CSE 3210

Computer Organization and Programming

Chapter 1 Basic Concepts

Dr. Zulkar Nine

mnine@gsu.edu

Georgia State University

Spring 2021

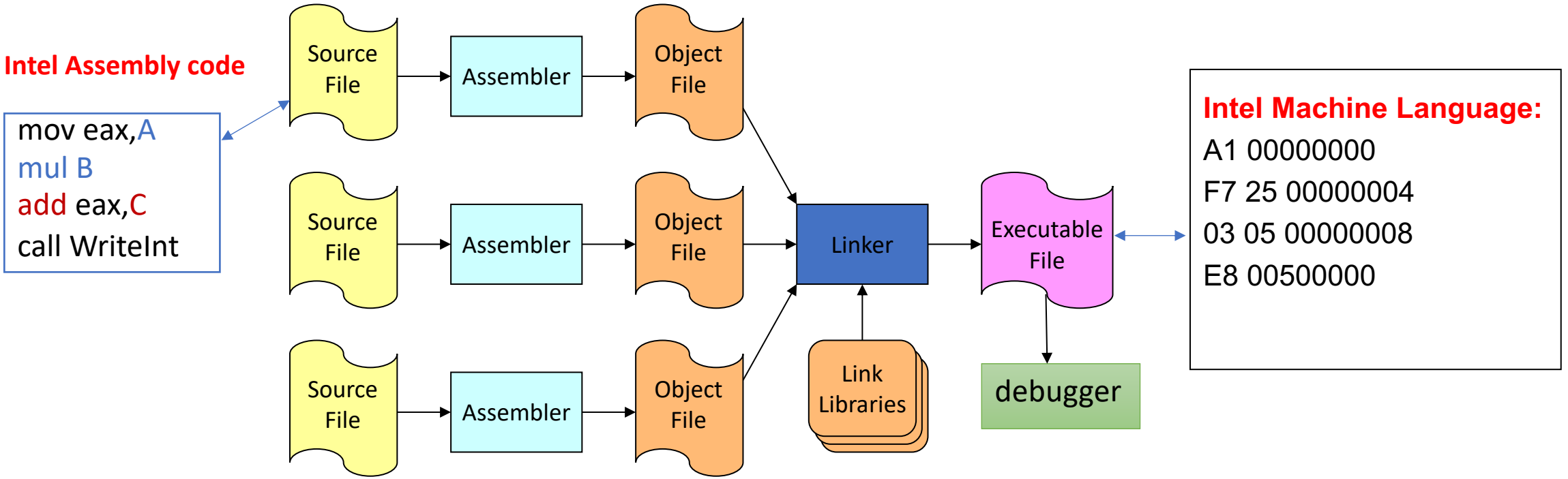
Outline

- Welcome to Assembly Language
  - Some concerns
  - Applications
- Virtual Machine Concepts
  - Virtual Machines
  - Machine Levels
  - Translating Language
- Data Representation
- Boolean Operations

Welcome to Assembly language!

Programming is fun!

What is an **assembler**, a **linker**, a debugger?



Some issues

- What **hardware/software** do I need?
  - Computer 32/64 bit
  - Windows + Microsoft Visual Studio
  - Mac + Virtualization software (Virtualbox) + Windows + Microsoft Visual Studio
- What **types of programs** will I create?
  - Mainly **32-Bit,** Some 64 bit
- What do I get with this book?
  - http://www.asmirvine.com/

More issues

- How does **assembly language** (AL) **relate** to **machine language**?
  - One-to-one relationship
- How do **C/C++** relates to **AL**?

C/C++

```
int Y;
int X = (Y + 4) * 3;
```

Assembly

```
mov eax,Y ;     - move Y to the EAX register
add eax,4 ;     - add 4 to the EAX register
mov ebx,3 ;     - move 3 to the EBX register
imul ebx ;      - multiply EAX by EBX
mov X,eax ;     - move EAX to X
```

- Is **AL** portable? No. Why?

Assembly Language Applications

- Some representative types of applications:
  - **Hardware device driver**
  - **Embedded systems** & **computer games**
  - Business application for single platform (No)
  - Business application for multiple platforms (No)

Think again!

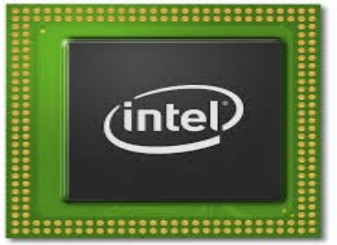
Is the **assembly language for x86** processors **the same as those** for computer systems such as the **Vax, Motorola 68x00, or SPARC**?

SPARC

```
mov 5,r1
mov 6,r2
add r1,r2,r3
```

Intel

```
mov eax,5
add eax,6
```



What's Next

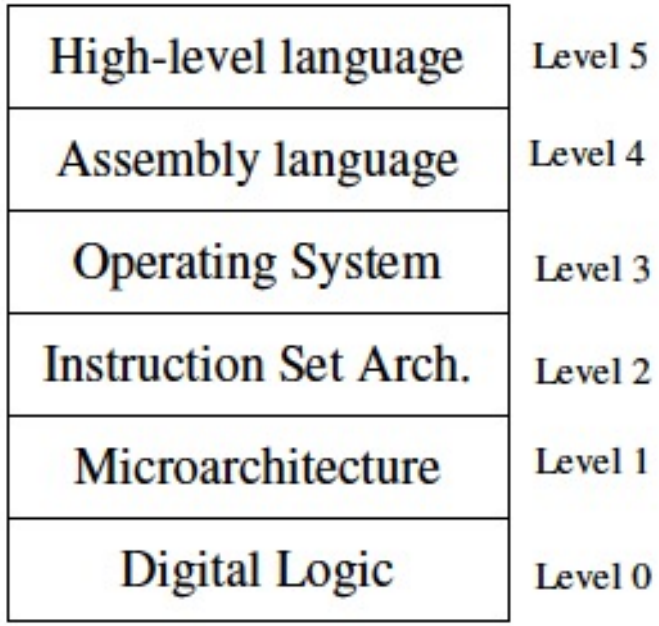
- - Welcome to Assembly Language
  - Some Good Questions to Ask
  - Assembly Language Applications
- **- Virtual Machine Concept**
  - Virtual Machines
  - Specific Machine Levels
  - Translating Languages
- - Data Representation
- - Boolean Operations

Virtual Machine Concepts

Abstraction that hides all the low level complexities

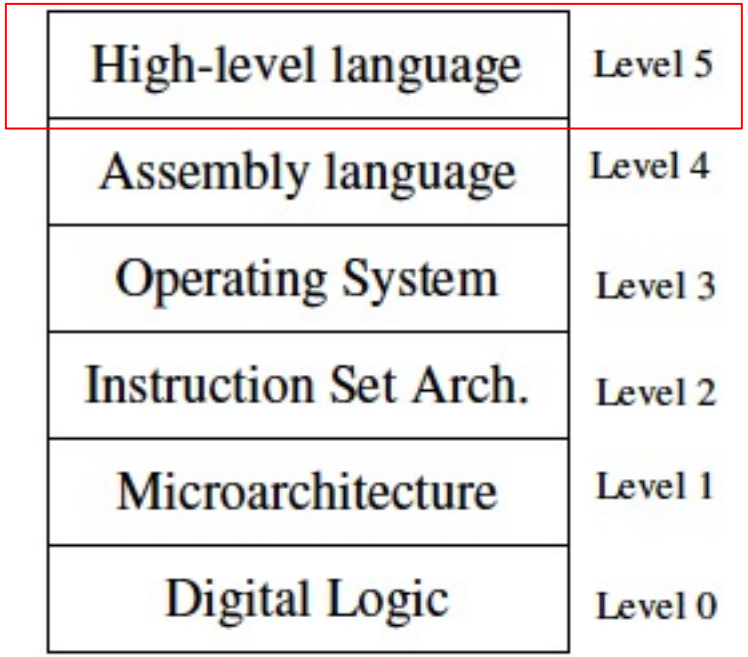
Specific Machine Levels

- **Virtual Machine Concept**
  - Explain how a computer's **hardware** and **software** are related.
  - The main thing a computer does is **executing programs**
  - A computer execute programs written in its **native machine language (ML).**
    - difficult to write programs in ML
      - What to do?
    - programs could be written in another language (Interpretation vs. Translation)



Specific Machine Levels: High-Level Language

- **Level 5**
- Application-oriented languages
  - **C++,** Java, Visual Basic . . .
- Programs **compile into assembly language** (Level 4)



Specific Machine Levels: High-Level Language

High-level language program (in C)

```
void main()
{
C = ( Y + 4 ) * 3 ;
}
```

C Compiler

Assembly language program

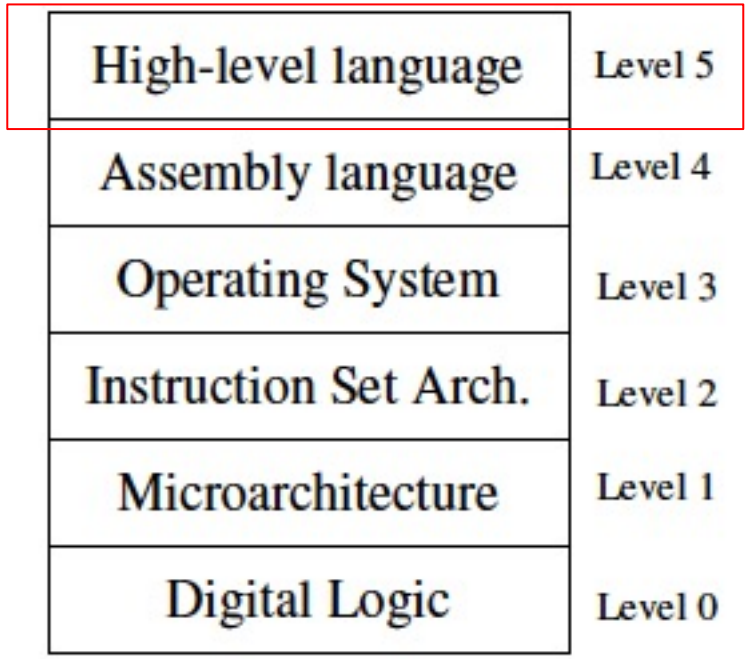
```
mov eax, Y
add eax, 4
mov ebx, 3
imul ebx
mov C, eax
```

Assembler

Linker

Binary machine language program

```
00000000100011100001100000100001
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000
00000011111000000000000000001000
```



What's Next

- - Welcome to Assembly Language
  - Some Good Questions to Ask
  - Assembly Language Applications
- **- Virtual Machine Concept**
  - Virtual Machines
  - Specific Machine Levels
  - Translating Languages
- **- Data Representation**
- - Boolean Operations

Data Representation

Lots of math!

Data Representation

- **Binary Numbers**
  - Translating between binary and decimal
- **Binary Addition**
- Integer Storage Sizes
- **Hexadecimal Integers**
  - Translating between decimal and hexadecimal
  - Hexadecimal subtraction
- **Signed Integers**
  - Binary subtraction
- **Character Storage**

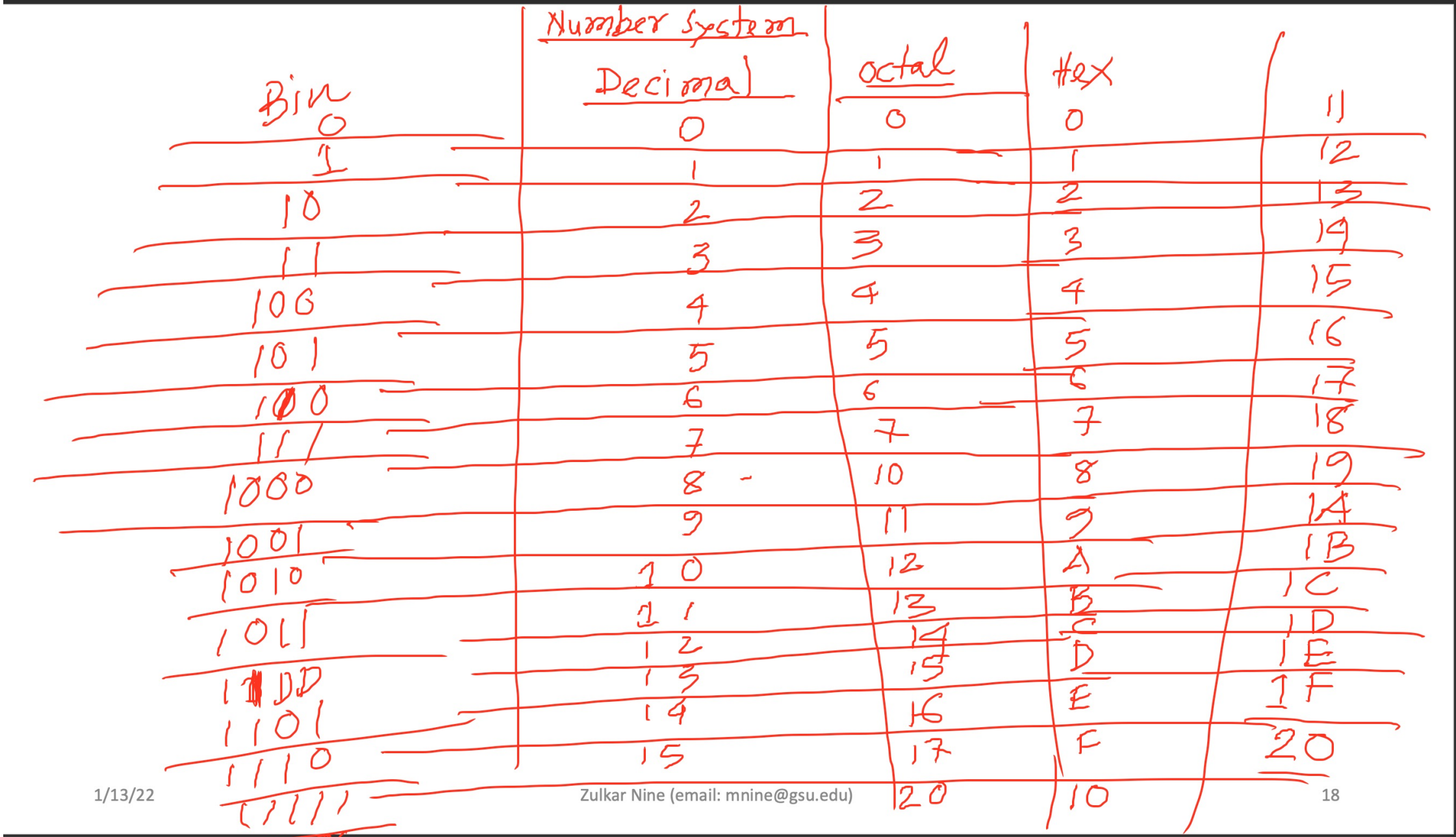
Data Representation: Numbering System

- Assembly language deals with **Data at the physical level**
  - so you need to examine **registers** and **memory**
- **Binary** and **hexadecimal** numbers are commonly used to describe those contents (other systems used as well)
- **Need to learn how to translate from one format to another**

TABLE 1-2 Binary, Octal, Decimal, and Hexadecimal Digits.

System	Base	Possible Digits
Binary	2	0 1
Octal	8	0 1 2 3 4 5 6 7
Decimal	10	0 1 2 3 4 5 6 7 8 9
Hexadecimal	16	0 1 2 3 4 5 6 7 8 9 A B C D E F

Base: maximum number of **symbols** assigned to every digit

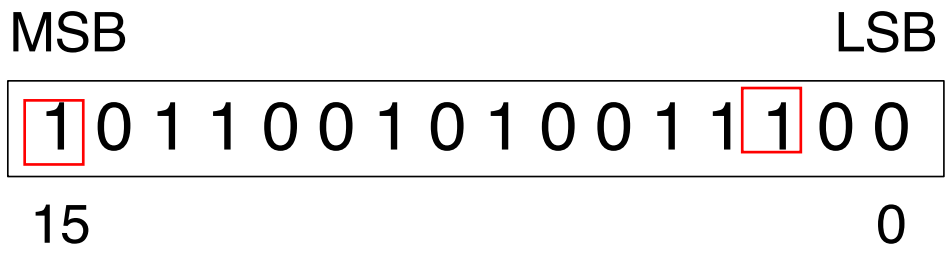


Data Representation: Binary Numbers **(Integers)**

- Binary integers can be **signed** or **unsigned**.
  - **A signed** integer is **positive** or **negative**.
  - **An unsigned** integer is by default **positive**.
    - **Zero** is considered **positive**.

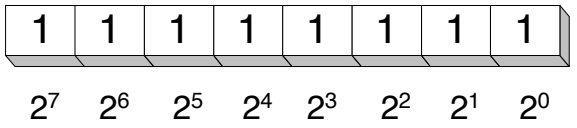
Data Representation: Binary Numbers **(Integers)**

- Digits are 1 and 0
  - 1 = true
  - 0 = false
- **MSB** – most significant bit
- **LSB** – least significant bit
- **Bit numbering:**



Data Representation: Binary Numbers **(Integers)**

- Each digit (bit) is either 1 or 0
- Each bit represents **a power of 2:**



Every binary number is a sum of powers of 2

Table 1-3 Binary Bit Position Values.

2^n	Decimal Value	2^n	Decimal Value
2^0	1	2^8	256
2^1	2	2^9	512
2^2	4	2^{10}	1024
2^3	8	2^{11}	2048
2^4	16	2^{12}	4096
2^5	32	2^{13}	8192
2^6	64	2^{14}	16384
2^7	128	2^{15}	32768

Data Representation: Translating **Binary to Decimal**

- Weighted positional notation shows how to calculate the decimal value of each binary bit:

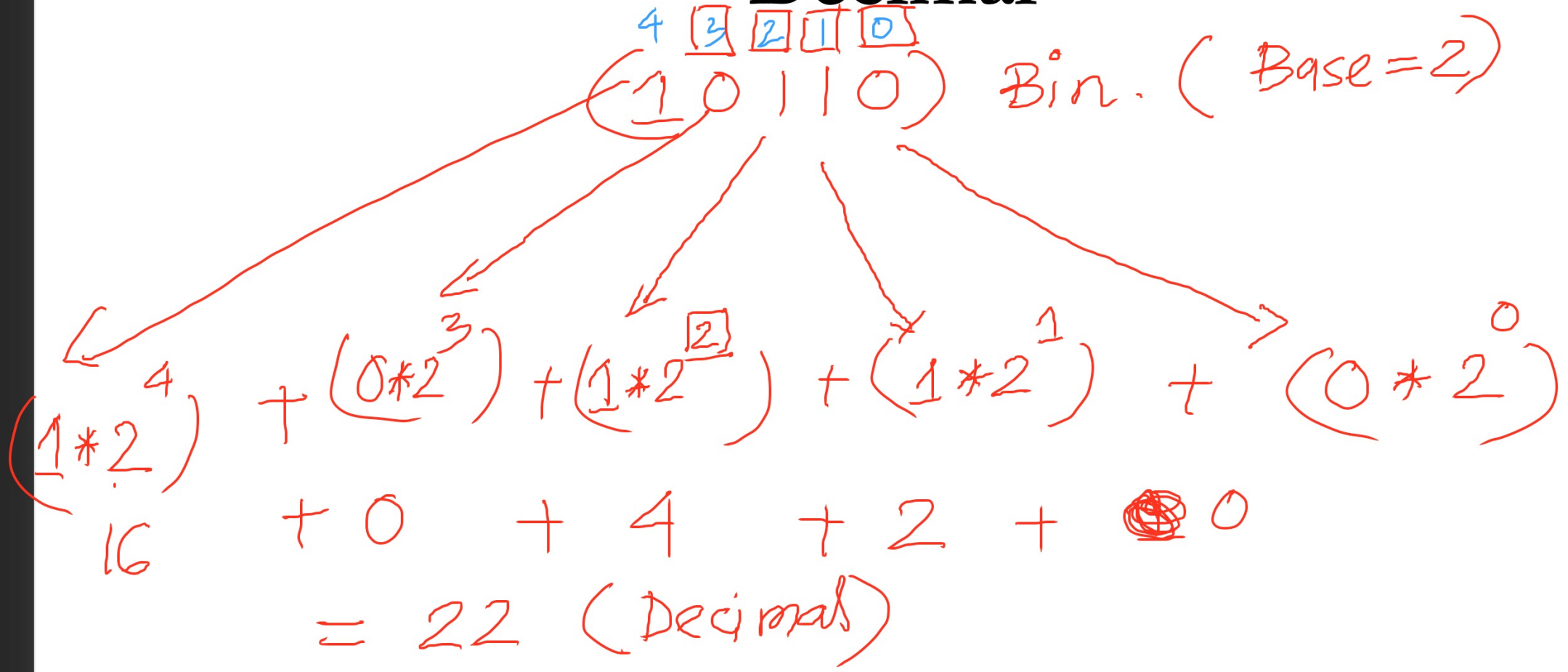
$$dec = (D_{n-1} \times 2^{n-1}) + (D_{n-2} \times 2^{n-2}) + \ldots + (D_1 \times 2^1) + (D_0 \times 2^0)$$

D = binary digit

binary 00001001 = decimal 9:

$$(1 \times 2^3) + (1 \times 2^0) = 9$$

Data Representation: Translating **Binary to Decimal**

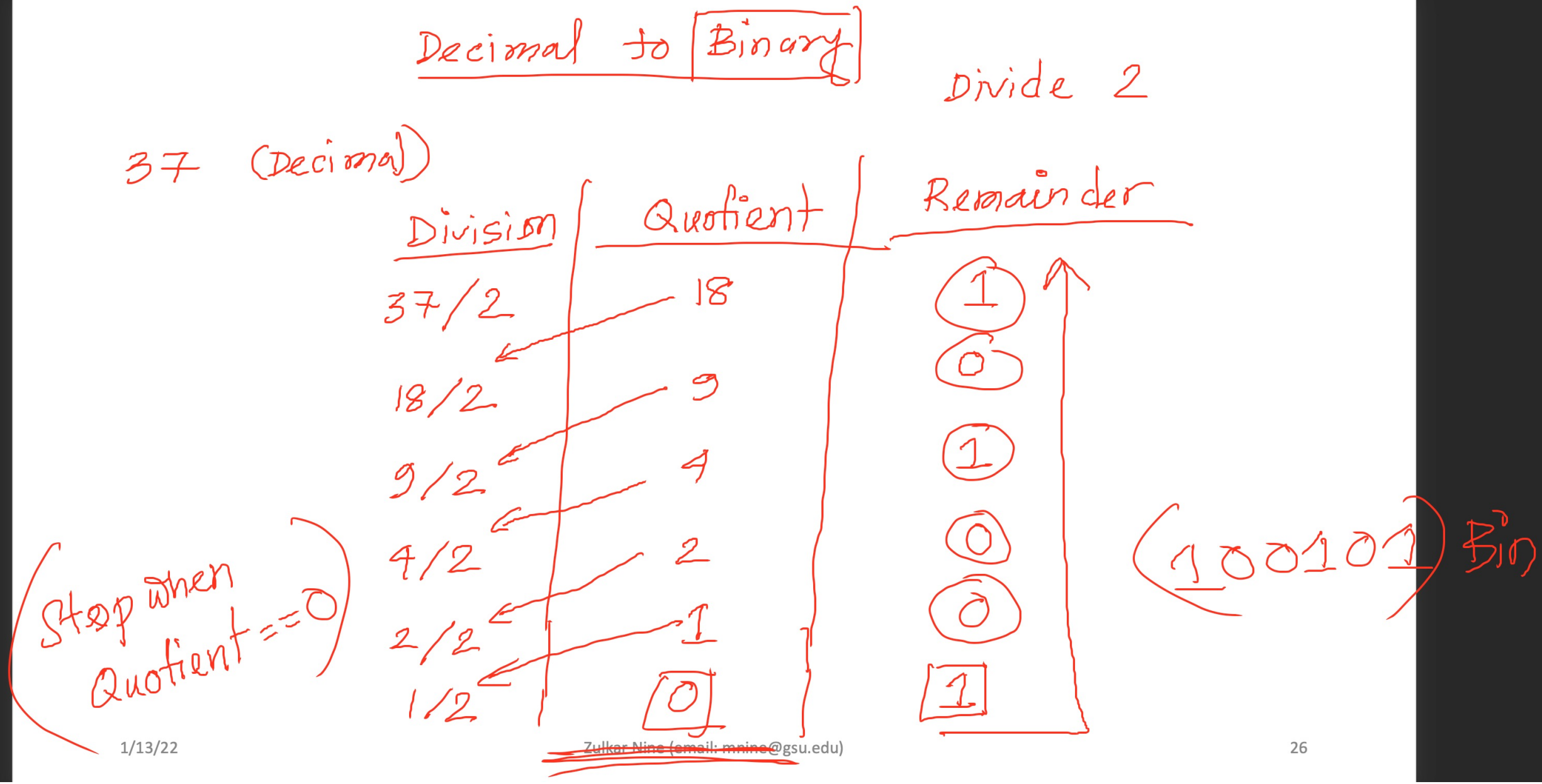


Data Representation: Translating **Unsigned Decimal to Binary**

- Repeatedly divide the decimal integer by 2.
- Each remainder is a binary digit in the translated value:

37 = 100101

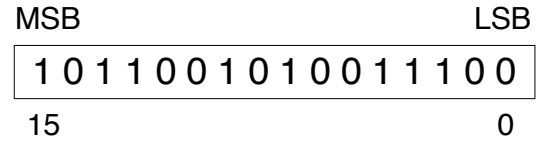
Division	Quotient	Remainder
37 / 2	18	1
18 / 2	9	0
9 / 2	4	1
4 / 2	2	0
2 / 2	1	0
1 / 2	0	1

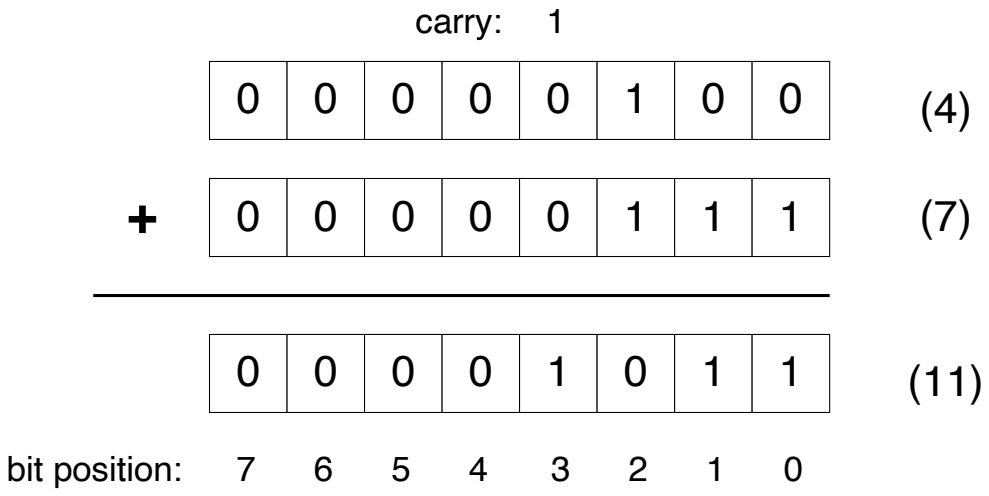


1/13/22 Zulkar Nine (email: mnine@gsu.edu) 26

Data Representation: Binary **Addition**

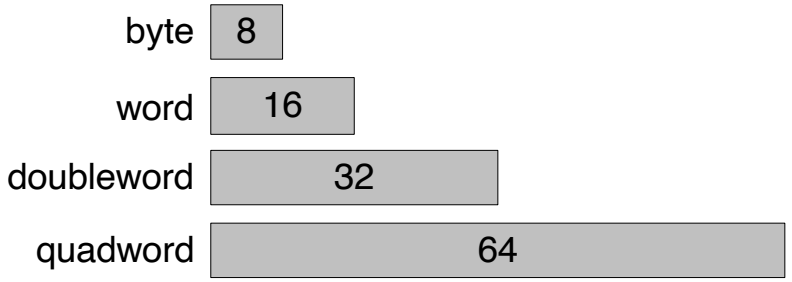
- Starting with the **LSB**, add each pair of digits, **include the carry if present**.





Data Representation: Integer **Storage Sizes**

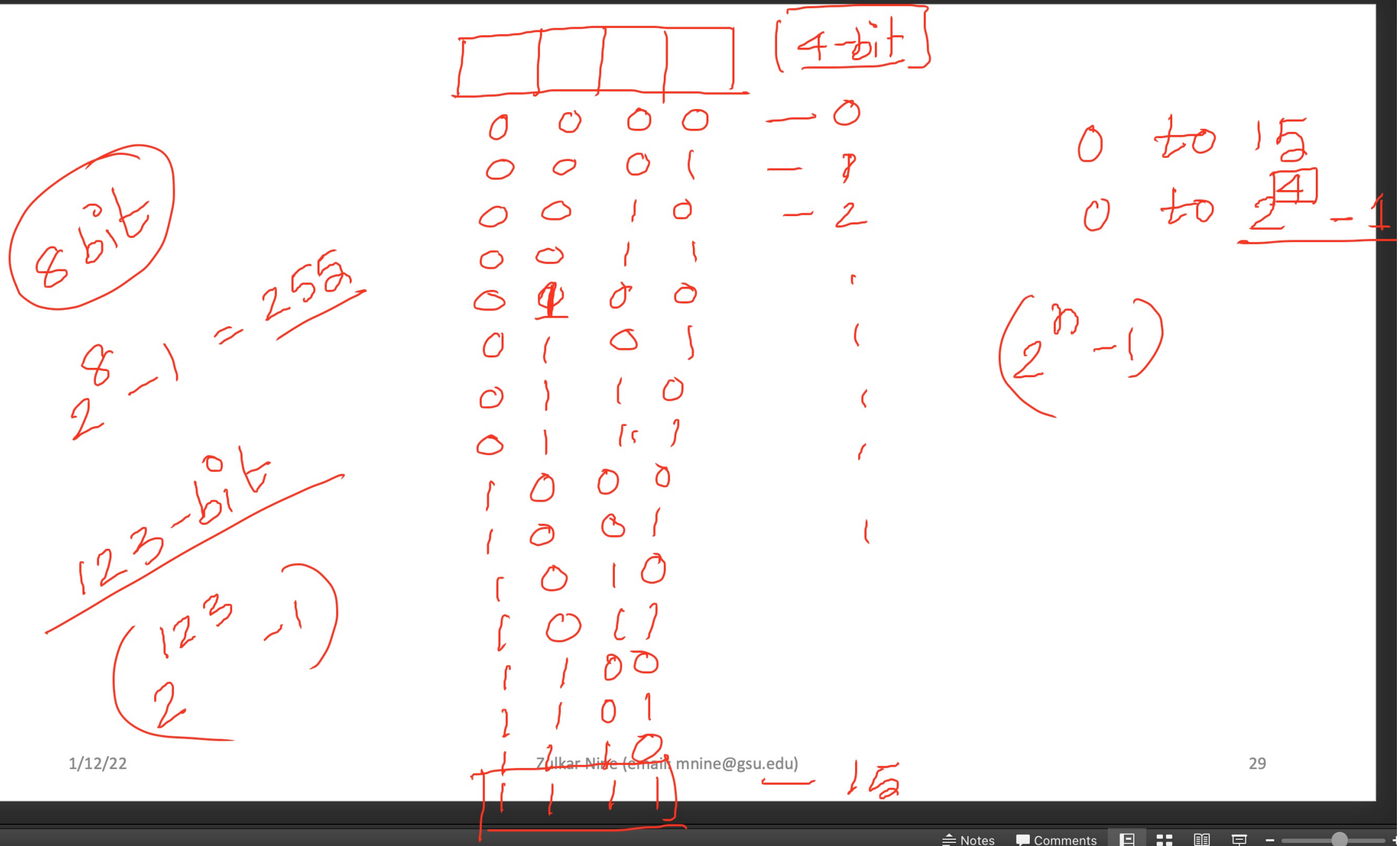
- Standard sizes (**x86**):



What is the largest unsigned integer that may be stored in 20 bits?

TABLE 1-4 Ranges and Sizes of Unsigned Integer Types.

Type	Range	Storage Size in Bits
Unsigned byte	0 to $2^8 - 1$	8
Unsigned word	0 to $2^{16} - 1$	16
Unsigned doubleword	0 to $2^{32} - 1$	32
Unsigned quadword	0 to $2^{64} - 1$	64
Unsigned double quadword	0 to $2^{128} - 1$	128


4-bit
0 0 0 0 — 0
0 0 0 1 — 1
0 0 1 0 — 2
1 1 1 1 — 15
0 to 15
0 to $2^4 - 1$
$(2^n - 1)$
8 bit
$2^8 - 1 = 255$
123-bit
$(2^{123} - 1)$

Data Representation: **Hexadecimal** Integers

- Binary values are represented in hexadecimal (Why).
  - **Large binary numbers are hard to read**

Table 1-5 Binary, Decimal, and Hexadecimal Equivalents.

Binary	Decimal	Hexadecimal	Binary	Decimal	Hexadecimal
0000	0	0	1000	8	8
0001	1	1	1001	9	9
0010	2	2	1010	10	A
0011	3	3	1011	11	B
0100	4	4	1100	12	C
0101	5	5	1101	13	D
0110	6	6	1110	14	E
0111	7	7	1111	15	F

Data Representation: Translating **Binary** to **Hexadecimal**

- Each hexadecimal digit corresponds to **4 binary bits** (Why?).
- **Example**: Translate the binary integer

  000101101010011110010100

  to hexadecimal:

1	6	A	7	9	4
0001	0110	1010	0111	1001	0100

Table 1-5 Binary, Decimal, and Hexadecimal Equivalents.

Binary	Decimal	Hexadecimal	Binary	Decimal	Hexac
0000	0	0	1000	8	
0001	1	1	1001	9	
0010	2	2	1010	10	
0011	3	3	1011	11	
0100	4	4	1100	12	
0101	5	5	1101	13	
0110	6	6	1110	14	
0111	7	7	1111	15	

Data Representation: Converting **Hexadecimal to Decimal**

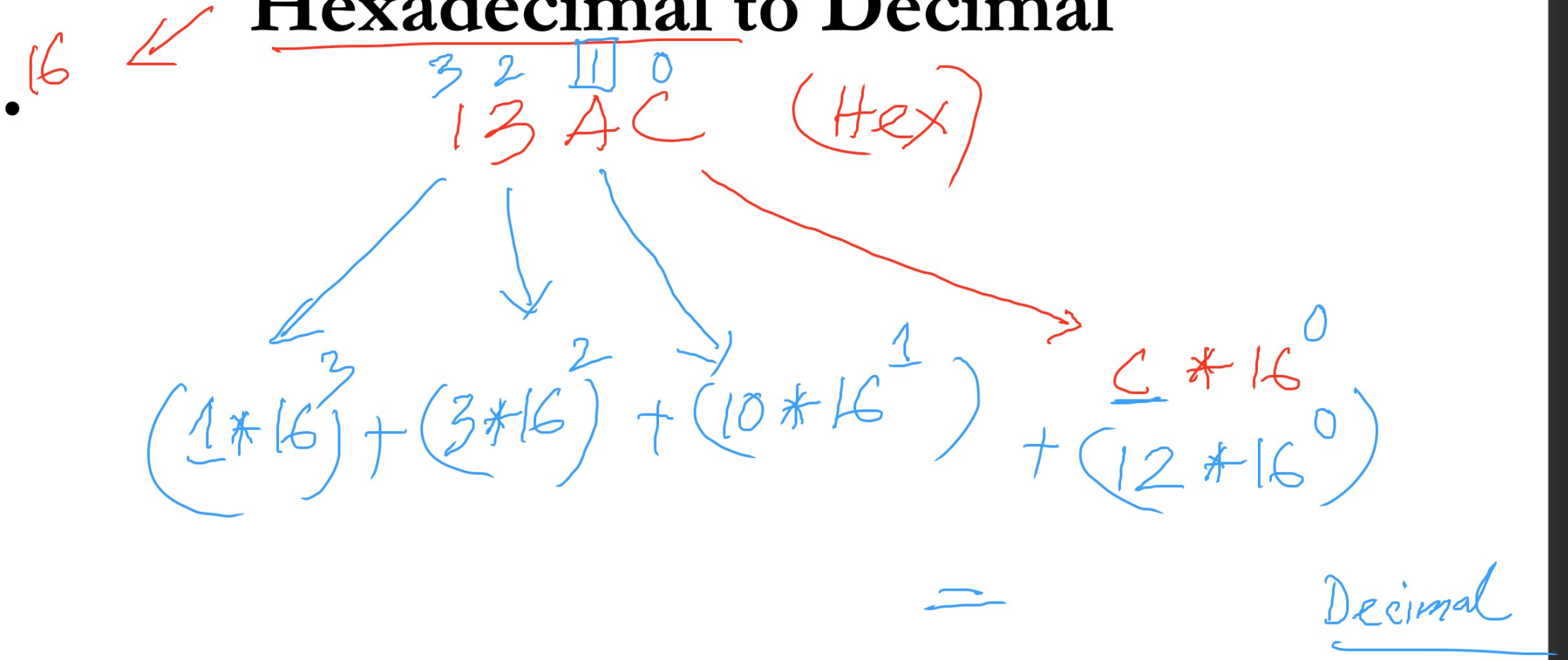
- Multiply each digit by its corresponding power of 16:

  $dec = (D_3 \times 16^3) + (D_2 \times 16^2) + (D_1 \times 16^1) + (D_0 \times 16^0)$

- Hex 1234 equals $(1 \times 16^3) + (2 \times 16^2) + (3 \times 16^1) + (4 \times 16^0)$, or decimal 4,660.

- Hex 3BA4 equals $(3 \times 16^3) + (11 * 16^2) + (10 \times 16^1) + (4 \times 16^0)$, or decimal 15,268.

Data Representation: Converting **Hexadecimal to Decimal**



Data Representation: Converting **Decimal to Hexadecimal**

422 Decimal

Division	Quotient	Remainder
422 / 16	26	6
26 / 16	1	(10)→A
1 / 16	0	1

(stop when Quotient == 0)

(1A6) Hex

decimal 422 = 1A6 hexadecimal

Data Representation: Hexadecimal Addition

- Divide the sum of two digits by the number base (16). The quotient becomes the carry value, and the remainder is the sum digit.

			1	1
⇒	36	28	28	6A
⇒	42	45	58	4B
	78	6D	80	B5

21 / 16 = 1, rem 5

Important skill: Programmers frequently add and subtract the addresses of variables and instructions.

Decimal

$$\begin{array}{r} {\scriptstyle 1} \\ 96 \\ +\ 37 \\ \hline 3 \end{array}$$

carry.

13

1 ← Result

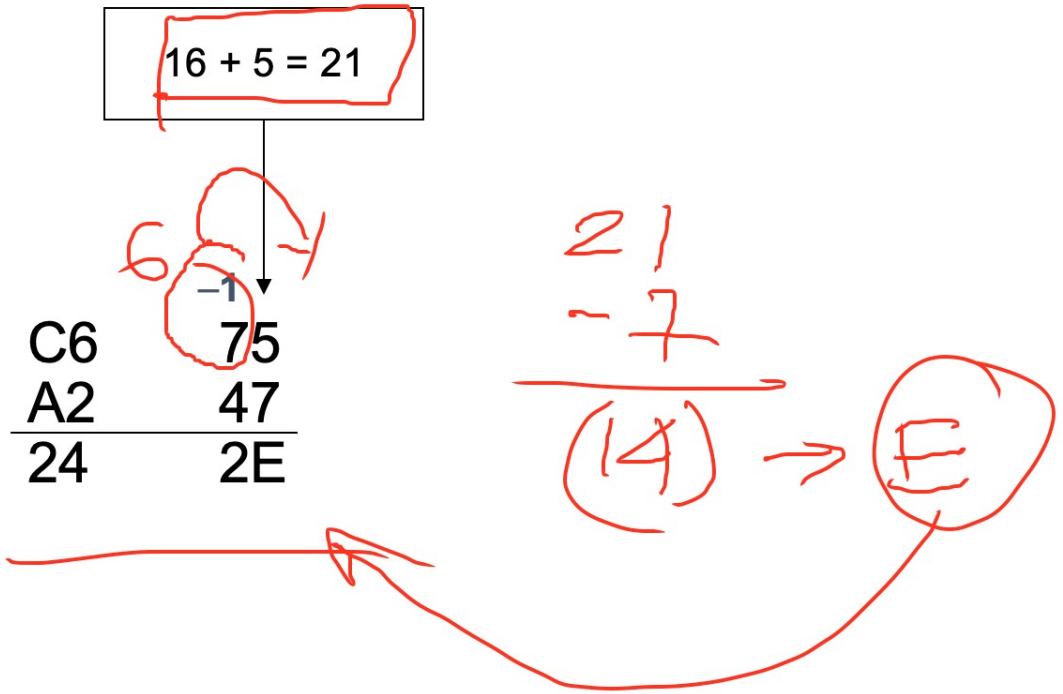
$$10\overline{)13}$$

10

3 ← Remainder

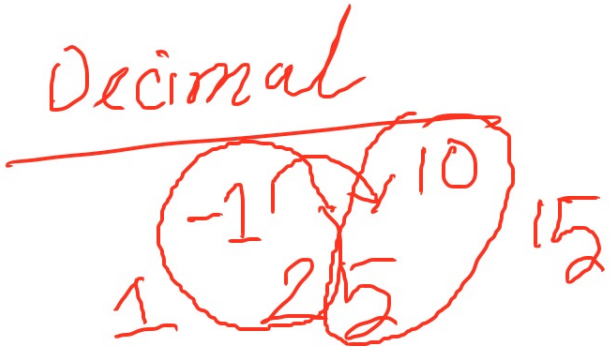
Data Representation: **Hexadecimal** **Subtraction**

- When a borrow is required from the digit to the left, **add 16 (decimal) to the current digit's value**:



21
−7
(14) → E

Decimal



− 18
07