

EE213M

Digital Circuits

Arun Tej M.



EE213M Digital Circuits

L2: Number Systems - 2

Decimal to Binary



Decimal to Binary – Take powers of 2 from highest to lowest possible values, subtract each

$$(625)_{10} = (??)_2$$

How many bits are required?

 $4 \to 0 \text{ to } 15$ $8 \to 0 \text{ to } 255$

n bits can represent values from 0 to $(2^n - 1)$

 $9 \to 0 \text{ to } 511$ $10 \to 0 \text{ to } 1023$

ي 512	2 ⁸ 256	128	64	32	2 ⁴ 16	8	2 ² 4	2 2	2° 1	
1	0	0	1	1	1	0	0	0	1	

$$625 - 512 = 113$$

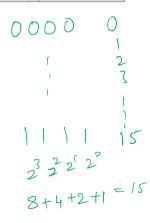
$$113 - 64 = 49$$

$$49 - 32 = 17$$

$$17 - 16 = 1$$

$$1 - 1 = 0$$

✓ 1001110001





Another example

$$(167)_{10} = (??)_2$$

128	64	32	16	8	4	2	1
1	0	1	0	0	1	1	1

$$167 - 128 = 39$$

$$39 - 32 = 7$$

$$7 - 4 = 3$$

$$3 - 2 = 1$$

$$1 - 1 = 0$$





- Base or Radix = $8 = 2^3$
- Every 3 bit binary number is represented by one octal number

$$2^{2} 2^{1} 2^{0}$$

$$(101)_{2} = 1x2^{2} + 0x2^{1} + 1x2^{0} = 5$$

Binary to Octal Conversion:

For the integer part, starting from LSB to MSB, make groups of 3 bits each (add leading zeros if necessary)

For the fraction part, starting from MSB to LSB, make groups of 3 bits each (add trailing zeros if necessary)

Binary
000
001
010
011
100
(101)
110
111

Octal System



$$\frac{(1111011001)_{2}}{\text{Grouping Direction}} = \frac{001}{1} \qquad \frac{111}{7} \qquad \frac{011}{3} \qquad \frac{001}{1} = (1731)_{8}$$

Octal to Binary Conversion: Represent each digit by its 3-bit binary value.

$$(721)_8 = (111 \ 010 \ 001)_2$$
 $(61.42)_8 = (110 \ 001 \ . \ 100 \ 010)_2$



Hexadecimal System

- Base or Radix = $16 = 2^4$
- Every 4 bit binary number is represented by one hexadecimal number

Conversion to and from binary:

Similar to octal \leftrightarrow binary, but with groups of 4 bits.

 $(0010\ 1100\ 0110\ 1011\ .\ 1111\ 0000\ 0110)_2$

 $(2 \quad C \quad 6 \quad B \quad F \quad 0 \quad 6)_{16}$

Hexa	Binary	Hexa	Binary
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	В	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111

THE STATE OF THE S

Some more examples

Binary \leftrightarrow Octal/Hex – Two approaches

- a. Binary \leftrightarrow Decimal \leftrightarrow Octal/Hex
- b. Partition the binary number into groups of three/four bits each, starting from the binary point, proceed to the left and to the right

$$ijkl$$
 \overrightarrow{l} \overrightarrow{l} \overrightarrow{m} \overrightarrow{n} \overrightarrow{l}

$$100110011_2 = 100 \ 110 \ 011 = 463_8$$

$$11100110010_2 = 11 \ 100 \ 110 \ 010 = 3462_8$$

$$000100110011_2 = 1 \quad 0011 \quad 0011 = 133_{16}$$

$$11100110010_2 = 111 \ 0011 \ 0010 = 732_{16}$$

 $1100110011.011101101_2 = 1$ 100 110 011 .011 101 101 = 1463.355₈

$$1100110011.011101101_2 = 11 0011 0011 .0111 0110 1000 = 333.768_{16}$$

Conversions



Octal to Hex (or vice-versa) – Use binary as intermediate

$$3541_8 = ??_{16}$$

$$011 \ 101 \ 100 \ 001 = ??_{16}$$
 $0111 \ 0110 \ 0001 = 761_{16}$

$$5647.6711_8 = ??_{16}$$

$$1F0A.A8_{16} = ??_{8}$$

0001 1111 0000 1010 . 1010
$$1000 = ??_8$$
0001 111 100 001 010 . 101 010 $00 = 17412.52_8$



Binary

Digital circuits – implemented with transistors acting as switches i.e., they have either on or off states Binary - convenient and compatible

$$(625)_{10} = (??)_{2} 625 - 512 = 113 = N_{1} 512 = 2^{9}$$

$$113 - 64 = 49 = N_{2} 64 = 2^{6}$$

$$49 - 32 = 17 = N_{3} 32 = 2^{5}$$

$$17 - 16 = 1 = N_{4} 16 = 2^{4}$$

$$1 - 1 = 0 = N_{5} 1 = 2^{0}$$

No. of bits 4	Term ⁢ Nibble
8	Byte
16/32/64	Word

3 digits
$$\leftarrow$$
 (625)₁₀ = $2^9 + 2^6 + 2^5 + 2^4 + 2^0 = (1001110001)_2 \rightarrow 10$ binary digits (bits)

- Large number of bits are required to represent numbers in binary than in decimal
- Main use of octal and hexadecimal systems: To represent binary numbers in a compact way
- Conversion from binary to octal and hexadecimal is straight forward (unlike from binary to decimal)

Ranges for Unsigned Numbers



For a computer processing 16 –bit unsigned integers?

n bits can represent values from 0 to $(2^n - 1)$

 \checkmark 16 → 0 to 65,535

For a computer processing 16 –bit unsigned fractions?

$$S_n = \left(\frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \dots + \frac{1}{2^{n-1}}\right) + \frac{1}{2^n}$$

$$2S_n = \frac{2}{2} + \frac{2}{4} + \frac{2}{8} + \dots + \frac{2}{2^{n-1}} + \frac{2}{2^n} = 1 + \left(\frac{1}{2} + \frac{1}{4} + \dots + \frac{1}{2^{n-1}}\right) = 1 + \left(S_n - \frac{1}{2^n}\right) \Rightarrow S_n = 1 - \frac{1}{2^n}$$

$$= 1 + \left(S_n - \frac{1}{2^n}\right) \quad \Rightarrow S_n = 1 - \frac{1}{2^n}$$

n bits can represent fraction values from 0 to $(2^n - 1)/2^n$

$$16 \rightarrow 0.0 \text{ to } 65,535/65,536$$

 \checkmark 16 \rightarrow 0.0 to 0.9999847412



Arithmetic Operations

- Arithmetic operations with numbers in base *b* follow the same rules as for decimal numbers.
- However, when a base other than the familiar base 10 is used, one must be careful to use only *b* allowable digits and perform all computations with base *b* digits.
- The sum of two binary numbers is calculated following the same rules as for decimal numbers, except that the sum digit in any position can be only 1 or 0.
- Also, a carry in binary occurs if the sum in any bit position is greater than 1. (A carry in decimal occurs if the sum in any digit position is greater than 9.)
- Any carry obtained in a given position is added to the bits in the column one significant position higher.
- The rules for subtraction are the same as in decimal, except that a borrow into a given column adds 2 to the minuend bit. (A borrow in the decimal system adds 10 to the minuend digit.)

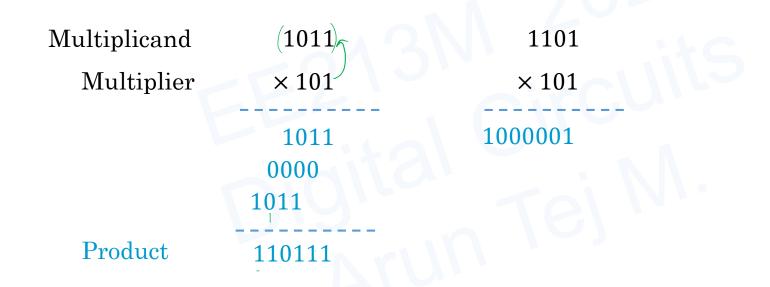


Binary Arithmetic Operations

Carries	00000	01100		
Augend	01100	10110		
Addend	+10001	+10111	4-9	
Sum	11101	101101	Jircon	
Borrows	00000	00110 00110		00110
Minuend	10110	10110	10011	11110 がかい
Subtrahend	-10010	-10011^{2}	−11110 · ✓	-10011
Difference	00100	00011	✓ - 01011	01011











- Arithmetic operations with octal, hexadecimal, or any other base *b* system will normally require the formulation of tables from which one obtains sums and products of two digits in that base.
- An easier alternative for adding two numbers in base *b* is to convert each pair of digits in a column to decimal, add the digits in decimal, and then convert the result to the corresponding sum and carry in the base *b* system.
- Since addition is done in decimal, we can rely on our memories for obtaining the entries from the familiar decimal addition table.
- In general, the multiplication of two base *b* numbers can be accomplished by doing all the arithmetic operations in decimal and converting intermediate results one at a time.





Convert to decimal, perform addition, convert to hexadecimal

$$(59F)_{16} + (E46)_{16}$$

$$5$$

$$9$$

$$15$$

$$14$$

$$4$$

$$6$$

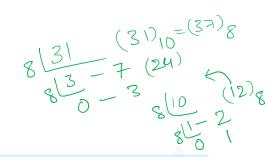
$$19 = 16 + 3$$

$$14$$

$$21 = 16 + 5$$

Instead of adding F + 6 in hexadecimal, we add the equivalent decimals, 15 + 6 = 21. We then convert back to hexadecimal by noting that 21 = 16 + 5. This gives a sum digit of 5 and a carry of 1 to the next higher-order column of digits. The other two columns are added in a similar fashion.

Octal Multiplication





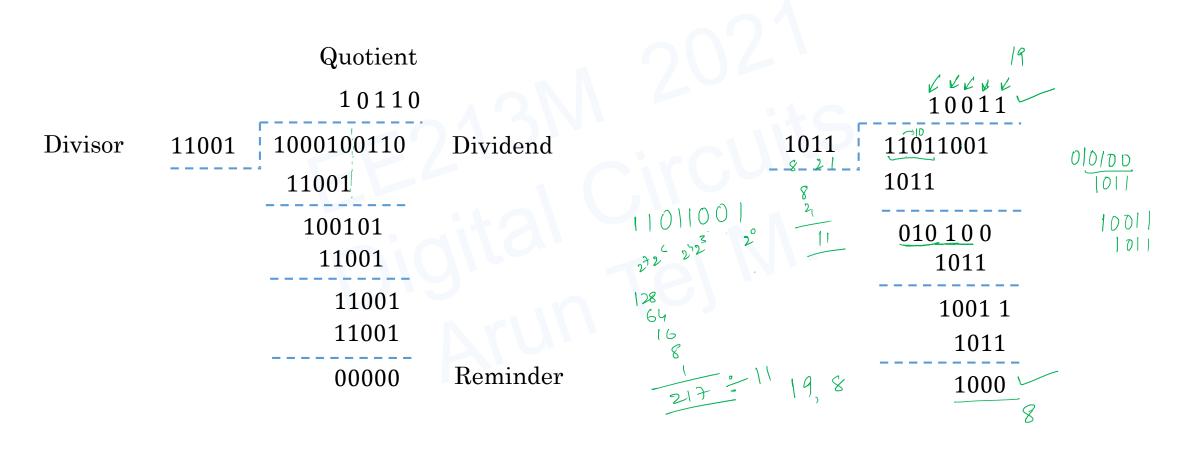
Perform operation in decimal, convert intermediate results one at a time

$(762)_8 \times (45)_8$	762
	45
[→ 14672
	3710
	√ 43 772
<u></u>	8 + 3
→8+3	
13	

Octal	Decimal	Octal
5 × 2	= 10 = 8 + 2	12
5 × 6 + 1	= 31 = 24 + 7	37
$5 \times 7 + 3$	= 38 = 32 + 6	46
4×2	= 8 = 8 + 0	10
4 × 6 + 1	= 25 = 24 + 1	31
$4 \times 7 + 3$	= 31 = 24 + 7	37



Binary Long Division



Convert to decimal, divide, convert back to binary: 217 \div 11 \Rightarrow 19,8 i.e., 10011, 1000





Perform long division for the following:

- i. Divide 1101111011 by 1111
- ii. Divide 1110011001 by 1101
- iii. Divide 1011110101 by 1011
- iv. Divide 1100111001 by 1001

KEVIN PATEL

kpatel@iitg.ac.in