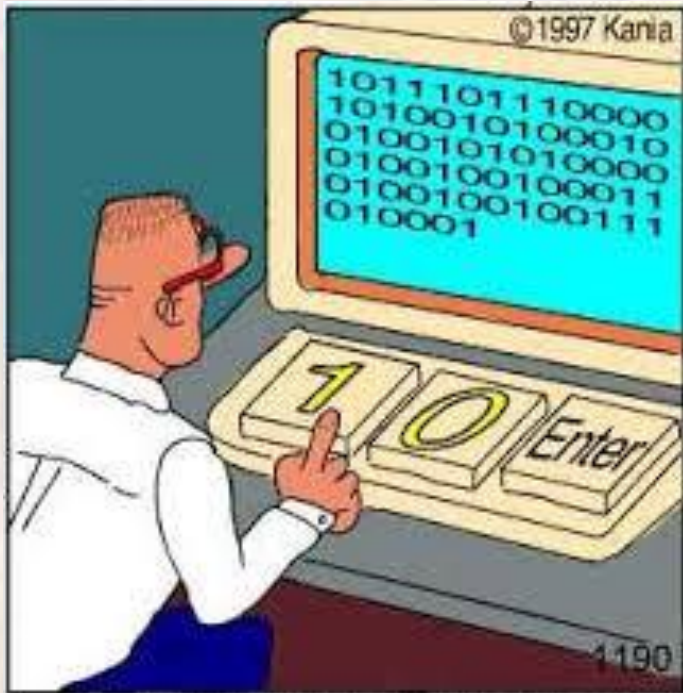


Bits and Bytes

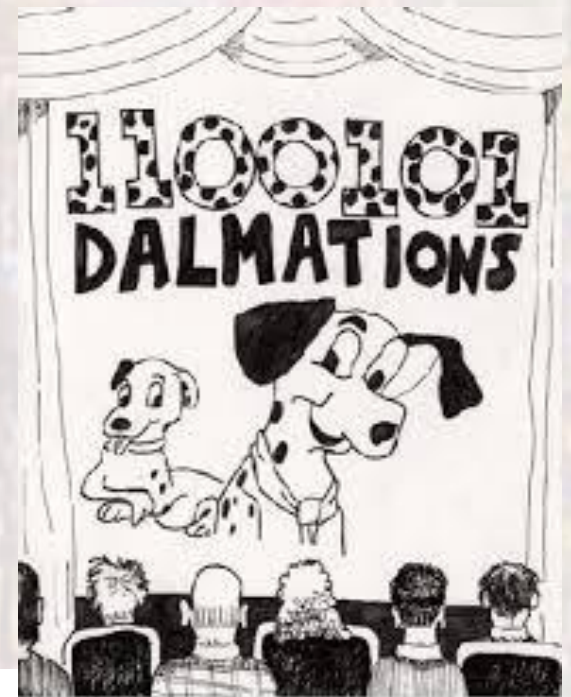
Some Digital Terminology

Last updated 6/13/23

These slides introduce basic digital number concepts



Real programmers code in binary.



Film Night at the Binary Society



www.clipartof.com - 443435

Bits and Bytes

- Base 10 (decimal)
 - The most familiar base for most people
 - ones, tens, hundreds, thousands
 - tenths, hundredths, thousandths
 - Base 10 → 10 individual digits
 - Range of individual digit: 0 → 9
 - Each position to the left of the decimal point is 10X the previous position
 - Each position to the right of the decimal point is 1/10th the previous position

1	2	3	4	.	5	6	7
Thousands	Hundreds	Tens	Ones	decimal point	tenths	hundredths	thousandths

1	2	3	4	.	5	6	7
digit x 10 ³	digit x 10 ²	digit x 10 ¹	digit x 10 ⁰	decimal point	digit x 10 ⁻¹	digit x 10 ⁻²	digit x 10 ⁻³

Bits and Bytes

- Base 2 (binary)
 - The most common base for digital electronics
 - ones, twos, fours, eights
 - halves, quarters, eighths
 - Base 2 → 2 individual digits
 - Range of individual digit: 0 → 1
 - Each position to the left of the binary point is 2X the previous position
 - Each position to the right of the binary point is 1/2 the previous position

1	1	0	1	.	1	0	1
Eights	Fours	Twos	Ones	binary point	Halves	Quarters	Eighths

1	1	0	1	.	1	0	1
digit x 2^3	digit x 2^2	digit x 2^1	digit x 2^0	binary point	digit x 2^{-1}	digit x 2^{-2}	digit x 2^{-3}

Bits and Bytes

- Base 16 (hexadecimal)
 - Used as a short hand for binary
 - ones, 16s, 256s, 4096s
 - 16ths, 256ths
 - Base 16 → 16 individual digits
 - Range of individual digit: 0 → 9, A → F
 - 10=A, 11=B, 12=C, 13=D, 14=E, 15=F
 - Each position to the left of the hexadecimal point is 16X the previous position
 - Each position to the right of the hexadecimal point is 1/16 the previous position

2	B	0	E	.	3	A	2
4096s	256s	16s	Ones	hexadecimal point	16ths	256ths	4096ths

2	B	0	E	.	3	A	2
digit x 16 ³	digit x 16 ²	digit x 16 ¹	digit x 16 ⁰	hexadecimal point	digit x 16 ⁻¹	digit x 16 ⁻²	digit x 16 ⁻³

Bits and Bytes

- Binary Terminology

- bit – logical interpretation

- The smallest logical quantity in a digital system
 - Can have a logical value of 0 or 1
 - It may be unknown to us, but it is always either 0 or 1

- bit – physical interpretation

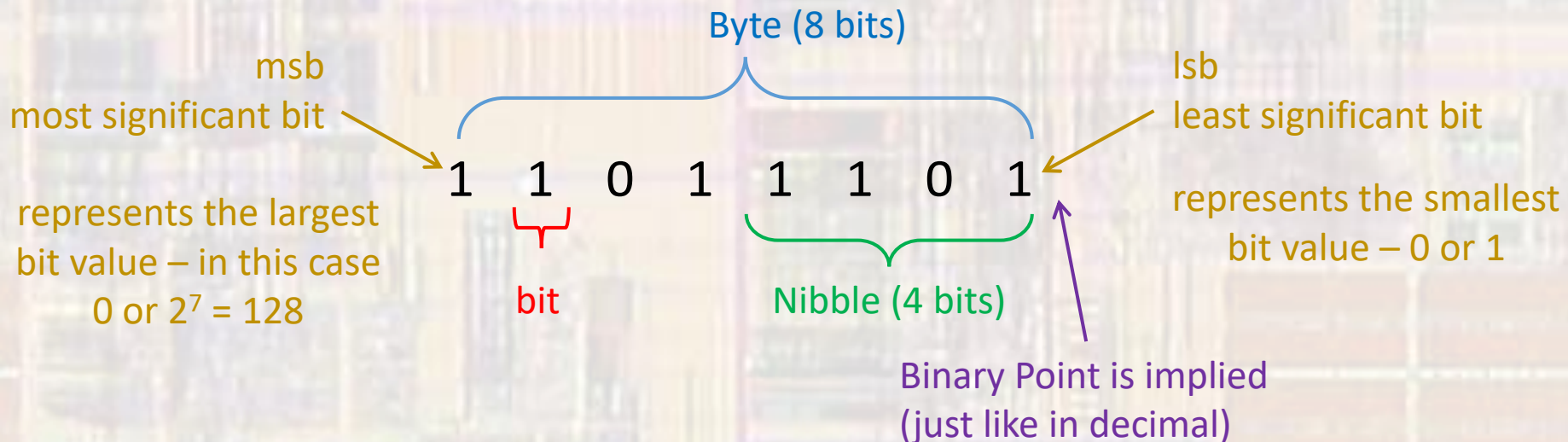
- Represents the value on a wire (or pin) in a digital system
 - The value is associated with a voltage
 - The physical value of a 0 is close to 0V
 - The physical value of a 1 is close to the supply voltage (Vdd)

System Vdd	5V	3.3V	1.8V
Physical 0	0V – 0.5V	0V – 0.3V	0V – 0.2V
Physical 1	4V – 5V	2.8V – 3.3V	1.5V – 1.8V

Simplified representation – more in your embedded systems and logic classes

Bits and Bytes

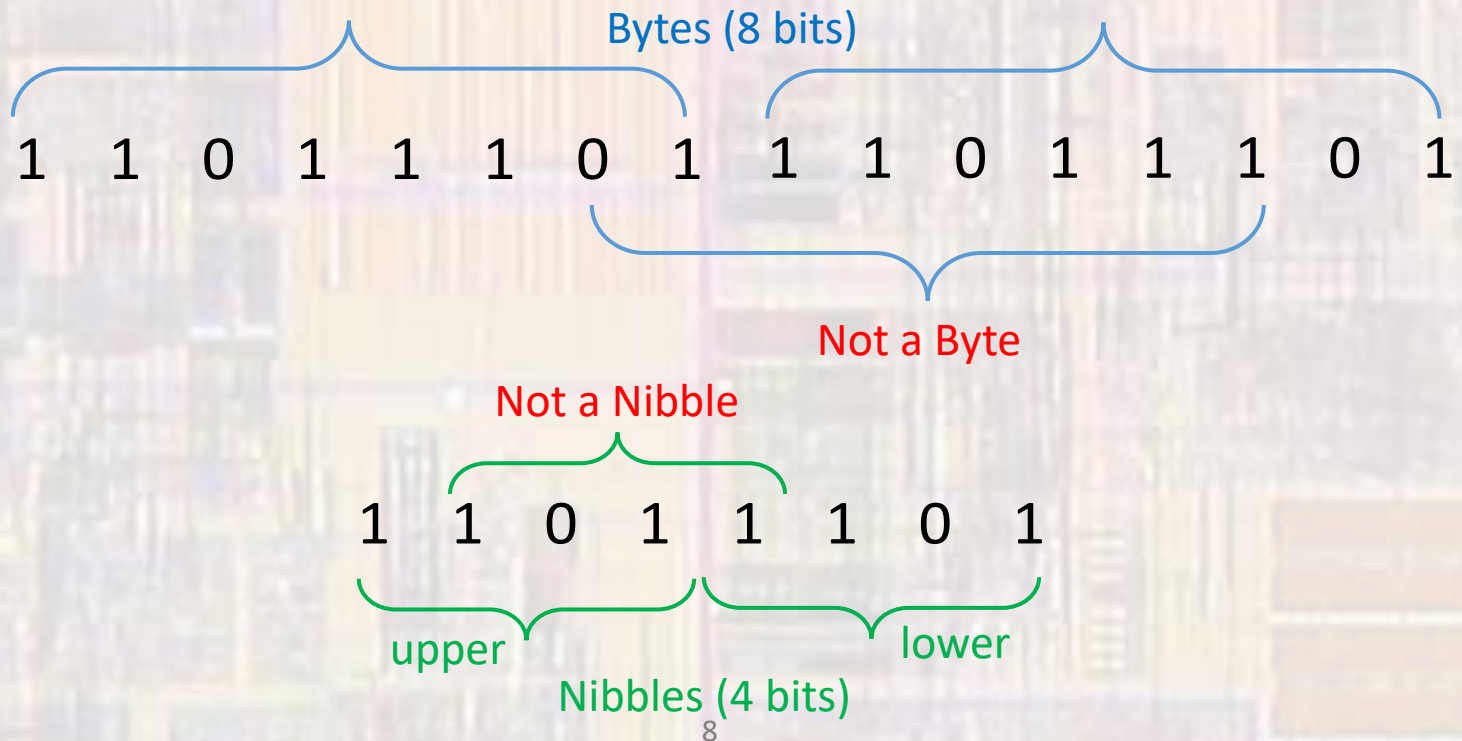
- Binary Terminology
 - **Bit**: single binary element
 - denoted with a small **b**
 - “0110 is a 4**b** number”
 - **Nibble**: a group of 4 bits
 - not normally referenced
 - **Byte**: a group of 8 bits
 - denoted with a capital **B**
 - “I need a 64**B** memory chip”



Bits and Bytes

- Binary Terminology

- **Bits** can be anywhere in the binary number
- **Bytes** are segmented from the binary point
 - Not just any set of 8 bits
- **Nibbles** are segmented within a byte
 - Upper and lower nibble



Bits and Bytes

- Binary Terminology
 - A **Word** is a logical collection of bytes

8 bit Word (1B)

1 1 0 1 1 1 0 1

16 bit Word (2B)

1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1

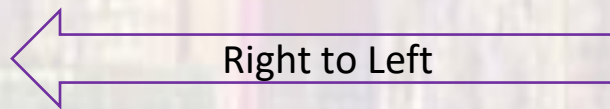
32 bit Word (4B)

1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1

64, 128, 256, 512, 1024 bit Words

Bits and Bytes

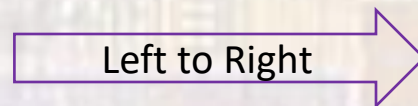
- Bit Values



bit #	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	1	0	1	1	1	0	1	1	1	0	0	0	1	0	0	0	0	1	1	0	0	1	0	0	1	1	0	0	0	1	1	
Value	2,147,483,648	1,073,741,824	536,870,912	268,435,456	134,217,728	67,108,864	33,554,432	16,777,216	8,388,608	4,194,304	2,097,152	1,048,576	524,288	262,144	131,072	65,536	32,768	16,384	8,192	4,096	2,048	1,024	512	256	128	64	32	16	8	4	2	1

bit #	8	7	6	5	4	3	2	1
	0	1	0	0	0	0	1	1
Value	0.00390625	0.0078125	0.015625	0.03125	0.0625	0.125	0.25	0.5

Binary Point →




Bits and Bytes

- Exponential shorthand
 - Use of K, M, G, T is **situationally dependent**
 - In science and math:
 - $K = \times 10^3$
 - $M = \times 10^6$
 - $G = \times 10^9$
 - $T = \times 10^{12}$
 - In computer and digital systems:
 - $K = \times 2^{10} = \times 1,024$ Kilo
 - $M = \times 2^{20} = \times 1,048,576$ Mega
 - $G = \times 2^{30} = \times 1,073,741,824$ Giga
 - $T = \times 2^{40} = \times 1,099,511,627,776$ Tera

Bits and Bytes

- Quick binary size calculation method

- Multiplying exponential numbers (with the same base) → adding the exponents


$$2^{12} = 2^2 2^{10} = 4 \text{ K}$$

Requires 12 bits

$$2^{16} = 2^6 2^{10} = 64 \text{ K}$$


Requires 16 bits

$$2^{23} = 2^3 2^{20} = 8 \text{ M}$$

Requires 23 bits

$$2^{35} = 2^5 2^{30} = 32 \text{ G}$$

Requires 35 bits


$$16\text{K} = 2^4 2^{10} = 2^{14}$$

Requires 14 bits

$$2\text{K} = 2^1 2^{10} = 2^{11}$$

Requires 11 bits

$$1\text{M} = 2^0 2^{20} = 2^{20}$$

Requires 20 bits

$$128\text{G} = 2^7 2^{30} = 2^{37}$$

Requires 37 bits

Bits and Bytes

- More Terminology

- Assume S is an 8 bit binary number

S = 10010110

- S[7:0] = 10010110 10010110
- S[3:0] = 0110 10010110
- S[7:6] = 10 10010110
- S[5] = 0 10010110
- S[6,3] = 00 10010110
- S[1] = 1 10010110
- S[0] = 0 10010110

Bits and Bytes

- Special note on binary numbers in C programming
 - Some **but not all** compilers allow binary numbers to be represented in C code directly

95 → **0b01011101**

- To be safe and ensure our code is portable we will **NOT** use this notation.
- Binary numbers can be represented with:
 - Their decimal equivalents **95**
 - Their hexadecimal equivalents **0x5D**