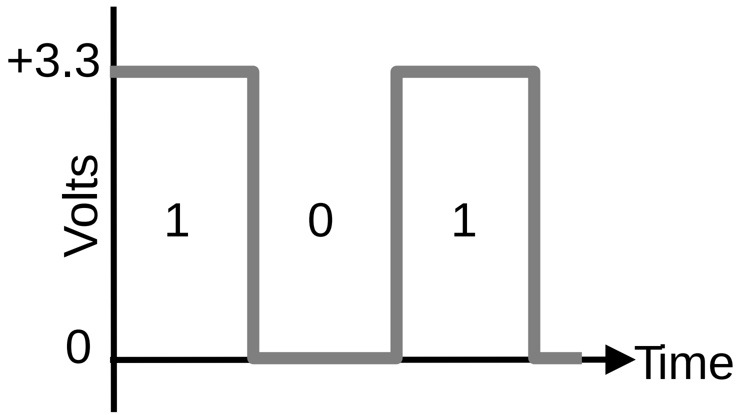
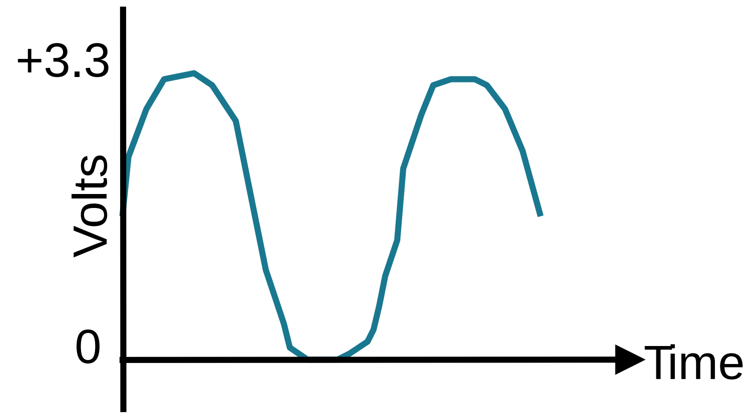
Course Motivation

- Smaller, faster, cheaper hardware has enabled so many advances in electronics
  - Computers & phones
  - Vehicles (cars, planes)
  - Robots
  - Portable & household electronics

- An *introduction* to digital logic design
  - **Lecture:** How to think about hardware, basic higher-level circuit design techniques – preparation for EE/CSE469
  - **Lab:** Hands-on FPGA programming using Verilog – preparation for EE/CSE371

Digital vs. Analog





Digital:

Discrete set of possible values

Analog:

Values vary over a continuous range

Binary (2 values):

On, 3.3 V, high, TRUE, "1"
Off, 0 V, low, FALSE, "0"

Digital vs. Analog Systems

- Digital systems are more reliable and less error-prone
  - Slight errors can cascade in Analog system
  - Digital systems reject a significant amount of error; easy to cascade
- Computers use digital circuits internally
  - CPU, memory, I/O
- Interface circuits with “real world” often analog
  - Sensors & actuators

This course is about logic design,
not system design (processor architecture),
and not circuit design (transistor level)

Digital Design: What's It All About?

- Create an implementation using a set of building blocks given a functional description and constraints
- Digital design is in some ways more art than a science
  - The creative spirit is in combining primitive elements and other components in new ways to achieve a desired function
- However, unlike art, we have objective measures of a design (*i.e.* constraints):
  - Performance
  - Power
  - Cost

Digital Design: What's It All About?

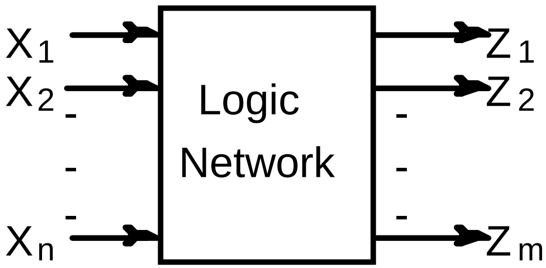
- How do we learn how to do this?
  - Learn about the building blocks and how to use them
  - Learn about design representations
  - Learn formal methods and tools to manipulate representations
  - Look at design examples
  - Use trial and error – CAD tools and prototyping (practice!)

Lecture Outline

- Course Logistics
- Course Motivation
- Combinational Logic Review
- Combinational Logic in the Lab

Combinational vs. Sequential Logic

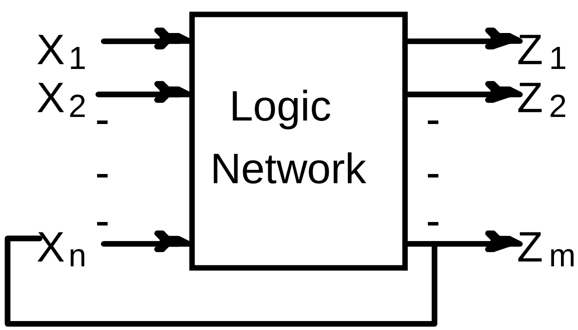
- Combinational Logic (CL)



Network of logic gates without feedback.

Outputs are functions only of inputs.

- Sequential Logic (SL)



The presence of feedback introduces the notion of “state.”

Circuits that can “remember” or store information.

Representations of Combinational Logic

- Text Description
- Circuit Description
  - ~~Transistors~~ Not covered in 369
  - Logic Gates
- Truth Table
- Boolean Expression

- *All are equivalent!*

Example: Simple Car Electronics

- Door Ajar (DriverDoorOpen, PassengerDoorOpen)

- High Beam Indicator (LightsOn, HighBeamOn)

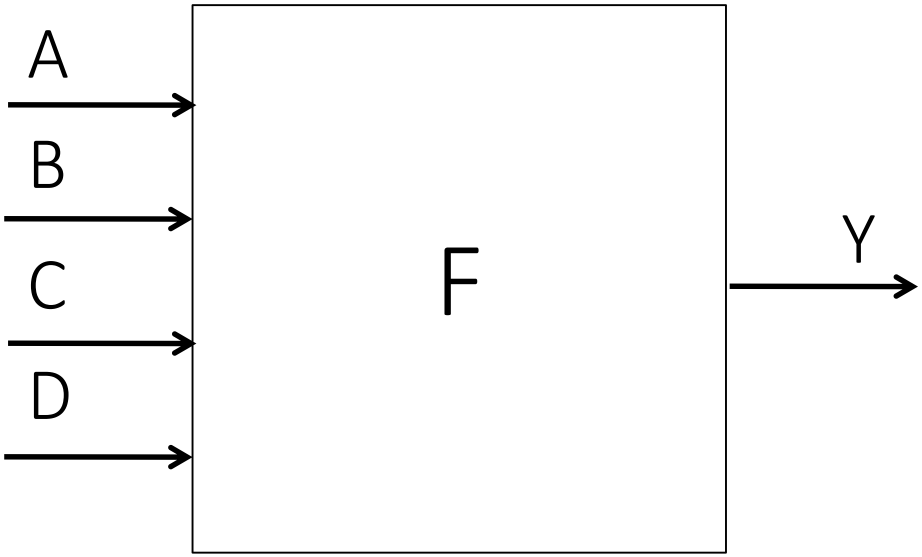
- Seat Belt Light (DriverBeltIn, PassengerBeltIn, Passenger)

Truth Tables

- Table that relates the inputs to a combinational logic (CL) circuit to its output
  - Output *only* depends on current inputs
  - Use abstraction of 0/1 instead of high/low voltage
  - Shows output for *every* possible combination of inputs ("black box" approach)

- How big is the table?
  - 0 or 1 for each of N inputs
  - Each output is a separate function of inputs, so don't need to add rows for additional outputs

CL General Form



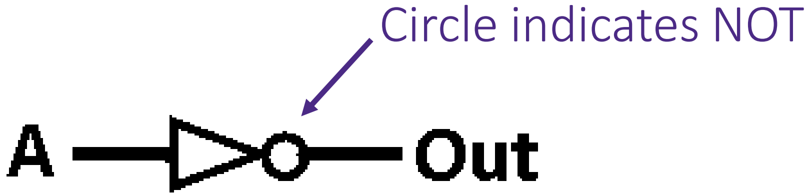
If N inputs, how many distinct functions F do we have?

a	b	c	d	y
0	0	0	0	F(0,0,0,0)
0	0	0	1	F(0,0,0,1)
0	0	1	0	F(0,0,1,0)
0	0	1	1	F(0,0,1,1)
0	1	0	0	F(0,1,0,0)
0	1	0	1	F(0,1,0,1)
0	1	1	0	F(0,1,1,0)
1	1	1	1	F(0,1,1,1)
1	0	0	0	F(1,0,0,0)
1	0	0	1	F(1,0,0,1)
1	0	1	0	F(1,0,1,0)
1	0	1	1	F(1,0,1,1)
1	1	0	0	F(1,1,0,0)
1	1	0	1	F(1,1,0,1)
1	1	1	0	F(1,1,1,0)
1	1	1	1	F(1,1,1,1)

Logic Gates (1/2)

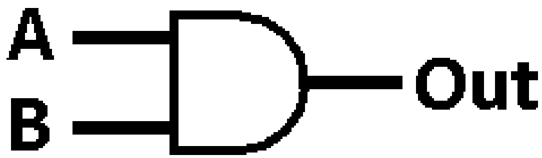
- Special names and symbols:

NOT



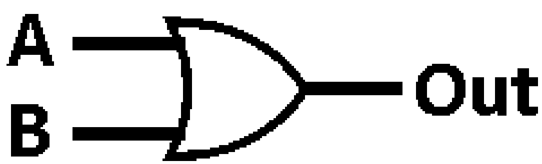
A	Out
0	1
1	0

AND



A	B	Out
0	0	0
0	1	0
1	0	0
1	1	1

OR

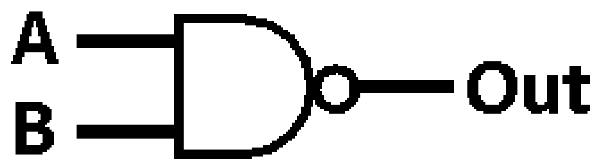


A	B	Out
0	0	0
0	1	1
1	0	1
1	1	1

Logic Gates (2/2)

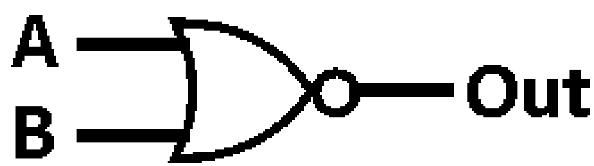
- Special names and symbols:

NAND



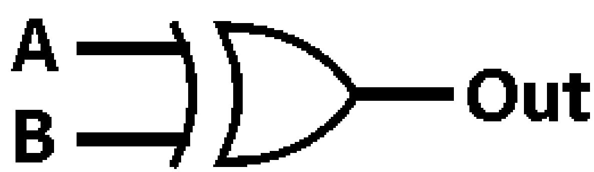
A	B	Out
0	0	1
0	1	1
1	0	1
1	1	0

NOR



A	B	Out
0	0	1
0	1	0
1	0	0
1	1	0

XOR



A	B	Out
0	0	0
0	1	1
1	0	1
1	1	0

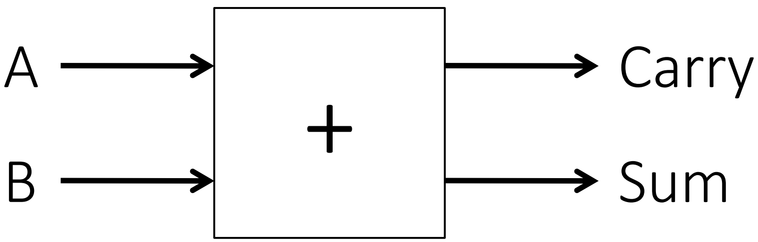
More Complicated Truth Tables

3-Input Majority

How many rows?

A	B	C	Out
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

1-bit Adder



A	B	Carry	Sum
0	0		
0	1		
1	0		
1	1		

Boolean Algebra

- Represent inputs and outputs as variables
  - Each variable can only take on the value 0 or 1
- Overbar is NOT: "logical complement"
  - If A is 0, then $\overline{\mathrm{A}}$ is 1 and vice-versa
- Plus $(+)$ is 2-input OR: "logical sum"
- Product $(\cdot)$ is 2-input AND: "logical product"
- All other gates and logical expressions can be built from combinations of these
  - e.g. $\mathrm{A\ XOR\ B} = \mathrm{A} \oplus \mathrm{B} = \overline{\mathrm{A}}\mathrm{B} + \overline{\mathrm{B}}\mathrm{A}$