ECE 152A ; [illegible]

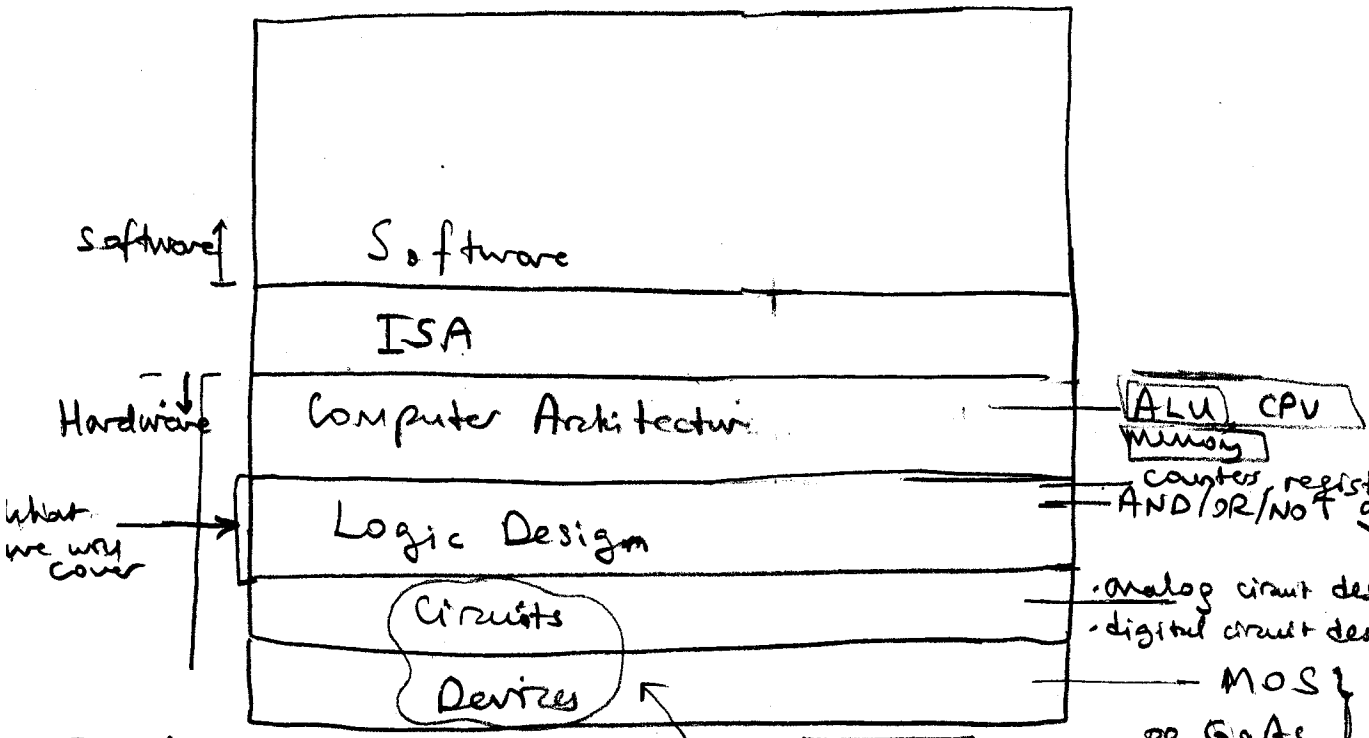
Lecture #1 [Reading: Ch 1.1, 2.1-2.8, 3.5]

(10 minutes)

- Welcome to ECE 152A.

— Digital design principles: First, let's place the content of this course in context.

Example.




(A) - Analog underlying world → Digital [0,1]. - for simplicity of design to build complex systems

(B) circuits, devices must expert to us abstraction of AND/OR/NOT gates.

underlying technology changes very rapidly.

1. "Moore's Law": Integrated circuit technology doubles the # transistors per $cm^2$ roughly every 1.5-2 years (exponential increase in time)

   Currently, we're at ≈ 10's of millions of transistors/$cm^2$. (2004)

   (gate length: ≈ 0.10 μm.)

   - Increase due to: 1) device innovations, 2) fabrication technology innovations.

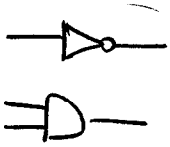
* Enormous implication because the fabrication cost of a chip

Scales with its area. So, if you can pack twice the number of transistors per area in 18 months / 2 years then the cost of digital functionality is driven down. Considered the main reason for ↓ of digital hardware over the last 3 years. —

2 New technologies emerging:

Ex1 Rick Weiss at Princeton: computing w/ bacteria. 1-2 sentences

- Biological devices are very slow, but capable of massively parallel operation.

- So: Ex1  — build using biological means (bacteria)

- Whatever new technology emerges, we expect that it export common abstractions such as AND, OR, NOT gates.

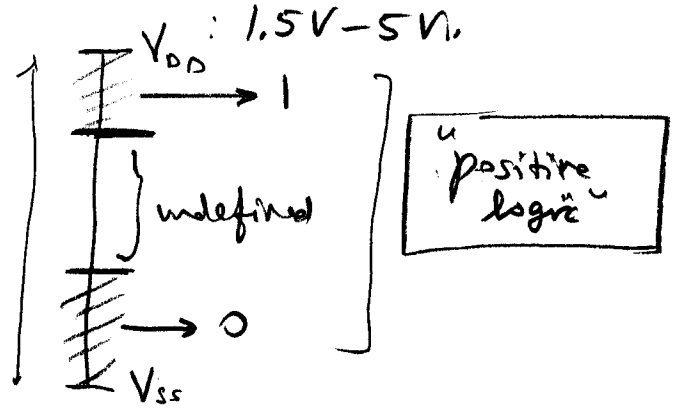
Ex2 IBM: • Amorphous computing. — check out their web site. "Biologically inspired".

---

- To get to the existing frontiers, start out at humble places: (10 minutes)

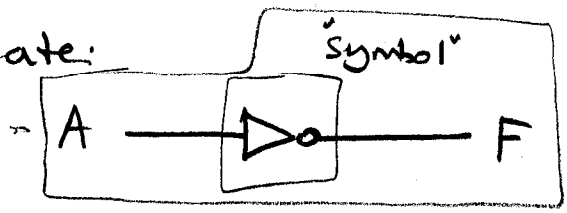
Digital logic: $\mathbb{B} = \{0, 1\}$.

Values are called Boolean



Basic logic operations: NOT, AND, OR

- NOT gate:

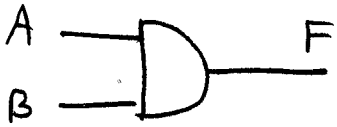



$F = \bar{A}$ (or $F = A'$)

Truth table

| A (input) | F (output) |
|---|---|
| 0 | 1 |
| 1 | 0 |

- AND gate:



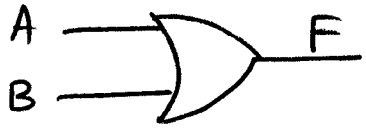

$F = A \cdot B$

(or $F = AB$)

| A | B | F |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

(ask class)

- OR gate:




$F = A + B$

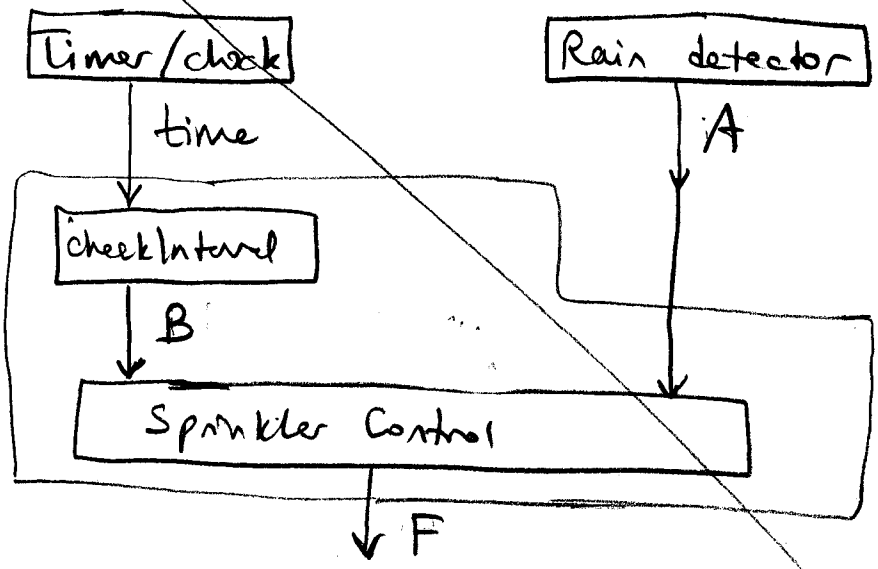
| A | B | F |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

skip

EX1 "Sprinkler system": [Logic design is widely used to build embedded systems / controllers of simple systems; not just computers (complex systems like ...)]

- Sprinkler system in the garden:
  - I want the sprinklers to turn on between 11:00 PM - 1:00 AM and if it's not raining.

[Ask class, what are the input devices?]




$A = \begin{cases} 1 & \text{if raining} \\ 0 & \text{otherwise} \end{cases}$ $B = \begin{cases} 1 & 11{:}00\ PM \le t \le 1{:}00\ AM \\ 0 & \text{otherwise} \end{cases}$

inputs

$F = A' \cdot B$

not raining

| A | B | F |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

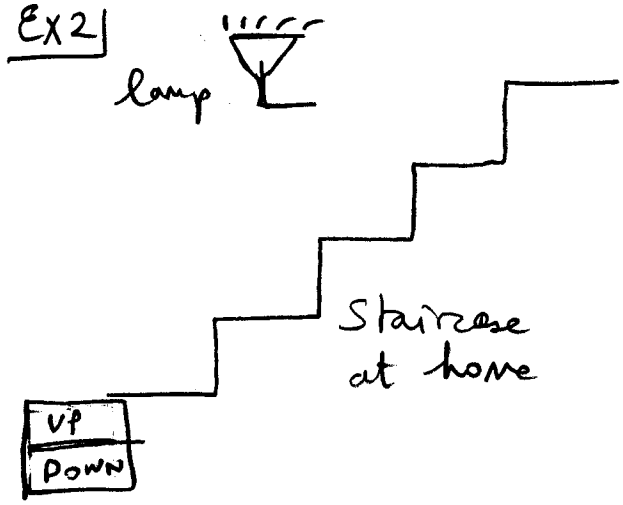
not raining

Skip

(15 minutes)

---

EX2




— Design the controller circuit for the lamp such that I can turn the lamp on or off from either switch.

1 Understand the problem statement by creating some examples.

a Very important to assign variables clearly to real-world events.

Ex] Let $F = \begin{cases} 1 & \text{lamp is on} \\ 0 & \text{lamp is off.} \end{cases}$

b When working on examples, you need to determine the initial conditions — OR convention used.

Ex] Decide that: if BOTH s_u and s_d are OFF, then the lamp will be off.

$s\_u = \begin{cases} 1 & \text{if switch upstairs is in up position} \\ 0 & \text{else} \end{cases}$

$s\_d = \begin{cases} 1 & \text{if switch downstairs is in up position} \\ 0 & \text{else} \end{cases}$

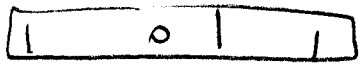
Answer

| s_u | s_d | F |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

c Now, play with it:

- Assume I go upstairs and flip the switch to turn it on: 1 → 0. Then, the lamp must turn on. 1 → 1

- Now, I go downstairs and want to turn off the lamp. — 1 → 0 I flip the switch. 1 → 1

Now, I go upstairs again and want to turn on the lamp. Flip the switch. 


- Try many other combinations. Make sure your truth table works.

* 2 This is the truth table of an (XOR) gate. ⟹ F=1 iff inputs are different

- The truth table gives a complete characterization of the input/output behavior of a combinational circuit. But it is not compact (grows exponentially in size with the # input variables) and is hard to manipulate.
- Tool that we use is Boolean algebra.

(1) Associative Law:

$$A + (B + C) = (A + B) + C$$

$$A \cdot (B \cdot C) = (A \cdot B) \cdot C$$

(2) Commutative Law:

$$A + B = B + A$$

$$A \cdot B = B \cdot A$$

(3) Distributive Law:

$$A \cdot (B + C) = A \cdot B + A \cdot C$$

$$\boxed{A + (B \cdot C) = (A + B) \cdot (A + C)} \leftarrow \text{Important.}$$

(4) $A + 0 = A$ $\quad A + 1 = 1$

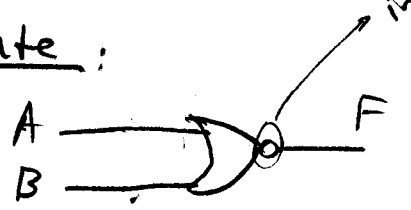
$A \cdot 0 = 0$ $\quad A \cdot 1 = A$

(Ask class)

(10 minutes)

## Other logic gates:

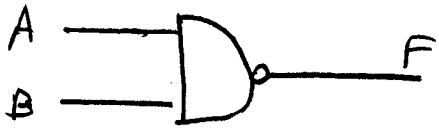
### NOR gate:




$F = (A+B)'$

| A | B | F |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

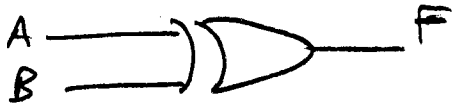
### NAND gate:




$F = (A \cdot B)'$

| A | B | F |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

### XOR gate:

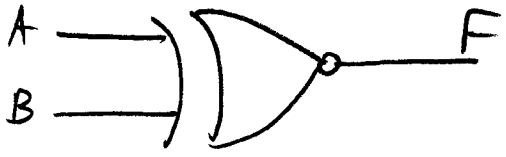



$F = A \oplus B$

(F is 1 iff A and B are different)

| A | B | F |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

### XNOR gate:

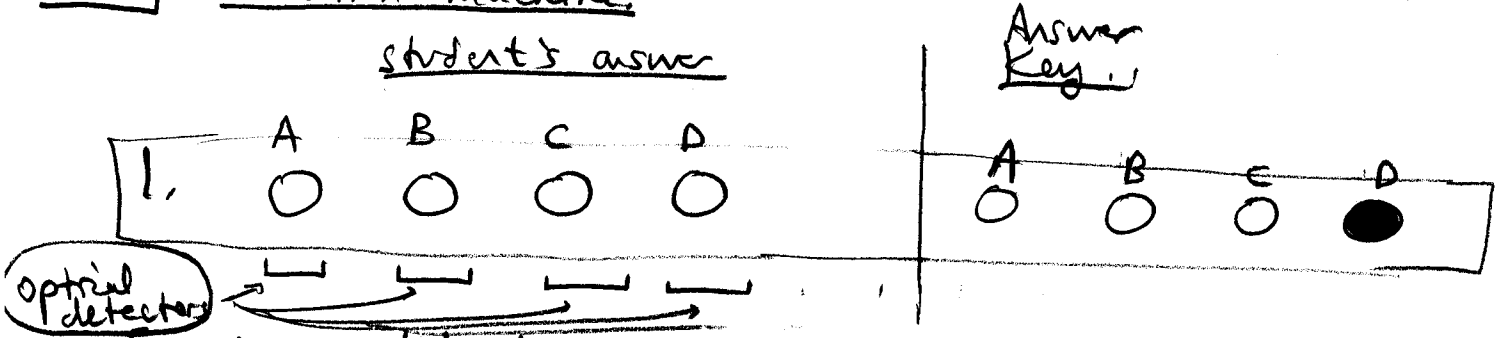



$F = (A \oplus B)'$

(F=1 iff A and B are the same)

| A | B | F |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

EX3] Scantron machine (10 minutes)




— Say we want to design part of this machine in digital hardware.

— Design the digital circuit that compares the student's answer with the key and tells you whether the answer is correct or not.

---

Inputs: [1] optical detectors,

$X1$ $X2$ $X3$ $X4$

$$X1 = \begin{cases} 1 & \text{if } \overset{\text{spot}}{A} \text{ is detected as "dark"} \\ 0 & \text{else.} \end{cases}$$

$$X4 = \begin{cases} 1 & \text{if } \overset{\text{spot}}{D} \text{ is detected as "dark"} \\ 0 & \text{else.} \end{cases}$$

[2] Answer key: $Y1$ .... $Y4$

$$Y1 = \begin{cases} 1 & \text{if correct answer has A as } \underline{\text{dark}} \\ 0 & \text{else} \end{cases}$$

$$Y4 = \begin{cases} 1 & \text{if } \quad D \\ 0 & \text{else.} \end{cases}$$

outputs:

Let $Z = \begin{cases} 1 & \text{if } \overset{\text{student}}{\text{answer}} \text{ is correct} \\ 0 & \text{else} \end{cases}$

Key step: [ASK CLASS]

$Z = 1$ iff "each choice matches the answer key"

$X1 == Y1$ and $X2 == Y2$, and ... $X4 == Y4$

$(X1 \oplus Y1)'$ · $(X2 \oplus Y2)'$ · $(X3 \oplus Y3)'$ · $(X4 \oplus Y4)'$

$$Z = [(X1 \oplus Y1)' \cdot (X2 \oplus Y2)'] \cdot [(X3 \oplus Y3)' \cdot (X4 \oplus Y4)']$$

Circuit Schematic: