.

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
 - Introduction to Verilog
- **Lab 2**
 - 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 3**
 - Design a complicated system with multiple FSMs
(Major/Minor FSM)
 - Voice recorder using AC97 codec and SRAMs
- **Lab 4**
 - Video circuits: a simple Pong game

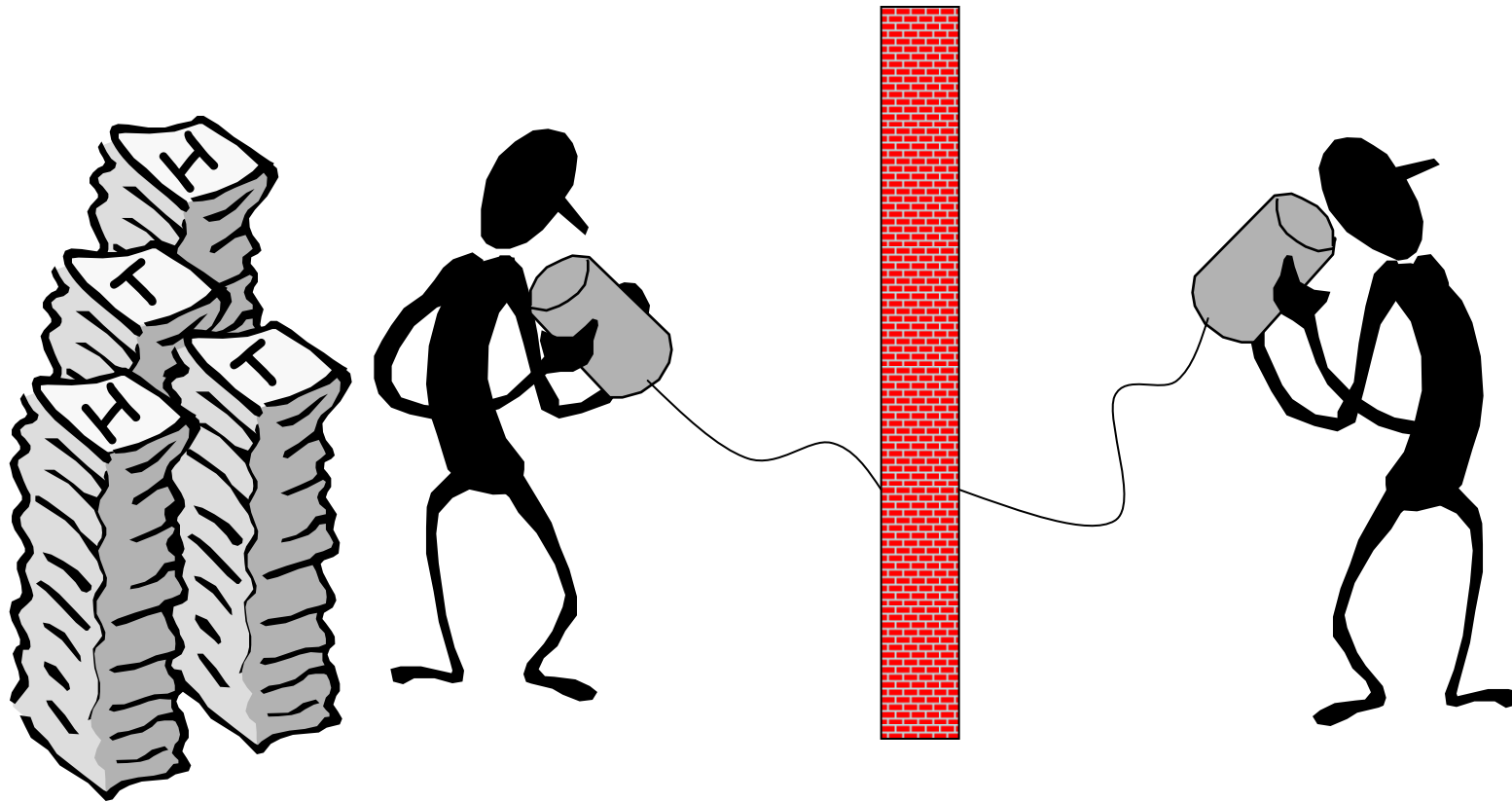
Final Project

- 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 in class (% of the final grade for the in-class presentation)
- 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

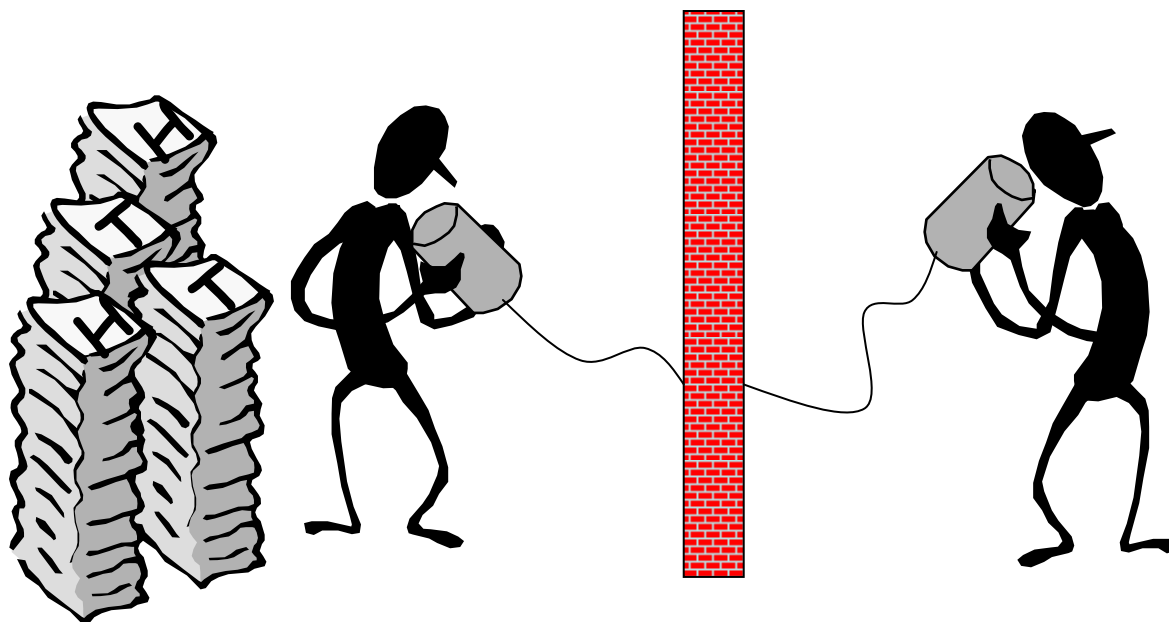
- **Midterm (11/2): 20%**
- **Labs: 30%**
 - Labs 1 & 2: 5%, Labs 3 & 4: 10%
- **CI-M paper: 10%**
- **Final Project: 40%**
 - Deadlines and participation: 5%
 - Quality and organization of presentation and report: 5%
 - Complexity, innovation and risk: 10%
 - Problem definition: 2%
 - Architecture: 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 = $1/2^{100} = \sim 8e-31$ volts



impossible to reliably transmit/receive voltages with that resolution

Rethink basic system architecture

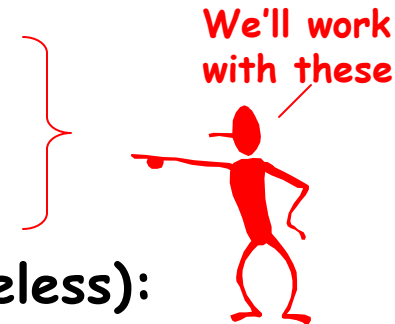
- Noise and inaccuracy are inevitable; we can't reliably transmit/receive/manipulate "infinite" information-- we must **design our system to tolerate some amount of error** if it is to process information reliably.
- A system is a structure that is guaranteed to exhibit a specified behavior, assuming 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.

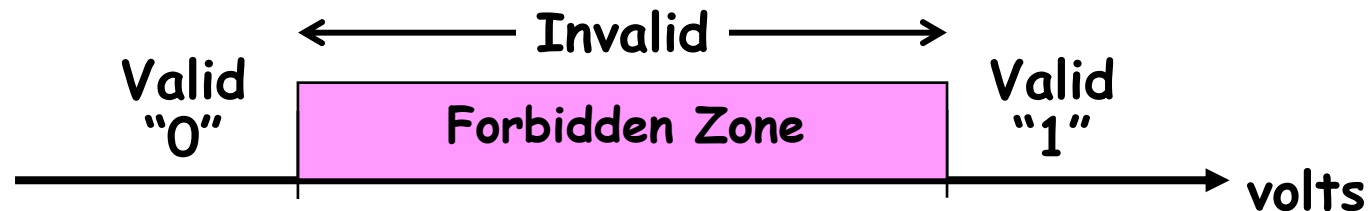
Going Digital

- 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
- 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".

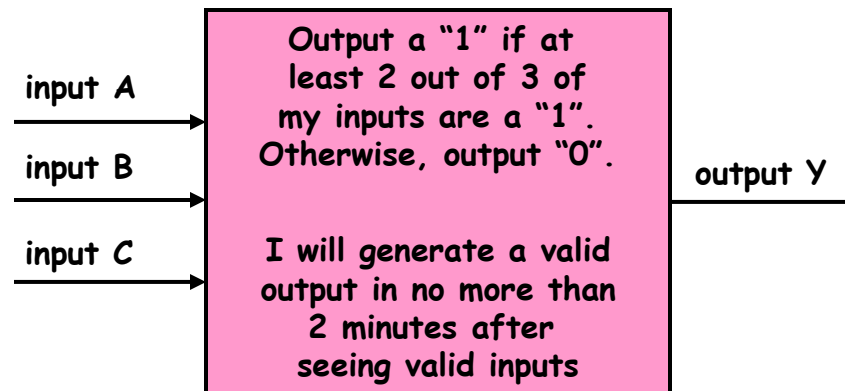


Consequence: notion of valid and invalid signals

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

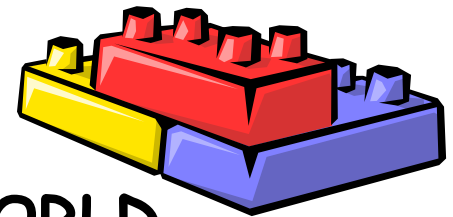


Why have processing blocks?

- 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

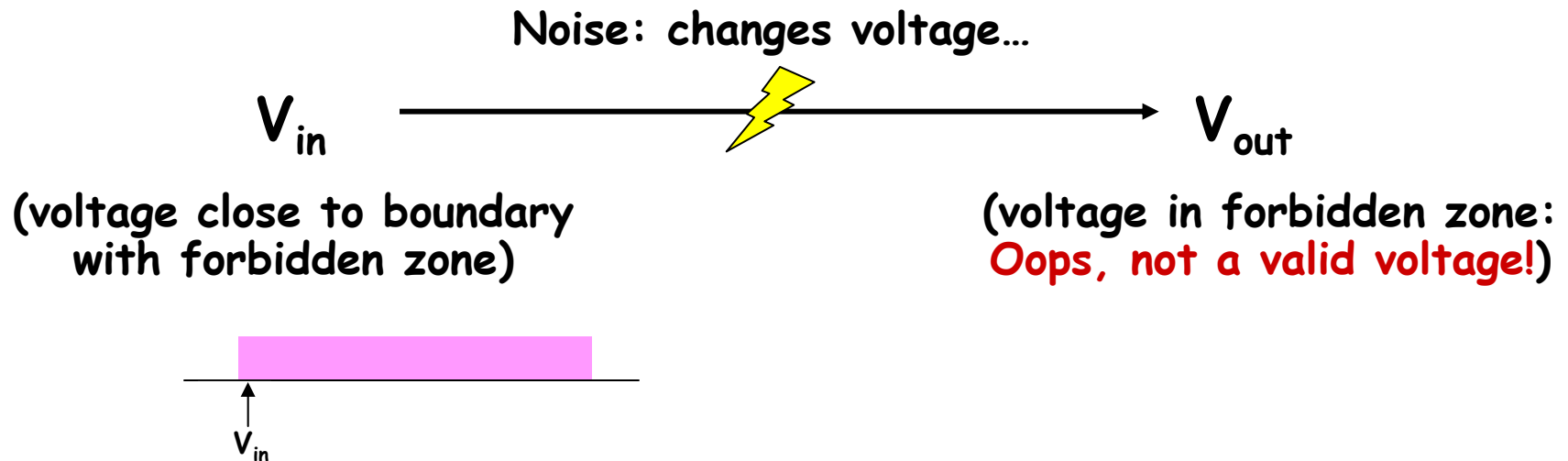


A Combinational Digital System

- 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

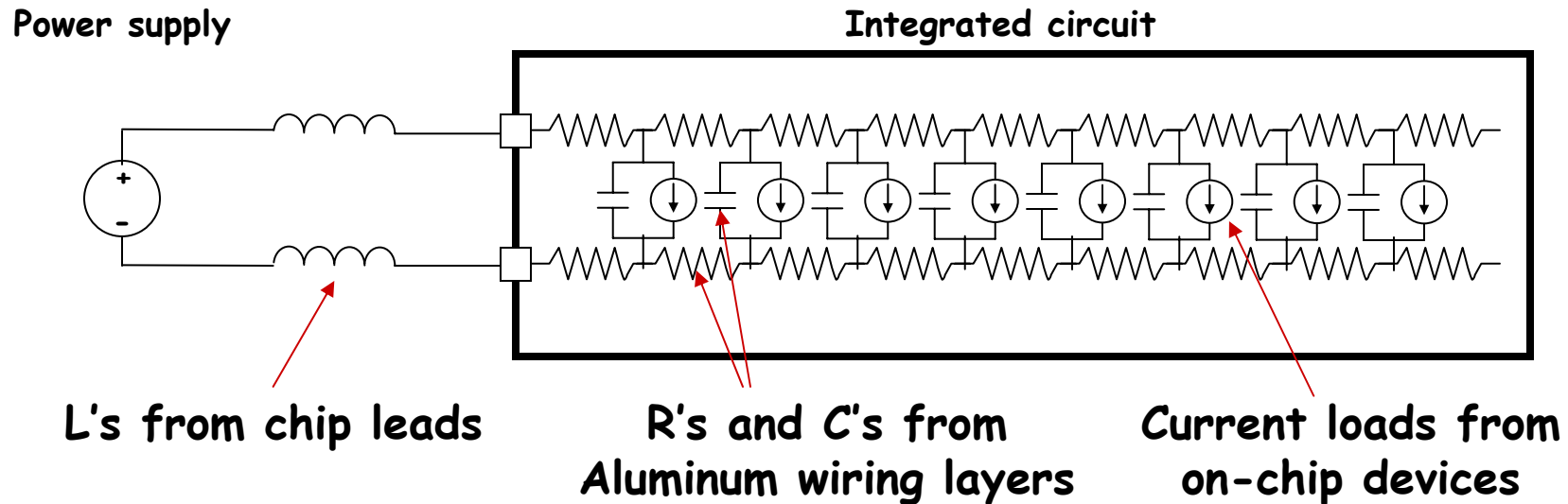
Does a wire obey the static discipline?



Questions to ask ourselves:

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

Power Supply Noise



ΔV from:

- **IR drop**

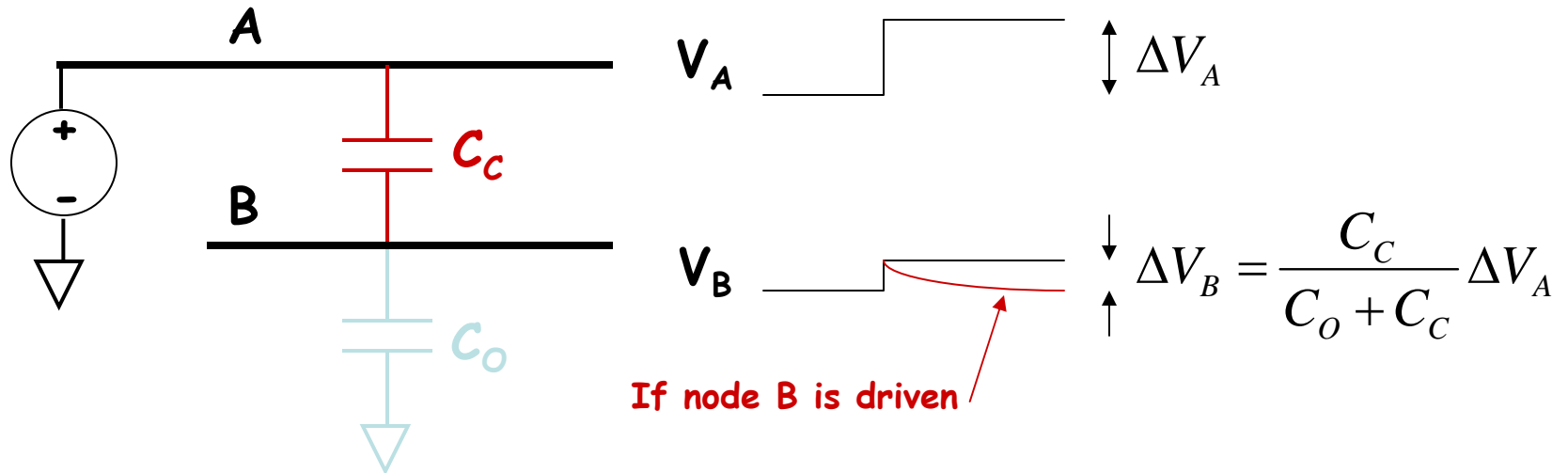
(between gates: 30mV, within module: 50mV, across chip: 350mV)

- **$L(dI/dt)$ drop**

(use extra pins and bypass caps to keep within 250mV)

- **LC ringing** triggered by current "steps"

Crosstalk

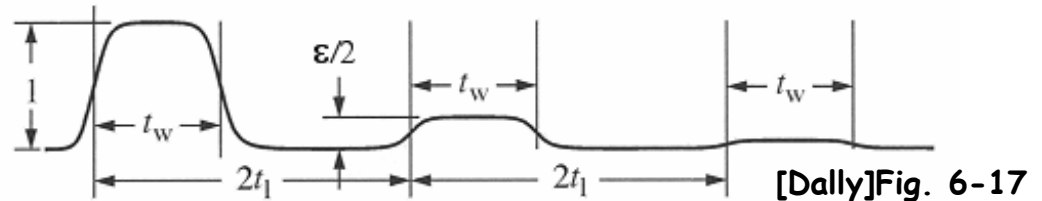
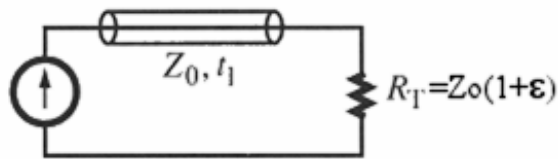


This situation frequently happens on integrated circuits where there are many overlapping wiring layers. In a modern integrated circuit ΔV_A might be **2.5V**, $C_o = 20\text{fF}$ and $C_c = 10\text{fF} \rightarrow \Delta V_B = 0.83\text{V}$! Designers often try to avoid these really bad cases by careful routing of signals, but some crosstalk is unavoidable.

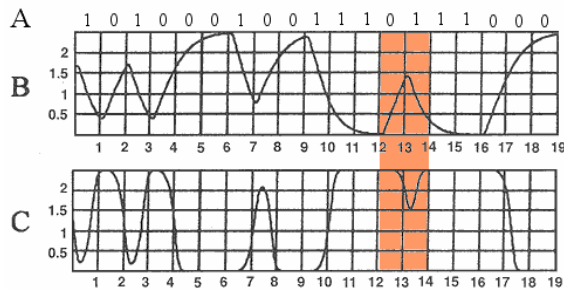
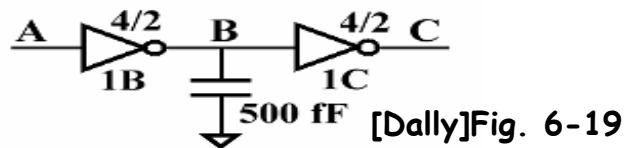
Intersymbol Interference

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

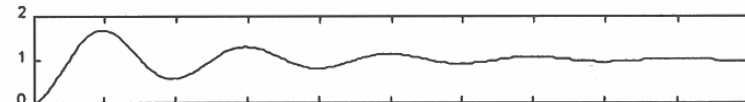
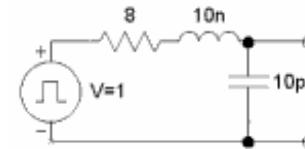
- **transmission line discontinuities**
(reflections off of impedance mismatches and terminations)



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



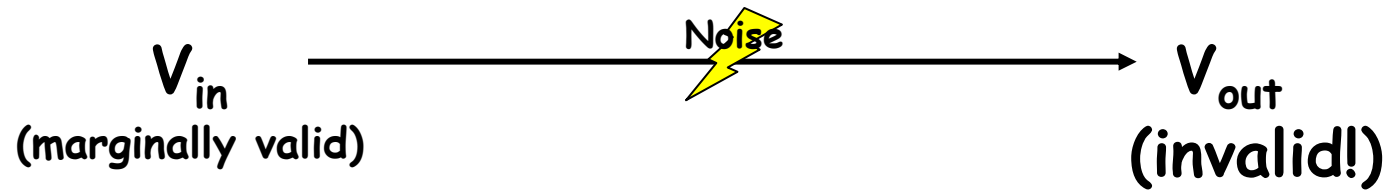
- **RLC ringing**
(triggered by voltage "steps")



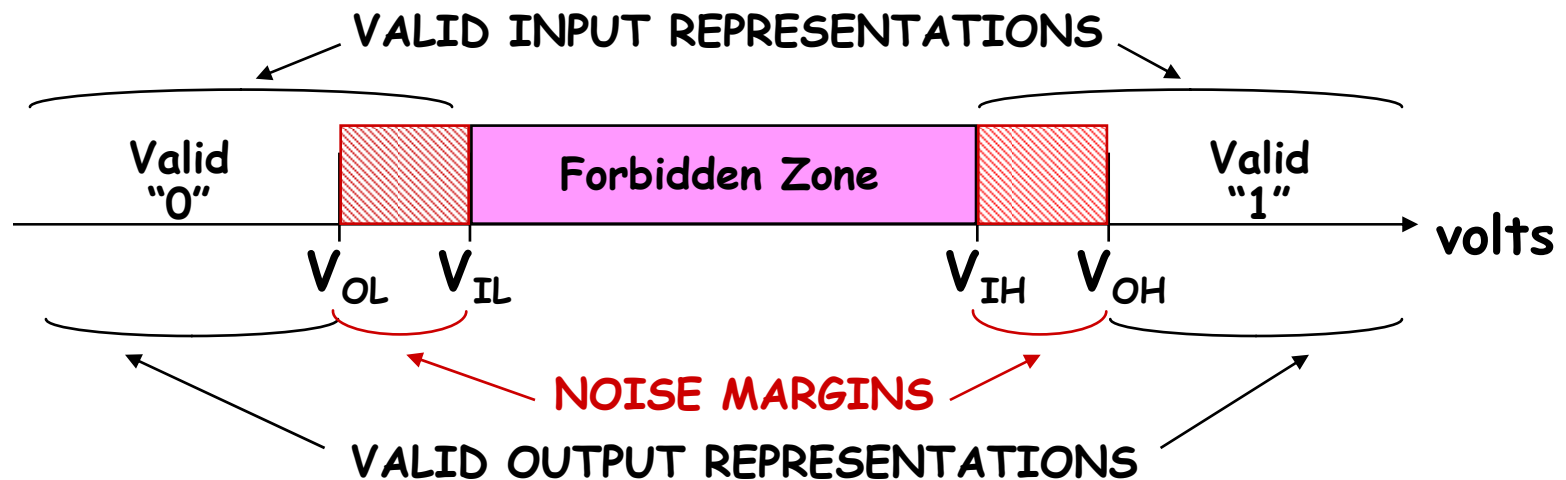
Fix: slower operation, limiting voltage swings and slew rates

Needed: Noise Margins!

Does a wire obey the static discipline?



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).



Sample DC (signalling) Specification

HCMOS family characteristics

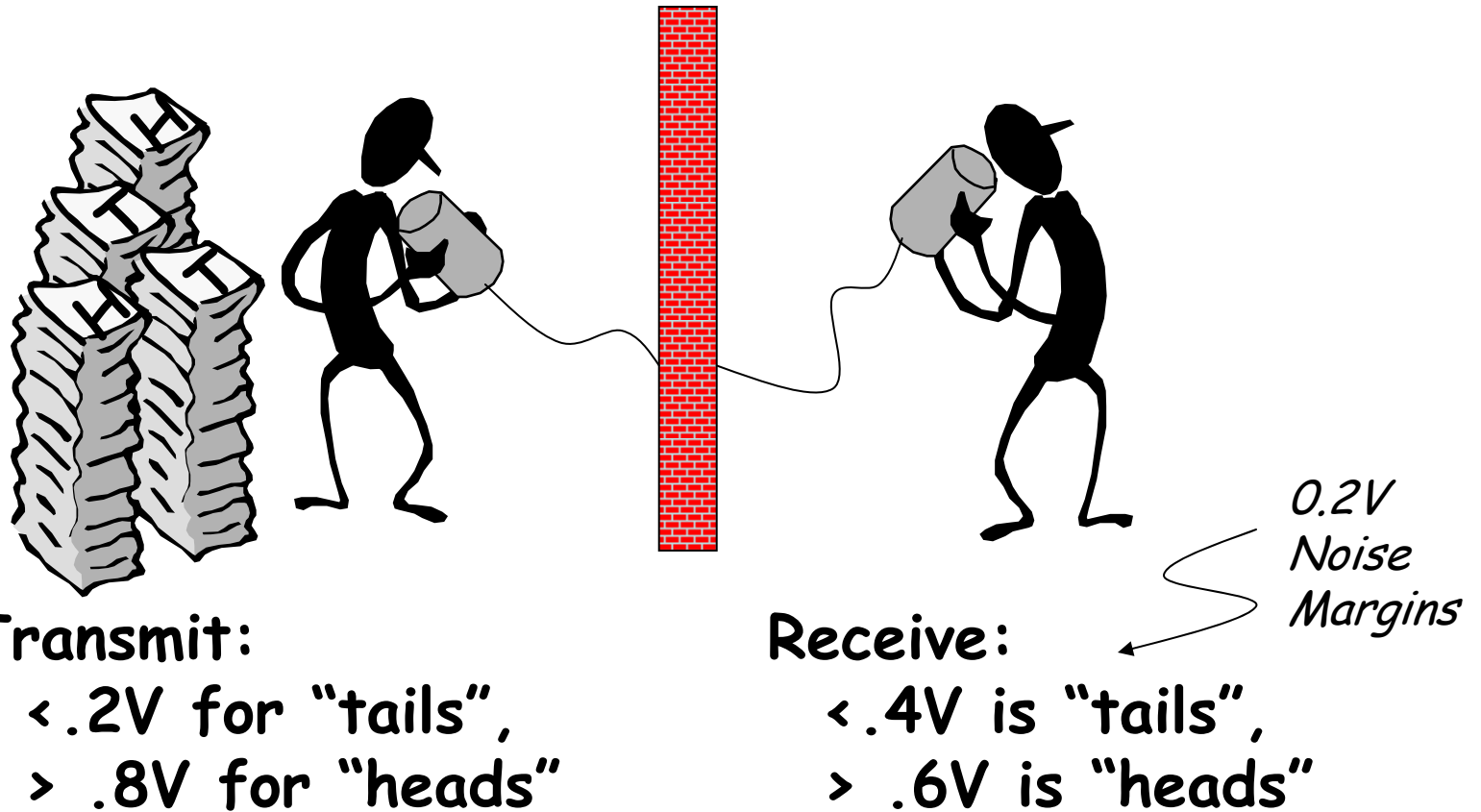
FAMILY SPECIFICATIONS

DC CHARACTERISTICS FOR 74HCT

Voltages are referenced to GND (ground = 0 V)

SYMBOL	PARAMETER	T _{amb} (°C)							UNIT	TEST CONDITIONS		
		74HCT								V _{CC} (V)	V _I	OTHER
		+25			-40 to +85		-40 to +125					
		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 μA
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 μA
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

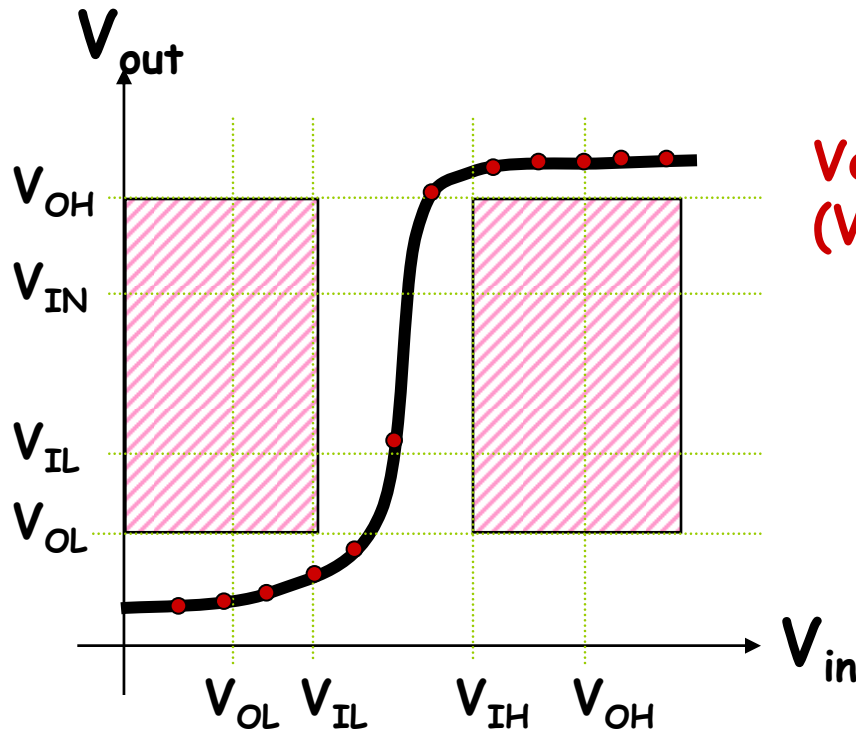
Experiment #2: Digital Encoding



100 coin flips → one transmission for each flip

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

Example device: A Buffer



Voltage Transfer Characteristic (VTC):

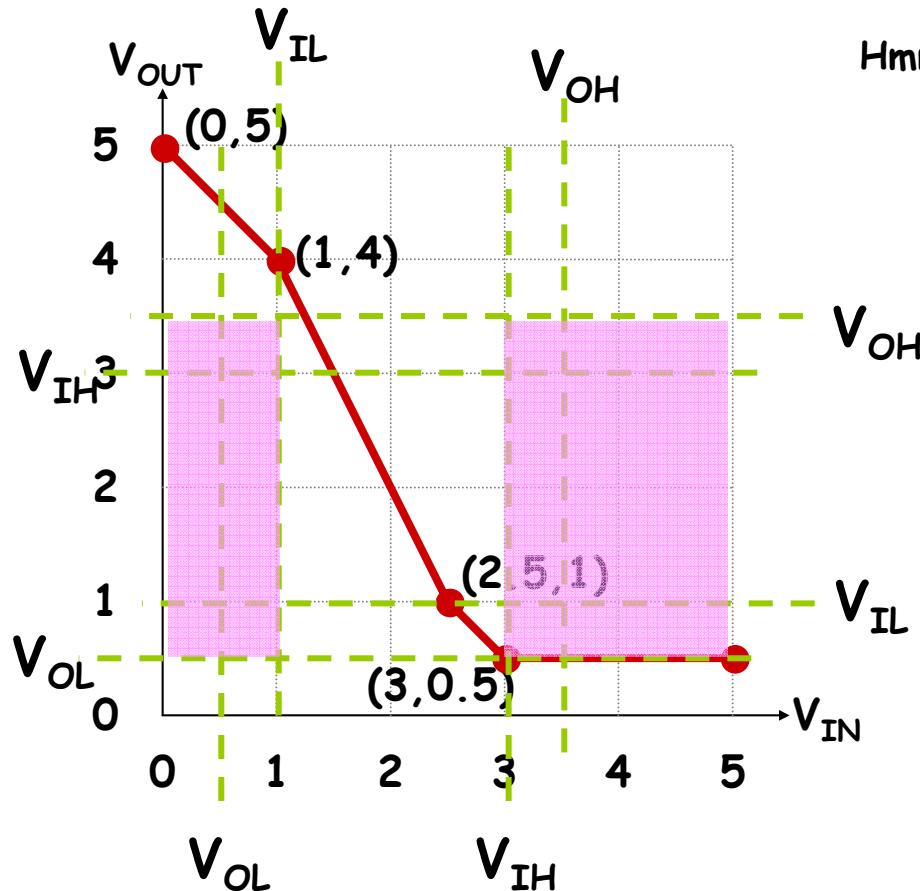
Plot of V_{out} vs. V_{in} where each measurement is taken after any transients have died out.

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 → whole thing works”
 - static discipline
 - digital inputs, outputs; restore marginal input voltages
 - complete functional spec
 - valid inputs lead to valid outputs in bounded time