Lect 03
Number System, Gates, Boolean Algebra

CS221: Digital Design

Dr. A. Sahu
Dept of Comp. Sc. & Engg.
Indian Institute of Technology Guwahati

Outline

- Number System
  - Decimal, Binary, Octal, Hex
  - Conversions
  - Operations: Add, Sub, Mul, Div, Complement
- Gates in Digital System
  - Basic Gates (AND, OR & NOT)
  - Universal Gates (NAND & NOR)
  - Others : XOR, XNOR
- Boolean Algebra
  - Axioms
- Boolean Functions

Number System

- Number System
  - Decimal, Binary, Octal, Hex
- Conversion (one to another)
  - Decimal to Binary, Octal, Hex & Vice Versa
  - Binary to HEX & vice versa
- Other representation
  - Signed, Unsigned, Complement
- Operation
  - Add, Sub, Mul, Div, Mod

What Digit? => Number System

- Famous Number System: Decimal, Binary, Octal, Hexa-decimal
- Decimal System: 0 -9
  - May evolves: because human have 10 finger
- Binary System, Others (Oct, Hex)

Significant Digits

Binary: 11101101

Most significant digit *Least significant digit*

Decimal: 1063079

Most significant digit *Least significant digit*

Decimal (base 10)

- Uses positional representation
- Each digit corresponds to a power of 10 based on its position in the number
- The powers of 10 increment from 0, 1, 2, etc. as you move right to left
  - $1,479 = 1 * 10^3 + 4 * 10^2 + 7 * 10^1 + 9 * 10^0$

Binary (base 2)

- Two digits: 0, 1
- To make the binary numbers more readable, the digits are often put in groups of 4
  - 1010 = $1 * 2^3 + 0 * 2^2 + 1 * 2^1 + 0 * 2^0$
    = 8 + 2
    = 10
  - 1100 1001 = $1*2^7 + 1*2^6 + 1*2^3 + 1*2^0$
    = 128 + 64 + 8 + 1
    = 201

How to Encode Numbers: Binary Numbers

- Working with binary numbers
  - In base ten, helps to know powers of 10
    - one, ten, hundred, thousand, ten thousand, ...
  - In base two, helps to know powers of 2
    - one, two, four, eight, sixteen, thirty two, sixty four, one hundred twenty eight
    - Count up by powers of two

2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
512	256	128	64	32	16	8	4	2	1

Octal (base 8)

- Shorter & easier to read than binary
- 8 digits: 0, 1, 2, 3, 4, 5, 6, 7,
- Octal numbers

$$136_8 = 1*8^2 + 3*8^1 + 6*8^0$$
$$= 1*64 + 3*8 + 6*1$$
$$= 94_{10}$$

Hexadecimal (base 16)

- Shorter & easier to read than binary
- 16 digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
- "0x" often precedes hexadecimal numbers

$$0x123 = 1*16^2 + 2*16^1 + 3*16^0$$
$$= 1*256 + 2*16 + 3*1$$
$$= 256 + 32 + 3$$
$$= 291$$

Counting

Decimal	Binary	Octal	Hexadecimal
0	00000	0	0
1	00001	1	1
2	00010	2	2
3	00011	3	3
4	00100	4	4
5	00101	5	5
6	00110	6	6
7	00111	7	7
8	01000	10	8

11

Counting

Decimal	Binary	Octal	Hexadecimal
9	01001	11	9
10	01010	12	A
11	01011	13	B
12	01100	14	C
13	01101	15	D
14	01110	16	E
15	01111	17	F
16	10000	20	10

12

Fractional Number

- Point: Decimal Point, Binary Point, Hexadecimal point
- Decimal

  $247.75 = 2\times10^{2}+4\times10^{1}+7\times10^{0}+7\times10^{-1}+5\times10^{-2}$

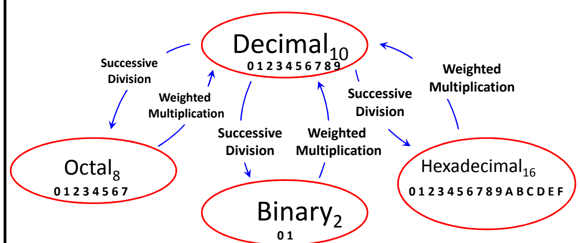
Fractional Number

- Point: Decimal Point, Binary Point, Hexadecimal point
- Binary

  $10.101 = 1\times2^{1}+0\times2^{0}+1\times2^{-1}+0\times2^{-2}+1\times2^{-3}$
- Hexadecimal

  $6A.7D = 6\times16^{1}+10\times16^{0}+7\times16^{-1}+D\times16^{-2}$

Converting To and From Decimal



15

Decimal ↔ Binary

a) Divide the decimal number by **2**; the remainder is the LSB of the **binary** number.

b) If the quotation is zero, the conversion is complete. Otherwise repeat step (a) using the quotation as the decimal number. The new remainder is the next most significant bit of the **binary** number.

Decimal ↔ Binary

a) Multiply each bit of the **binary** number by its corresponding bit-weighting factor (i.e., Bit-0→2^{0}=1; Bit-1→2^{1}=2; Bit-2→2^{2}=4; etc).

b) Sum up all of the products in step (a) to get the decimal number.

Decimal to Binary : Subtraction Method

- Goal
  - Good for human
  - Get the binary weights to add up to the decimal quantity
    - Work from left to right
    - (Right to left – may fill in 1s that shouldn't have been there – try it).

Desired decimal number: **12**

32	16	8	4	2	1	
1						=32 too much
0	1					=16 too much
0	0	1				=8 ok, keep going
0	0	1	1			=8+4=12 DONE
0	**0**	**1**	**1**	**0**	**0**	answer

Decimal to Binary : Division Method

- Good for computer: Divide decimal number by 2 and insert remainder into new binary number.
  - Continue dividing quotient by 2 until the quotient is 0.
- Example: Convert decimal number 12 to binary

12 div 2 = (Quo=6 , Rem=0) LSB
6 div 2 = (Quo=3, Rem=0)
3 div 2 = (Quo=1,Rem=1)
1 div 2 = (Quo=0, Rem=1) MSB

12_{10}= 1 1 00 $_2$

Conversion Process Decimal ↔ $Base_N$

$Base_{10}$ DECIMAL → Successive Division → $Base_N$ ANY BASE

a) Divide the decimal number by **N**; the remainder is the LSB of the **ANY BASE** Number .

b) If the quotient is zero, the conversion is complete. Otherwise repeat step (a) using the quotient as the decimal number. The new remainder is the next most significant bit of the **ANY BASE** number.

$Base_N$ ANY BASE → Weighted Multiplication → $Base_{10}$ DECIMAL

a) Multiply each bit of the **ANY BASE** number by its corresponding bit-weighting factor (i.e., Bit-0→N^0; Bit-1→N^1; Bit-2→N^2; etc).

b) Sum up all of the products in step (a) to get the decimal number.

Decimal ↔ Octal Conversion

The Process: Successive Division

- Divide number by **8**; R is the LSB of the **octal** number
- While Q is 0
  - Using the Q as the decimal number.
  - New remainder is MSB of the **octal** number.

8) 94 → 11   r = 6 ← LSB
8) 11 → 1   r = 3
8) 1 → 0   r = 1 ← MSB

$94_{10} = 136_8$

21

Decimal ↔ Hexadecimal Conversion

The Process: Successive Division

- Divide number by **16**; R is the LSB of the **hex** number
- While Q is 0
  - Using the Q as the decimal number.
  - New remainder is MSB of the **hex** number.

16) 94 → 5   r = E ← LSB
16) 5 → 0   r = 5 ← MSB

$94_{10} = 5E_{16}$

Example: Hex → Octal

Example:

Convert the hexadecimal number $5A_H$ into its octal equivalent.

Solution:

First convert the hexadecimal number into its decimal equivalent, then convert the decimal number into its octal equivalent.

5	A
16_1	16_0
16	1

80 + 10 = 90_{10}

8) 90 → 11   r = 2 ← LSB
8) 11 → 1   r = 3
8) 1 → 0   r = 1 ← MSB

∴ $5A_H = 132_8$

23

Example: Octal → Binary

Example:

Convert the octal number 132_8 into its binary equivalent.

Solution:

First convert the octal number into its decimal equivalent, then convert the decimal number into its binary equivalent.

1	3	2
8^2	8^1	8^0
64	8	1

64 + 24 + 2 = 90_{10}

2) 90 → 45   r = 0 ← LSB
2) 45 → 22   r = 1
2) 22 → 11   r = 0
2) 11 → 5   r = 1
2) 5 → 2   r = 1
2) 2 → 1   r = 0
2) 1 → 0   r = 1 ← MSB

$132_8 = 1011010_2$

24

Binary ↔ Octal ↔ Hex Shortcut

- Relation
  - Binary, octal, and hex number systems
  - All powers of two
- Exploit (This Relation)
  - Make conversion easier.

Substitution Code

Convert $010101101010111001101010_2$ to hex using the 4-bit substitution code :

5	6	A	E	6	A
0101	0110	1010	1110	0110	1010

$\boxed{56AE6A_{16}}$

Substitution Code

Substitution code can also be used to convert binary to octal by using 3-bit groupings:

2	5	5	2	7	1	5	2
010	101	101	010	111	001	101	010

$\boxed{25527152_8}$

Conversion of Fractional Number

- Convert 163.875_{10} to binary
- Integer part

  163/2=>81/2=>40/2=>20/2=>10/2=>5/2=>2/2=>1

  LSB 1 1 0 0 0 1 0 1

  Integer part: 1010001
- Fractional part
  - 0.875*2=1.750 => 0.75*2=1.50 => 0.5*2=1.0
  - Fractional part : .111
- Total : 10100011.111

Conversion of Fractional Number

- Convert 0.15_{10} to binary
- Fractional part
  - 0.15*2=0.30 => 0
  - 0.30*2=0.60 => 0
  - 0.6*2 = 1.2 => 1
  - 0.2*2 = 0.4 => 0
  - 0.4*2 = 0.8 => 0
  - 0.8*2 =1.6 => 1
  - **0.6*2 =>.... Repeating sequences......**
- Total : 0.001001....

Other Representation

- Signed & Unsigned Number
- Signed number last bit (one MSB) is signed bit

  Assume: 8 bit number

  Unsigned 12 : 0000 1100

  Signed +12 : 0000 1100

  Signed -12 : 1000 1100
- Complement number

  Unsigned binary 12 = 00001100

  1's Complement of 12 = 1111 0011

Operations on Numbers

- Addition
- Substraction
- Multiplication
- Division

Binary Addition

- One bit

0	+	0 =	0
1	+	0 =	1
0	+	1 =	1
1	+	1 =	0 1 (Carry bit)

- Multibit (consider carry)

		1	1	0	1
	+	1	0	0	1
=	1	0	**1**	1	-

Binary Subtraction

- One bit

0	-	0 =	0	
1	-	0 =	1	
0	-	1 =	1	1 (Carry bit)
1	-	1 =	0	

- Multibit (consider carry)

		1	1	1	0
	-	1	0	0	1
=		0	**1**	0	1

Subtraction

- 9's complement and 10's complement
- Suppose Decimal two digit System
  - Max number 99, Min =0
- Substation :

  55-33= 22

  [55 + (100-33)] %100= [55+67]%100= [122]%100=22
- Tow 's complement in Dec

  99-33+1 == 100-33

Binary Subtraction

- Multibit (consider carry)

	1	1	1	0
-	1	0	0	1
=	0	**1**	0	1

- Add 2's complement= (1001)'+1= 0110+1=0111
- Other way (add 2's complement & discard carry)

	1	1	1	0
+	**0**	**1**	**1**	1
= ~~1~~	0	**1**	0	1

2's complement examples

- Express -45 in 8 bit 2's complement form

- +45 in 8 bit form 0010 1101
- Complement it = 1101 0010
- Add 1 to it = +1

2's complement = 1101 0011

Binary Multiplication

- Repeated addition
- Many improved technique (Famous Partial Sum)

			1	0	0	0	$=8_{10}$	
			X	0	1	1	1	$=7_{10}$
=			+	1	0	0	0	
		+	1	0	0	0		
	+	1	0	0	0			
+ 0	0	0	0					
0	1	1	1	0	0	0	=	56_{10}

37

Binary Multiplication

- Repeated addition
- Many improved technique [7 is 8-1]

			1	0	0	0	$=8_{10}$	
		X	0	1	1	1	$=7_{10}$	
=		+	1	0	0	0		
	+	1	0	0	0			
+ 1	0	0	0					
+ 0	0	0	0					
0	1	1	1	0	0	0	=	56_{10}

38

Binary Multiplication

- Repeated addition
- Many improved technique [7 is 8-1]

			1	0	0	0	$=8_{10}$	
		X	+1	0	0	-1	$=7_{10}$	
=		-	1	0	0	0		
	+	0	0	0	0			
+ 0	0	0	0					
+ 1	0	0	0					
0	1	1	1	0	0	0	=	56_{10}

39

Binary Multiplication

- Booth Methods

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

+1 0 -1 0 +1 0 0 0 -1 +1 0 0 0 -1 0 0

Binary Division & Modulus

- 0110010 ÷ 011 $= 50_{10} \div 3_{10}$

```
011 ) 0 1 1 0 0 1 0 (
      0 1 1        ( 1
      0 0 0
        0 0 0     ( 0
        0 0 0
          0 0 0     ( 0
          0 0 0
            0 0 1
            0 0 0   ( 0          Q=1000=16_10
              0 1 0                R=10= 2_10
              0 0 0   ( 0
```

41

Hex Addition

- Addition

Carry Value to higher significant one is 1

	+1		+1	
	1	A	2	B
+	7	C	A	6
=	9	6	D	1

- Subtraction

	-1	$+16_{10}$	
A	2	B	9
1	C	F	3
8	5	C	6

42

Boolean Algebra

Boolean Algebra

- Computer hardware using binary circuit greatly simply design
- George Boole (1813-1864): developed a mathematical structure in **1847**
  - To deal with binary operations with just two values
- Binary circuits: To have a conceptual framework to manipulate the circuits algebraically
  - **Claude Shannon : 1937, Master Thesis**

Basic Gates in Binary Circuit

- Element 0 : “FALSE”. Element 1 : “TRUE”.
- ‘+’ operation “OR”,‘*’ operation “AND” and ’ operation “NOT”.

OR	0	1
0	0	1
1	1	1

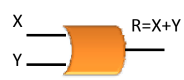
AND	0	1
0	0	0
1	0	1

NOT	
0	1
1	0

45

OR Gate

- ‘+’ operation “OR”



OR	*0*	*1*
0	*0*	*1*
1	*1*	*1*

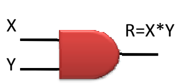
X	Y	R=X OR Y R= X + Y
0	0	0
0	1	1
1	**0**	**1**
1	1	1

1 + Y = 1

46

AND Gate

- ‘*’ operation “AND”



AND	0	1
0	0	0
1	0	1

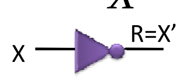
X	Y	R=X AND Y R= X * Y
0	**0**	**0**
0	**1**	**0**
1	0	0
1	1	1

0 * Y = 0

47

NOT Gate

- ‘ operation “NOT” or use BAR
- R= $\overline{X}$



X	R=X’ R= NOT X
0	1
1	0

48

Boolean Algebra Defined

- Boolean Algebra B : 5-tuple

  {B, +, *, ’, 0, 1}
- **+** and ***** are *binary* operators,
- **‘** is a *unary* operator.

Boolean Algebra Defined

- ***Axiom #1: Closure***

  If a and b are Boolean

  (a + b) and (a * b) are Boolean.
- ***Axiom #2: Cardinality/Inverse***

  if a is Boolean then a’ is Boolean
- ***Axiom #3: Commutative***

  (a + b) = (b + a)

  (a * b) = (b * a)

Boolean Algebra Defined

•***Axiom #4: Associative :*** **If a and b are Boolean**

(a + b) + c = a + (b + c)

(a * b) * c = a * (b * c)

•***Axiom #6: Distributive***

a * (b + c) = (a * b) + (a * c)

a + (b * c) = (a + b) * (a + c)

2nd one is Not True for Decimal numbers System
5+(2*3) ≠ (5+2)*(5+3)
11 ≠ 56

Thanks

52