EE1005 – Digital Logic Design

- **Lecture Slides**
- **Week 01**

Course Instructor:

Dr. Arslan Ahmed Amin

FAST National University of Computer and Emerging Sciences CFD Campus

Instructor's Introduction

- **Dr. Arslan Ahmed Amin**
  - PhD (EE) from UET Lahore
  - MS (EE) from UET Lahore
  - MBA (Management) from VU
  - BS (EE) from UET Lahore (Main Campus)
  - Fulbright PhD USA Award Winner
  - Overall Top in Intermediate F.Sc. Examination BISE Faisalabad
  - Represented as Talent of Pakistan Youth in China
  - Gold Medal Winner in Faisalabad Distrcit Science Quiz Competition
  - Worked for 07 years as Engineer (Electrical / Instrumentation) at Pakistan Petroleum Limited (PPL)
  - Worked for 1.5 years as Lecturer (EE) at National Textile University, Faisalabad
  - Working at FAST NUCES CFD Campus since January 2019 and now Associate Professor (EE)

Research Interests

- Fault Tolerant Control
- Non Linear Control
- Automation and Control
- Power Electronic Control

Course Guidelines

- Course Outlines
- Class Norms
- Assignment Norms
- Attendance
- Deadlines
- Professional Behavior

Books

- Text Book
  - ***"Digital Design"*** by M. Morris Mano, Michael Ciletti, 5th Edition

- Reference Book
  - ***"Logic and Computer Design Fundamentals"*** by M. Morris Mano, Charles Kime

Course Outline

Week	Topics	Chapter
1, 2	Introduction, Number Systems (Binary, Octal and Hexadecimal), Number Ranges, Arithmetic Operations, Conversion from Decimal to Other Bases, Negative number representations	1
3, 4, 5	Binary Logic and Gates, Boolean Algebra, Standard Forms, Map Simplification, Map Manipulation, Don't-Care Conditions, NAND, NOR & Exclusive-OR Gates & Circuits, Integrated Circuits, Positive & Negative Logic	2, 3
6, 7, 8	Combinational Circuits, Analysis Procedure, Design Procedure, Decoders, Encoders, Priority Encoders, Multiplexers, Demultiplexers, Binary Adders (Half, Full, Ripple Carry, Carry Lookahead), Binary Subtraction, Signed Binary Numbers, Overflow	4
9, 10, 11	Sequential Circuits, Latches, Flip-Flops, Sequential Circuit Analysis, State Diagram, Sequential Circuit Design (with D Flip-Flops, JK Flip-Flops, T flip flops), word problems	5
12, 13, 14	Registers and Counters, Register with Parallel Load, Shift Registers, Shift Register with Parallel Load, Bidirectional Shift Register, Ripple Counter, Synchronous Binary Counters, Serial and Parallel Counters, Up-Down Binary Counter, Binary Counter with Parallel Load. Miscellaneous Counters (BCD Counters, Arbitrary Sequence Counters)	6
15, 16	Memory and Programmable Logic Devices, Random-access Memory, Dynamic RAM ICs, Programmable Logic Technologies (Read-only Memory, Programmable Logic Arrays, Programmable Array Logic Devices, FPGAs)	7

Marks Distribution

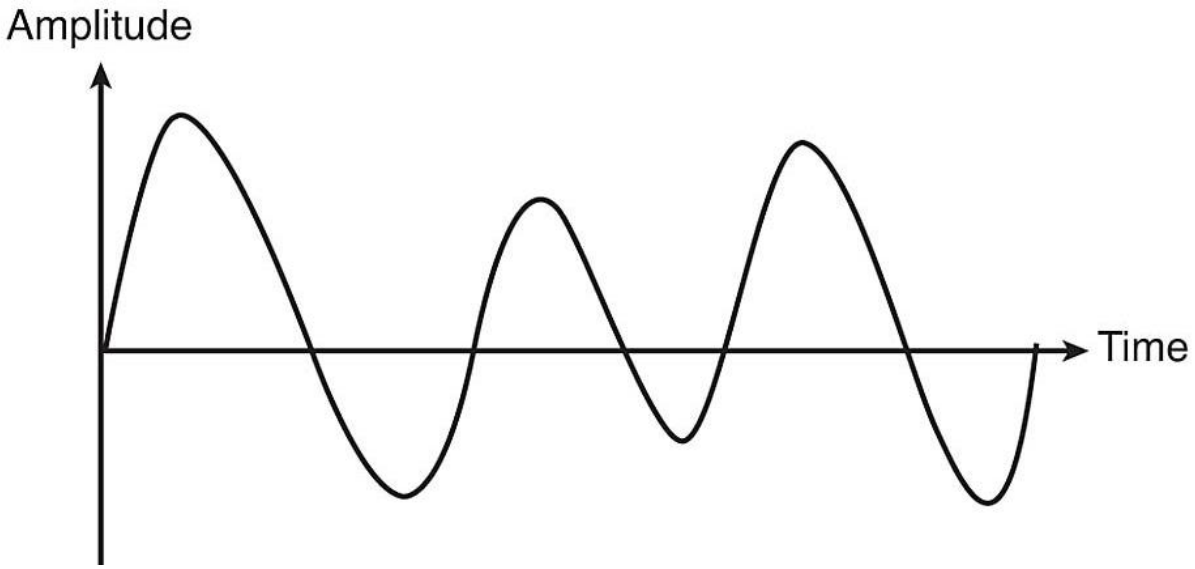
Mid Term I	15%
Mid Term II	15%
Quizzes	10 %
Assignment/Project	10%
Final Exam	50%

Information Processing

- In our daily life we have to process the following types of data
  - Text
  - Numbers
  - Formulas and Equation
  - Pictures
  - Audio
  - Video
- All above can be divided in two categories
  - Analog
  - Digital

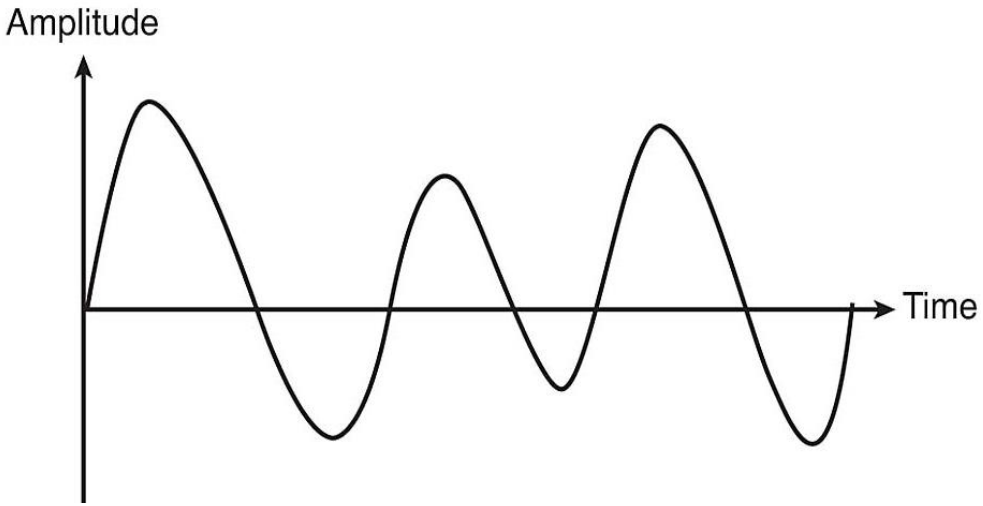
Analog Quantities

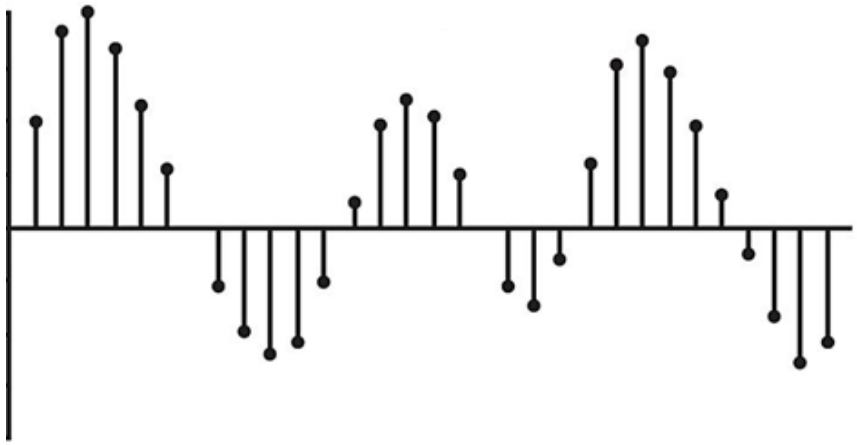
- Continuous in nature
  - Intensity of light
  - Temperature
  - Velocity

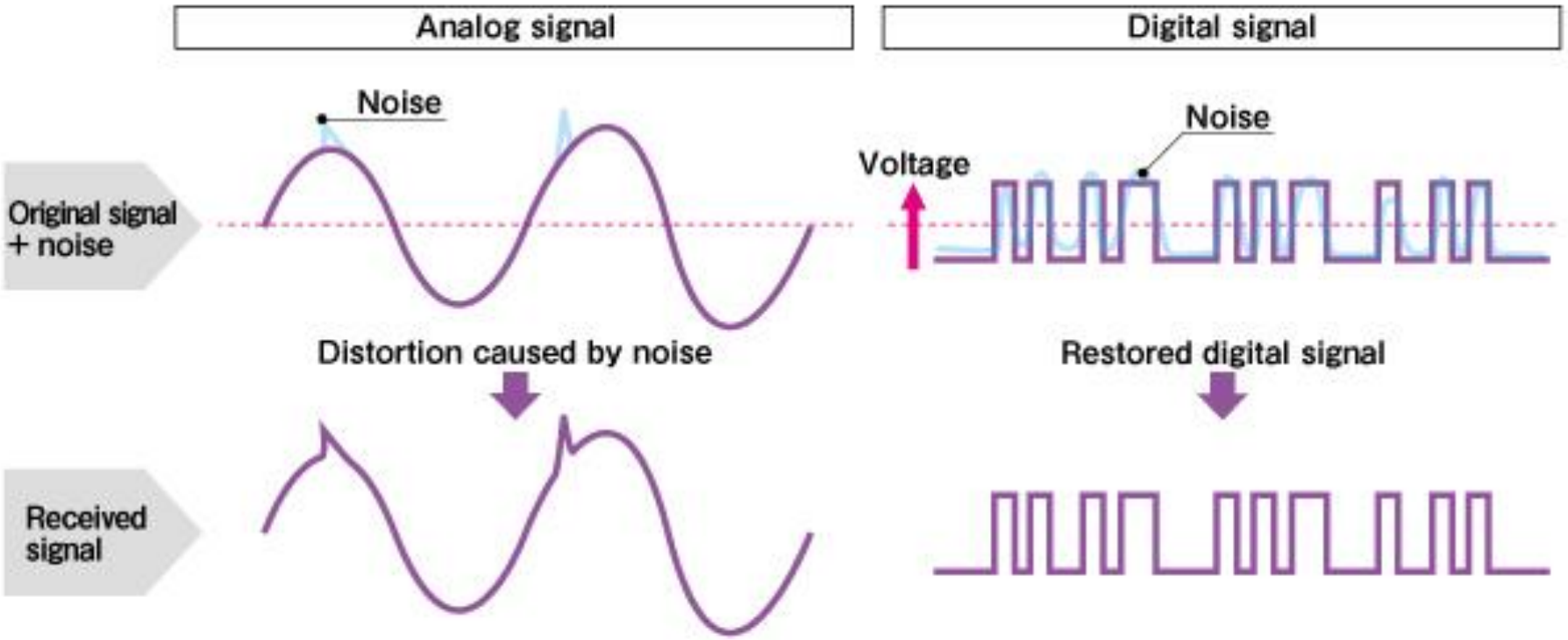


Digital Quantities

- Discrete set of values
- Can be obtained by sampling of analog signals






Analog signal
Digital signal
Noise
Noise
Voltage
Original signal + noise
Distortion caused by noise
Restored digital signal
Received signal

Advantages of using Digital System

- Efficient Processing & Data Storage
- Efficient & Reliable Transmission
- Detection and Correction of Errors
- Precise & Accurate Reproduction
- Easy Design and Implementation
- Occupy minimum space
- Digital Systems are programmable
- Overall cost of system is less

Why Digital Logic Design is Important?

- A digital system is an interconnection of digital modules
- **To understand the operation of each digital module, it is necessary to have a basic knowledge of digital circuits and their logical function**

Numbers with Different Bases

Decimal	Binary	Octal	Hex
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Data Representation

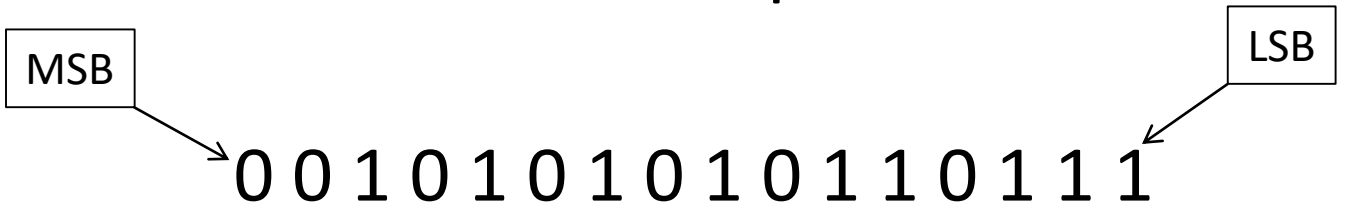
- Four basic data representation techniques
  - Binary (base 2)
  - Octal (base 8)
  - Decimal (base 10)
  - Hexadecimal (base 16)

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

For a valid representation the digits used must be less than the Base

Binary Number System (1/2)

- Data is represented with two possible states
- Digits 1 and 0 are used to represent
  - 1 → True
  - 0 → False
- Numbers are stored as sequence of 1s and 0s



- Leftmost bit is call Most Significant Bit (MSB)
- Rightmost bit is called Least Significant Bit (LSB)

Binary Number System (2/2)

- Each bit is either 0 or 1
- The weight of each bit is the power of 2
- Bit number starts from 0
- And counting starts from LSB

Bits	1	1	1	1	1	1	1	1
Weight	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1

Signed and Unsigned Binary Numbers

- Binary integers can be signed or unsigned
- Signed integers, as evident from name, are either positive or negative
- Unsigned integers are by default positive

Hexadecimal Number System

- Used to represent large binary numbers
- Digits 0 to 15 are used in hexadecimal notation where 10 is A, 11 is B, 12 is C, 13 is D, 14 is E, and 15 is F
- Commonly used to represent memory addresses
- The weight of each digit is the power of 16

Decimal and Octal Number System

- In decimal number system there are 10 possible digits (0 – 9)
- The weight of each digit is the power of 10
- For example $(9874)_{10}$ is a number in decimal number system
- In Octal number system there are 8 digits (0 – 7)
- The weight of each digit is the power of 8
- For example the number $(7301)_8$ is a number is octal system

Base Conversions

- We will discuss the following base conversions during this course
  - Conversions of unsigned numbers
  - Conversions of fractions
  - Conversions of signed numbers

Unsigned Binary Integers to Decimal

- Weighted Positional Notation method

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

  - D = binary digit
  - n = bit position number in binary number
- Example
  - Convert 11010 to decimal
  - Convert 11001110 to decimal

As noted before, the digits in a binary number are called *bits*. When a bit is equal to 0, it does not contribute to the sum during the conversion. Therefore, the conversion from binary to decimal can be obtained by adding only the numbers with powers of two corresponding to the bits that are equal to 1. For example,

$$(110101)_2 = 32 + 16 + 4 + 1 = (53)_{10}$$

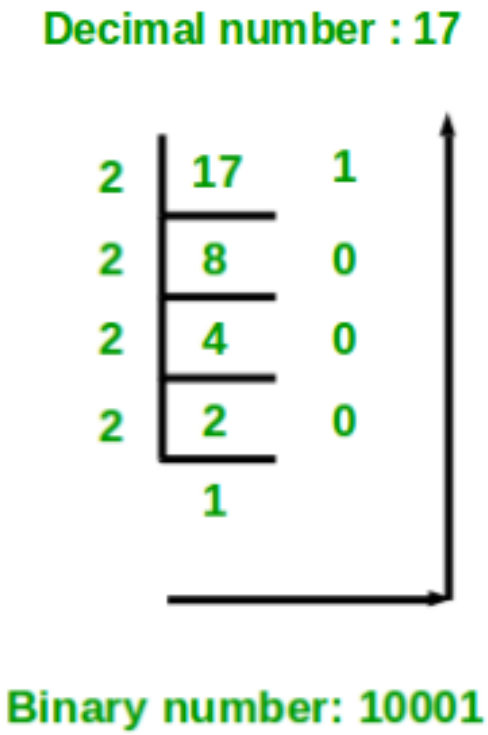
There are four 1's in the binary number. The corresponding decimal number is the sum of the four powers of two. The first 24 numbers obtained from 2 to the power of n are listed in Table 1.1. In computer work, 2^{10} is referred to as K (kilo), 2^{20} as M (mega), 2^{30} as G (giga), and 2^{40} as T (tera). Thus, $4K = 2^{12} = 4{,}096$ and $16M = 2^{24} = 16{,}777{,}216$. Computer capacity is usually given in bytes. A *byte* is equal to eight bits and can accommodate (i.e., represent the code of) one keyboard character. A computer hard disk with four gigabytes of storage has a capacity of $4G = 2^{32}$ bytes (approximately 4 billion bytes).

Decimal to Unsigned Binary Conversion

- Repeatedly divide the decimal integer by 2 until the quotient is 0
- The combination of remainders makes the binary number
- The first remainder goes at LSB position and last remainder goes at MSB position
- Example
  - Convert 25 from decimal to binary
  - Convert 115 from decimal to binary

Decimal to binary conversion using

Sum of weight



Therefore, the answer is $(41)_{10} = (a_5a_4a_3a_2a_1a_0)_2 = (101001)_2$.

The arithmetic process can be manipulated more conveniently as follows:

Integer	Remainder	
41		
20	1	
10	0	
5	0	
2	1	
1	0	
0	1	101001 = answer

Conversion from decimal integers to any base-r system is similar to this example, except that division is done by r instead of 2.

Decimal-Binary Conversion

- Binary to Decimal Conversion
  - Sum-of-Weights
  - Adding weights of non-zero terms

$$10011_2 = 16 + 2 + 1 = 19$$

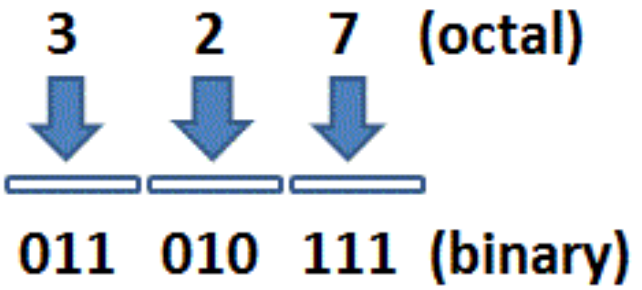
$$1011.101_2 = 8 + 2 + \frac{1}{2} + \frac{1}{8}$$

$$= 11 + \frac{5}{8}$$

$$= 11.625$$

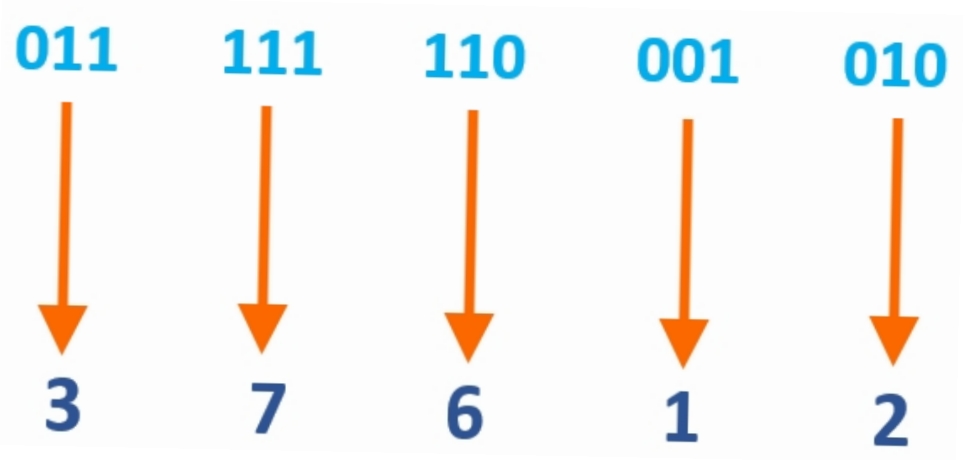
Octal to Binary

- Each octal integer corresponds to 3 binary bits
- Convert each octal number to corresponding binary number
- Example
  - Convert 327 to binary



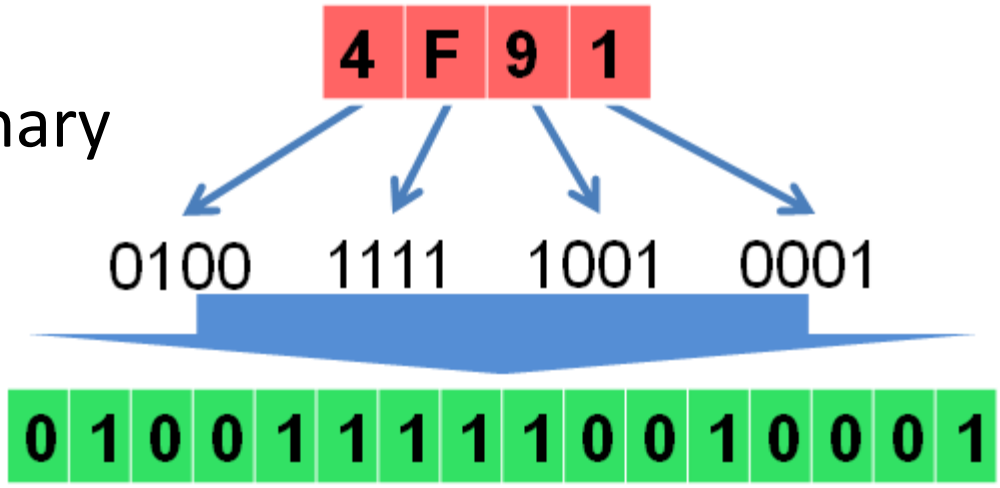
Binary to Octal

- Starting from right convert each 3 bits of binary into its corresponding octal value
- If there are less than 3 bits remaining in the end then append zeros to complete the group of 3
- Example
  - Convert 11 111 110 001 010 to Octal



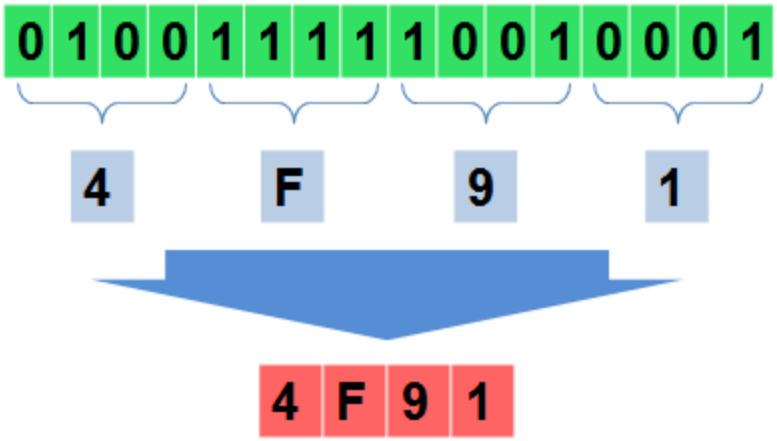
Hexadecimal to Binary

- Each hexadecimal integer corresponds to 4 binary bits
- Convert each hexadecimal number to corresponding binary number
- Example
  - Convert 4F91 to binary



Binary to Hexadecimal

- Starting from right convert each 4 bits of binary into its corresponding hexadecimal
- If there are less than 4 bits remaining in the end then append zeros to complete the group of 4
- Example
  - Convert 0100 1111 1001 0001 to hexadecimal



Hexadecimal to Decimal

- Multiply each hexadecimal digit with its corresponding power of 16

$$\text{Dec} = (D_{n-1} \times 16^{n-1}) + (D_{n-2} \times 16^{n-2}) + \ldots + (D_1 \times 16^1) + (D_0 \times 16^0)$$

  - D = hexadecimal digit
  - n = digit position number in hexadecimal number
- Example
  - Convert 3BA4 to decimal
  - Convert 23AB to decimal

EXAMPLE 1.2

Convert decimal 153 to octal. The required base r is 8. First, 153 is divided by 8 to give an integer quotient of 19 and a remainder of 1. Then 19 is divided by 8 to give an integer quotient of 2 and a remainder of 3. Finally, 2 is divided by 8 to give a quotient of 0 and a remainder of 2. This process can be conveniently manipulated as follows:

153	
19	1
2	3
0	$2 = (231)_8$

Decimal to Hexadecimal

- Repeatedly divide the decimal integer by 16 until last quotient is 0
- Each remainder is a hex digit
- First remainder goes at least significant position and last remainder goes at most significant position
- Example
  - Convert 140 to Hexadecimal
  - Convert 396 to Hexadecimal

Table 1.2
Numbers with Different Bases

Decimal (base 10)	Binary (base 2)	Octal (base 8)	Hexadecimal (base 16)
00	0000	00	0
01	0001	01	1
02	0010	02	2
03	0011	03	3
04	0100	04	4
05	0101	05	5
06	0110	06	6
07	0111	07	7
08	1000	10	8
09	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Table 1.1
Powers of Two

n	2^n	n	2^n	n	2^n
0	1	8	256	16	65,536
1	2	9	512	17	131,072
2	4	10	1,024	18	262,144
3	8	11	2,048	19	524,288
4	16	12	4,096	20	1,048,576
5	32	13	8,192	21	2,097,152
6	64	14	16,384	22	4,194,304
7	128	15	32,768	23	8,388,608

Converting Fractions to Decimal

- In any number system the base is also called the radix, and is represented by r
- We can convert the fractions from any base to decimal by using the following relation

$$\textbf{Dec} = (a_n \times r^n + a_{n-1} \times r^{n-1} + \ldots + a_1 \times r^1 + a_0 \times r^0) \,.\, (a_{-1} \times r^{-1} + a_{-2} \times r^{-2} + \ldots.)$$

Example

Convert the following numbers to equivalent decimal numbers

i. $(11010.11)_2$
ii. $(4021.2)_5$
iii. $(127.4)_8$

The coefficients a_j range in value from 0 to $r - 1$. To distinguish between numbers of different bases, we enclose the coefficients in parentheses and write a subscript equal to the base used (except sometimes for decimal numbers, where the content makes it obvious that the base is decimal). An example of a base-5 number is

$$(4021.2)_5 = 4 \times 5^3 + 0 \times 5^2 + 2 \times 5^1 + 1 \times 5^0 + 2 \times 5^{-1} = (511.4)_{10}$$

The coefficient values for base 5 can be only 0, 1, 2, 3, and 4. The octal number system is a base-8 system that has eight digits: 0, 1, 2, 3, 4, 5, 6, 7. An example of an octal number is 127.4. To determine its equivalent decimal value, we expand the number in a power series with a base of 8:

$$(127.4)_8 = 1 \times 8^2 + 2 \times 8^1 + 7 \times 8^0 + 4 \times 8^{-1} = (87.5)_{10}$$

Note that the digits 8 and 9 cannot appear in an octal number.

It is customary to borrow the needed r digits for the coefficients from the decimal system when the base of the number is less than 10. The letters of the alphabet are used to supplement the 10 decimal digits when the base of the number is greater than 10. For example, in the *hexadecimal* (base-16) number system, the first 10 digits are borrowed from the decimal system. The letters A, B, C, D, E, and F are used for the digits 10, 11, 12, 13, 14, and 15, respectively. An example of a hexadecimal number is

$$(B65F)_{16} = 11 \times 16^3 + 6 \times 16^2 + 5 \times 16^1 + 15 \times 16^0 = (46,687)_{10}$$

Converting Fractions from Decimal to Other Bases

- Multiply the fractional part by radix (r), the integer part will be the first digit
- Take the fractional part of the immediate product and multiply it by r again and note the next digit
- Continue this process until the fractional part of the subsequent product is 0 or starts to repeat itself
- For example lets convert 0.5625 to binary
  - The answer is $(0.1001)_2$
- If there is an integer part then convert it separately by using the division technique

EXAMPLE 1.3

Convert $(0.6875)_{10}$ to binary. First, 0.6875 is multiplied by 2 to give an integer and a fraction. Then the new fraction is multiplied by 2 to give a new integer and a new fraction. The process is continued until the fraction becomes 0 or until the number of digits have sufficient accuracy. The coefficients of the binary number are obtained from the integers as follows:

	Integer		Fraction	Coefficient
$0.6875 \times 2 =$	1	+	0.3750	$a_{-1} = 1$
$0.3750 \times 2 =$	0	+	0.7500	$a_{-2} = 0$
$0.7500 \times 2 =$	1	+	0.5000	$a_{-3} = 1$
$0.5000 \times 2 =$	1	+	0.0000	$a_{-4} = 1$

Therefore, the answer is $(0.6875)_{10} = (0.a_{-1}a_{-2}a_{-3}a_{-4})_2 = (0.1011)_2$.

To convert a decimal fraction to a number expressed in base r, a similar procedure is used. However, multiplication is by r instead of 2, and the coefficients found from the integers may range in value from 0 to $r - 1$ instead of 0 and 1.

■

EXAMPLE 1.4

Convert $(0.513)_{10}$ to octal.

$$0.513 \times 8 = 4.104$$

$$0.104 \times 8 = 0.832$$

$$0.832 \times 8 = 6.656$$

$$0.656 \times 8 = 5.248$$

$$0.248 \times 8 = 1.984$$

$$0.984 \times 8 = 7.872$$

The answer, to seven significant figures, is obtained from the integer part of the products:

$$(0.513)_{10} = (0.406517\ldots)_8$$

Arithmetic operations with numbers in base r follow the same rules as for decimal numbers. When a base other than the familiar base 10 is used, one must be careful to use only the r-allowable digits. Examples of addition, subtraction, and multiplication of two binary numbers are as follows:

augend:	101101	minuend:	101101	multiplicand:	1011
addend:	+100111	subtrahend:	−100111	multiplier:	× 101
sum:	1010100	difference:	000110		1011
					0000
					1011
				product:	110111

The sum of two binary numbers is calculated by the same rules as in decimal, except that the digits of the sum in any significant position can be only 0 or 1. Any carry obtained in a given significant position is used by the pair of digits one significant position higher. Subtraction is slightly more complicated. The rules are still the same as in decimal, except that the borrow in a given significant position adds 2 to a minuend digit. (A borrow in the decimal system adds 10 to a minuend digit.) Multiplication is simple: The multiplier digits are always 1 or 0; therefore, the partial products are equal either to the multiplicand or to 0.

Conversion from binary to hexadecimal is similar, except that the binary number is divided into groups of *four* digits:

$$\begin{array}{cccccccc} (10 & 1100 & 0110 & 1011 & \cdot & 1111 & 0010)_2 = (2C6B.F2)_{16} \\ 2 & C & 6 & B & & F & 2 \end{array}$$

The corresponding hexadecimal (or octal) digit for each group of binary digits is easily remembered from the values listed in Table 1.2.

Conversion from octal or hexadecimal to binary is done by reversing the preceding procedure. Each octal digit is converted to its three-digit binary equivalent. Similarly, each hexadecimal digit is converted to its four-digit binary equivalent. The procedure is illustrated in the following examples:

$$\begin{array}{cccccccc} (673.124)_8 = (110 & 111 & 011 & \cdot & 001 & 010 & 100)_2 \\ 6 & 7 & 3 & & 1 & 2 & 4 \end{array}$$

and

$$\begin{array}{cccccc} (306.D)_{16} = (0011 & 0000 & 0110 & \cdot & 1101)_2 \\ 3 & 0 & 6 & & D \end{array}$$

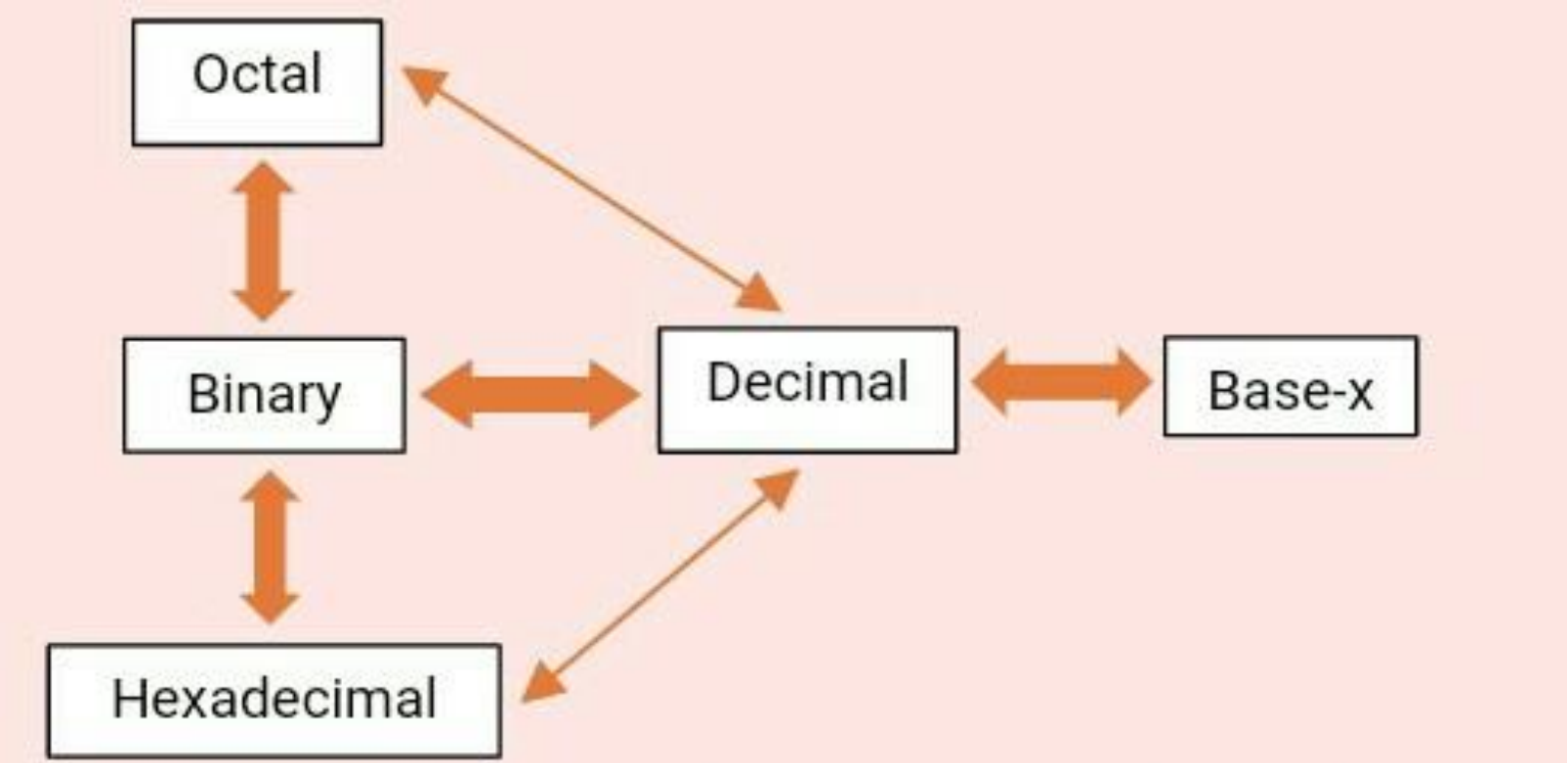
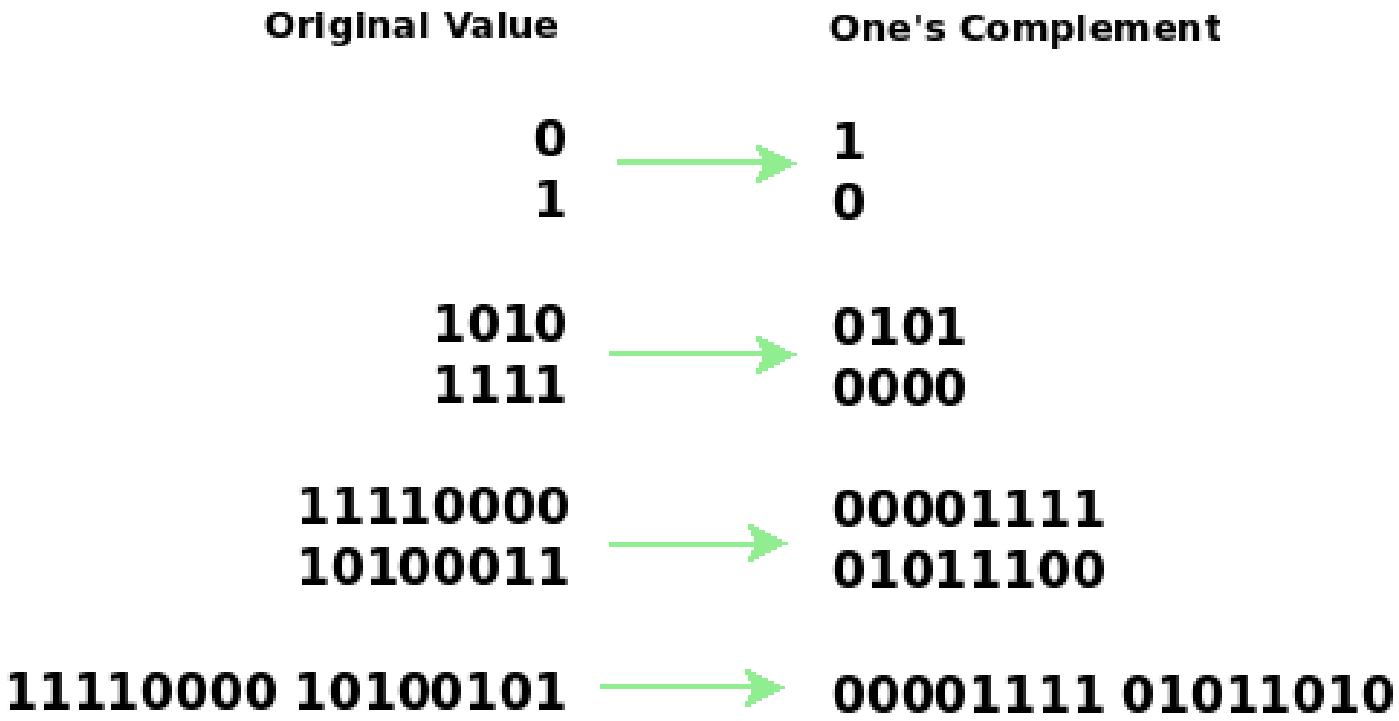


Figure 1. Number systems conversion pathways

One's Complement

Invert all bits. Each 1 becomes a 0, and each 0 becomes a 1.



Binary representation of 5 is: 0 1 0 1

1's Complement of 5 is: 1 0 1 0

2's Complement of 5 is: (1's Complement + 1) i.e.

1 0 1 0 (1's Compliment)

\+ 1

1 0 1 1 (2's Complement i.e. -5)

Two's Complement Addition

$27 - 6 = 27 + (-6)$

$27 = 16 + 8 + 2 + 1 = 2^4 + 2^3 + 2^1 + 2^0 = 011011$

$6 = 4 + 2 = 2^2 + 2^1 = 000110$

-6 $\quad 111001 + 1 = 111010$

```
  1 1 1
  011011
  111010
--------
X010101
```

$= 2^4 + 2^2 + 2^0 = 16 +$

Two's Compliment of Addition

- Two's complement addition follows the same rules as **binary addition**.
- *For example,*

$5 + (-3) = 2$

```
  0000 0101 = +5
+ 1111 1101 = -3
-----------
  0000 0010 = +2
```

Practice Problem 1

- Convert the following numbers to decimal
  i. $(123)_4$
  ii. $(32401)_5$
  iii. $(1100101)_2$
- Convert the following decimals numbers to indicated bases
  i. $(1254)_{10}$ to base 3
  ii. $(4096)_{10}$ to base 16
  iii. $(0362)_{10}$ to base 8
- Convert $(2376)_8$ to hexadecimal equivalent

Practice Problem 2

- Carry out the following conversions
  i. 0.513 to Octal
  ii. 41.6875 to binary
  iii. $(10010.100101)_2$ to decimal
  iv. 153.125 to octal
  v. $(10110001101011 \ . \ 11110000011)_2$ to Octal
  vi. $(306.D)_{16}$ to binary
  vii. $(673.124)_8$ to binary
  viii. $(10110001101011 \ . \ 1111001)_2$ to Hexadecimal