Intro to Digital Design
Combinational Logic

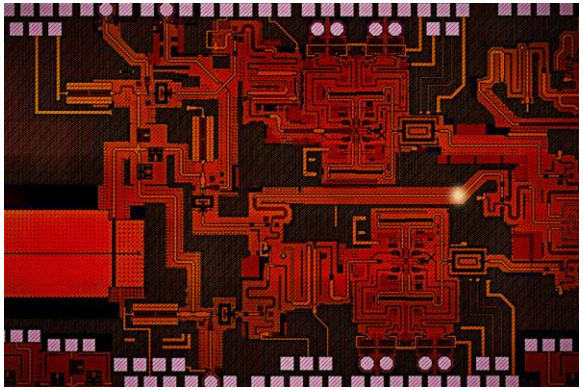
Instructor: Clarice Larson

Data transfer system connects silicon chips with a hair's-width cable

"Researchers have developed a data transfer system that can transmit information 10 times faster than a USB. The new link pairs high-frequency silicon chips with a polymer cable as thin a strand of hair. The system may one day boost energy efficiency in data centers and lighten the loads of electronics-rich spacecraft."

"The researchers suggest "data-dense" applications, like server farms, could be early adopters of the new links, since they could dramatically cut data centers' high energy demands. The link could also be a key solution for the aerospace and automotive industries, which place a premium on small, light devices …"

https://news.mit.edu/2021/data-transfer-system-silicon-0224



Lecture Outline

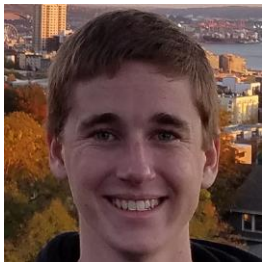
- Course Logistics
- Course Motivation
- Combinational Logic Review
- Combinational Logic in the Lab

Course Staff

- **Instructor:** Clarice Larson
- **TAs:**

Hafsa

Laksh



Matthew

Noah

Sahar

Suliman

Virtual Reality

- Zoom will be used for Lectures, Office Hours, Demos
- Students will not have access to CSE lab spaces
  - Install Quartus on your machine
- Kits can be picked up in person or shipped based on your preference (**Pre-Course Survey**)

Course Information

- **Instructor:** Clarice Larson (clarice@cs.uw.edu)
- **TAs:** Matthew Cinnamon (cinnam@cs.uw.edu)
  Laksh Gupta (lg2000@cs.uw.edu)
  Hafsa Khan (hafsa@cs.uw.edu)
  Sahar Osman (saharo@cs.uw.edu)
  Suliman Osman (osman92@cs.uw.edu)
  Noah Ponto (ponton@cs.uw.edu)
- **Lab Hours:** Wed & Thu 2:30-5:20 pm (Zoom)
- **Canvas:** https://canvas.uw.edu/courses/1448815
- **Discussion:** https://edstem.org/us/courses/4902/discussion/

Grading

- Labs (66%)
  - 6 regular labs – 1 week each
    - Labs 2-3: 60 points each, Labs 1, 4-6: 100 points each
  - 1 "final project" – 2 weeks
    - Lab 7 Check-In: 10 points, Lab 7: 150 points
- 3 Quizzes (no final exam)
  - Quiz 1 (10%): 20 min in class on April 27
  - Quiz 2 (10%): 25 min in class on May 18
  - Quiz 3 (14%): 40 min in class on June 1

Labs

- Labs are a combination of report + demo
  - Submit materials via Canvas Wednesdays before 2:30 pm
    - This is *before* your demo
  - 10-minute demos done in lab sections (Pre-course Survey)
- Each student will get a lab kit for the quarter
  - Install software on laptop (Windows or VM)
  - Lab sections are for access to TAs
- Penalties on lab submissions and demos:

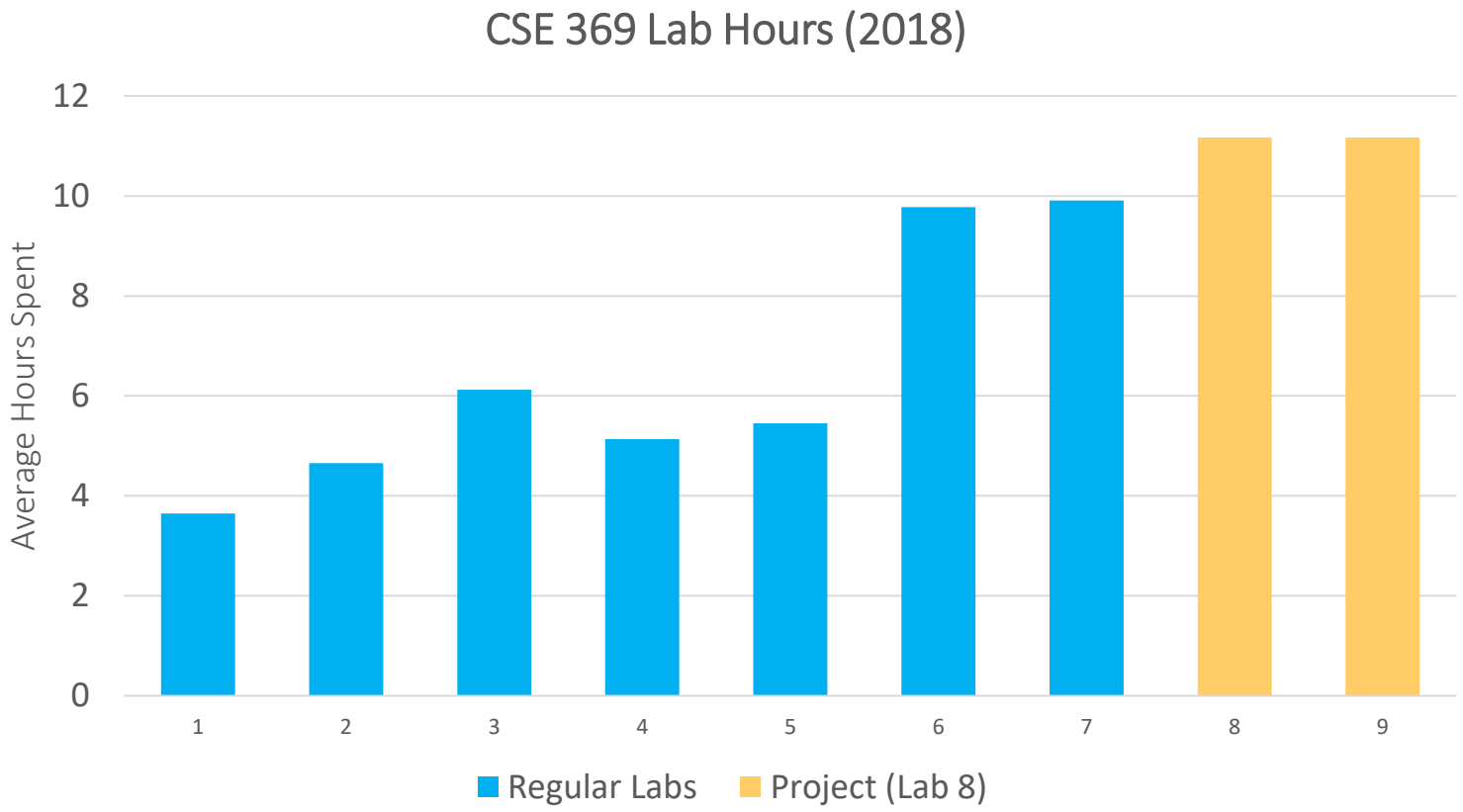
Lateness	$<$ 24 hr	$<$ 48 hr	$<$ 72 hr	$\geq$ 72 hr
Penalty	10%	30%	60%	100%

Collaboration Policy

- Labs and final project are to be completed *individually*
  - Learn by doing
  - Violation of these rules is grounds for failing the class
- OK:
  - Discussing lectures and/or readings, studying together
  - *High-level* discussion of general approaches
  - Help with debugging, peculiarities with tools, etc.
- Not OK:
  - Developing a lab together
  - Giving away solutions or having someone else do your lab for you

Course Workload

- The workload ramps up significantly towards the end of the quarter:



Lecture Outline

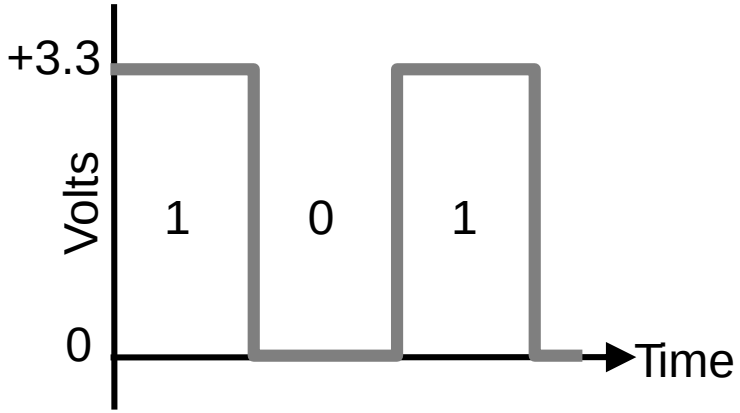
- Course Logistics
- Course Motivation
- Combinational Logic Review
- Combinational Logic in the Lab

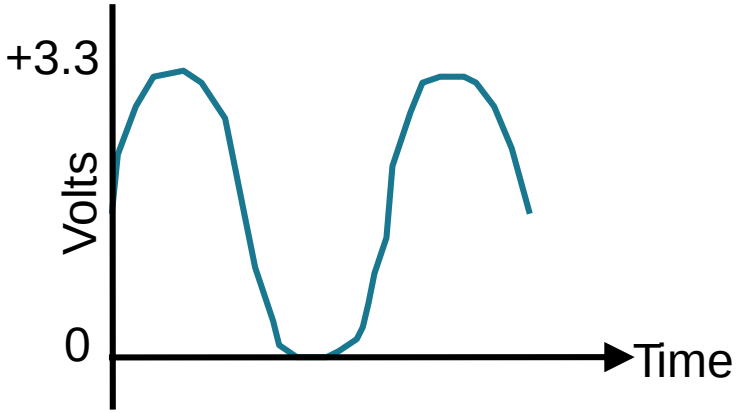
Course Motivation

- 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

Binary (2 values):

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

Analog:

Values vary over a continuous range

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?

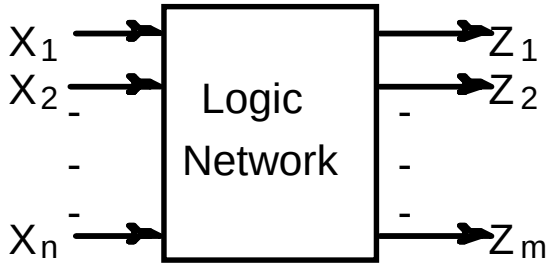
- How do we learn how to do this?
  - Learn about the building blocks and how to use them
  - Learn about design representations
  - Learn formal methods and tools to manipulate representations
  - Look at design examples
  - Use trial and error – CAD tools and prototyping (practice!)

Lecture Outline

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

Combinational vs. Sequential Logic

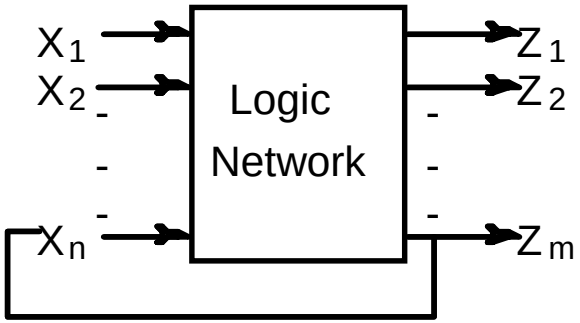
- Combinational Logic (CL)



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

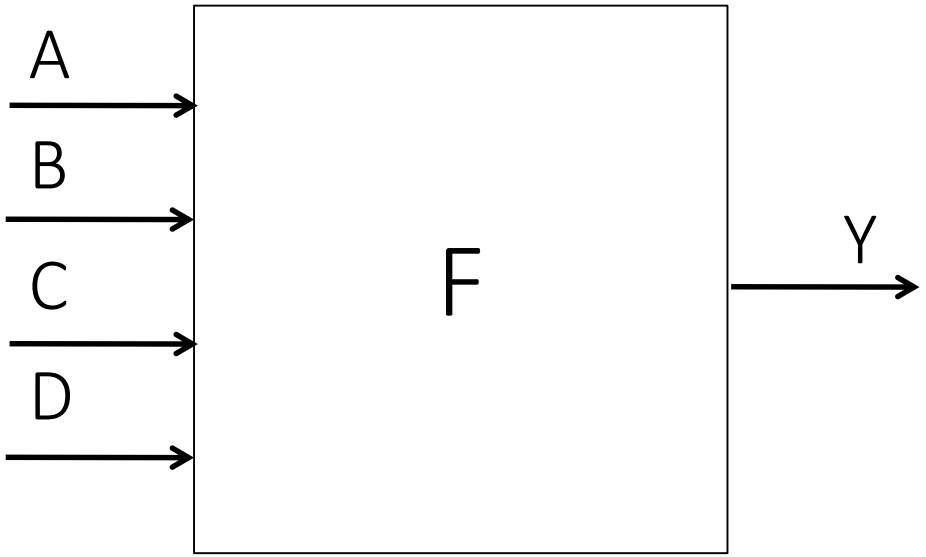
- Door Ajar (DriverDoorOpen, PassengerDoorOpen)
- High Beam Indicator (LightsOn, HighBeamOn)
- Seat Belt Light (DriverBeltIn, PassengerBeltIn, Passenger)

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



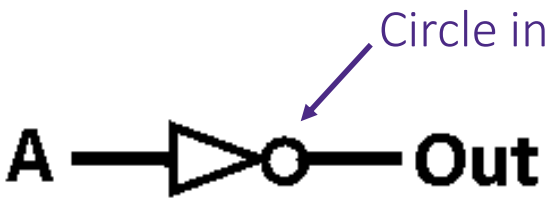
If N inputs, how many distinct functions F do we have?

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

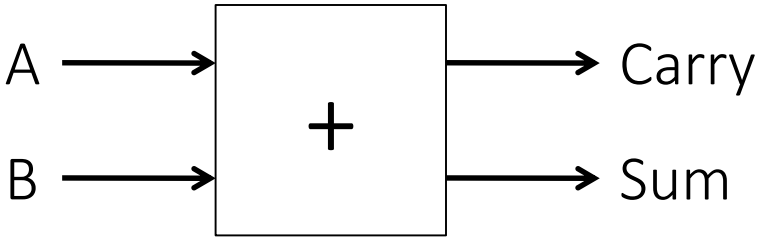
More Complicated Truth Tables

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. $\mathrm{A\ XOR\ B} = \mathrm{A} \oplus \mathrm{B} = \overline{\mathrm{A}}\mathrm{B} + \overline{\mathrm{B}}\mathrm{A}$

Truth Table to Boolean Expression

- Read off of table
  - For 1, write variable name
  - For 0, write complement of variable
- *Sum of Products (SoP)*
  - Take rows with 1's in output column, sum products of inputs
  - $C = \overline{A}B + \overline{B}A$
- *Product of Sums (PoS)*
  - Take rows with 0's in output column, product the sum of the *complements of the inputs*
  - $C = (A + B) \cdot (\overline{A} + \overline{B})$

a	b	c
0	0	0
0	1	1
1	0	1
1	1	0

We can show that these are equivalent!

Basic Boolean Identities

- $X + 0 = X$
- $X + 1 = 1$
- $X + X = X$
- $X + \overline{X} = 1$
- $\overline{\overline{X}} = X$
- $X \cdot 1 = X$
- $X \cdot 0 = 0$
- $X \cdot X = X$
- $X \cdot \overline{X} = 0$

Basic Boolean Algebra Laws

- Commutative Law:

$$X + Y = Y + X \qquad X \cdot Y = Y \cdot X$$

- Associative Law:

$$X+(Y+Z) = (X+Y)+Z \qquad X \cdot (Y \cdot Z) = (X \cdot Y) \cdot Z$$

- Distributive Law:

$$X \cdot (Y+Z) = X \cdot Y + X \cdot Z \qquad X+YZ = (X+Y) \cdot (X+Z)$$

Advanced Laws (Absorption)

- $X + XY = X$
- $XY + X\overline{Y} = X$
- $X + \overline{X}Y = X + Y$
- $X(X + Y) = X$
- $(X + Y)(X + \overline{Y}) = X$
- $X(\overline{X} + Y) = XY$

Practice Problem

- Boolean Function: $F = \overline{X}YZ + XZ$

Truth Table:

X	Y	Z	F
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Simplification:

Technology Break

Lecture Outline

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

Why Is This Useful?

- Logic minimization: reduce complexity at gate level
  - Allows us to build smaller and faster hardware
  - Care about both # of gates, # of literals (gate inputs), # of gate levels, and types of logic gates

Why Is This Useful?

- Faster hardware?
  - Fewer inputs implies faster gates in some technologies
  - Fan-ins (# of gate inputs) are limited in some technologies
  - Fewer levels of gates implies reduced signal propagation delays
  - # of gates (or gate packages) influences manufacturing costs
  - Simpler Boolean expressions → smaller transistor networks → smaller circuit delays → faster hardware
  - Does the type of gate matter?

Does the Type of Gate Matter?

- Yes!

2-Input Gate Type	# of transistors
NOT	2
AND	6
OR	6
NAND	4
NOR	4
XOR	8
XNOR	8

- Can recreate all other gates using only NAND or only NOR gates
  - Called "universal" gates
  - *e.g.* A NAND $A = \overline{A}$, B NOR $B = \overline{B}$
  - DeMorgan's Law helps us here!

DeMorgan's Law

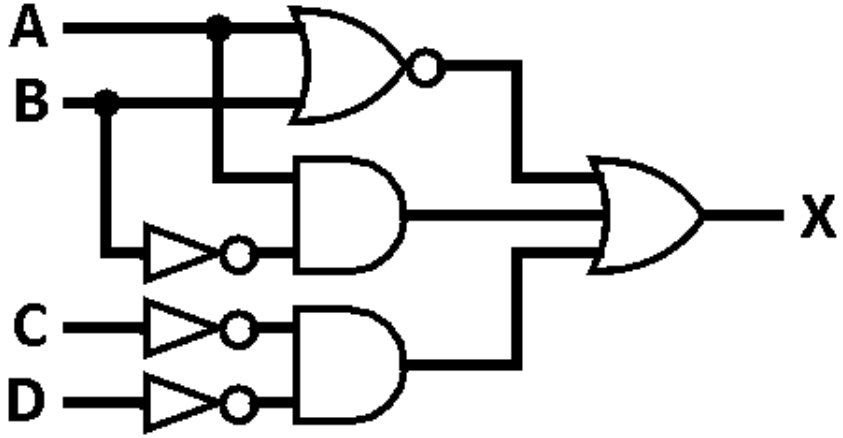
- $\overline{X + Y} = \overline{X} \cdot \overline{Y}$
- $\overline{X \cdot Y} = \overline{X} + \overline{Y}$

				NOR		NAND	
X	Y	$\overline{X}$	$\overline{Y}$	$\overline{X + Y}$	$\overline{X} \cdot \overline{Y}$	$\overline{X \cdot Y}$	$\overline{X} + \overline{Y}$
0	0	1	1	1		1	
0	1	1	0	0		1	
1	0	0	1	0		1	
1	1	0	0	0		0	

- In Boolean Algebra, converts between NAND/NOR and OR/AND expressions
  - $Z = \overline{(A + B + \overline{C}) \cdot (A + \overline{B} + \overline{C}) \cdot (\overline{A} + B + \overline{C})}$
  - $Z = \overline{A}\overline{B}C + \overline{A}BC + A\overline{B}C$
  - At gate level, can convert from NAND/NOR to OR/AND gates
  - "Flip" all input/output bubbles and "switch" gate

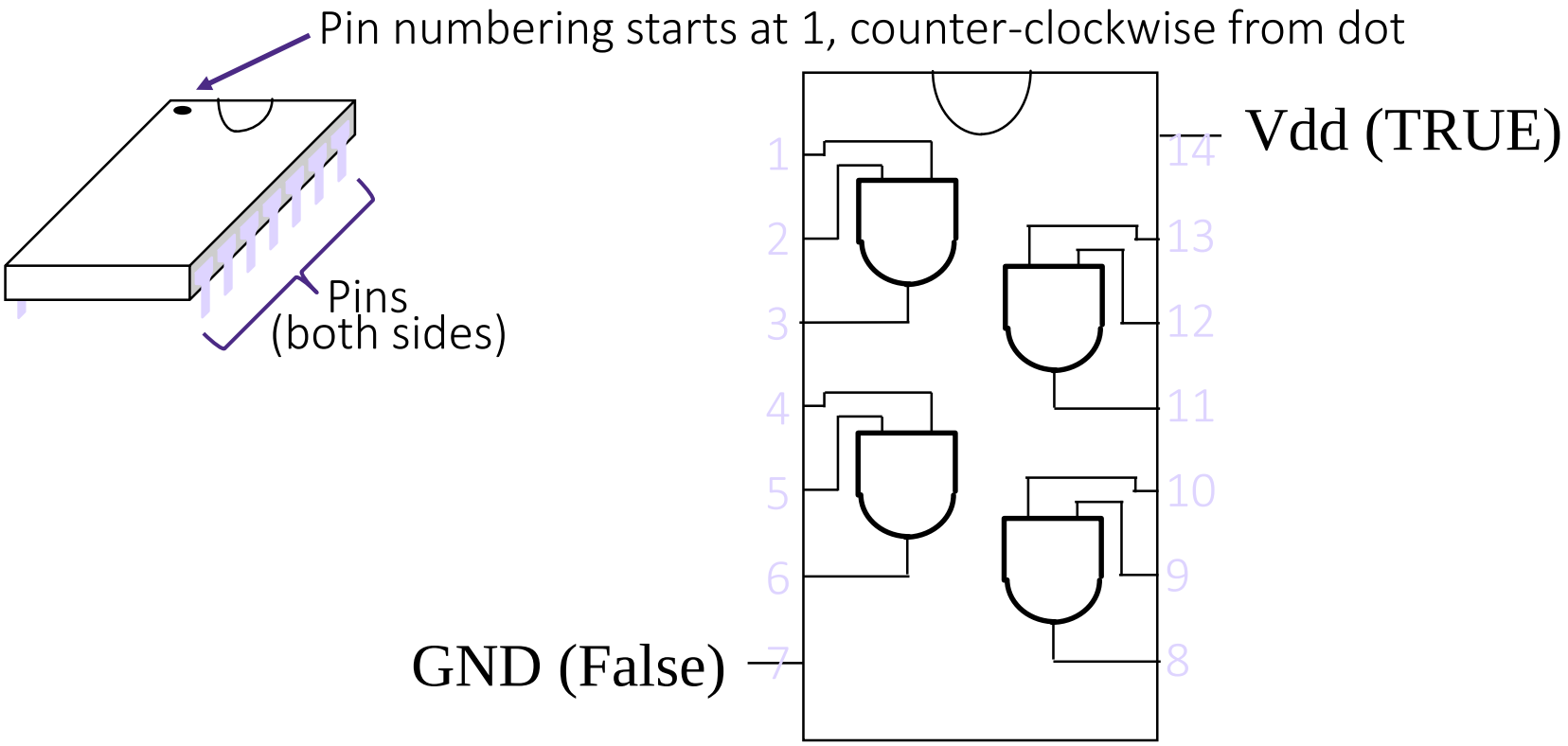
DeMorgan's Law Practice Problem

- Simplify the following diagram:



- Then implement with only NAND gates:

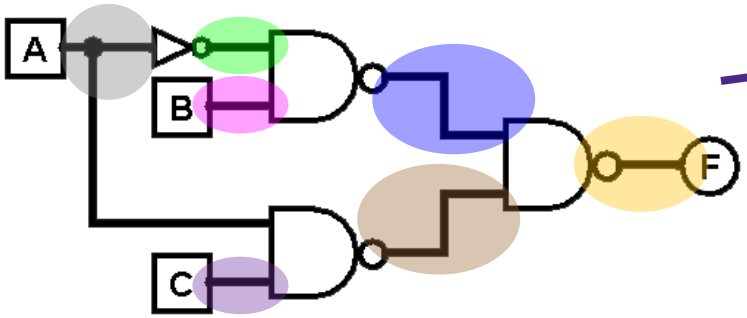
Transistor-Transistor Logic (TTL) Packages



- Diagrams like these and other useful/helpful information can be found on part data sheets
  - It's really useful to learn how to read these

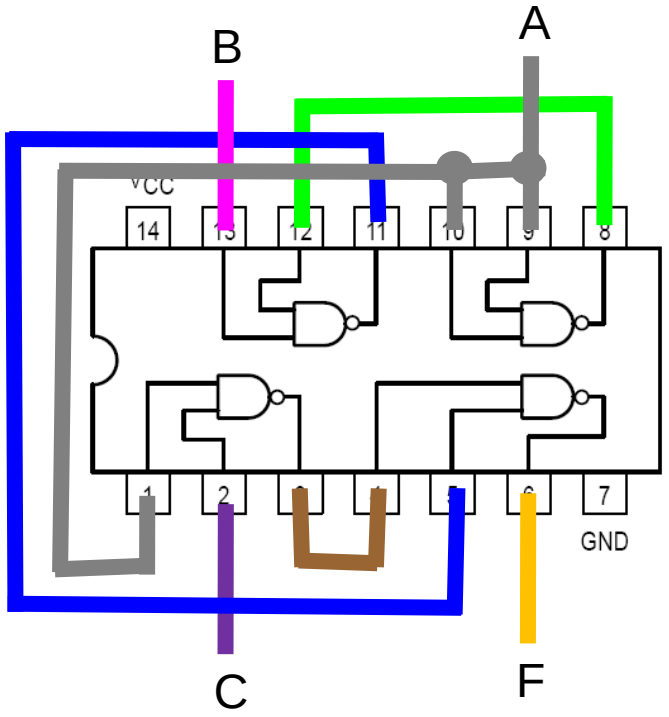
Mapping truth tables to logic gates

- Given a truth table:
  1) Write the Boolean expression
  2) Minimize the Boolean expression
  3) Draw circuit diagram with gates
  4) Map to available gates
  5) Determine # of packages and their connections

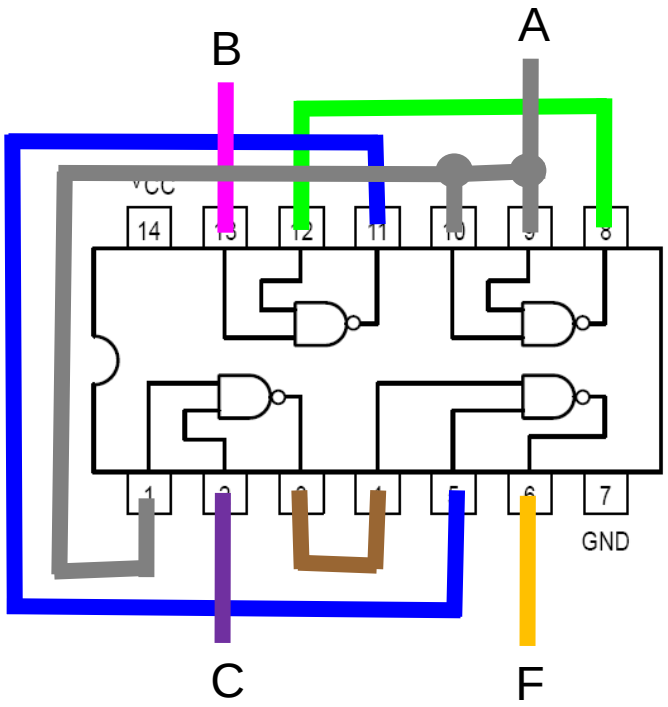


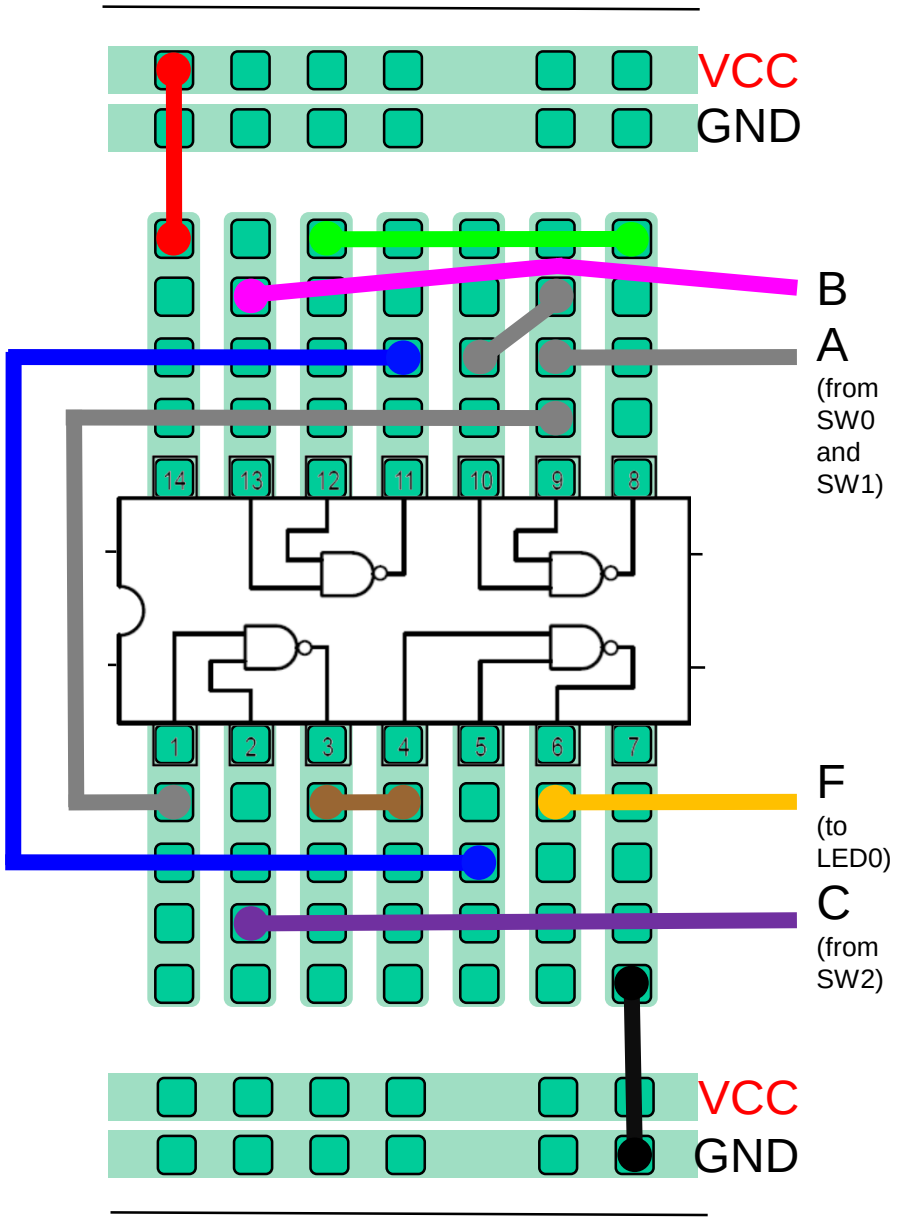
7 nets (wires) in this design

(4)



Breadboarding circuits





Summary

- Digital systems are constructed from Combinational and Sequential Logic
- Logic minimization to create smaller and faster hardware
- Gates come in TTL packages that require careful wiring