CM1604
Computer Systems Fundamentals

Number Systems & Binary Operation

Week No 01| Rathesan Sivagnanalingam

ROBERT GORDON UNIVERSITY ABERDEEN
INFORMATICS INSTITUTE OF TECHNOLOGY

In this week lecture..

- Categories of Numbers
- Positional Number System
- Conversion of numbers from other bases to base 10 and vice versa
- Ranges of values (Positive Integers)
- Binary Operations
- Shift Left & Shift Right

By the end of this lecture, you will be able to:

- Distinguish among categories of numbers
- Describe positional notation
- Convert numbers from other bases to base 10 and vice versa
- Work out the range of values of Positive (Unsigned) Integers of different bases
- Perform Binary Operations
- Understand primitive arithmetic functions of the CPU

Numbers

- **Natural Numbers**
  Zero and any numbers obtained by repeatedly adding 1 to it
  Eg: 45875, 0, 1254, 12
- **Negative numbers**
  A value less than 0, with a '-'sign
  Eg: -4581, -45, -1, -8
- **Integers**
  Either a natural number or a negative number
  Eg: 4587, 5, 0, -4, -4543
- **Rational Number**
  An integer or a quotient of two integers
  Eg: 458, 0, -754, 8/25, -2/5

Positional Notation

495

$$= 4 * 10^2 + 9 * 10^1 + 5 * 10^0$$

$$= 4 * 100 + 9 * 10 + 5 * 1$$

$$= 400 + 90 + 5$$

Power indicates the position

Different Bases

Decimal

Base 10, has 10 different digit symbol

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Binary

Base 2 has 2 different digit symbol

0, 1

Different Bases

Octal

Base 8, has 8 different digit symbol

0, 1, 2, 3, 4, 5, 6, 7

Hexadecimal

base 16 has 16 different digit symbol

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

Converting Binary to Decimal

- What is the decimal equivalent of 1101010_2

	$1*2^6$	$= 1*\ 64$	=	64
+	$1*2^5$	$= 1*\ 32$	=	32
+	$0*2^4$	$= 0*\ 16$	=	0
+	$1*2^3$	$= 1*\ 8$	=	8
+	$0*2^2$	$= 0*\ 4$	=	0
+	$1*2^1$	$= 1*\ 2$	=	2
+	$0*2^6$	$= 0*\ 1$	=	0

= 106 in base 10

Converting Hexadecimal to Decimal

- What is the decimal equivalent of ABC_{16}

	$A*16^2$	= 10* 256	=	2560
+	$B*16^1$	= 11* 16	=	176
+	$C*16^0$	= 12* 1	=	12
		=		2748 in base 10

Converting Octal to Decimal

- What is the decimal equivalent of 367_8

	$3*8^2$	**$= 3*64$**	**=**	**192**
+	**$6*8^1$**	**$= 6*8$**	**=**	**48**
+	**$7*8^0$**	**$= 7*1$**	**=**	**7**
			=	**247 in base 10**

Converting Decimal to Binary

43 / 2	**=**	**21 remainder 1**
21 / 2	**=**	**10 remainder 1**
10 / 2	**=**	**5 remainder 0**
5 / 2	**=**	**2 remainder 1**
2 / 2	**=**	**1 remainder 0**
1 / 2	**=**	**0 remainder 1**

1

0 1 0 101 1_2

Converting Decimal to Hexadecimal

298 / 16 = 18 remainder 10

18/ 16 = 1 remainder 2

2 A $_{16}$

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

Converting Binary to Hexadecimal

- **Separate the number in to group of 4 (from right)**

- **Convert each group individually**

11101011_2 → 1110 1011

	1110	**1011**
	------	**-----**
	14	**11**

EB_{16}

Ranges of values Decimal (Positive Integers)

Decimal

1 digit $0 \rightarrow 9$ $(10^1 - 1)$ 10^1 values

2 digit $0 \rightarrow 99$ $(10^2 - 1)$ 10^2 values

n digit $0 \rightarrow 10^n$ values

Ranges of values Binary (Positive Integers)

Binary

1 digit	0 → 1	$(2^1 - 1)$	2^1 values
2 digit	0 → 11	$(2^2 - 1)$	2^2 values2

n digit 0 → 2^n values

Ranges of values Hexadecimal (Positive Integers).....

Hexadecimal

1 digit	0 → F	$(16^1 - 1)$	16^1 values
2 digit	0 → FF	$(16^2 - 1)$	16^2 values

n digit 0 → 16^n values

Why use Binary?

- Computer contain only 2 states
  - low-voltage
  - high-voltage

Why use Hexadecimal?

- More efficient to store large numbers
- Quick conversion between binary

Binary Operation

Binary Operations

NOT Operation

- NOT(0) = 1
- NOT(1) = 0

Input 1	**0**	**1**
Input 2	**1**	**0**

Binary Operations ...

OR Operation

Input 1	0	0	1	1
Input 2	0	1	0	1
Output	0	1	1	1

Binary Operations …

AND Operation

Input 1	0	0	1	1
Input 2	0	1	0	1
Output	0	0	0	1

Binary Operations …

Binary Addition

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

Shift Left Operation

Shift Left

- Shifting binary value one position to left
- Multiplying by 2

Shift Left

binary					
			1	1	3
		1	1	0	6
	1	1	0	0	12
1	1	0	0	0	24

Shift Right Operation

Shift Right

- Shifting binary value one position to right
- Dividing by 2

Shift Right

binary						
1	0	1	0	0		20
	1	0	1	0		10
		1	0	1		5
			1	0	1	2

Primative arithmatic functions of CPU

- **Addition**
- **Subtraction**
- **Multiplication (Shift Left)**
- **Division (Shift Right)**

Composite arithmatic

- **Multiplication by 5**
  - **Multiply by 2 (Shift Left)**
  - **Multiply by 2 (Shift Left)**
  - **Add original number**

REFERENCE

- Dale, N.B. and Lewis, J., 2007. Computer science illuminated. Jones & Bartlett Learning.

READING

Chapter # 2, 4

- Computer science illuminated. Jones & Bartlett Learning.