CS/EE-220: Lecture-2

Digital Logic Circuits

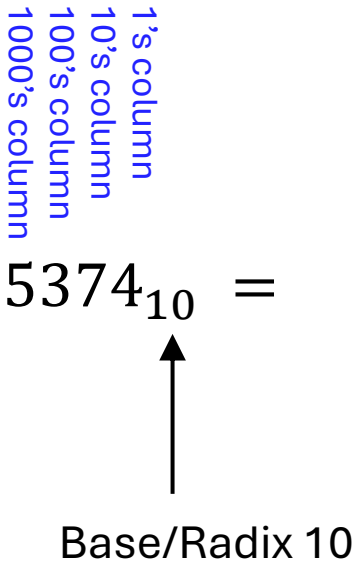
Number Systems and Codes

Number Systems

Decimal, Binary, Hexadecimal Numbers
And their Conversions

Number Systems

- Positional Number Systems
  - Positional of a digit determines its value
  - Example
    - 888 – Each digit ‘eight’ is unique
- For instance, the decimal system



Number Systems

- Positional Number Systems
  - Positional of a digit determines its value
  - Example
    - 888 – Each digit ‘eight’ is unique
- For instance, the decimal system

1’s column
10’s column
100’s column
1000’s column

$$5374_{10} = 5 \times 10^3 + 3 \times 10^2 + 7 \times 10^1 + 4 \times 10^0$$

Five thousands | Three hundreds | Seven tens | Four ones

Base/Radix 10

Place Values in the Decimal System

- Integer Part
  - Moving from right to left the associated value of a digit is increased in powers of 10 (decimal multiplication)
- Fraction Part
  - Moving from left to right the associated value of a digit is reduced in powers of 10 (decimal division)

M	Hth	TTh	T	H	T	O	$\frac{1}{10}$	$\frac{1}{100}$	$\frac{1}{1000}$
0	0	0	0	0	0	0 .	0	0	0
Millions	Hundred Thousands	Ten Thousands	Thousands	Hundreds	Tens	Ones	Tenths	Hundredths	Thousandths

$$5374.5_{10} = 5 \times 10^3 + 3 \times 10^2 + 7 \times 10^1 + 4 \times 10^0 + 5 \times 10^{-1}$$

Five thousands | Three hundreds | Seven tens | Four ones | Five tenths

Positional Number System

- Generic Notation
  - A number with radix r is represented by a string of n integral digits and m fractional digits

  - $A_{n-1}\ A_{n-2} \cdots A_1\ A_0 . A_{-1}\ A_{-2} \cdots A_{-m+1}\ A_{-m}$ (1)
  - Where $0 \leq A_i < r$ and . is the radix point

  - $\sum_{i=0}^{n-1} A_i \times r^i + \sum_{j=-m}^{-1} A_j \times r^j = \sum_{i=-m}^{n-1} A_i \times r^i$ (2)

- Coefficients? (value)
- Subscripts? (place)

Positional Number Systems

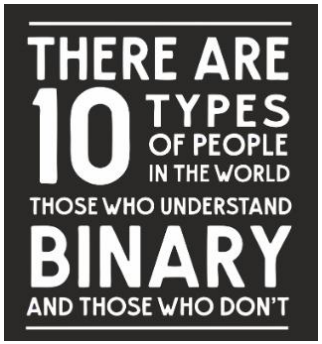
- Generic Notation
  - A number with radix r is represented by a string of n integral digits and m fractional digits

  - $A_{n-1}\ A_{n-2}\ \cdots\ A_1\ A_0 . A_{-1}\ A_{-2}\ \cdots\ A_{-m+1}\ A_{-m}$ (1)
  - Where $0 \leq A_i < r$ and . is the radix point

- A_{n-1} is referred to as the most significant digit (MSD) and A_{-m} as the least significant digit (LSD) of the number

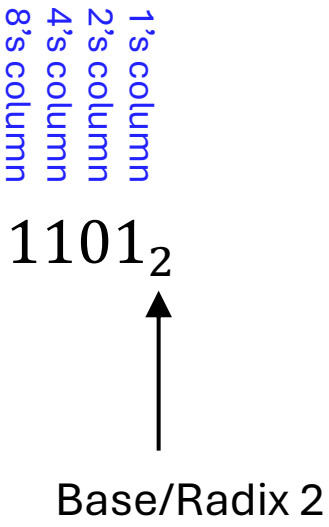
Binary Number System

- Coefficient / binary digit can take [0 to 1], i.e., 0 or 1
  - **Binary digit : Bit (shorter form)**
- Have you heard the term before?
  - One of the basic data units in a digital systems like a computer
- How to represent a 2 in binary?



Binary Numbers

- **Base 2** (binary numbers)



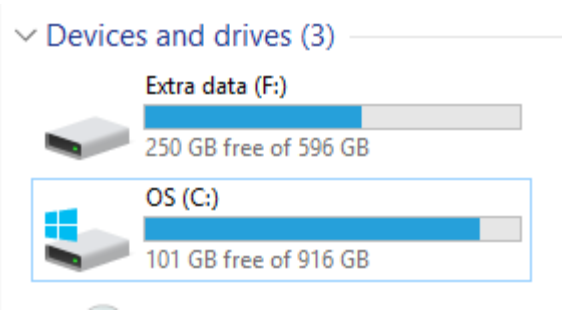
Binary Number System

		General	Decimal	Binary
Radix (base)		**r**	**10**	**2**
Digit Range		**0 → r-1**	**0 → 9**	**0 → 1**
		Weight (Value)		
Powers of Radix (Place)	**0**	r^0	1	1
	1	r^1	10	2
	2	r^2	100	4
	3	r^3	1000	8
	4	r^4	10,000	16
	5	r^5	100,000	32
	-1	r^{-1}	0.1	0.5
	-2	r^{-2}	0.01	0.25
	-3	r^{-3}	0.001	0.125
	-4	r^{-4}	0.0001	0.0625
	-5	r^{-5}	0.00001	0.03125

Special Powers of 2 in Binary Number System

- $2^{10} = 1024_{10}$ is kilo (Ki) $\approx 1000_{10}$
- $2^{20} = 1{,}048{,}576_{10}$ is Mega (Mi) $\approx (1000)^2{}_{10}$
- $2^{30} = 1{,}073{,}741{,}824_{10}$ is Giga (Gi) $\approx (1000)^3{}_{10}$
- $2^{40} = 1{,}099{,}511{,}627{,}776_{10}$ is Tera (Ti) $\approx (1000)^4{}_{10}$
- Example: 8 Mi bits = $2^3 \times 2^{20} = 2^{23} = 8{,}388{,}608$ bits $\approx$ 8 Million bits

Giga vs Gibi



The class action lawsuit against Kodak, Sandisk, Lexar Media, and other memory card makers alleges that the defendants intentionally misrepresented the capacity of their flash memory devices by using decimal definitions, in which a megabyte is 1,000,000 bytes. The suit says a binary definition is appropriate, meaning that one megabyte equals 1,048,576 bytes and that the memory card sizes were overstated by 4 percent to 5 percent.

Converting from Any Base to Decimal?

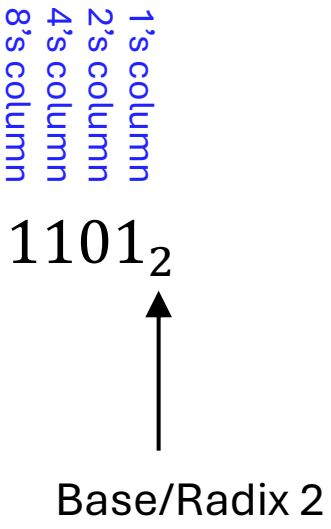
Power Series Expansion

(Recall Expanded Form in Primary School ☺)

M	HTh	TTh	T	H	T	O	$\frac{1}{10}$	$\frac{1}{100}$	$\frac{1}{1000}$
0	0	0	0	0	0	0	0	0	0
Millions	Hundred Thousands	Ten Thousands	Thousands	Hundreds	Tens	Ones	Tenths	Hundredths	Thousandths

Binary Numbers

- **Base 2** (binary numbers)



		General	Decimal	Binary
Radix (base)		**r**	**10**	**2**
Digit Range		**0 → r-1**	**0 → 9**	**0 → 1**
		Weight (Value)		
Powers of Radix (Place)	**0**	r^0	1	1
	1	r^1	10	2
	2	r^2	100	4
	3	r^3	1000	8
	4	r^4	10,000	16
	5	r^5	100,000	32
	-1	r^{-1}	0.1	0.5
	-2	r^{-2}	0.01	0.25
	-3	r^{-3}	0.001	0.125
	-4	r^{-4}	0.0001	0.0625
	-5	r^{-5}	0.00001	0.03125

Binary Numbers

- **Base 2** (binary numbers)

8's column
4's column
2's column
1's column

$$1101_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 13_{10}$$

one eight | one four | no two | one one

Base/Radix 2

ICP 2-1 Convert $(110101.11)_2$ to decimal

Hexadecimal to Decimal Conversion

- Hexadecimal = Radix-16 Number System
- Hexadecimal digits can take on [0 to ?] value?
  - [0 to 15] but how to represent a 15 in single digit?
  - We use English alphabets for hex digits > 9
    - `10 = A, 11 = B, 12 = C, 13 = D, 14 = E, 15 = F`

ICP 2-2

$B65F.7_{16} \rightarrow ??_{10}$

		General	Hexadecimal
Radix (base)		r	16
Digit Range		0 → r-1	0 → 15 (f)
		Weight (Value)	
Powers of Radix (Place)	0	r^0	1
	1	r^1	16
	2	r^2	256
	3	r^3	4096
	4	r^4	65,536
	5	r^5	1,048,576
	-1	r^{-1}	0.0625
	-2	r^{-2}	0.00390625
	-3	r^{-3}	0.000244140

Decimal to any other Base?

1. Weightage Method
2. Successive Division and Multiplication

Decimal to Binary Conversion

- **Two methods:**
  - **Method 1:** Find the **largest power of 2 that fits**, subtract and repeat.
  - **Method 2:** Repeatedly **divide by 2**, remainder goes in next most significant bit

Decimal to Binary Conversion

Method 1: Find the **largest power of 2 that fits**, subtract and repeat.

53_{10} **32**×1

53-32 = 21 **16**×1

21-16 = 5 **4**×1

5-4 = 1 **1**×1 **= 110101_2**

ICP 2-3 Convert $(625)_{10}$ to binary

n	2^n
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8	256
9	512
10	1024

Decimal to Binary Conversion

Method 1: Find the **largest power of 2 that fits**, subtract and repeat.

53_{10} 32×1

53-32 = 21 16×1

21-16 = 5 4×1

5-4 = 1 1×1 = 110101_2

Method 2: **Repeatedly divide by 2,**
remainder goes in next most significant bit.

53_{10}

53/2 = 26 R1
26/2 = 13 R0
13/2 = 6 R1
6/2 = 3 R0
3/2 = 1 R1
1/2 = 0 R1

= 110101_2

n	2^n
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8	256
9	512
10	1024

ICP 2-4 Convert $(105)_{10}$ to binary

Converting Fractions

- To convert from one base to another:

  1) Convert the Integer Part
  2) Convert the Fraction Part
  3) Join the two results with a radix point

Converting Fractions

- To Convert the Integral Part:

  Repeatedly divide the number by the new radix and save the remainders. The digits for the new radix are the remainders in *reverse order* of their computation.

- To Convert the Fractional Part:

  Repeatedly multiply the fraction by the new radix and save the integer digits that result. The digits for the new radix are the integer digits in *order* of their computation.

Converting Fractions

Example: Convert 0.6875_{10} To Base 2

$$0.6875 = A_{-1} \times 2^{-1} + A_{-2} \times 2^{-2} + A_{-3} \times 2^{-3} + A_{-4} \times 2^{-4} + \cdots$$

Multiply by 2

$$1.375 = A_{-1} + A_{-2} \times 2^{-1} + A_{-3} \times 2^{-2} + A_{-4} \times 2^{-3} + \cdots$$

Assign $A_{-1} = 1$. Subtract A_{-1} and Multiply again by 2

$$0.75 = A_{-2} + A_{-3} \times 2^{-1} + A_{-4} \times 2^{-2} + \cdots$$

Assign $A_{-2} = 0$. Subtract A_{-2} and Multiply again by 2

$$1.5 = A_{-3} + A_{-4} \times 2^{-1} + \cdots$$

Assign $A_{-3} = 1$. Subtract A_{-3} and Multiply again by 2

$$1.0 = A_{-4} + \cdots$$

Assign $A_{-4} = 1$. Subtract A_{-4} and Multiply again by 2

Since the fractional part is now zero, the process is terminated

$$(0.6875)_{10} = (0.1011)_2$$

ICP 2-5 Convert $(0.65)_{10}$ to binary

Converting Fractions

- Note that in the first conversion, the fractional part became 0 as a result of the repeated multiplications.
- In general, it may take many bits to get this to happen or it may never happen.
- Example: Convert 0.65_{10} to binary
  - 0.65 = 0.10100011001001 …
  - The fractional part begins repeating every 4 steps yielding repeating 1001 forever!
- Solution: Specify number of bits to right of radix point and round or truncate to this number.

The Patriot Missile Failure

On February 25, 1991, during the Gulf War, an American Patriot Missile battery in Dharan, Saudi Arabia, failed to track and intercept an incoming Iraqi Scud missile. The Scud struck an American Army barracks, killing 28 soldiers and injuring around 100 other people.

- The system's internal timer incremented in tenths of a second (0.1_{10}).
- The decimal fraction 0.1translates to an infinitely repeating binary sequence ($0.000110011\ldots_2$).
- The missile battery was left running continuously for over 100 hours. The microscopic truncation error multiplied over millions of clock ticks, resulting in a 0.34-second time shift.
- In 0.34 seconds, the missile moved over 500 meters outside the predicted tracking gate.

Decimal to Hexadecimal Conversion

- Integer Part:
  - Successive Division by 16 until Quotient is 0
- Fraction Part:
  - Successive Multiplication by 16 until fraction part is 0
  - Or until the required fractional precision

ICP 2-6

$153_{10} \rightarrow ??_{16}$

$0.515625_{10} \rightarrow ??_{16}$

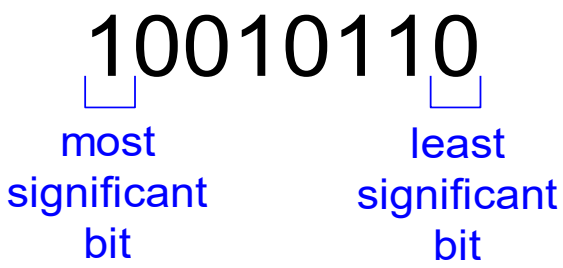
$153.515625_{10} \rightarrow ??_{16}$

Binary ⟺ Hexadecimal

Shortcut Method

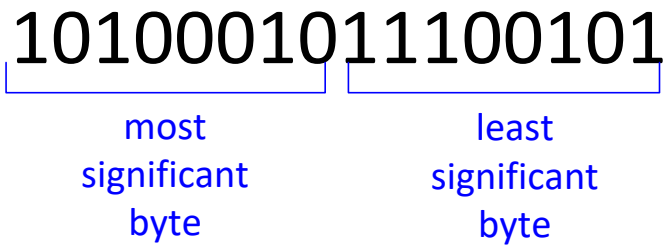
Bits, Bytes, Nibbles

- **Bits**
  - **msb:** most significant bit
  - **lsb:** least significant bit



- **Bytes & Nibbles**

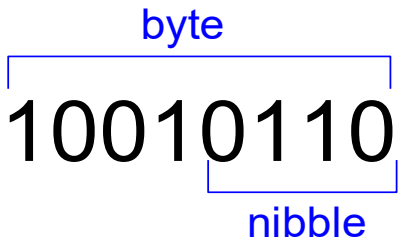
- **Bytes**
  - **MSB:** most significant byte
  - **LSB:** least significant byte



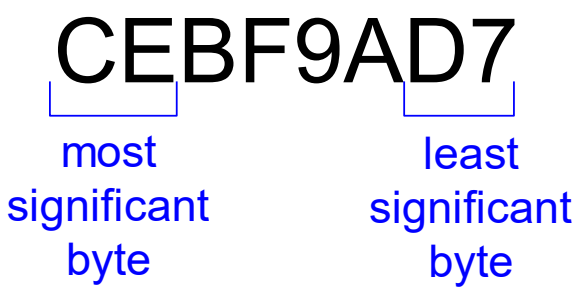
Bits, Bytes, Nibbles

- **Bits**
  - **msb:** most significant bit
  - **lsb:** least significant bit

- **Bytes & Nibbles**



- **Bytes**
  - **MSB:** most significant byte
  - **LSB:** least significant byte
  - Each hex digit represents a nibble (4 bits)



Binary to/from Hexadecimal Conversion

- Long Method

- Short Method
  - Direct Binary to Hexadecimal
    - Starting from least significant bit (LSB) position, i.e., the fractional point, each group of 4-bits correspond to a hexadecimal number
- Example:
  - $1\,0110\,1100\,1010\,1011_2 \rightarrow ??_{16}$

    1 6 C A B

    $16CAB_{16}$

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

Better memorize this basic conversion table

Binary to/from Hexadecimal Conversion

- Direct Hexadecimal to Binary
  - Inverse of Binary to Hexadecimal conversion
  - Represent each hexadecimal number as a 4-bit binary number from the table

ICP 2-7

$3A6.C_{16} \rightarrow ??_2$

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

Better memorize this basic conversion table

Binary Codes

Non-numeric Binary Codes

- Given *n* binary digits (called bits), a binary code is a mapping from a set of represented elements to a subset of the 2^n binary numbers.
- Example: A binary code for the seven colors of the rainbow
- Code 100 is not used

Color	Binary Number
Red	000
Orange	001
Yellow	010
Green	011
Blue	101
Indigo	110
Violet	111

ICP 2-8 How many bits are required to represent decimal digits with a binary code?

Binary Codes for Decimal Digits

There are over 8,000 ways that you can chose 10 elements from the 16 binary numbers of 4 bits. A few are useful:

Decimal	8,4,2,1	Excess3	8,4,-2,-1	Gray
0	0000	0011	0000	0000
1	0001	0100	0111	0100
2	0010	0101	0110	0101
3	0011	0110	0101	0111
4	0100	0111	0100	0110
5	0101	1000	1011	0010
6	0110	1001	1010	0011
7	0111	1010	1001	0001
8	1000	1011	1000	1001
9	1001	1100	1111	1000

Binary Coded Decimal (BCD)

- The BCD code is the 8,4,2,1 code.
- This code is the simplest, most intuitive binary code for decimal digits and uses the same powers of 2 as a binary number, but only encodes the first ten values from 0 to 9.

$(396)_{10}$ is represented in BCD with 12 bits as

$0011\ 1001\ 0110$

BCD is not the same as the binary equivalent

Compare the value of $(185)_{10}$ in BCD and binary

Alphanumeric Codes

American Standard Code for Information Interchange (ASCII)

	$B_7B_6B_5$							
$B_4B_3B_2B_1$	000	001	010	011	100	101	110	111
0000	NULL	DLE	SP	0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

QUIZ 1

- Wednesday, September 9, 2026
- Syllabus: Number systems and binary codes.