Lezione 2

Conversioni tra basi

All'interno del calcolatore abbiamo i registri (insime di dispositivi che memorizzano bit), quindi importante sapere che lavora in binario.

! La base binaria indica in numero di valori che ogni cifra può assumere.

Es. Base 2 → Base 10

è possibile convertire un numero binario in decimale facendo questa piccola operazione, ovvero moltiplicare ogni

3-bit message		
A	B	C
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

cifra binaria (partendo da destra) per $2^{posizione}$.

Es. 101 → $1 * 2^2 + 0 * 2^1 + 1 * 2^0 = 5$

Conversione Base X → Base 10

Come precendentemente operato bisognerà solamente sostituire X base al posto di 2 e quindi moltiplicare ogni cifra per $x^{posizione}$.

Es. b = 3 | n = 102 →

$1 * 3^2 + 0 * 3^1 + 2 * 3^0 = 11$

Conversione Base 10 → Base X

Per convertire codificare invece il valore in base 10 in una base diversa bisognerà:

- Dividere il valore dato per X
- Ricordare il Resto
- Ripetere usando il quoziente fino a quanto si ha quoziente 0

La rappresentazione in base X è data dalla sequenza dei resti in ordine inverso.

Es. b = 2 | n = 21

21 / 2 = 10 con R=1

10 / 2 = 5 con R=0

5 / 2 = 2 con R=1

2 / 2 = 1 con R=0

1 / 2 = 0   con R=1

Quindi avremo 10101 come risultato (sempre invertito)

LSB E MSB

LSB = Less significant bit (Bit più a destra)

MSB = Most significant bit (Bit più a sinistra)

Operazioni tra numeri naturali

Addizione

```
0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 0 con riporto di 1
```

Es. Base 2

0011 + (3)

0010   (2)

0101

Esempio di overflow (carry)

L'overflow avviene nel momento in cui il riporto supera il massimo numero di bit avviene questo femonemo.

Esempio se andiamo a sommare:

b = 2 | N. Bit = 4 | {0 ... 15}

1010 + (10)

0111   (7)

10001

> Questo 1 nei numeri naturali vale come overflow ma anche come riporto, nei numeri interi per esempio non vale come riporto.

Moltiplicazione per la base o per una potenza della base (aggiungere zero)

b = 10 → 35 * 100 = 3500 (aggiungiamo gli zeri)

La stessa cosa vale se moltiplichiamo per la base anche se non decimale, esempio in binario:

b = 2 → 101 * 2 = 1010 (aggiungiamo zeri quanto la potenza del numero 2, in questo caso 2^1 quindi 1 zero)

Naturalmente vale anche per la potenza della base, se vogliamo moltiplicare 16 * 101 sappiamo che 16 è 2^4 quindi:

b = 2 → 101 * 2^4 = 1010000

Questa operazione (moltiplicazione) viene chiamata anche operazione di shift a sinistra.

Divisione per la base o potenza della base (rimuovere zero)

Come nella moltiplicazione vale la stessa regola.

Es. 1010000 / 16 (2^4) = 101~~0000~~ (togliamo gli zeri quanto la potenza del divisore)

> Naturalmente se avessimo un numero che non finisce solo con gli zero ma per esempio avremo 1010001, subentrerà il bisogno di utilizzare e disporre il numero con la virgola

Moltiplicazione standard

La moltiplicazione precedente (trucchetto) poteva avvenire solamente se il nostro numero finiva con 0 quante le cifre del numero della potenza.

Nella moltiplicazione avremo bisogno per rappresentare il risultato tra questa operazione con il doppio di bit

La moltiplicazione si esegue semplicemente moltiplicando il dividento per ogni singola cifra del divisore.

Es. 1

0111 x (7)

0110 (6)

0000

0111

0111

0000

0101010

Es. 2

> In questo caso avremo 3 (1) + 1 riporto (1) → 4_{10} = 100_2

> quindi se avremo 1 + 1 + 1 +1 il riporto andra non sulla prossima colonna, ma bensì sulla prossima ancora.

0111 x (7)

0111 (7)

0111

0111

0111

0000

0110001

Sottrazione e numeri negativi

Con la sottrazione dobbiamo introdurre i numeri negativi (interi), perchè naturalmente se avremo un sottraendo maggiore del minuendo il risultato sarà negativo.

Se c'è bisogno di fare la sottrazione tra 0 e 1, dovremmo prendere in prestito un 1 nella colonna successiva, quindi nell'esempio 0101 / 0011 → 00(10)1 / 0011.

In particolar modo quello che andremo a fare è aggiungere un 1 davanti allo 0, cosicchè diventa 10 e se sottraiamo ad esso 1 il risultato sarà comunque 1 (2-1)

Es. 1

0101 -

0011

0010

In questo caso prendendo in prestino l'unico 1 del dividendo comporterà l'aggiunzione di un prestito di valore uno sulle caselle di destra (meno importanza), nell'esempio

0100/0001

→ 001(1)0/0001

Es. 2

0100 -

0001

0011

Numeri interi

Esistono 3 metodi per rappresentare i numeri negativi:

- Modulo e segno
- Complemeno a 1 (Binario)
- Complemento a 2 (alla base)

In tutti questi casi comunque il numero disponibile di rappresentazioni no cambia, sempre $(N^{bit} - 1)$ rimane.

Modulo e segno

Dedichiamo il bit più significativo al segno:

- 0 = +
- 1 = -

I rimanenti danno il valore.

000 = +0

001 = +1

010 = +2

011 = +3

100 = -0

101 = -1

110 = -2

111 = -3

Complemento a 1

I negativi si ottendono complementando i valori che hanno il bit più significativo = 0

Complementare signifca invertire il valore 0→1 e 1→0.

Se inizia con 0 è positivo, se inizia con 1 allora è negativo e lo si complementa.

000 = +0

001 = +1

010 = +2

011 = +3

100 → 011 = -3

101 → 010 = -2

110 → 001 = -1

111 → 000 = -0

Complemento a 2

❗ Metodo più utilizzato e migliore, importante saperlo.

Il Bit più significativo ha peso negativo. Ovvero non ha solo il valore di bit di segno.

> $-Q_{n-1} * 2^{n-1} + \sum_{i=0}^{n-2} Q_i * 2^i$

Andremo a considerare il primo bit 1 ed il suo valore e poi a sommarlo con il valore dei restanti bit

000 = +0

001 = +1

010 = +2

011 = +3

100 = -4 + 0 → -4

101 = -4 + 1 → -3

110 = -4 + 2 → -2

111 = -4 + 3 → -1

Ciò scaturisce però che non avremo un'intervallo simmetrico. Nell'esempio non avremo un +4.

Opposto di un numero (complemento a 2)

Positivo → Negativo o viceversa.

Si ottiene facendo:

- Complemento a 1
- Sommando poi 1

Es.

1011 (-5) → 0100 +

0001 (1)

0101

> Un trucco sarebbe quello di complementare tutti i bit tranne l'ultimo se è uguale a 1.

Somma (complemento a 2)

Possiamo notare che se facciamo la somma tra (-5) e (-2) per esempio:

1011 +

1110

~~1~~1001

Il numero -7 lo possiamo rappresentare comunque e non subentra il concetto di overflow

Però il concetto di overflow può rimanere se superiamo comunque l'intervallo massimo rappresentabile dato dai bit.

con 4 bit = {-8 ... +7}

> Es. con negativi:

Stessa cosa vale con i positivi

> Es. con positivi

Sottrazione (complemento a 2)

Per la sottrazione basta fare la somma dell'opposto del divisore.

$A - B = A + (-B)$

Es. 5 - 6

0101 +(5)

1010   (-6)

1111 → -1

Moltiplicazione (complemento a 2)

Proprietà dell'estensione con il bit più significativo.

Per esempio per portare a 8 bit un numero di 4 bit basta aggiungere 4 volte davanti al numero binario il bit più significativo.

0101 → 0000 0101

1010 → 1111 1010