

COMP311: *COMPUTER ORGANIZATION!*

Lecture 1: Introduction

tinyurl.com/comp311-sp26

Welcome! 😊

- Target audience: COMP majors
- Pre-reqs: 210, 211

What is the course about?

High-level programming
languages

```
// High-level (C)
int add3(int a, int b)
{
    return a + b + 3;
}
```

Easy for humans to
read/write

Assembly
Low-level languages

```
# Matching RISC-V assembly
# a0 = a, a1 = b; return in a0

add    a0, a0, a1    # a0 = a0 + a1
addi   a0, a0, 3     # a0 = a0 + 3
ret                               # return
```

Human readable

Machine code

```
00000000 01011 01010
000 01010 0110011

0000000000011 01010
000 01010 0010011

0000000000000 00001
000 00000 1100111
```

Machine-readable

Why should we care?



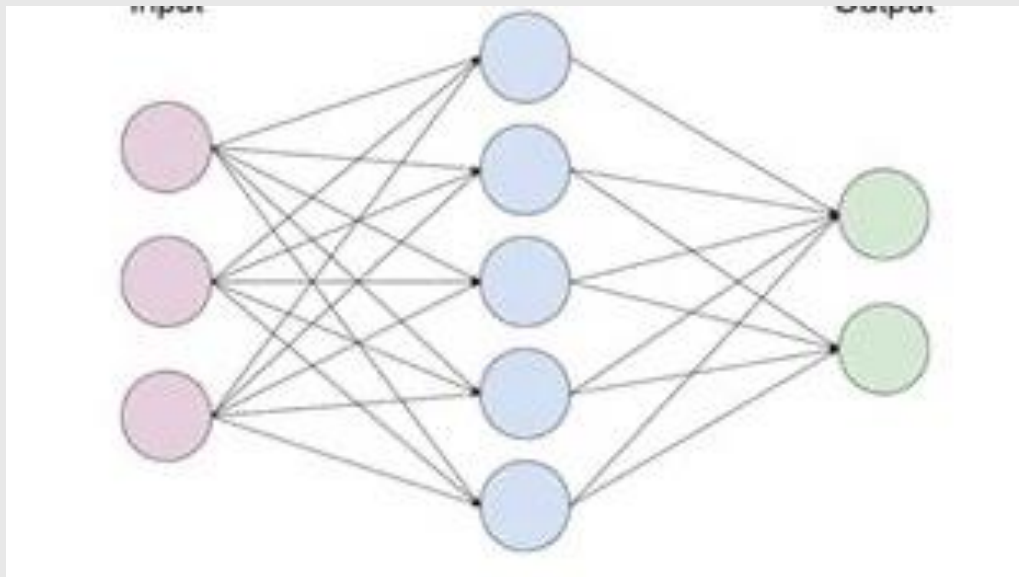
Intel Core i9-13900K



Apple M1 Pro

Artificial Intelligence

Advancements in computer architecture have been a driving force behind the advancement of AI technologies



Goals of this Course

- How does a computer work?
- How is data represented in a computer?
 - Numbers, strings, arrays, photos, music
- How is a **program** represented in a computer?
- What does a computer do with my program?
- How is data stored? How is data processed?
- Are there any limits to what a computer can do?



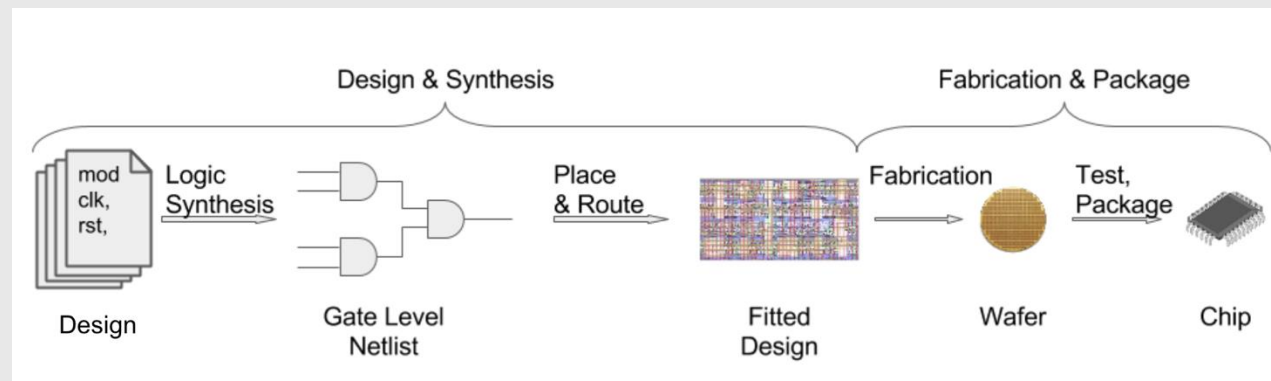
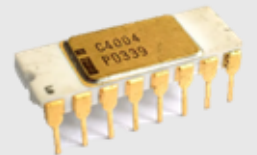
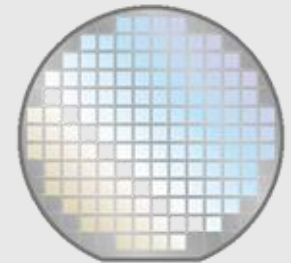
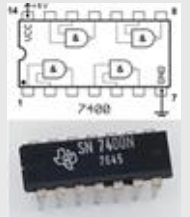
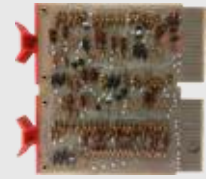
Logic Circuits and Digital Hardware

- Logic circuits are used to build computer hardware, as well as many other types of systems. All such systems are broadly classified as digital hardware.
- ***Digital*** derives from how ***information*** is represented as discrete ranges of electrical signals. Using discrete ranges provides robustness and scalability.



Digital Logic is Relatively New + Revolutionary!

- Digital hardware technology has evolved dramatically over the past 75 years!
 - *In the 1960s logic was built with discrete components, transistors and resistors.*
 - *Integrated circuits (ICs) made it possible to place a number of transistors on a single chip.*
 - *By 1970 it was possible to implement all circuitry needed to realize a microprocessor on a single chip.*
 - *Complexity has exploded!*



Moore's Law!

Number of Transistors in CPUs over Time



Course Theme #1: Demystify Computers!

Strangely, most people seem to be afraid of computers.

People only fear things they do not understand!



“I do not fear computers, I fear the lack of them.”

- Isaac Asimov (1920 - 1992)

“Fear is the main source of superstition, and one of the main Sources of cruelty. To conquer fear is the beginning of wisdom.”

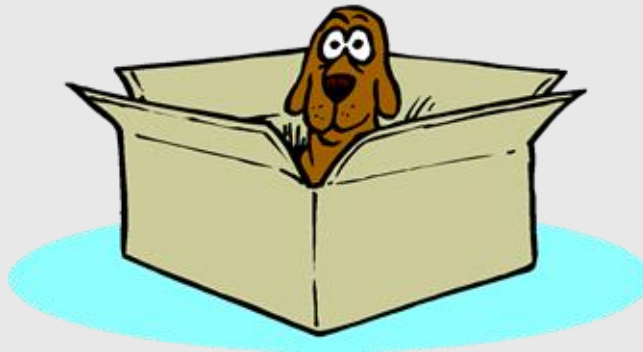
- Bertrand Russell (1872 -1970)

“Nobody knows exactly what’s going on because of computers!”

- Donald Trump

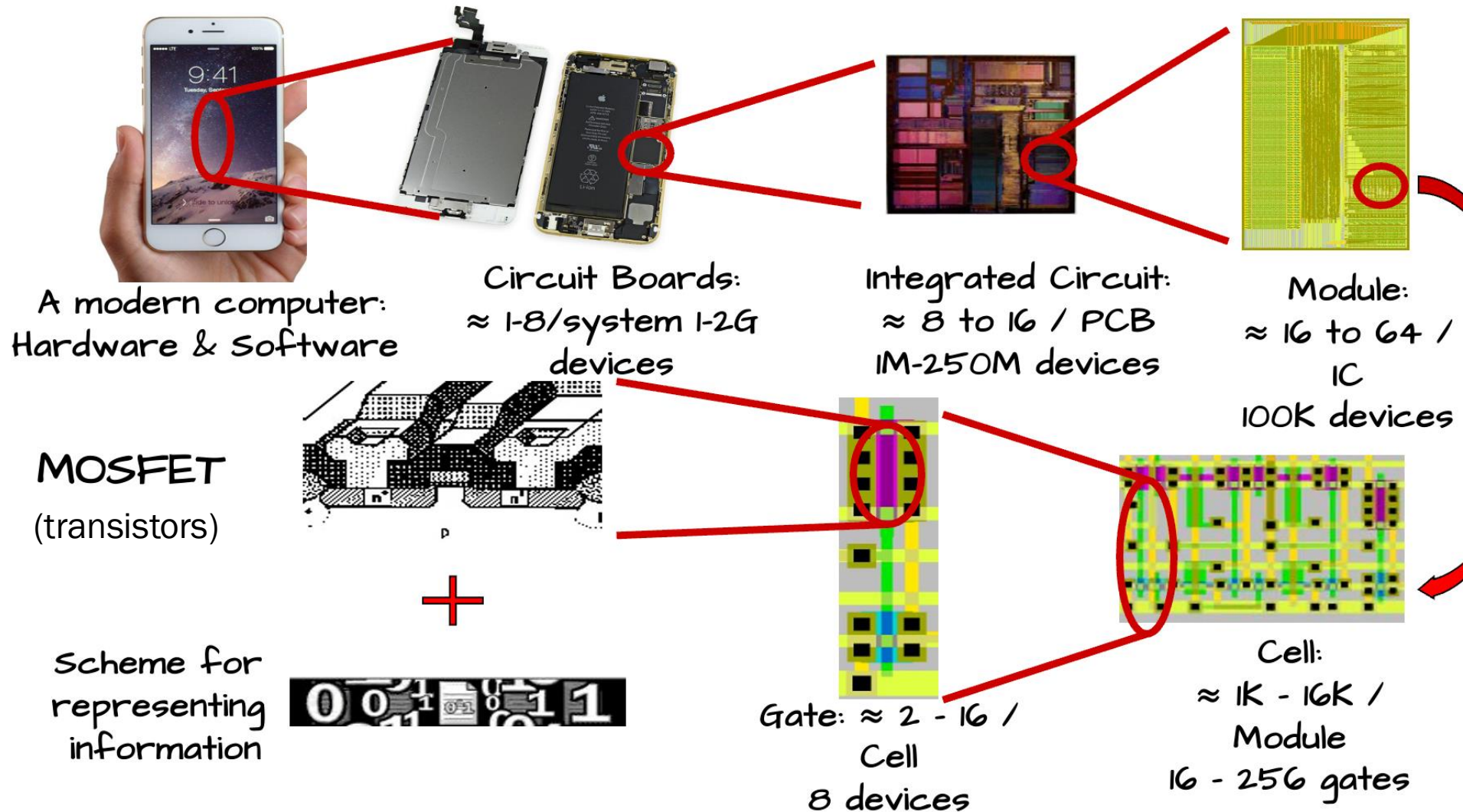
Course Theme #2: The Power of Abstraction!

Define a function, develop a roust implementation, and then put a box around it, and then reuse it (over and over!).



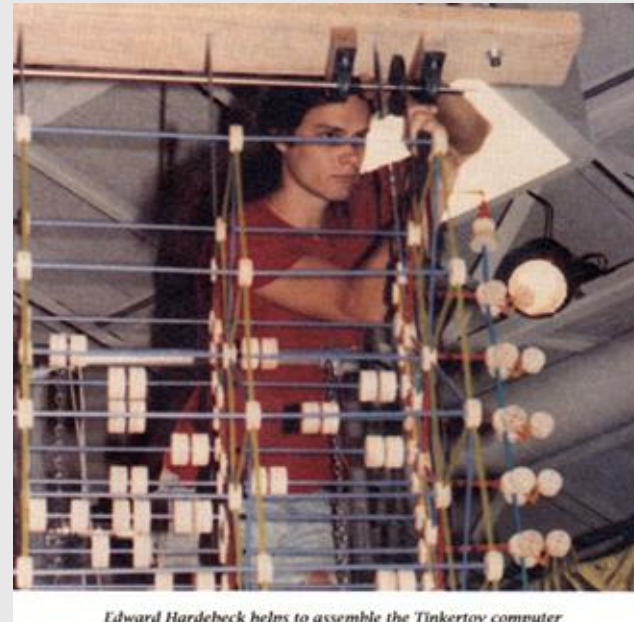
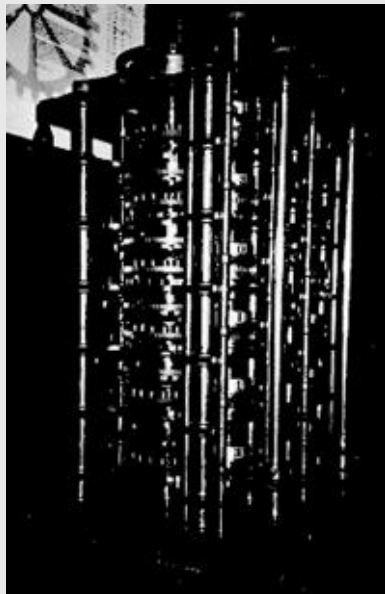
Abstraction enables us to create unfathomable systems, including computer hardware and software.

Course Theme #2: The Power of Abstraction!



Computational structures!

- What are the fundamental elements of computation?
- Can we define computation independent of implementation or the technology that it is implemented with?



Edward Hardebeck helps to assemble the Tinkertoy computer

Recall: Goals of this Course

- How does a computer work?
- How is data represented in a computer?
 - Numbers, strings, arrays, photos, music
- How is a **program** represented in a computer?
- What does a computer do with my program?
- How is data stored? How is data processed?
- Are there any limits to what a computer can do?

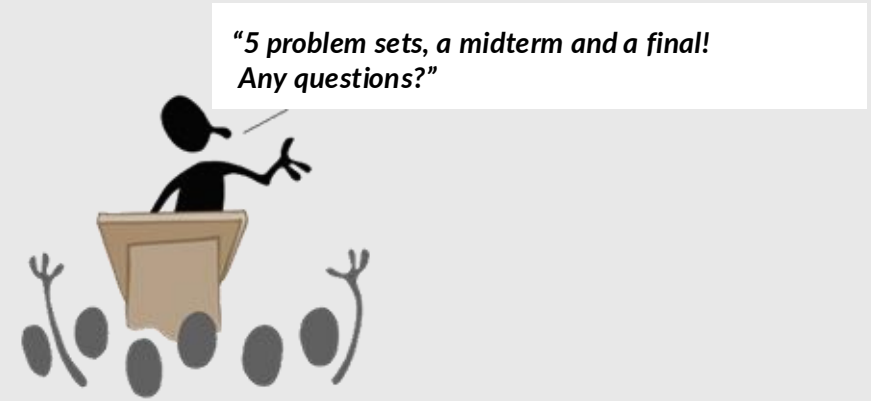


What does Digital Logic do?

- It processes bits: 1s and 0s
- It implements abstract logic functions
- It is robust, stable, and repeatable
- It doubles every two years.
- It represents, transforms, and stores *information*

What is “Information”?

information, n. *Knowledge communicated or received concerning a particular fact or circumstance.*



A Computer Engineer's definition:
Information resolves uncertainty.
Information is simply that "stuff" which cannot be predicted. The less likely a message is, the more information it conveys.

Quantifying Information

Suppose you're faced with N equally probable choices, and I give you a fact that narrows it down to M possibilities. Then you've been given:

$\log_2(N/M)$ bits of information

Examples:

- Outcome of a coin flip: $\log_2(2/1) = 1$ bit
- The roll of one die?
- Someone tells you that their 8-digit phone number is a palindrome?

*Information is measured
in bits (binary digits) =
number of 0/1's required
to encode choice(s)*



Information is the theoretical underpinning of why digital computers use bits! 😊

An exercise: How many bits?

With 1-2 classmates, take 5 minutes to answer question 1 on your worksheet.

You can take “N” to represent the number of students in this class, and “M” to be the number of students with the last name Ryan.

An exercise: How many bits?

With 1-2 classmates, take 5 minutes to answer question 1 on your worksheet.

You can take “N” to represent the number of students in this class, and “M” to be the number of students with the last name Ryan.

1.1 $\text{bits} = \lceil \log_2(N) \rceil$, where N is the number of students in the class.
if there are 100 students in the class, $\log_2(100) \sim 6.64 \Rightarrow 7$ bits!

1.2 N/M

What do programs really do?

At this point in your CS career, you look at a program specification and figure out what it does!

Q: What does this do? What is your strategy for figuring it out?

Hint: Try `f(36)`, `f(64)`, `f(100)`

```
int f(int x) {  
    int r;  
    int odd = 1;  
    for (r = 0; x >= odd; r++) {  
        x -= odd;  
        odd += 2;  
    }  
    return r;  
}
```

What do programs really do?

At this point in your CS career, you look at a program specification and figure out what it does!

Q: What does this do? What is your strategy for figuring it out?

Hint: Try `f(36)`, `f(64)`, `f(100)`

This is computing the square root

```
int f(int x) {  
    int r;  
    int odd = 1;  
    for (r = 0; x >= odd; r++) {  
        x -= odd;  
        odd += 2;  
    }  
    return r;  
}
```

How does a computer do it?

What would a computer do with this program specification?

```
int f(int x) {  
    int r;  
    int odd = 1;  
    for (r = 0; x >= odd; r++) {  
        x -= odd;  
        odd += 2;  
    }  
    return r;  
}
```

```
f:      mv      t0, a0  
        li      t1, 1  
        li      a0, 0  
        blt     t0, t1, return  
loop:   sub     t0, t0, t1  
        addi    t1, t1, 2  
        addi    a0, a0, 1  
        bge     t0, t1, loop  
return: ret
```

It translates it into a series of simple instructions!

Are there limits to computation?

- Will some new instruction be invented that fundamentally changes how fast computers solve problems?
- Can computers solve **any** well-specified problem?
 - COMP455
- Can we predict how long it will take for a computer to solve a given problem?
 - COMP210, COMP550
- Does there exist a new model of computation?





ADMINISTRATIVE DETAILS

Syllabus!

Course site:

<https://www.cs.unc.edu/~kakiryan/teaching/311-sp26/311-sp26.html>

Or

<https://tinyurl.com/comp311-sp26>

Course Structure

■ Assessment (65%):

- *6x Quizzes (QZ): 50%*
- *Final Exam (FN): 15%*

} Your final can replace up to two of your lowest quiz scores.

■ Practice and Preparation (30%):

- *6x Written Assignments (WA): 15%*
- *5x Labs (LAB): 15%*

■ In-Class Participation (5%):

- *1x Submission to Gradescope per class.*
 - 4 can be dropped, no questions asked.

Extensions + Late Days

- You have 8 days to use on any assignment in the course, with no explanation needed.
- You may request extensions for illness or personal life events so you don't have to use up your late days.
 - Extension form is linked in the syllabus
- If you are not sure if something qualifies for an extension, just fill out the form anyway, and we will get back to you.
- *You will not be penalized for submitting the assignment within the provided grace period if you receive an extension!*

Gradescope

- Make sure you are enrolled!
 - *For the correct section!*

The Team!

- LAs:

- *Zhi Yang, Jules Kim, Ben Chesser, Trey Anderson, Aashvi Jain, Joe Shanahan, Christopher McClanahan, Grace Odondi, Akil Dickerson, Gabrielle Stewart, Kiara Smith, Nathan Santiago, Betsy George, Matthew Thornton, Celine Keles, Rachit Pajbai, Joaquim D'Silva, Joseph Zhen, Sven Reyes, Chenghe Kong, Wai Phyo, Danny Xia, Issa Masumbuko*

- LA office hours will be held in CSXL RM SN143. Time TBD!

- I will send out an announcement once we schedule them!

- My office hours are Tuesdays 1-3pm in FB114.

Classroom Community

- Let's **all work together** to create a supportive and welcoming learning environment for **everyone** in this room!
- I expect all of you to be present, respectful & engaged in class, every day!
- We will transition between groupwork & lecture frequently.
 - *During group work:* the more conversation the better!
 - *During lecture:* Please lock in! Sound carries more than you think.
- Ask questions! Utilize the TAs and office hours!
 - *Come to office hours for conceptual help, too! We love talking to you all!*
- I am **always** open to feedback!



HEALTH & SAFETY

Face Masks

- Optional
- Use a mask often!

Emergency Procedures

- Evacuate (fire, chemical hazard)
- Shelter in place (weather related)
- Secure in place (armed and dangerous assailant)
 - run, hide (secure in place), fight

Acknowledgements

- Optional Textbook 1: Computer Organization and Design RISC-V Edition: The Hardware Software Interface by David Patterson and John Hennessey
- Textbook 2: The RISC-V Reader: An Open Architecture Atlas by David Patterson and Andrew Waterman.
- Connor McMahon, Leonard McMillan, Montek Singh

211 REVIEW 😊

Base

- A base is the number of different digits that a system can use to represent numbers
- Binary is base 2 because it has the digits 0 and 1
- Decimal base 10 because it has the digits 0-9
- Hex is base 16 because it has the digits 0-F

Binary Notation

- Always prepend binary numbers with “0b” or add a subscript of 2
 - *Ex: 0b 1001 or 1001₂*
- Place binary numbers in groups of four to make it easier to read
 - *Ex: 0b 1010 1111 0101 or 0b1010_1111_0101*
 - *If you cannot evenly break up the number into groups of four, start making the groups from the right*
 - *Ex: 0b10_1010_0011*

Hex Notation

- Always prepend hex numbers with “0x” or add a subscript of 16
 - *Ex: 0xE2A or $E2A_{16}$*

Decimal Notation

- You can indicate that a number is a decimal value by adding a subscript of 10
 - *Ex: 54_{10}*
- If a number does not have a leading “0b” or “0x” and does not have a subscript, it is assumed to be a decimal value

Overflow

- Overflow occurs when the result of the operation cannot be represented with the given number of bits
- When adding two unsigned numbers, overflow occurs when the carry-out of the operation is 1

Binary Addition and Overflow

Take 5 minutes to answer question 3 on your worksheets!

Review: Binary Addition and Overflow

$$\begin{array}{r} 1010_2 \\ + 0101_2 \\ \hline 1111 \end{array} \quad \text{no}$$

$$\begin{array}{r} 1111 \\ 1001_2 \\ + 0111_2 \\ \hline 0000 \end{array} \quad \text{yes}$$

$$\begin{array}{r} 111 \\ 1010_2 \\ + 0111_2 \\ \hline 0001 \end{array} \quad \text{yes}$$

$$\begin{array}{r} 11 \\ 0110_2 \\ + 0110_2 \\ \hline 1100 \end{array} \quad \text{no}$$

Two's Complement

Take 5 minutes to answer question 4 on your worksheets!

Two's Complement

4.1: Represent -15 in two's complement with 7 bits

Unsigned: 0b 000 1111

Flip bits: 0b 111 0000

Add one: 0b 111 0001

4.3: Convert the following two's complement number to decimal:
0b 1111 1110

Flip bits: 0b 0000 0001

Add one: 0b 0000 0010 → -2

4.2: What is the minimum number of bits needed to represent 20 in two's complement

0b 010100

6 bits

4.4: Convert the following two's complement number to decimal:
0b 0011 1000

Positive, so no need to flip bits

56