BASE CONVERSIONS

Converting from Base 10 to Base 2

Method I

E.g. Convert the decimal number 10 to base 2.

2	10
2	5 Remainder 0
2	2 Remainder 1
2	1 Remainder 0
	0 Remainder 1

Answer: 1010. The bits are reversed.

Method II

Use the headings 16 8 4 2 1 etc. (the last heading should not exceed the number). Place a 1 under all headings so that when they are added together we get the required number. Place a 0 under all headings not used. E.g. Convert 35 base 10 to base 2

32 16 8 4 2 1
1 0 0 0 1 1 Answer is 100011 base 2

Converting from Base 2 to Base 10

E.g. Convert 11001 to base 10

16	8	4	2	1
1	1	0	0	1 Ans = 16+8+1 = 25 base 10

Converting from Base 10 to Base 8 or 16

Divide the number by 8 or 16 and hold the remainder. Divide the quotient by 8 or 16 and hold the remainder, repeat this process until the quotient is 0. Reverse the remainder to give the answer.

Converting from Base 8 or 16 to Base 10

Starting from the rightmost digit use the below formula to convert

E.g. 1 Convert 130 base 8 to base 10:

$1 \times 8^2 + 3 \times 8^1 + 0 \times 8^0 = 1 \times 64 + 3 \times 8 + 0 \times 1 = 64 + 24 + 0 = 88$ base 10

E.g. 2 Convert A10 base 16 to base 10:

$10 \times 16^2 + 1 \times 16^1 + 0 \times 16^0 = 10 \times 256 + 1 \times 16 + 0 \times 1 = 2560 + 16 + 0 = 2576$ base 10

Converting Between Bases 2,8 and 16

Since $2^3 = 8$, $2^4 = 16$ then groups of 3's or 4's may be used when converting between bases 2,8, and 16

E.g. 1 Convert 11001 base 2 to (a) base 8 (b) base 16

Step 1 Starting from the rightmost bit group in 3's to give 011 001

Step 2 Convert each group to decimal to give the digits for the number (011 = 3, 001 = 1)
Answer: 31 base 8

(Use the same method for base 16 but group in 4's - 0001 1001 (0001 = 1, 1001 = 9)

Converting Binary Fractions to Decimal

Starting from the leftmost digit, use the following headings ½ ¼ ⅛ etc.

E.g. Convert .101 base 2 to decimal

Step 1 Starting from the leftmost bit we get 1 x ½ + 0 x ¼ + 1 x ⅛ = .5 + 0 + .125 = .625

NB. To convert 10.101, 10 is converted as usual (2) and the answer would be 2.625

Converting Decimal Fractions to Binary

Multiply the value to the right of the decimal point by 2 and hold the value to the left of the point. This process is repeated until the value to the right of the decimal point is 0 or repeats itself. The values to the left of the decimal point represent the binary answer (Read the answer from the top to the bottom, no reversing)

E.g. Convert .75 base 10 to binary.

.75
x 2
1.50
x 2
1.00 Ans .11

NB. To convert 5.75, 5 is converted as usual (101) and the answer would be 101.11

REPRESENTATION OF NEGATIVE NUMBERS

Sign and Magnitude/Explicit Sign Method

The leftmost bit represents the sign of a number (0 for a positive number and 1 for a negative number) and the remaining bits represent the size of the number.

E.g. 1 Using an 8 bit system represent (a) 13 and (b) -13 in sign and magnitude

Step 1: Convert the number to binary (13 in binary is 1101)

Step 2: Determine the sign (**0** in this case)

Step 3: Add zeroes (**0**) before the leftmost digit of the binary number to makeup the correct number of bits. Answer**: 0 0001101**. NB. -13 would be 1 001101

E.g. 2 Using an 8 bit system give the decimal representation of the following sign and magnitude numbers: (a) 10001011 (b) 00000110

Step 1: The leftmost bit is the sign (- since it is 1)

Step 2: The remaining bits represent the size of the number (0001011), convert them to base 10 (11) Answer -11, in the case of (b) the answer is 6.

One's Complement

This is obtained by changing the 0's to 1's and the 1's to 0's.

E.g1. Using an 8 bit system find the representation of - 19.

Step 1: Convert 19 to binary (10011)

Step 2: Add zeroes before the leftmost bit to makeup the correct number of bits (00010011)

Step 3: Change the 0's to 1's and the 1's to 0's (11101100, this is the answer)

Two's Complement

This is obtained by adding 1 to the 1's complement.

NB. In Two(s complement, all negative values begin with a 1 and all positive values with a 0.

Binary Coded Decimal (BCD)

Each digit of the decimal number is converted into its 4 bit binary representation. 1010 is used as the positive sign and 1011 as the negative sign.

E.g. 1 Represent (a) 23 and (b) -59 using BCD

Step 1 Convert each digit to its 4 bit binary representation (use zeroes before the leftmost bit to makeup 4 digits) (a) 23 = 0010 0011 (b) 101101011001

E.g. 2 What is the decimal representation of 101101011001?

Step 1 Starting from the rightmost digit, form groups of 4 bits (1011 0101 1001)

Step 2 Convert each group to decimal, remember that 1010 and 1011 are use for the sign)

Answer: -59

Representation of Characters

ASCII (American Standard Code for Information Interchange) - This system uses 7 bits to represent up to 128 characters and special symbols. It is a consecutive code, thus you add 1 to the code for "A" to get the code for "B". The ASCII system uses a different representation for lower case and upper case letters.

EBCDIC (Extended Binary Coded Decimal Interchange Code) - This code was developed by IBM and is also consecutive.

ANSI (American National Standards Institute)- This extends the ASCII set by using 8 bits instead of 7 and therefore covers 256 characters.

E.g. 1 Given that the ASCII representation for letter "C" is 1000100, what is the representation of (F(?

Step 1 Determine how far "F" is from "C" in the alphabet (C,D,E,F - 3 places)
Step 2 Add the number of places to the representation of (C(1000100 + 11 = 1000111

FIXED POINT REPRESENTATION

The computer treats the numbers as having a binary point placed among the bits. If a number is stored using 8 bits, it may be determined that the last 4 bits are to the right of the point. This method must be used for all numbers on this system. The point is not actually stored, however the number 10110010 is considered to be 1011.0010.

FLOATING POINT REPRESENTATION

The position of the decimal point changes and the number is represented by a sign, exponent (power of 2) and a mantissa. The exponent may be represented as 2's complement or sign and magnitude. A sign bit of 0 means a positive number and a 1 represents a negative number.

Positive Floating Point Calculations

Example

Using an 8 bit system where 1 bit is for the sign, 4 bits for the mantissa and 3 bits for the exponent (sign and magnitude). Give the representation of 5.75.

Step 1 Convert the number to binary to give 101.11

Step 2 Identify the sign (**0** since the number is positive)

Step 3 Identify the mantissa (**10111**, since we need 4 bits then this is stored as **1011**. The rightmost bit is dropped; hence this is an inexact representation of 5.75.

Step 4 Identify the exponent. This is 3 because we have to move the point three places to the right to get back the original numbers (.10111 moved 3 places to the right to gives 101.11). Next convert the exponent (3) to sign and magnitude with the correct number of bits to give **011**(Exponent is represented using sign and magnitude)

Step 5 The representation is the Sign (**0**) Mantissa (**1011**) Exponent (**011**) which is

0 1011 011

NB. The decimal number for the representation 0 1011 011 is found as follows:

Sign: **0**

Mantissa: **1011**

Exponent: **011**

Expression in standard form is $0.1011 \times 2^3 = 101.1 = 5.5$. Hence the number 5.75 is approximated to 5.5 when it is stored.

Negative Floating Point Calculations

Example

Given that 1 bit represents the sign, 3 bits represent the exponent and 4 bits the mantissa, what is the representation of -3.75?

Solution:

Step 1:
Determine the sign; this is 1 because the number is negative

Step 2:
Convert the number to binary to give -11.11
Step 3: Determine the exponent and convert it to binary, this is 010

Step 4:
Determine the mantissa, 1111 but since the number is negative we must find the two's complement of the mantissa. The one's complement is 0000, hence the two's complement is 0000 + 1 = 0001, the exponent is 0001.

Answer: Sign: 1 Exponent: 010 Mantissa: 0001

Testing the solution by doing the reverse. What decimal number is represented by 1 010 0001?

Solution:

Step 1:
Identify the sign: 1 which means that the number is negative and we have to find the two's complement of the mantissa.

Step 2:
Identify the exponent: 010, since this is positive we convert it to base 10 to get 2

Step 3
Identify the mantissa: 0001 but since the number is negative we must find the two's complement of the mantissa. The one's complement is 1110, hence the two's complement is 1110+1=1111

Step 4
Represent the number in standard form to give $-.1111 \times 2^2 = -11.11$

Step 5
Convert the standard form to base 10, -11.11 = -3.75

NORMALISATION

Normalization is used to overcome the problem of many different representations for the same number. For positive numbers the first bit of the mantissa, excluding the sign bit, is **1**. Thus 0000011111 would be represented as .**1**11110000. The exponent would have to be altered to compensate for the change. In this example the exponent has to move 5 places to the left to provide the original number, hence the exponent is -5.

For negative numbers the first bit of the mantissa, excluding the sign bit, is **0**. Thus 1111100100 is represented as .**0**01000000. The exponent would have to be altered to compensate for the change. In this example the exponent has to move 5 places to the left to provide the original number, hence the exponent is -5.

Example

Using a 16 bit register where 1 bit is for the sign, 9 bits for the mantissa and 6 bits for the exponent, represent 0.1875 in normalised form when both the mantissa and exponent are in two's complement.

Solution:

1. Convert 0.1875 to binary gives .0011
2. Represent the binary number with the correct number of bits (9). Five zeroes are added after the rightmost digit to give 001100000.
3. Place the assumed point to give 0.001100000
4. The Normalized form is 0.110000**00** Two zeroes are added after the rightmost digit to maintain the correct number of bits.
5. Since the point has to move two places to the left to get the original number then the exponent is -2. The two's complement representation for -2 is 111110
6. The normalized representation is sign: 0, mantissa: 110000000 and exponent: 111110 which gives: 0|110000000|111110

ERRORS IN COMPUTER ARITHMETIC

Precision: This refers to the number of bits available to represent a number.

Accuracy is a measure of the closeness of an approximation to the exact value.

Range: the set of all numbers which can be represented by a particular system.

Truncation: This is used to deal with situations when the precision is not adequate to represent all the digits in the number to be stored. E.g. suppose we wish to store 3.141592653 but only 5 digits are available then truncating the number means that all the digits after the fifth are dropped to give 3.1415

Overflow: This occurs when a computation has produced an answer that is too big to be represented in the system. E.g. a 4 bit system has a range of -8 to 7. However, 7 + 7 = 14 which is a number outside the range: 111 + 111 = 1110 which is a negative number (-2). An overflow also occurs if adding two negative numbers gives a positive result.

Error Classification:

Actual error = Exact Value – Computed Value.

E.g. Find the actual error when 5.75 is approximated to 5.5.

Actual Error = 5.75- 5.5 = 0.25

Relative Error = Actual Error/Exact Value.

E.g. Find the relative error when 5.75 is approximated to 5.5.

Relative Error = (5.75-5.5)/5.75 = 0.0434

Percentage Error = Relative Error x 100

E.g. Find the percentage error when 5.75 is approximated to 5.5.

Percentage Error = (5.75-5.5)/5.75 * 100 = 4.34