CSE 260M / ESE 260
Intro. To Digital Logic & Computer Design

Bill Siever
&
Michael Hall

5W+H

(Questions welcome at any time)

Who?

- Us: Bill Siever & Michael Hall
  - Bill: Teaching Prof. In CSE
  - Michael: Lecturer in CSE/ESE

Who?

- You?
  - Mix of Computer Engineering, Electrical Engineering, and C.S. Majors
  - Many in Dual Degree program
- Prerequisites: Intro. To Computer Science (Programming)
- Other related courses? 1302? 3601? 3602?

What?

- Digital Logic!
  - Digital: Usually about binary-based systems
    - Q: Why binary?
- Computer Design
  - Focus on Architecture: How Digital Logic is Used for a Modern Computer

When?

- Class (now): Tues/Thurs 2:30-3:50
- Instructor & TA Office Hours: TBD

Where?

- Hillman 60 (ish)
  (May be different on future Tuesdays — TBD)

Why?

- Digital logic is critical to
  - All of computing
  - Recent advances in A.I./M.L.
  - Understanding system-level behavior of computers

Why?

- Deep understanding benefits:
  - Design at all levels (hardware, software/API)
  - Debugging
- Integration of knowledge
  - Bring together lots of classes / topics

How?

- Overview of Syllabus / Schedule / Webpage
  - https://wustl.instructure.com/courses/154176

How?

- Summary:
  - For credit: Exams, Homework, Studios, Studio Lead duties, Prep work summaries
  - For prep: Lectures/discussion, Prep work (reading, videos, etc.)

Tools / Resources

- Website vs. Canvas
- Canvas, Gradescope, Github
- Forum: Piazza

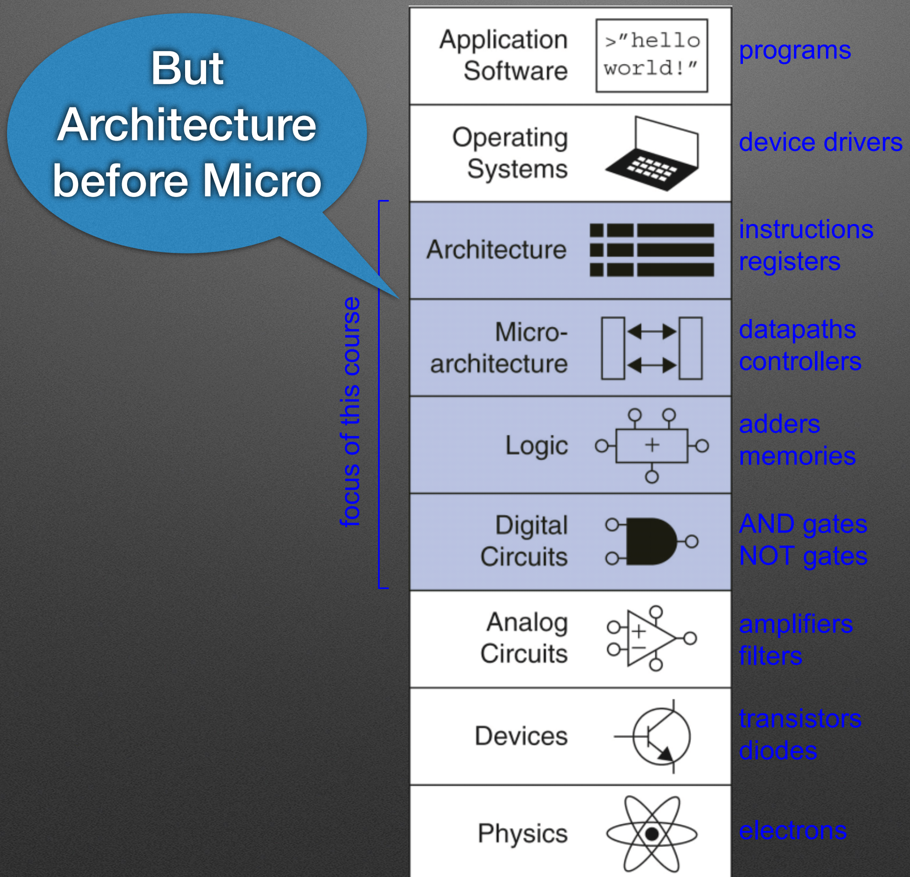
Challenges

- Significant change in content from some prior years
  - Still being refined — you will be a part of continued refinement
- There will be some challenges & problems
  - That’s common in engineering
  - We’ll focus on helping you learn the critical concepts despite setbacks

Chapter 1 Sections

1. The Game Plan
2. Managing Complexity
3. Digital Abstraction
4. Number Systems
5. Logic Gates

Course


But Architecture before Micro
focus of this course
Application Software
>"hello world!"
programs
Operating Systems
device drivers
Architecture
instructions
registers
Micro-architecture
datapaths
controllers
Logic
adders
memories
Digital Circuits
AND gates
NOT gates
Analog Circuits
amplifiers
filters
Devices
transistors
diodes
Physics
electrons

Abstraction

- Digital discipline
  - Discrete values
    - Moreover, *binary* (0/1; false/true; Off/On; 0v/3v; No/Yes; …)
      - Smallest unit of information: a binary digit. Also-know-as a ***Bit***
  - (Mostly) Starting at gate level

Goals Today

- Review / Learn (Unsigned) Binary Representations
- Learn Binary Addition
- Review Binary Operations
  - Consider Machines for Binary Operations

Counting

Decimal	
0	
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	

Counting

Decimal
00
01
02
03
04
05
06
07
08
09
10

Counting

Decimal	Binary
00	
01	
02	
03	
04	
05	
06	
07	
08	
09	
10	

Counting

Decimal	Binary
00	0
01	
02	
03	
04	
05	
06	
07	
08	
09	
10	

Counting

Decimal	Binary
00	0
01	1
02	
03	
04	
05	
06	
07	
08	
09	
10	

Counting

Decimal	Binary
00	0
01	1
02	?
03	
04	
05	
06	
07	
08	
09	
10	

Counting

Decimal	Binary
00	00
01	01
02	10
03	
04	
05	
06	
07	
08	
09	
10	

Counting

Decimal	Binary
00	0000
01	0001
02	0010
03	
04	
05	
06	
07	
08	
09	
10	

Counting

Decimal	Binary
00	0000
01	0001
02	0010
03	0011
04	
05	
06	
07	
08	
09	
10	

Counting

Decimal	Binary
00	0000
01	0001
02	0010
03	0011
04	0100
05	
06	
07	
08	
09	
10	

Counting

Decimal	Binary
00	0000
01	0001
02	0010
03	0011
04	0100
05	0101
06	
07	
08	
09	
10	

Counting

Decimal	Binary
00	0000
01	0001
02	0010
03	0011
04	0100
05	0101
06	0110
07	
08	
09	
10	

Counting

Decimal	Binary
00	0000
01	0001
02	0010
03	0011
04	0100
05	0101
06	0110
07	0111
08	
09	
10	

Counting

Decimal	Binary
00	0000
01	0001
02	0010
03	0011
04	0100
05	0101
06	0110
07	0111
08	1000
09	
10	

Counting

Decimal	Binary
00	0000
01	0001
02	0010
03	0011
04	0100
05	0101
06	0110
07	0111
08	1000
09	1001
10	

Counting

Decimal	Binary
00	0000
01	0001
02	0010
03	0011
04	0100
05	0101
06	0110
07	0111
08	1000
09	1001
10	1010

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: Base 10
Example: 123

Digits	1	2	3
Place Value	100	10	1
Place Value In terms of Base	10^2	10^1	10^0
Expansion	1×10^2	$+2\times10^1$	$+3\times10^0$

Place Value: Base 2
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	1×2^2	$+1 \times 2^1$	$+0 \times 2^0$

Easy Conversion: Binary to Decimal

Place Value (Decimal)	128	64	32	16	8	4	2	1
Place Value In terms of Base	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Problem: What is the decimal value of 5'b10011

Place Value (Decimal)	128	64	32	16	8	4	2	1
Place Value In terms of Base	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Easy Conversion: Decimal to Binary
Greedy Algorithm Approach: Right to Left

1. Start with value n

2. Find the exponent, k, of the *largest* power of 2 that is *smaller* than n.
   (I.e., first power of 2 that can be subtracted without going negative)

3. For k down to 0:

   1. If $2^k \leq n$

      1. Write down a 1 (and move right)

      2. $n = n - 2^k$

   2. Else

      1. Write down a 0 (and move right)

Example: Convert 27 to binary (With the greedy approach)

- First power of 2 less than 27
  - **16** (2^4)
- $n = 27 - 16 = 11$
- $n = 11 - 8 = 3$
- $n = 3 - 2 = 1$
- $n = 1 - 1 = 0$

Place Value	128	64	32	16	8	4	2	1
Place Value	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Result				1	1	0	1	1

Arithmetic

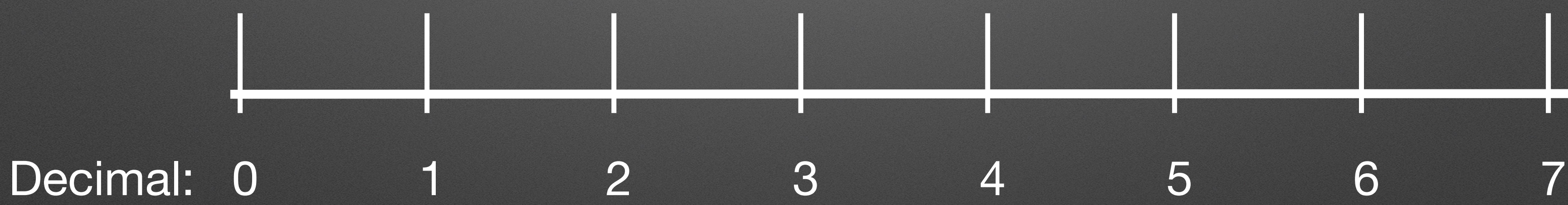
Decimal Addition

+	1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10	11
2	3	4	5	6	7	8	9	10	11	12
3	4	5	6	7	8	9	10	11	12	13
4	5	6	7	8	9	10	11	12	13	14
5	6	7	8	9	10	11	12	13	14	15
6	7	8	9	10	11	12	13	14	15	16
7	8	9	10	11	12	13	14	15	16	17
8	9	10	11	12	13	14	15	16	17	18
9	10	11	12	13	14	15	16	17	18	19
10	11	12	13	14	15	16	17	18	19	20

Decimal Addition: Bunch of Rules

Rules just “encode” moving right on the number line

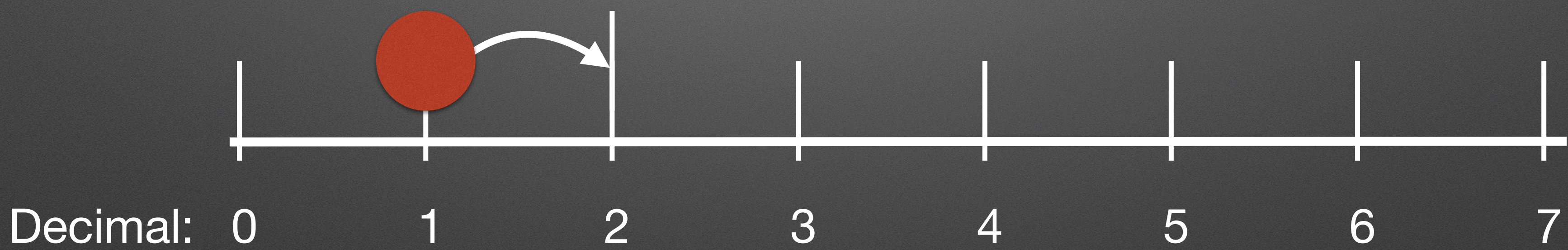
Ex: 1+2



Decimal Addition: Bunch of Rules

Rules just "encode" moving right on the number line

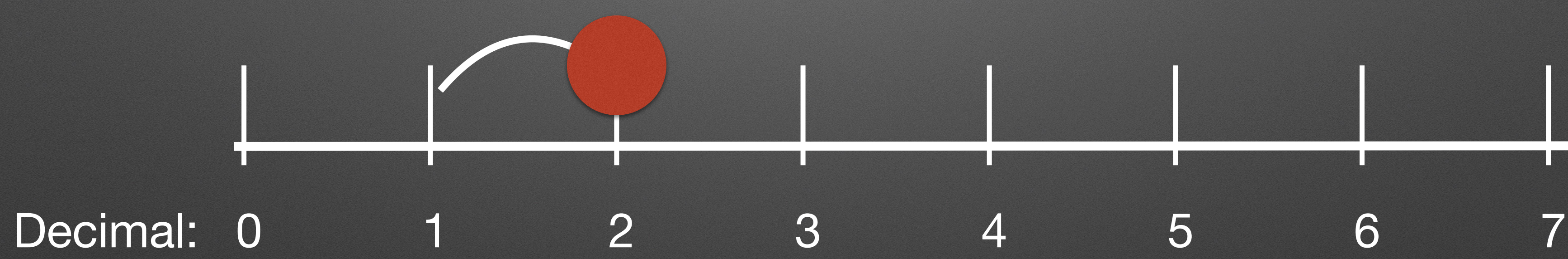
Ex: 1+2



Decimal Addition: Bunch of Rules

Rules just “encode” moving right on the number line

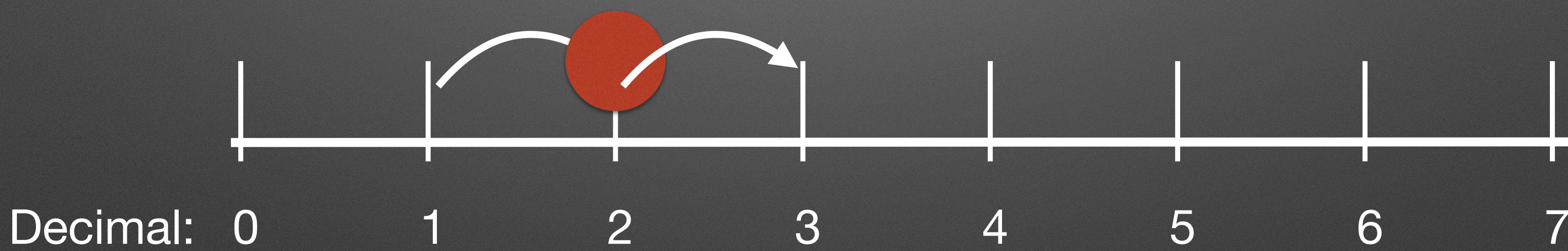
Ex: 1+2



Decimal Addition: Bunch of Rules

Rules just “encode” moving right on the number line

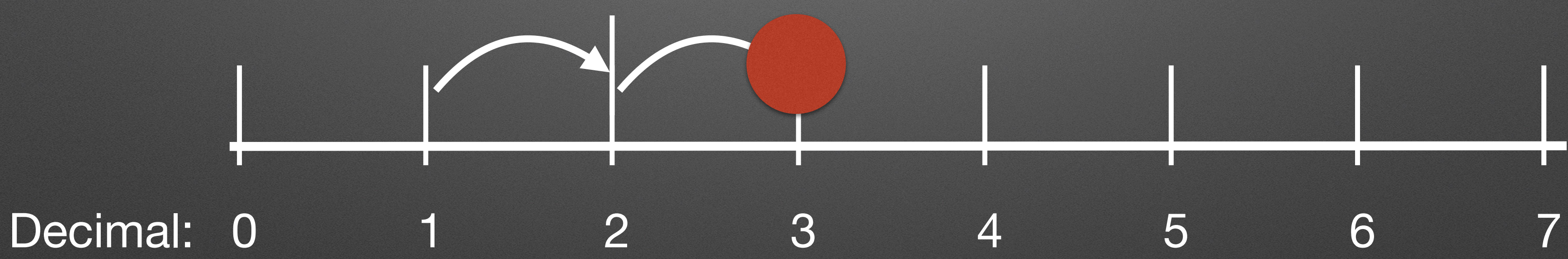
Ex: 1+2



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

Problem

- Add 4’b1010 + 4’b0011

Review: Operations on Booleans

Review: Boolean Logic Operations

LOGIC OPERATION	COMMON PROG. LANG. SYMBOLS	FIRST-ORDER LOGIC	DIGITAL LOGIC
And	&&, and	^	* (multiplication)
Or	||, or	v	+
Not / Negation	!, not	¬	/ (also line over)

Gates: Conceptual Machines for Boolean Ops

LOGIC OPERATION	COMMON PROG. LANG. SYMBOLS	FIRST-ORDER LOGIC	DIGITAL LOGIC	GATE
And	&&, and	^	* (multiplication)	See here
Or	||, or	v	+	See here
Not / Negation	!, not	¬	/ (also line over)	See here

Gates: Machines for Boolean Ops

(A look at “Computer Engineering for Babies”)

For Thursday

- Read Chapter 1: 1.1-1.5
  - Complete the questions (Canvas) before 11am (not officially due)
  - Future prep work questions are 11:59pm on Mondays
  - Reading is almost all of Chapters 1-7. Can work ahead!

Homework #1 Posted!
Dropbox available on Thursday (28th)
Due next Wednesday (September. 3rd)

What’s the operation?

- Consider the following problems:
  - 123 ? 10 = 3
  - 7 ? 10 = 7
  - 29 ? 10 = 9

- Consider the following problems:
  - 123 ? 100 = 23
  - 7 ? 100 = 7
  - 29 ? 100 = 29

Why is that important?

- We'll often work with fixed-width numbers
  - Ex: our rules of addition are just for 1 column of digits
- Multi-digit numbers are handled via chaining together fixed width operations
- Truncation to fixed width numbers is a special case of modular arithmetic (which has some cool properties)

Fixed Width / Truncation in Decimal

- We'll often work with fixed-width numbers
  - Ex: our rules of addition are just for 1 column of digits
- Multi-digit numbers are handled via chaining together fixed width operations
- Truncation to fixed width numbers is a special case of modular arithmetic (which has some cool properties)

What’s the operation?

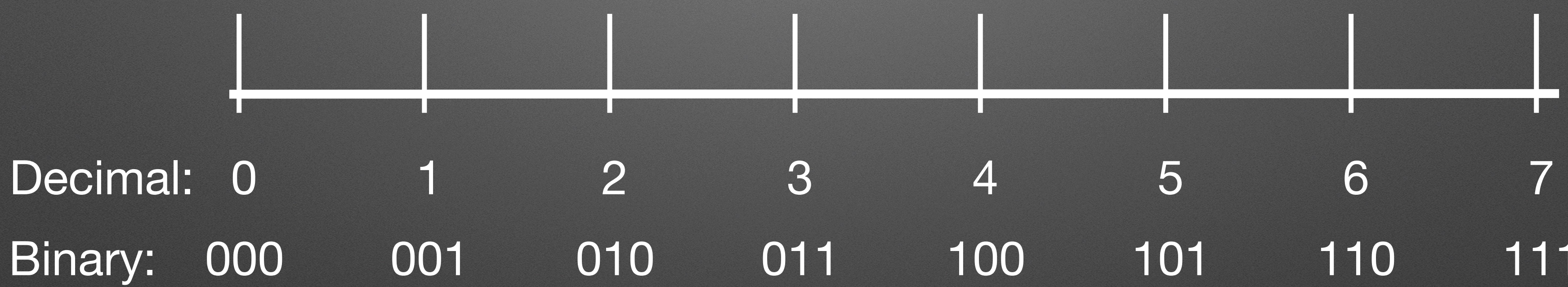
- Consider the following problems:
  - 123 ? 10 = 3
  - 7 ? 10 = 7
  - 29 ? 10 = 9

- Consider the following problems:
  - 123 ? 100 = 23
  - 7 ? 100 = 7
  - 29 ? 100 = 29

What's the operation?

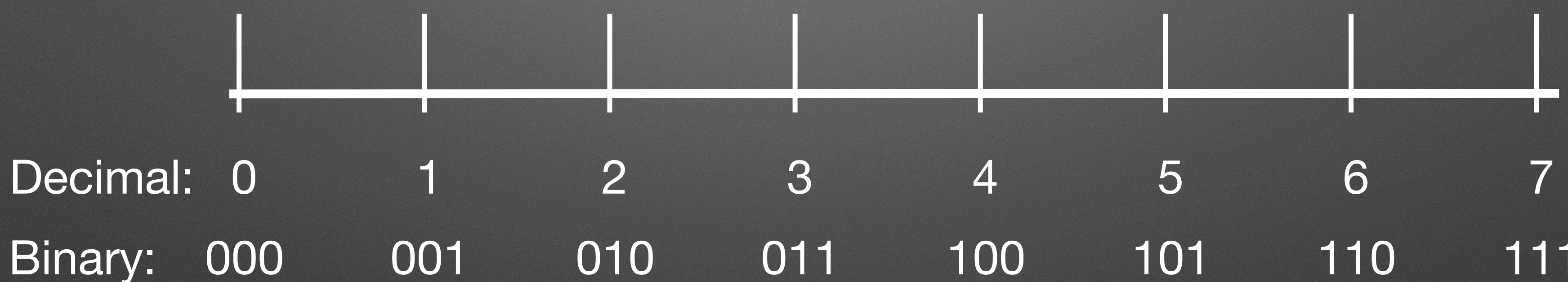
- What is the 1 digit result of:
  - 122 + 1 = 3
  - 3+4 = 7
  - 15+14 = 9
- What is the 2 digit result of:
  - 3+120 = 23
  - 2+5 = 7
  - 28+1 = 29

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



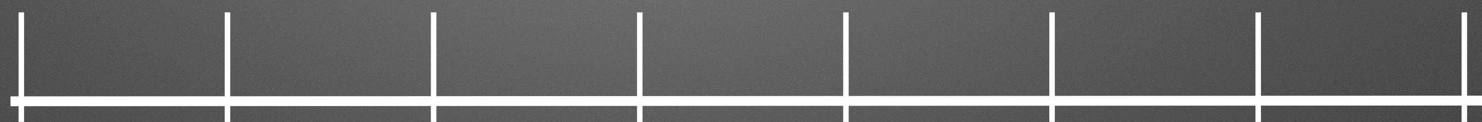
What’s 1+2?

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



What’s 6+2?

Challenge: Describe the result of n+7



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

Challenge: How can you emulate n-2?

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

The Magic of 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

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

Hexadecimal

- Convenient, compact way to deal with binary
- Each hex digit = exactly 4 binary digits