# CSE 2600
# Intro. To Digital Logic & Computer Design

Bill Siever
&
Michael Hall

# Content Review

- Site
- Graded parts
  - Prep 1: “Due” earlier today.  Still accepted a bit longer
  - Hw 1: Posted / Due next Wed.
    - Gradescope access available Friday
  - Prep 2A: Posted / Due Sep 2 @ 11am
    - Will be available on Canvas Friday

# Review & Expansion

# Chapter 1 Sections

3. **Digital Abstraction**
4. **Number Systems**
5. Logic Gates

# Abstraction

- Digital discipline
  - Discrete values: 0 and 1
    - Smallest unit of information: a binary digit.  Also-know-as a *Bit*
  - (Mostly) Starting at gate level

# Counting

| Decimal | Binary |
|---|---|
| 00 | 000 |
| 01 | 001 |
| 02 | 010 |
| 03 | 011 |
| 04 | 100 |
| 05 | 101 |
| 06 | 110 |
| 07 | 111 |

# Binary Basics: Number Line

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Decimal: | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Binary: | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |

# Conversions: Place Value

# Place Value: Base 2
# To *Decimal*
# Example: $110_2$ (or 3'b110)

| | | | |
|---|---|---|---|
| Digits | 1 | 1 | 0 |
| Place Value (Decimal) | 4 | 2 | 1 |
| Place Value In terms of Base | $2^2$ | $2^1$ | $2^0$ |
| Expansion | $1\times2^2$ | $+1\times2^1$ | $+0\times2^0$ |

**Digit in decimal**

**Value of place in decimal**

**Sum them all**

# Hexadecimal

- Convenient, compact way to deal with binary
- Each hex digit = exactly 4 binary digits
- Sixteen digits:
  0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

# Counting

| Decimal | Binary | Hex |
|---|---|---|
| 00 | 0000 | |
| 01 | 0001 | |
| 02 | 0010 | |
| 03 | 0011 | |
| 04 | 0100 | |
| 05 | 0101 | |
| 06 | 0110 | |
| 07 | 0111 | |
| 08 | 1000 | |
| 09 | 1001 | |
| 10 | 1010 | |
| 11 | 1011 | |
| 12 | 1100 | |
| 13 | 1101 | |
| 14 | 1110 | |
| 15 | 1111 | |

# Counting

| Decimal | Binary | Hex |
|---|---|---|
| 00 | 0000 | 0 |
| 01 | 0001 | 1 |
| 02 | 0010 | 2 |
| 03 | 0011 | 3 |
| 04 | 0100 | 4 |
| 05 | 0101 | 5 |
| 06 | 0110 | 6 |
| 07 | 0111 | 7 |
| 08 | 1000 | 8 |
| 09 | 1001 | 9 |
| 10 | 1010 | A |
| 11 | 1011 | B |
| 12 | 1100 | C |
| 13 | 1101 | D |
| 14 | 1110 | E |
| 15 | 1111 | F |

# Hex Notations

- Programming:  0x followed by digits, like 0xAB7
- Verilog: *n*'h*digits*

12'hAB7

Decimal number of BITS  h for Hex  Hex digits

# Hex to Binary

- What is 12’hAB7 in binary?

# Hex to Decimal

- What is 12'hAB7 in decimal?
  - Option 1: Convert to binary and then the binary to decimal
  - Option 2: Use place-value approach

# Hex to Decimal

- What is 12’hAB7 in decimal?

# What is 123 in hex?

- Greedy approach
  - A lot like binary, but have to ask “how many times does this go in…”

| | | | |
|---|---|---|---|
| Place Value (Decimal) | 256 | 16 | 1 |
| Place Value In terms of Base | $16^2$ | $16^1$ | $16^0$ |

# Why hex?

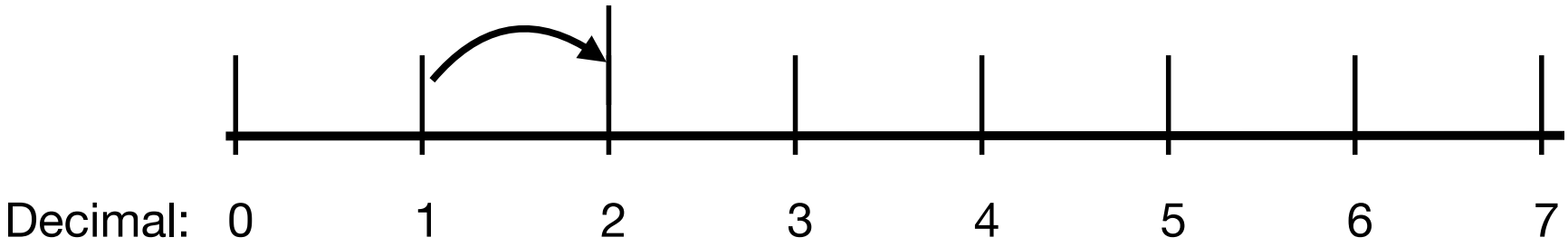
- Consider 32-bit numbers
  - 0xA764 1CD8
  - Vs.  1010 0111 0110 0100 0001 1100 1101 1000

# Review & Expansion: Arithmetic

# Decimal Addition: Bunch of Rules

Rules just “encode” moving right on the number line

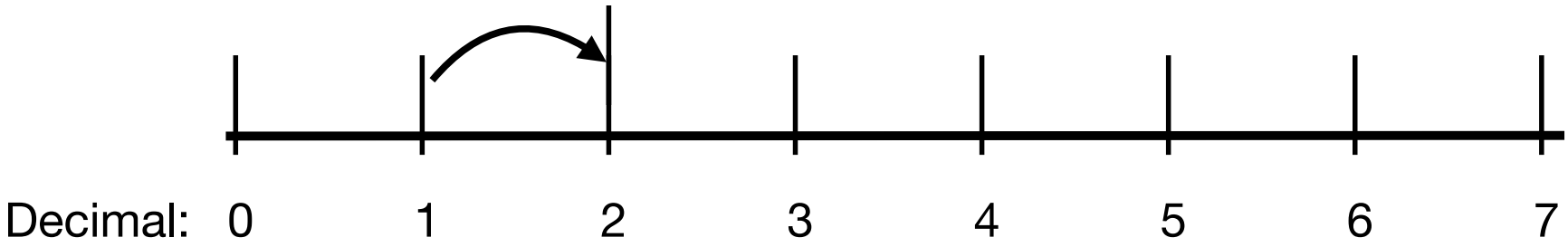
Ex: 1+2



# Decimal Addition: Bunch of Rules

Rules just “encode” moving right on the number line

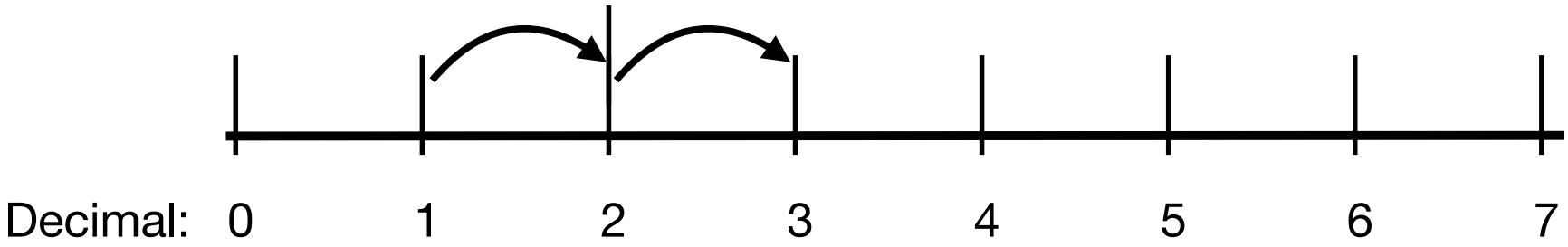
Ex: 1+2



# Decimal Addition: Bunch of Rules

Rules just “encode” moving right on the number line

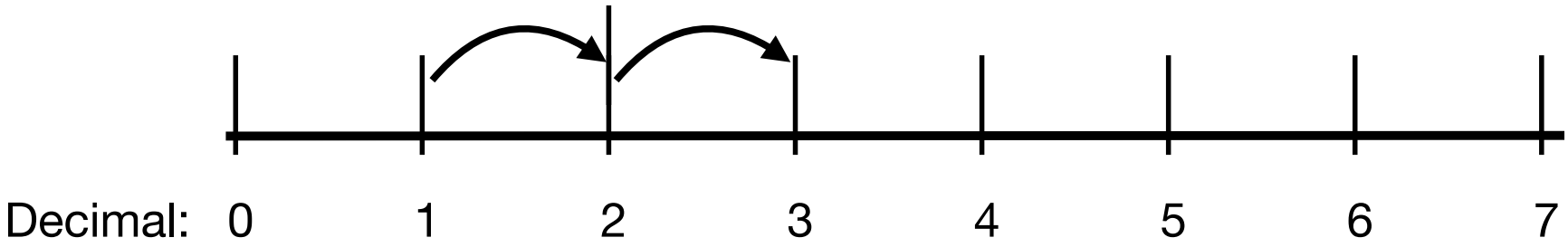
Ex: 1+2



# Decimal Addition: Bunch of Rules

Rules just “encode” moving right on the number line

Ex: 1+2



# Binary Addition Rules

- Condensed
  - No ones: 0+0+0 = 00
  - One one: 0+0+1 = 01
  - Two Ones: 0+1+1 = 10
  - Three Ones: 1+1+1 =11

# Binary Addition Rules: Fully Elaborated

| | | |
|---|---|---|
| 0+ 0+ 0 | = | 00 |
| 0+ 0+ 1 | = | 01 |
| 0+ 1+ 0 | = | 01 |
| 0+ 1+ 1 | = | 10 |
| 1+ 0+ 0 | = | 01 |
| 1+ 0+ 1 | = | 10 |
| 1+ 1+ 0 | = | 10 |
| 1+ 1+ 1 | = | 11 |

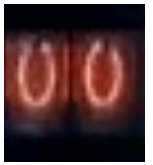
# Review Problem

- Add 4’b0110 + 4’b0010

# Fixed Width Arithmetic

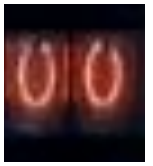
# Fixed Width Arithmetic

# Fixed Width Arithmetic



# Fixed Width Arithmetic

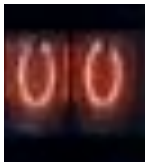
- Consider the following problems:
  - 1 + 2 =
  - 95 + 6 =
  - 80 + 20 =



# Fixed Width Arithmetic

- Consider the following problems:
  - 01 + 02 = 03  => 03
  - 95 + 06 = 101 => 01
  - 80 + 20 = 100 => 00

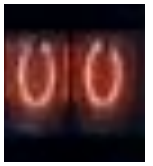
- Consider the following problems:
  - 03 op number  = 03
  - 101 op number = 01
  - 100 op number = 00



# Fixed Width Arithmetic

- Consider the following problems:
  - 01 + 02 = 03 => 03
  - 95 + 06 = 101 => 01
  - 80 + 20 = 100 => 00

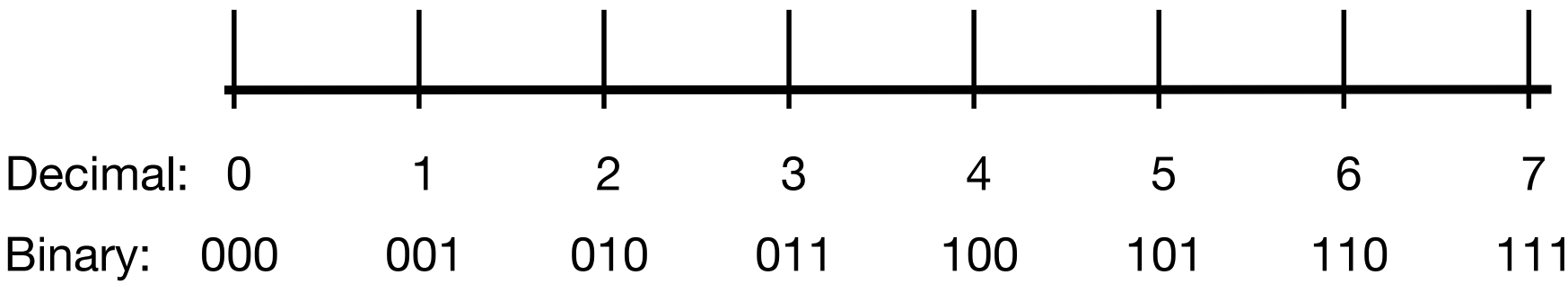
- Consider the following problems:
  - 03 % 100 = 03
  - 101 % 100 = 01
  - 100 % 100 = 00



# Fixed Width Arithmetic

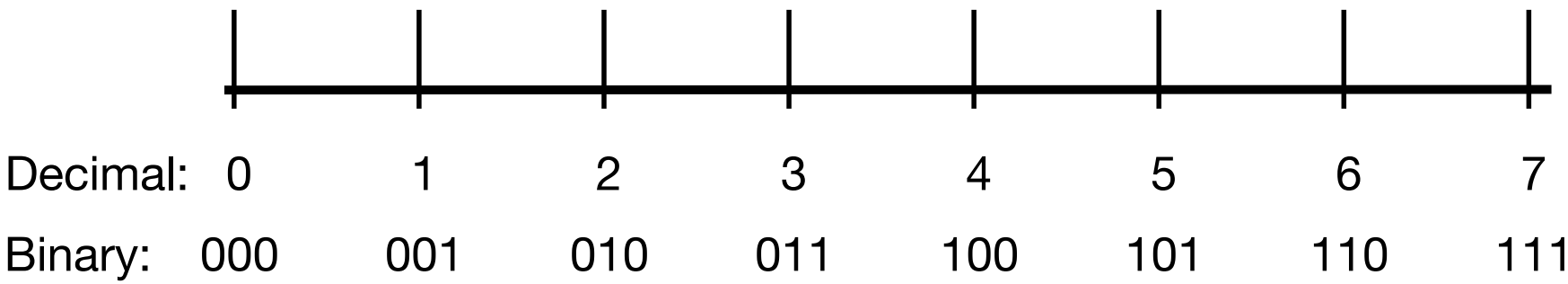
- $A + B$ With $n$-digit inputs and result in base $b$
- Result is $(A + B) \% b^n$

# Modular Arithmetic & The Number Line (Binary, 3-bit)



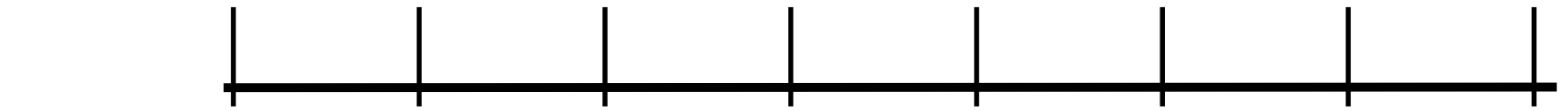

**What's 1+2?**

# Modular Arithmetic & The Number Line (Binary, 3-bit)




**What's 6+2?**

# Full, binary number line ($2^n$ values): Exceeding End Wraps!



| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Decimal: | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Binary: | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |

**What's 6+2?**

# Going Negative

- What about negative numbers?
  - Easy option: Encode the concept of a sign
    - 0 = positive
    - 1 = negative
  - Pick a bit to represent the sign

# Example: 4-bit sign / magnitude

- Format

| Place: | Sign | 2^2 | 2^1 | 2^0 |
|---|---|---|---|---|

  - Sign first (common choice)

- What is the decimal value of the 4-bit, sign-magnitude

  - 4’b1001

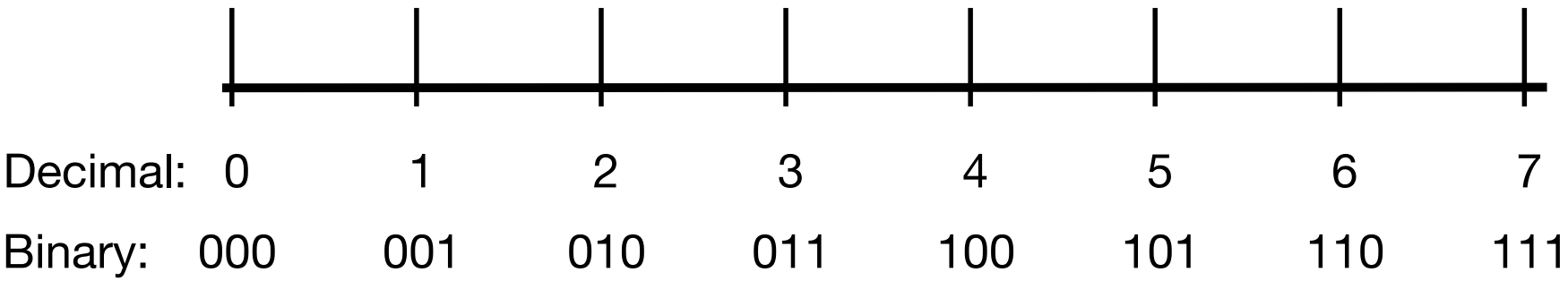
  - 4’b0110

# Really Negative

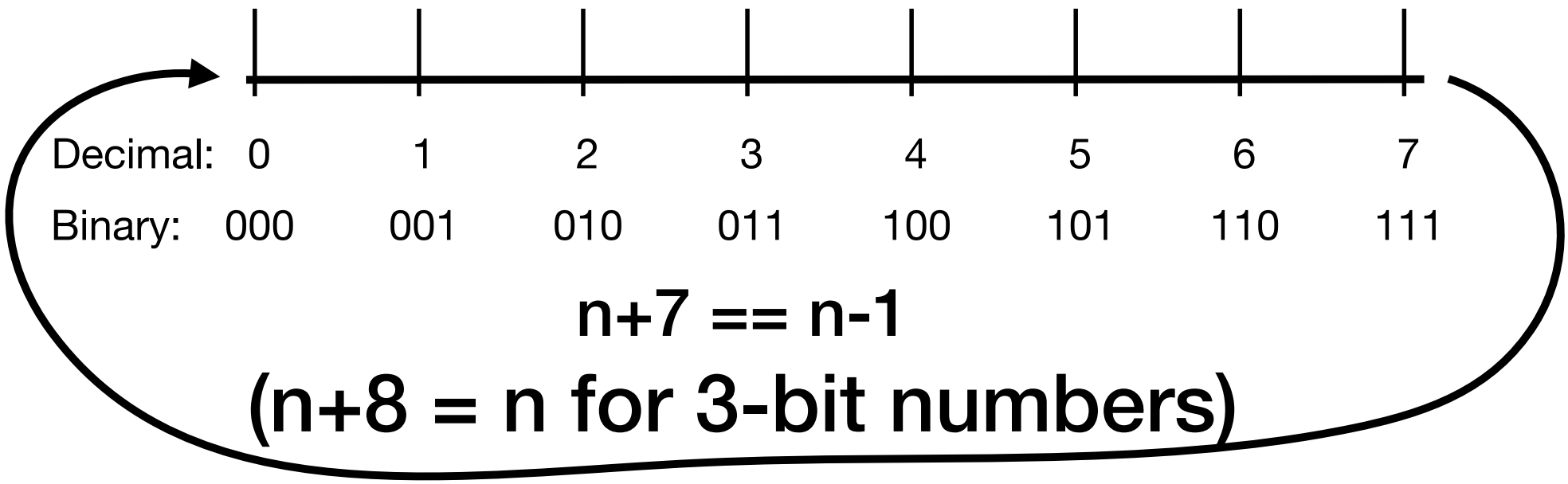
- What is the decimal value of the 4-bit, sign-magnitude
  - 4’b1000
  - 4’b0000
  - Weird
- What is 4’b1001 + 4’b0110?
  - Arithmetic is messy with sign/magnitude!

# UNSIGNED Again…

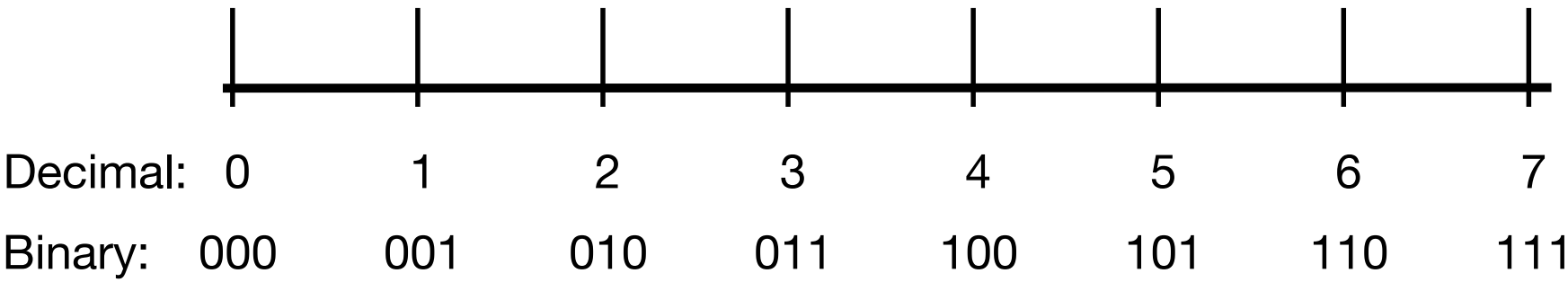
# Challenge: Describe the result of n+7?



# Challenge: Describe the result of n+7?



# Challenge: How can you emulate n-2?



# The Magic of Fixed Width numbers (modular arithmetic): Addition can emulate subtraction!

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Decimal: | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Binary: | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
| 2's comp behavior: | | | | | -4 | —3 | -2 | -1 |

# Two's Complement Sign Representation

# The “Two’s Complement”

## Dictionary

Definitions from Oxford Languages · Learn more

## com·ple·ment

*noun*

/ˈkämpləmənt/

1. a thing that completes or brings to perfection.
   "the libretto proved a perfect **complement to** the music"

   Similar: accompaniment | companion | addition | supplement | accessory

2. a number or quantity of something, especially that required to make a group complete.
   "at the moment we have **a full complement of** staff"

   Similar: amount | total | aggregate | contingent | company | capacity

# Dividing the Line

- It'll be nice if we have about half as many negatives as positives. We'll split the line in half, like we've already done.
  - How can we identify if a binary number is positive or negative?
    - Ex: 010? Or 110?

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Decimal: | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Binary: | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
| 2's comp behavior: | | | | | -4 | —3 | -2 | -1 |

# What about conversions & place/value stuff?

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Decimal: | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Binary: | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
| 2's comp behavior: | | | | | -4 | —3 | -2 | -1 |

# Consider the Upper Bit to be Negative

| | | | |
|---|---|---|---|
| Place Value (Decimal) | -4 | 2 | 1 |
| Place Value In terms of Base | $-(2^2)$ | $2^1$ | $2^0$ |

# Consider the Upper Bit to be Negative

| | | | |
|---|---|---|---|
| Place Value (Decimal) | -4 | 2 | 1 |
| Place Value In terms of Base | $-(2^2)$ | $2^1$ | $2^0$ |

**What is the decimal value of the 3-bit, 2's complement numbers:**

**110**

**011**

# Consider the Upper Bit to be Negative

| Place Value (Decimal) | -4 | 2 | 1 |
|---|---|---|---|
| Place Value In terms of Base | $-(2^2)$ | $2^1$ | $2^0$ |

**What is the 3-bit, 2's complement representation of:**

**2**

**-4**

**-5**

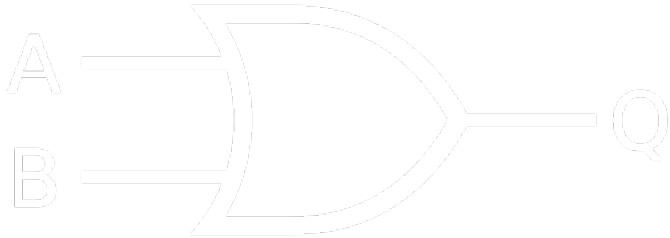
# 4-bit Two’s Complement

- What is the decimal value of the 4-bit, Two’s Complement Number
  - 0xA
  - 0x7

# Gates & Tables

- We can describe the *final behavior* simple machines, like gates, with tables
  - Truth table
    - Row for each possible combination of inputs
      (Binary; Fixed width counting)
    - Column for each input and each output

# OR Gate



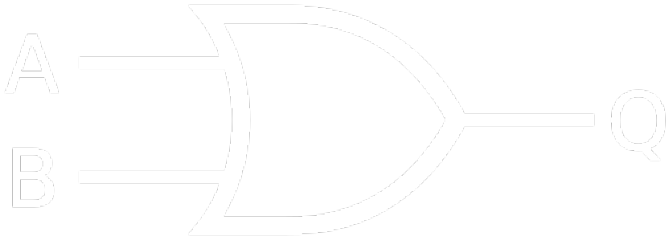

- How many rows in truth table?

- How many columns?

# OR: Gate…A real machine

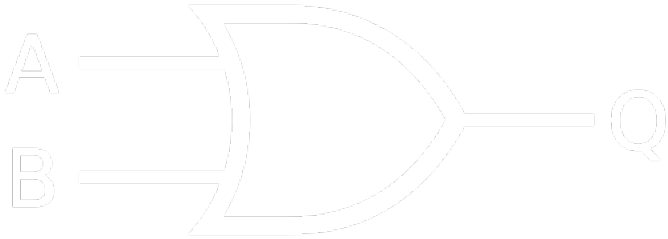
- Spintronics game: https://upperstory.com/en/spintronics/

# New Idea: Behavior OVER TIME




- Gate represents a machine of some sort machine
  - in the real world
- They are not instantaneous

# The timing diagram



# Intro to JLS

# Coming Due Dates

- Prep 1: “Due” earlier today.  Still accepted a bit longer
- Hw 1: Posted / Due next Wed.
  - Gradescope access available Friday
- Prep 2A: Posted / Due Sep 2 @ 11am
  - Will be available on Canvas Friday

# Questions

- “Will we make a basic physical computer in this course?”
  - Yes…ish
- How does this stuff apply?
  - Binary: Represent concepts, like numbers (2,3,4) and operations (+ or -) with off/on
  - Gates: Will use them to build more and more complex machines
- How will quantum…:  Talk to Prof. Cytron (CSE 4608 next semester)
- Is Hex used? Is it practical?  Yep.

# Questions

- What's "architecture"?
  - Instruction Set Architecture (ISA): The numeric codes and meanings used for a specific processor
    - Ex: RISC-V has a different ISA than ARM, IA64, etc.
  - Microarchitecture: Largely about how a processor is build and organized
  - Tabled until modules/chapters 5-7