DIGITAL LOGIC DESIGN
Course Code: 328356(28)

Mr. Manjeet Singh Sonwani
Assistant Professor
Department of Electronics & Telecomm.
Government Engineering College Raipur

Course Contents

UNIT-I NUMBER SYSTEMS, CODES AND BOOLEAN ALGEBRA

UNIT-II MINIMIZATIONTECHNIQUES

UNIT-III COMBINATIONAL CIRCUITS

UNIT-IV SEQUENTIAL CIRCUITS

UNIT-V DIGITAL LOGIC FAMILIES

Text Books:

1. Fundamentals of Digital Circuits: A. Anand Kumar, PHI.(Unit – I to V)
2. Digital Electronics-Principles and Integrated Circuits, A.K. Maini, 1st Edition, Wiley India.

Reference Books:

1. Digital Fundamentals: Floyd & Jain: Pearson Education.
2. Digital Electronics: A. P. Malvino: Tata McGraw Hill. 3. Digital Circuits & Logic Design–LEE, PHI.

SCHEME OF TEACHING

Course Code	**328356(28)**
Course Title	Digital Logic Design
Credits	4
Contact Periods (L-T-P)	3-1-0
Course Objective	1.To Design, Analyze and Interpret Combinational Circuits 2. To Design, Analyze and Interpret Sequential Circuits
Course Outcome	**C206.1** Apply the knowledge of number system, various codes and laws of Boolean Algebra in digital electronics. **C206.2** Analyze the different minimization techniques and designing of programmable logic design **C206.3** Design combinational circuits such as adders, subtractors, multiplexers, decoder etc. **C206.4** Design sequential circuits such as counters, finite state machines,etc. **C206.5** Explain the properties and operation of digital logic family and select a suitable logic family for the required application.

UNIT-I NUMBER SYSTEMS, CODES AND BOOLEAN ALGEBRA

- Introduction
- Number Systems:Decimal,Binary,Octal,Hexadecimal
- Representation of Signed Numbers and Binary Arithmetic in Computers.

Introduction

Digital Electronics : Digital Electronics or digital circuits are the branch of Electronics that operates on digital signal.

Analog Signal : It gives its value at any instant of time.

Digital Signal : A signal that is sampled at discrete points and the sampled values are quantized into a set of discrete values.

At a given time it can only take on one of a finite no. Of values.

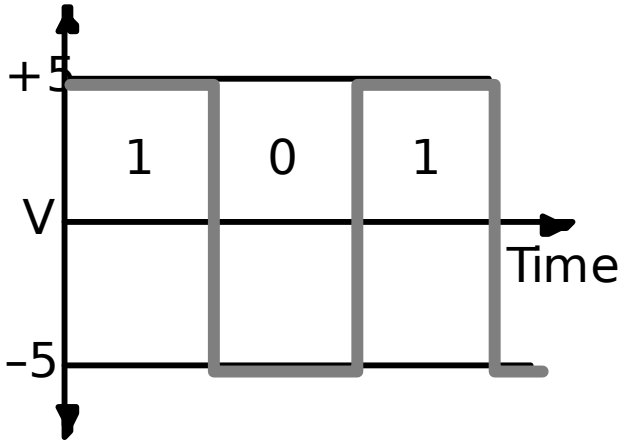
Discrete Signal : It gives its value at a particular instant of time.

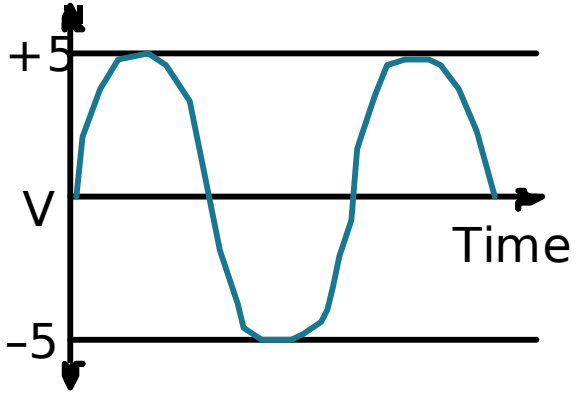
Digital Computers: Imply that the computer deals with digital information, i.e., it deals with the information that is represented by binary digits

- Why BINARY ? instead of Decimal or other number system ?

Digital Systems

Digital vs. Analog Waveforms





Digital:
only assumes discrete values

Analog:
values vary over a broad range continuously

Introduction : Number Systems

Number Systems : In Digital Electronics , the number system is used for representing the information.

The NS has different bases and the most common of them are decimal ,binary,octal,Hexadecimal .

The base and radix of any NS is the total no. of digit used in that NS.

Example: Four number system

Decimal (10): 0,1,2,3,4,5,6,7,8,9

Binary (2) : 0,1

Octal (8) : 0,1,2,3,4,5,6,7

Hexadecimal (16) : 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

Application: (i)To communicate (ii) To perform task (iii) To quantify (iv) To measure

Characteristics of Numbering Systems

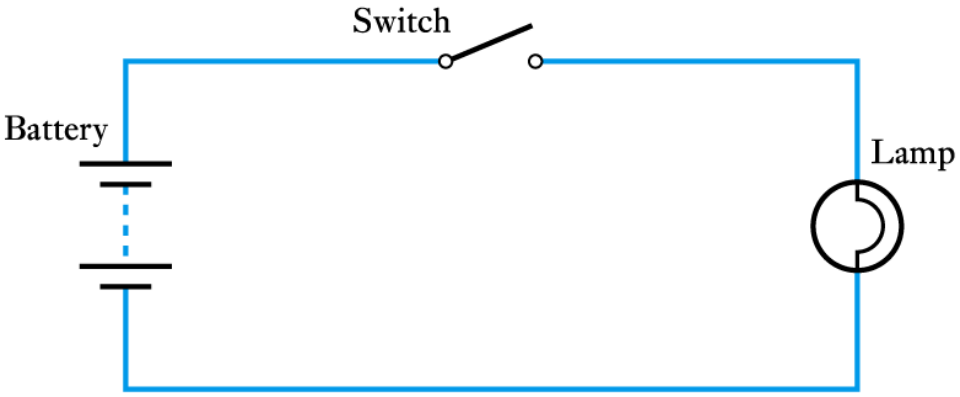
1) The digits are consecutive.
2) The number of digits is equal to the size of the base.
3) Zero is always the first digit.
4) The base number is never a digit.
5) When 1 is added to the largest digit, a sum of zero and a carry of one results.
6) Numeric values are determined by the implicit positional values of the digits.

Binary numbers?

- Computers work only on two states
  - On
  - Off
- Basic memory elements(transistor/MOSFET) hold only two states
  - Zero / One
- Thus a number system with two elements {0,1}
- A binary digit – bit !

Binary Quantities and Variables

- A **binary quantity** is one that can take only 2 states



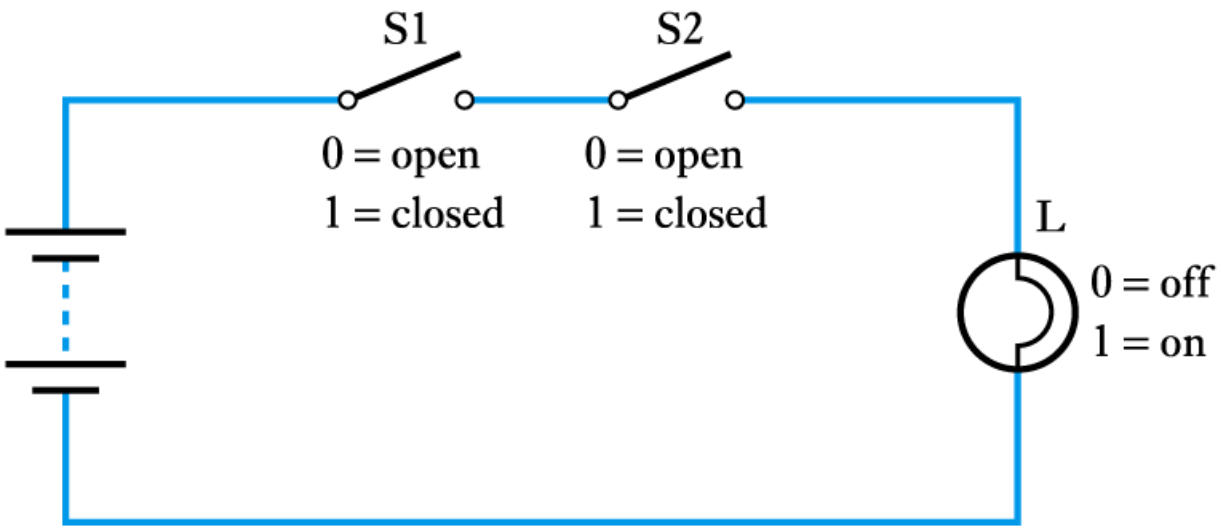
A simple binary arrangement

S	*L*
OPEN	OFF
CLOSED	ON

S	*L*
0	0
1	1

A truth table

- A binary arrangement with two switches in series



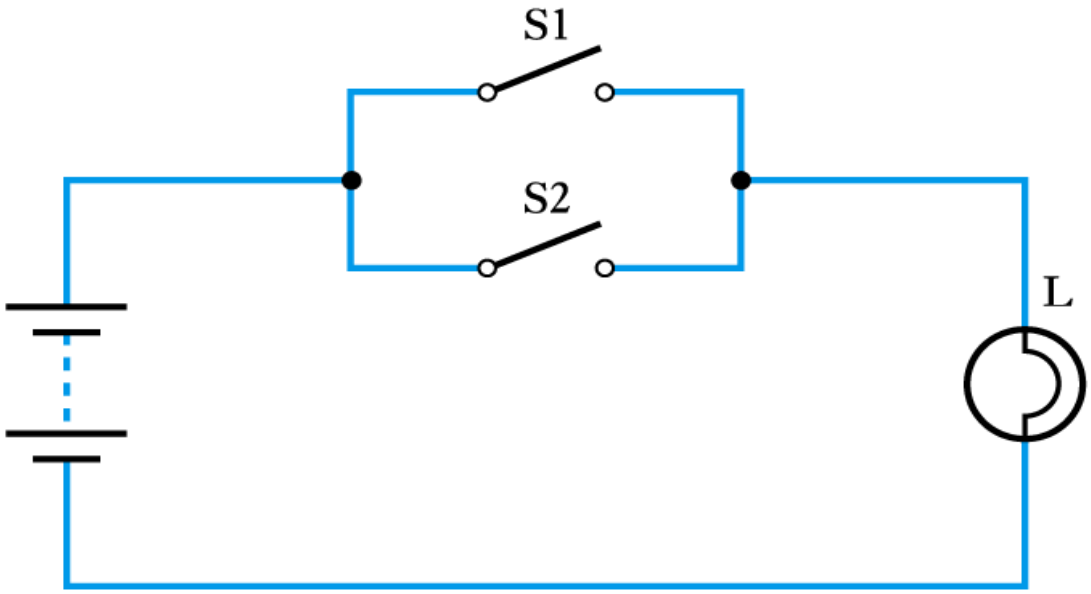
(a) Circuit

S1	S2	L
0	0	0
0	1	0
1	0	0
1	1	1

(b) Truth table

L = S1 AND S2

- A binary arrangement with two switches in parallel



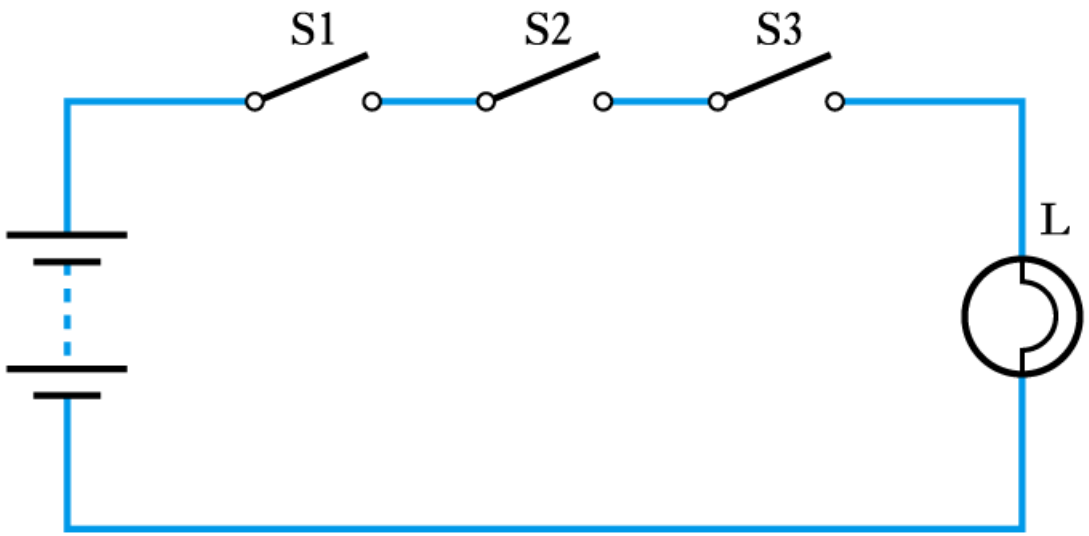
(a) Circuit

S1	S2	L
0	0	0
0	1	1
1	0	1
1	1	1

(b) Truth table

L = S1 OR S2

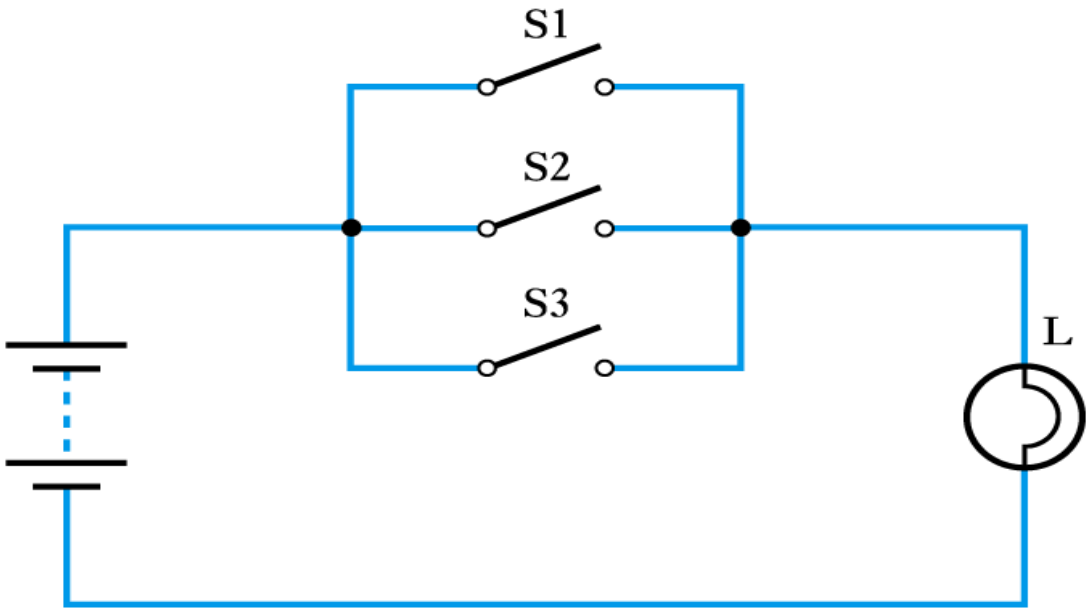
- Three switches in series



S1	S2	S3	L
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

L = S1 AND S2 AND S3

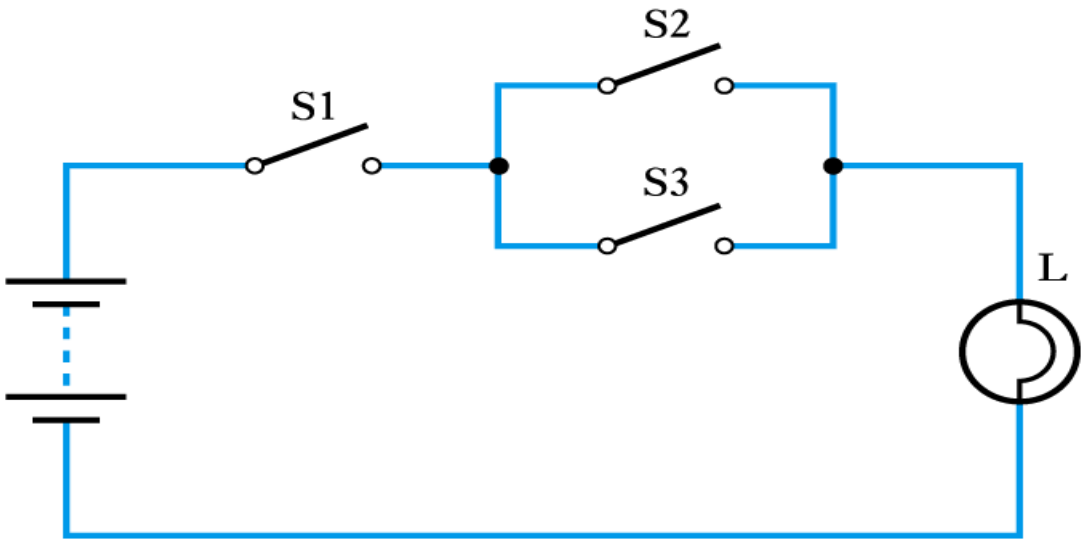
- Three switches in parallel



S1	S2	S3	L
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

L = S1 OR S2 OR S3

- A series/parallel arrangement



S1	S2	S3	L
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

L = S1 AND (S2 OR S3)

Numbers

Natural Numbers

Zero and any number obtained by repeatedly adding one to it.
Examples: 100, 0, 45645, 32

Negative Numbers

A value less than 0, with a – sign
Examples: -24, -1, -45645, -32

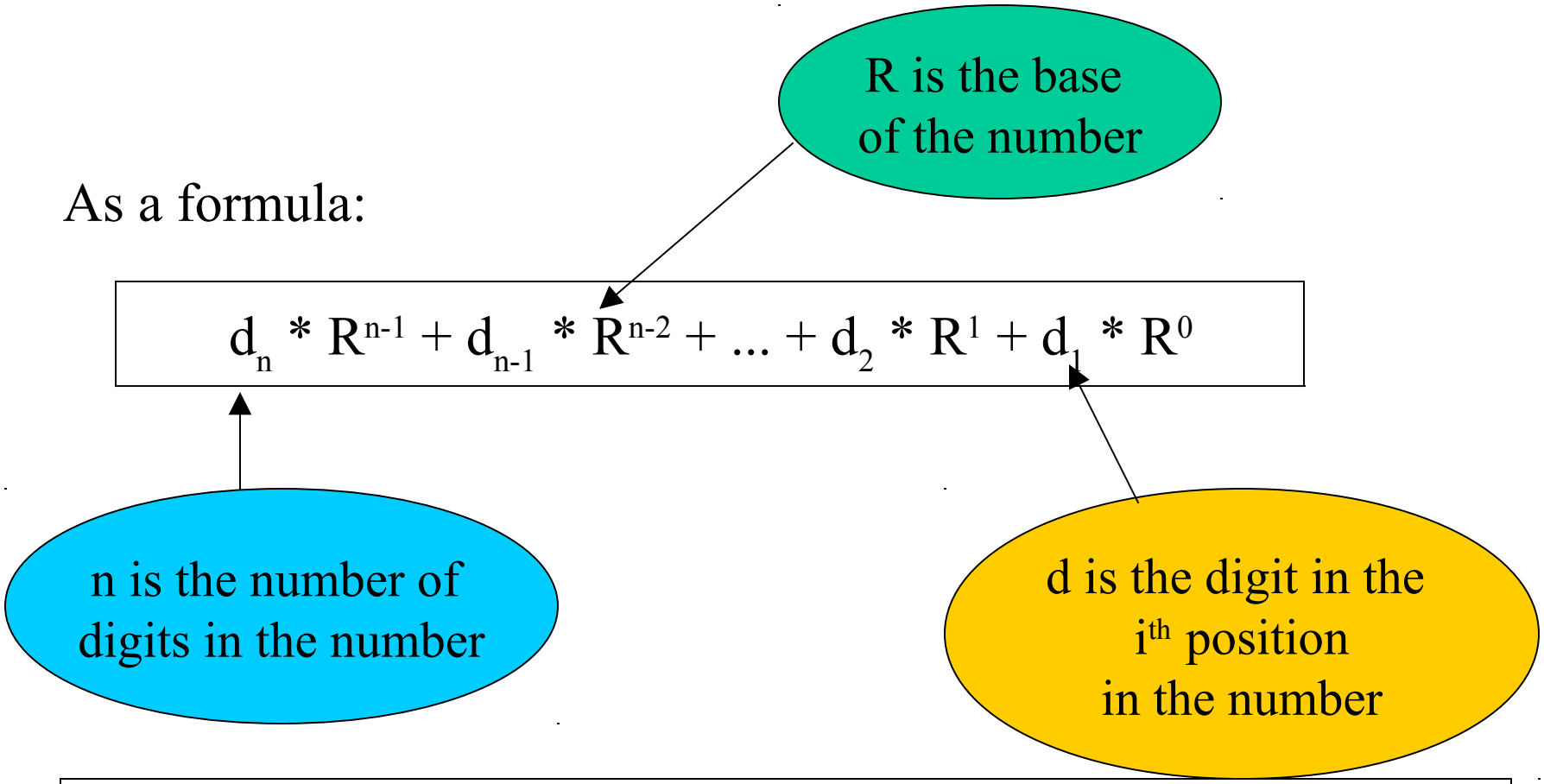
Integers

A natural number, a negative number
Examples: 249, 0, - 45645, - 32

Rational Numbers

An integer or the quotient of two integers
Examples: -249, -1, 0, 3/7, -2/5

Positional Notation

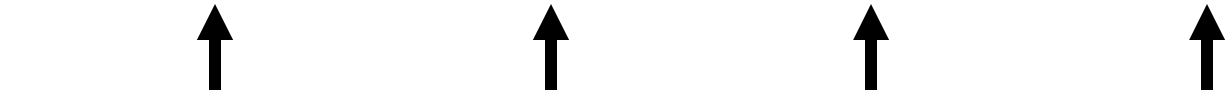


$1439 = 1 \times 10^3 + 4 \times 10^2 + 3 \times 10^1 + 9 \times 10^0$

$642 = 6 * 10^2 + 4 * 10 + 2 * 1$

Decimal numbers

$$(1439)_{10} = 1 \times 10^3 + 4 \times 10^2 + 3 \times 10^1 + 9 \times 10^0$$



Thousands Hundreds Tens Ones

- Base / Radix = 10

Binary→Decimal

$$1101 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

$$= 1 \times 8 + 1 \times 4 + 0 \times 2 + 1 \times 1$$

$$= 8 + 4 + 0 + 1$$

$$(1101)_2 = (13)_{10}$$

1, 2, 4, 8, 16, 32, 64, 128, 256, 512, ….

Significant Digits

Binary: 11101101

Most significant digit *Least significant digit*

Hexadecimal: 1D63A7A

Most significant digit *Least significant digit*

Binary Number System

- Also called the **"Base 2 system"**
- The binary number system is used to model the series of electrical signals computers use to represent information
- 0 represents the no voltage or an **off state**
- 1 represents the presence of voltage or an **on state**

Binary Numbering Scale

Base 2 Number	Base 10 Equivalent	Power	Positional Value
000	0	2^0	1
001	1	2^1	2
010	2	2^2	4
011	3	2^3	8
100	4	2^4	16
101	5	2^5	32
110	6	2^6	64
111	7	2^7	128

Converting Decimal to Other Bases

Algorithm for converting number in base 10 to other bases

Step 1. Divide the decimal number by the new base

Step 2. Make the remainder the next digit to the left in the answer

Step 3. Replace the original decimal number with the quotient While

Step 4.Repeat Step 1 to Step 3 till the quotient is not zero.

Division Algorithm

Decimal to Binary Conversion

Convert 67 to its binary equivalent:

$67_{10} = x_2$

Step 1: 67 / 2 = 33 R 1 — *Divide 67 by 2. Record quotient in next row*

Step 2: 33 / 2 = 16 R 1 — *Again divide by 2; record quotient in next row*

Step 3: 16 / 2 = 8 R 0 — *Repeat again*

Step 4: 8 / 2 = 4 R 0 — *Repeat again*

Step 5: 4 / 2 = 2 R 0 — *Repeat again*

Step 6: 2 / 2 = 1 R 0 — *Repeat again*

Step 7: 1 / 2 = 0 R 1 — **STOP** *when quotient equals 0*

$1\ 0\ 0\ 0\ 0\ 1\ 1_2$

Binary to Decimal Conversion

- The easiest method for converting a binary number to its decimal equivalent is to use the ***Multiplication Algorithm***
- Step 1. Multiply the binary digits by increasing powers of two, starting from the right(LSB)
- Step 2. Then, to find the decimal number equivalent, sum those products

Multiplication Algorithm

Convert $(10101101)_2$ to its decimal equivalent:

Binary 1 0 1 0 1 1 0 1

Positional Values 2^7 2^6 2^5 2^4 2^3 2^2 2^1 2^0

Products → $1x2^7 + 0x2^6 + 1x2^5 + 0x2^4 + 1x2^3 + 1x2^2 + 0x2^1 + 1x2^0$

$128 + 32 + 8 + 4 + 1$

173_{10}

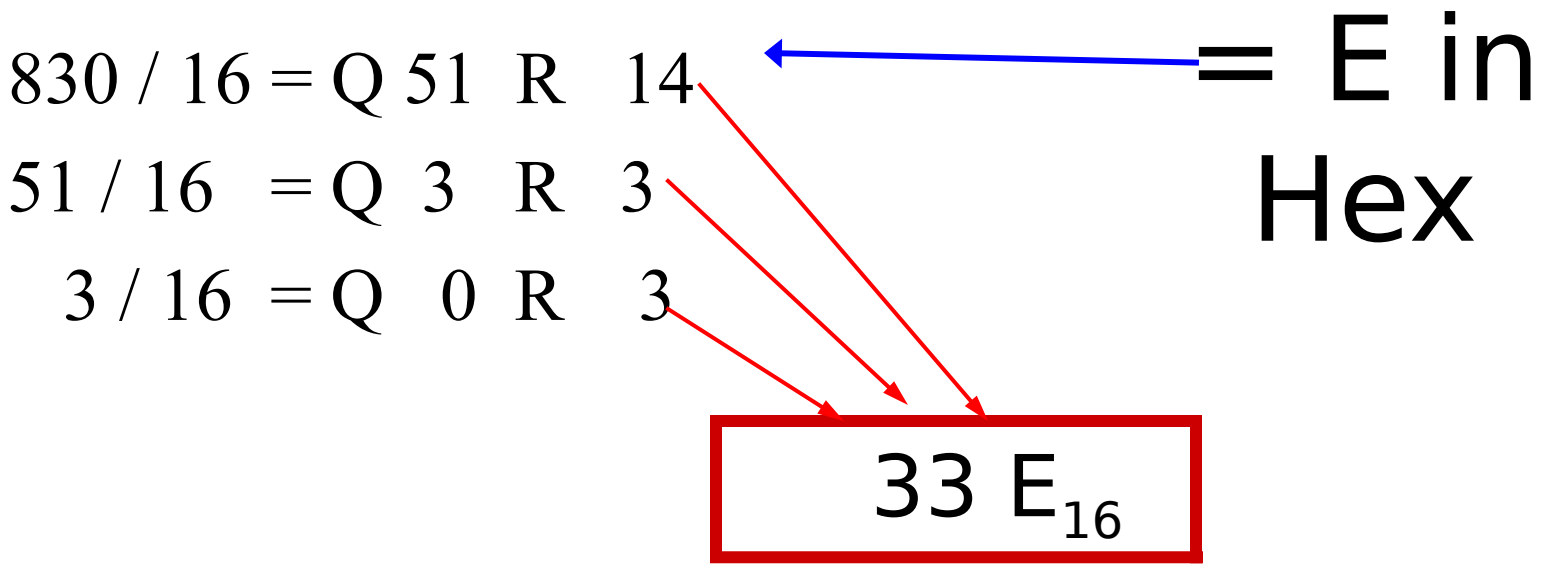
Hexadecimal Number System

- Base 16 system
- Uses digits 0-9 & letters A,B,C,D,E,F
- Groups of four bits represent each base 16 digit

Decimal	Hexadecimal
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	A
11	B
12	C
13	D
14	E
15	F

Decimal to Hexadecimal Conversion

Convert 830_{10} to its hexadecimal equivalent:



Hexadecimal to Decimal Conversion

Convert 3B4F16 to its decimal equivalent:

Hex Digits → 3 B 4 F

Positional Values → X 16^3 X 16^2 X 16^1 X 16^0

Products → 12288 + 2816 + 64 + 15

$15{,}183_{10}$

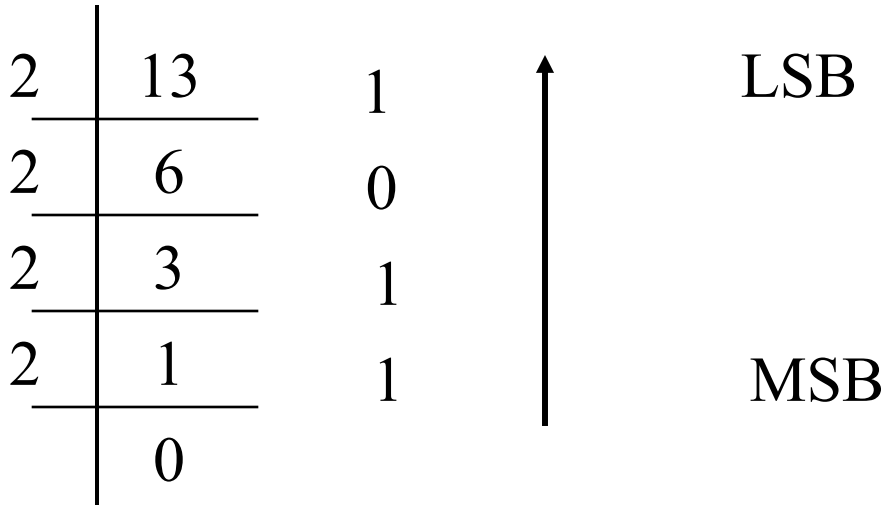
Binary to Hexadecimal Conversion

- The easiest method for converting binary to hexadecimal is to use a substitution code
- Each hex number converts to 4 binary digits

Substitution Code

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

Decimal → Binary



$$(13)_{10} = (1101)_2$$

Octal→Decimal

$$137 = 1 \times 8^2 + 3 \times 8^1 + 7 \times 8^0$$

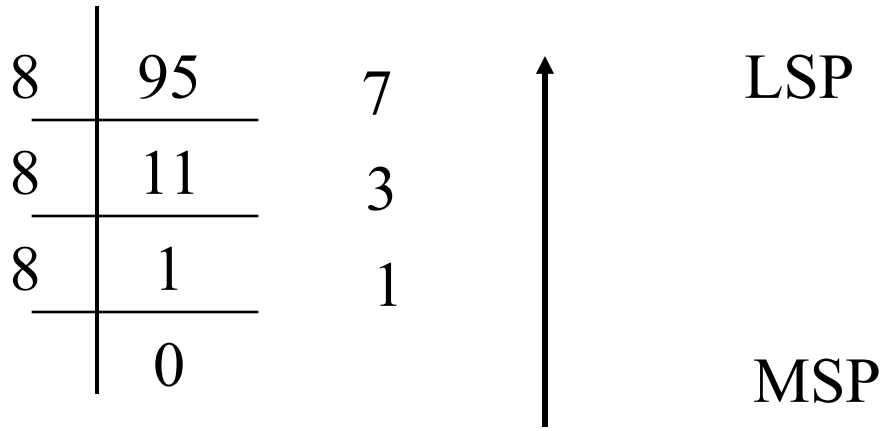
$$= 1 \times 64 + 3 \times 8 + 7 \times 1$$

$$= 64 + 24 + 7$$

$$(137)_8 = (95)_{10}$$

- Digits used in Octal number system – 0 to 7

Decimal → Octal



$$(95)_{10} = (137)_{8}$$

Hex→Decimal

$$BAD = 11 \times 16^2 + 10 \times 16^1 + 13 \times 16^0$$

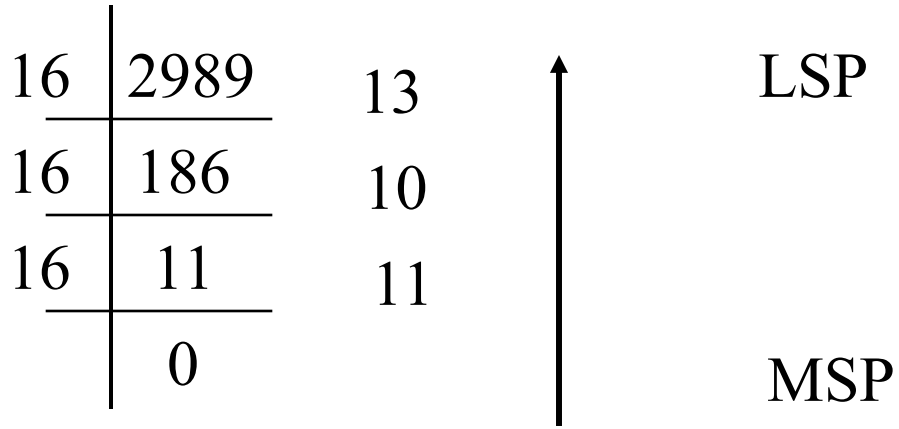
$$= 11 \times 256 + 10 \times 16 + 13 \times 1$$

$$= 2816 + 160 + 13$$

$$(BAD)_{16} = (2989)_{10}$$

A = 10, B = 11, C = 12, D = 13, E = 14, F = 15

Decimal → Hex



$$(2989)_{10} = (BAD)_{16}$$

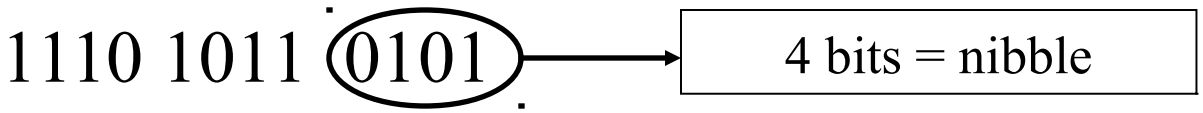
Why octal or hex?

- Ease of use and conversion
- Three bits make one octal digit

  111 010 110 101

  7 2 6 5 => 7265 in octal

- Four bits make one hexadecimal digit



E B 5 => EB5 in hex

$0_{hex} = 0_{dec} = 0_{oct}$	0	0	0	0
$1_{hex} = 1_{dec} = 1_{oct}$	0	0	0	1
$2_{hex} = 2_{dec} = 2_{oct}$	0	0	1	0
$3_{hex} = 3_{dec} = 3_{oct}$	0	0	1	1
$4_{hex} = 4_{dec} = 4_{oct}$	0	1	0	0
$5_{hex} = 5_{dec} = 5_{oct}$	0	1	0	1
$6_{hex} = 6_{dec} = 6_{oct}$	0	1	1	0
$7_{hex} = 7_{dec} = 7_{oct}$	0	1	1	1
$8_{hex} = 8_{dec} = 10_{oct}$	1	0	0	0
$9_{hex} = 9_{dec} = 11_{oct}$	1	0	0	1
$A_{hex} = 10_{dec} = 12_{oct}$	1	0	1	0
$B_{hex} = 11_{dec} = 13_{oct}$	1	0	1	1
$C_{hex} = 12_{dec} = 14_{oct}$	1	1	0	0
$D_{hex} = 13_{dec} = 15_{oct}$	1	1	0	1
$E_{hex} = 14_{dec} = 16_{oct}$	1	1	1	0
$F_{hex} = 15_{dec} = 17_{oct}$	1	1	1	1

Representation of Signed Numbers

Negative numbers

Three representations

- Signed magnitude
- 1's complement
- 2's complement

Sign magnitude

- Make MSB represent sign
- Positive = 0
- Negative = 1
- E.g. for a 3 bit set
  - "-2"

Sign	Bit	Bit
1	1	0
MSB		LSB

DRC(Diminished Radix Complement):(r-1)'s complement

1's complement

- MSB as in sign magnitude
- Complement all the other bits
- Given a positive number complement all bits to get negative equivalent
- E.g. for a 3 bit set
  - "-2"

Sign	Bit	Bit
1	0	1
0	1	0

RXC(Radix Complement): r ’s complement

2’s complement

If MSD is a 0 :

- The number is positive
- Remaining (n-1) bits directly indicate the magnitude

If the MSD is 1 :

- The number is negative
- Magnitude is obtained by complementing all the remaining (n-1) bits and adding a 1

- 1’s complement plus one
- E.g. for a 3 bit set
  - “-2”

Sign	Bit	Bit
1	1	0
0	1	0

Range of n-bit numbers

Twos complement numbers:

0111111 + 63

0000110 + 6

0000000 + 0

1111010 – 6

1000001 - 63

1000000 - 64

0 is represented by 000.....0

7- bit number covers the range from +63 to -64.

n-bit number has a range from +(2n-1 - 1) to -(2n-1)

Decimal number	Signed magnitude	2's complement	1's complement
3	011	011	011
2	010	010	010
1	001	001	001
0	000	000	000
-0	100	---	111
-1	101	111	110
-2	110	110	101
-3	111	101	100
-4		100	

No matter which scheme is used we get an even set of numbers but we need one less (odd: as we have a unique zero)

Binary Arithmetic in Computers

- Addition / subtraction
- Unsigned
- Signed
  - Using negative numbers

Unsigned: Addition

Like normal decimal addition

A \ B	+	0	1
	0	0	1
	1	1	10

```
  0101 (5)
+ 1001 (9)
----------
  1110 (14)
```

The carry out of the MSB is neglected

Unsigned: Subtraction

Like normal decimal subtraction

A	-	B: 0	B: 1
	0	0	11
	1	1	0

1001 (9)

- 0101 (5)

0100 (4)

A borrow (shown in red) from the MSB implies a negative

Signed arithmetic

- Use a negative number representation scheme
- Reduces subtraction to addition

2’s complement

Negative numbers in 2’s complement

$$\begin{array}{l} 001 \quad (\ 1)_{10} \\ 101 \quad (-3)_{10} \\ \hline 110 \quad (-2)_{10} \end{array}$$

The carry out of the MSB is lost/Discard the carry

Overflow / Underflow

- Maximum value N bits can hold : $2^n - 1$
- When addition result is bigger than the biggest number of bits can hold.
  - Overflow
- When addition result is smaller than the smallest number the bits can hold.
  - Underflow
- Addition of a positive and a negative number cannot give an overflow or underflow.

Overflow example

$$
\begin{array}{r}
011\ (+3)_{10} \\
011\ (+3)_{10} \\
\hline
110\ \ (+6)_{10} \qquad ????
\end{array}
$$

1's complement computer interprets it as –1 !!

$(+6)_{10} = (0110)_2$ requires four bits !

Underflow examples

Two's complement addition

$$\begin{array}{lll} & 101 & (-3)_{10} \\ & 101 & (-3)_{10} \\ \hline \text{Carry} \quad 1 & 010 & (-6)_{10} \quad ???? \end{array}$$

The computer sees it as +2.

$(-6)_{10} = (1010)_2$ again requires four bits !