Chapter One

Number Systems and Base Conversion

The binary system is fundamental to the study of digital electronics. In this lecture we will examine binary numbers and their relationship to other number systems such as decimal, octal, and hexadecimal. Arithmetic operations with binary numbers are covered to provide a basis for understanding how computers and other digital systems operate.

1.1 Decimal Numbers

Decimal system based on powers of 10. A number is written as a row of digits, with each position in the row corresponding to a certain power of 10. A decimal point in the row divides it into those powers of 10 equal to or greater than 0 and those less than 0, i.e., negative powers of 10. Positions farther to the left of the decimal point correspond to increasing positive powers of 10 and those farther to the right to increasing negative powers, i.e., to division by higher positive powers of 10.

<u>Example:</u> 4,309 = 3 2 1 0 (4×10)+(3x10)+(0×10)+(9×10^{0})=4,000+300+0+9

<u>Example:</u> 4.309= (4×10^{0})+ (3×10^{-1})+(0×10^{-2})+(9×10^{-3})=4+3/10+0/100+9/1000.

1.2 Binary Numbers

Binary system based on powers of 2. Binary uses two digits, so each column is worth twice the one before. This fact, coupled with expanded notation, can be used convert between from binary to decimal. In the binary system, the columns are worth 1, 2, 4, 8, 16, 32, 64, 128, 256, etc. To convert a number from binary to decimal, simply write it in expanded notation.

Example: $\mathbf{(101101)_2} = 2^5 \times 1 + 2^4 \times 0 + 2^3 \times 1 + 2^2 \times 1 + 2^1 \times 0 + 2^0 \times 1 = (45)_{10}$.

Example: Determine the value of the binary fractional number 0.1011.

$=1 \times 1^{-1} + 0 \times 1^{-2} + 1 \times 1^{-3} + 1 \times 1^{-4}$

$= 1 \times 0.5 + 0 \times 0.25 + 1 \times 0.125 + 1 \times 0.0625 = 0.5+0+0.125+ 0.0625 = 0.687510$

The following formula tells us how high we can count in decimal, beginning with zero, with n bits:

Highest decimal number = $2^n - 1$,

For example if n=2, then we can count from 0 to 3,

$2^2 - 1 = 3$

With four bit, we can count from 0 to 15,

$2^4 - 1 = 15$

1.3 Octal Numbers

Octal system based on powers of 8. The octal number system is composed of eight digits, which are: 0, 1, 2, 3, 4, 5, 6, 7 To count above 7, we begin another column and start over. 10, 11, 12, 13, 14, 15, 16, 17, 20, 21, 22, etc. Counting in octal is the same as counting decimal, except any number with 8 or 9 is omitted. To distinguish octal number from decimal number, we use the subscript 8 to indicate an octal number.

Example: $(127.4)_8=1x8^2+2x8^1+7x8^0+4x8^{-1}=(85.5)_{10}$

1.4 Hexadecimal Numbers

Hexadecimal based on powers of 16. The hexadecimal system has base sixteen; that is composed of 16 digit and characters (each hexadecimal digit represent by four bit binary number). The first ten digitis are borrowed from the decimal system. The letters A, B, C, D, E, and F are used for digits 10,11,12,13, 14,and 15.

For example: $(B65F)_{16}=11x16^3+6x16^2+5x16^1+15x16^0=(64687)_{10}$

This table shows number with different Bases:

Decimal (base 10)	Binary (base 2)	Octal (base 8)	Hexadecimal (base 16)
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
16	10000	20	10
17	10001	21	11
18	10010	22	12

19	10011	23	13
20	10100	24	14
21	10101	25	15
22	10110	26	16
23	10111	27	17
24	11000	30	18
25	11001	31	19
26	11010	32	1A
27	11011	33	1B
28	11100	34	1C
29	11101	35	1D
30	11110	36	1E
31	11111	37	1F

1.5 Number Based Conversions

1.5.1 Decimal to Binary Conversion

There are two ways of converting from decimal to binary number. Sum-of-Weights Method, and Repeated Division-by-2 Method,

a. Sum -of-Weights Method

One way to find the binary number equivalent to a given decimal number is to determine the set of binary weight values whose sum is equal to the decimal number.

Example: Convert the decimal numbers 12, 58, and 82 to binary number.

$(12_{10} = 8 + 4 = 2^3 + 2^2) = (1100)_2$

$(58_{10} = 32 + 16 + 8 + 2 = 2^5 + 2^4 + 2^3 + 2^1) = (111010)_2$

$(82_{10} = 64 + 16 + 2 = 2^6 + 2^4 + 2^1) = (1010010)_2$

b. Repeated Division-by-2 Method

A more systematic method of converting from decimal to binary is the repeated division by two processes until we have 0. The reminder generated by each division forms the binary number. The first reminder to be produced is the least significant bit (LSB) in the binary number.

Example:

6

2√12

12

0 0 (LSB)

. 3

2√6

6

0 0

1

2√3

2

1 1

. 0

2√1

0

1 1 (MSB) **$(1100)_2$**

1.5.2 Decimal to Octal

A method to converting from decimal number to octal number is the repeated division by eight method. Each successive division by 8 yields reminder which is a digit in the equivalent octal number. The first reminder is the least significant digit (LSD).

Example: Convert the number $359)_{10}$ from decimal number to octal number.

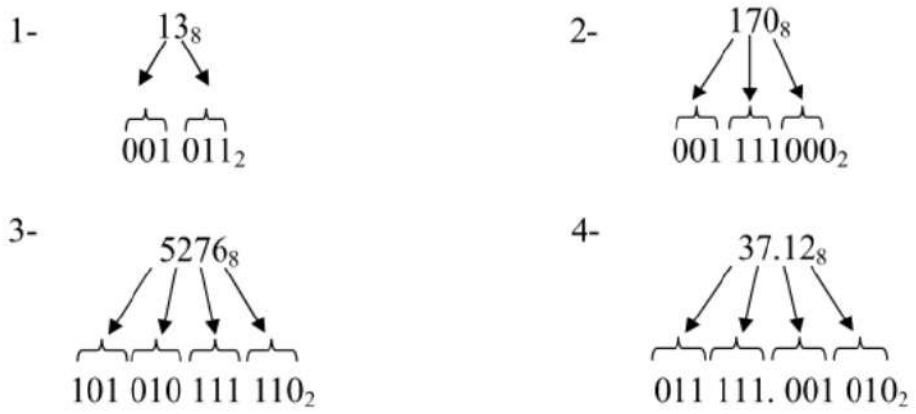
```
  44
8√359
  32
   39
   32
    7                    7 (LSD)


   5
 8√44
  40
   4                     4


   0
 8√5
   0
   5                     5  (MSD)
```

$359)_{10} = 547)_8$

1.5.3 Octal to Binary

To convert the octal number to binary number, simply replace each octal digit by the appropriate three bits.

Example: Convert each of the following octal numbers to binary: $13)_8$, $170)_8$, $5276)_8$, and $37.12)_8$.

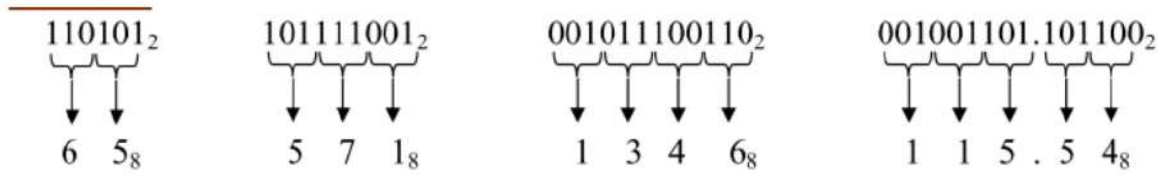


1.5.4 Binary to Octal

Beginning at right most bit, and if there is not three bit in the left most bit, zeros implied.

Example: Convert each of the following binary numbers to octal: 110101)$_2$, 101111001)$_2$, 1011100110)$_2$ and 1001101.1011)$_2$.

Solution:



1.5.4 Decimal to Hexadecimal

A method to converting from decimal number to hexadecimal number is the repeated division by 16 method. Each successive division by 16 yields reminder which is a digit in the equivalent formed hexadecimal number. The first reminder is the least significant digit (LSD).

Example: Convert the number 650)$_{10}$ from decimal number to hexadecimal number.

Step 1.

$$16\overline{)650} = 40, \quad 650 - 640 = 10 \longrightarrow \text{A (LSD)}$$

Step 2.

$$16\overline{)40} = 2, \quad 40 - 32 = 8 \longrightarrow 8$$

Step 3.

$$16\overline{)2} = 0, \quad 2 - 0 = 2 \longrightarrow 2 \text{ (MSD)}$$

$$\mathbf{650_{10} = 28A_{16}}$$

1.6 Binary Codes

1.6.1 Binary coded Decimal

BCD code is composed of four bits representing the decimal digit 0 through 9. So the code from 0000 to 1001are used and the six codes from 1010 to 1111 are not used. 8421 code is a type of BCD code and the designation "8421" indicate the binary weight of four bit (2^3, 2^2 , 2^1, 2^0).the table below shows the BCD code.

Example: Convert each the following number to BCD: 3, 18, 150, and 1472.

Solution:

3 = 0011

18 = 0001 1000

150 = 0001 0101 0000

1472 = 0001 0100 0111 0010

Example: Convert BCD cods to decimal numbers: 10000110, 100101110100, and 1100001100000.

Solution:

1000 0110	1001 0111 0100	1 1000 0110 0000
8 6	9 7 4	1 8 6 0

1.6.2. Other Decimal Codes

This table shows four different binary codes for the decimal digits.

Decimal digit	BCD 8421	2421	Excess-3	8,-4,-2,-1	Gray Code
0	0000	0000	0011	0000	0000
1	0001	0001	0100	0111	0001
2	0010	0010	0101	0110	0011
3	0011	0011	0110	0101	0010
4	0100	0100	0111	0100	0110
5	0101	1011	1000	1011	0111
6	0110	1100	1001	1010	0101
7	0111	1101	1010	1001	0100
8	1000	1110	1011	1000	1100
9	1001	1111	1100	1111	1101
Unused bit combinations	1010	0101	0000	0001	
	1011	0110	0001	0010	
	1100	0111	0010	0011	
	1101	1000	1101	1100	
	1110	1001	1110	1101	
	1111	1010	1111	1110	

1.6.3 Gray Code

The advantage of Gray code over the straight number sequence is that only one bit in the code group changes when going from one number to the next.

The code is called reflected because it can be generated in the following manner. Take the Gray code 0, 1. Write it forwards, then backwards: 0, 1, 1, 0. Then prepend 0s to the first half and 1s to the second half: 00, 01, 11, 10. Continuing, write 00, 01, 11, 10, 10, 11, 01, 00 to obtain: 000, 001, 011, 010, 110, 111, 101, 100,). Each iteration therefore doubles the number of codes.

The Gray code is used in applications where the normal sequence of binary numbers may produce an error or ambiguity during the transition from one number to the next.

Gray Code	Decimal equivalent
0000	**0**
0001	**1**
0011	**2**
0010	**3**
0110	**4**
0111	**5**
0101	**6**
0100	**7**
1100	**8**

1101	**9**
1111	**10**
1110	**11**
1010	**12**
1011	**13**
1001	**14**
1000	**15**