libri di testo : Harris & Harris, Digital Design and Computer Architechture, Arm Ed., Morgan Kaufmann 2016

Binary number system

1101 = 1*2^3 + 1*2^2 + 0 + 1*2^0 = 13

imparare a memoria le potenze del 2 da 2^0 a 2^15

2^23 = 2^10 * 2^10 * 2^3 = 1024* 1024 * 8 = 10^3 * 10^3 * 8 = 10^6 * 8 = 8000000 circa

Converti 10011 binario in decimale :

2^4 + 2 +1 = 19

Converti 47 decimale in binario :

Metodo 1 : Si divide per 2, e si vede se il resto è 0 o 1

101111

Metodo 2 : Si parte dalla cifra meno significativa, si prende la potenza di 2 immediatamente più piccola del numero decimale da convertire, in questo caso 32.

47 = 32 + x → x = 15 si ripete il metodo: 15 = 8 + x → x = 7

trova la potenza 2 che centra nel numero, se non trovi 0, ripeti

converti 53

53 – 32 = 21
21 – 16 = 5
5- 4 = 1
1 -1 = 0
32 + 16 + 4 + 1 = 110101

Binary values and range

supponiamo che abbiamo un numero di N bit, dove N = 4, la domanda è, quanti valori possiamo rappresentare con tali bit? Si possono rappresentare 2^N valori, in questo caso, con 4 bit possiamo rappresentare 16 valori. Il più grande numero rappresentabile è 2^N – 1, in questo caso è 15, difatti 1111 binario = 15 decimale.

Hexadecimal Numbers

Il sistema di numerazione esadecimale utilizza le regole del sistema binario ma con 16 cifre. I numeri da 0 a 9 corrispondono con il sistema decimale, da 10 in poi si utilizzano le lettere :
10 = A
11 = B
12 = C
13 = D
14 = E
15 = F

Tale sistema è comodo perché possiamo raggruppare un numero binario in blocchetti da 4 bit e scrivere per ogni blocco il valore esadecimale

Conversione da binario a esadecimale

Esempio : 10101101 = 1010 – 1101 → 1010 = A, 1101 = D

AD

Conversione da esadecimale a binario

4AF

F = 1111
A = 1010
4 = 0100

numero binario = 10010101111

Conversione da esadecimale a decimale

4AF = 4*16^2 + 10*16^1 + 15 * 16^0 = 1024 + 128 + 32 + 15 = 1199

8 bit formano un byte. Un byte è composto da 8 cifre binarie oppure da 2 cifre decimali.

32 bit sono 4 byte

un nibble è la perfetta metà di un byte.

1000 bit sono 1 byte

Addition

1011 + 0011

  11   riporto
1011 +
0011

1110

Addizione esadecimale

101
3A09 +
1B17

5520

Overflow

Se vogliamo rappresentare con 4 bit la somma 1011 + 0110 incappiamo in un problema, dato che il risultato 10001 richiede 5 bit, per questo non possiamo rappresentare il risultato della somma. Tale eccezione viene chiamato Overflow, il risultato è al di fuori del range disponibile.

Rappresentare numeri con il segno in binario

È convenzione utilizzare l'ultimo bit sulla sinistra per definire il segno del numero, se 0 il numero è positivo, se 1 il numero è negativo.
Esempio :

+6 = 0110   ->   0 (segno positivo) 110 (6 in numero binario)

-6 = 1110   ->   1 (segno negativo) 110 (6 in numero binario)

Questa rappresentazione si chiama SIGNED. Per rappresentare un valore binario bisogna quindi mettersi d'accordo sul tipo di convenzione da utilizzare, altrimenti uno stesso numero può avere 2 valori decimali diversi.

Nella convenzione SIGNED, il range rappresentabile è [-(2^(N-1) -1 ; -(2^(N-1)]

Usando tale convenzione il numero 0 può essere scritto in due modi : 10 o 00.

Tale convenzione è scomoda perché incompatibile con la versione UNSIGNED per le operazioni, è stata quindi creata una convenzione valida sia per i numeri con segno che senza segno, essa si chiama COMPLEMENTO A 2.

Two's Complement Numbers

In tale rappresentazione, il numero binario viene calcolato normalmente, eccetto per il fatto che l'ultima cifra sulla sinistra assume un valore di -2^N-1.

In tal modo, il valore più piccolo in un range è sempre 1 nel bit di sinistra, e 0 per tutti gli altri bit, contrariamente il valore maggiore è quando 0 si trova nel bit di sinistra, ed 1 in tutti gli altri. Ogni numero avente 1 come ultimo bit di sinistra è negativo. Il range in questo caso è :

$$\left[-2^{N-1},\ 2^{N-1}-1\right]$$

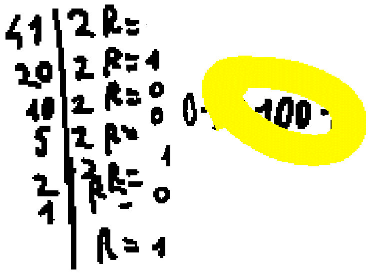
Esempio di conversione :

01101 = 0 + 1*2^3 + 1*2^2 + 0 + 1*2^0 = **13**
11010 = -2^4 + 1*2^3 +0 + * 1*2^1 + 0 = -16 + 8 + 2 = **-6**

L'ultima cifra a sinistra è il risultato della formula applicata -2^N-1. Il resto del numero viene calcolato come un binario semplice.

Per convertire un numero decimale in un numero complemento a 2, esistono due metodi :

Metodo 1 :
Supponiamo di voler calcolare il numero -41, di esso possiamo prima calcolare il suo valore positivo, cioè 41 :



Una volta ottenuto il valore binario del numero con segno positivo (aggiungendo uno 0 nell’ultimo bit a sinistra), invertiamo ogni bit, dove è 1 diventa 0 e viceversa :

0101001 inversione → 1b010110
Una volta ottenuto il numero invertito, sommiamo ad esso 1.

$$
\begin{array}{r}
1010110 \; + \\
0000001 \\
\hline
1010111
\end{array}
$$

Il risultato è 1010111, cioè -41.