Architettura degli Elaboratori

Rappresentazione e aritmetica in Complemento a 2

Barbara Masucci
UNIVERSITA DEGLI STUDI DI SALERNO
DIPARTIMENTO DI INFORMATICA
DIPARTIMENTO DI ECCELLENZA

Punto della situazione

- Finora abbiamo considerato solo la rappresentazione dei numeri positivi
- Oggi vedremo alcune tra le varie rappresentazioni per numeri con il segno
  - Modulo e segno
  - Complemento a 2

Rappresentazione con modulo e segno

- Per rappresentare un numero con segno si usa
  - Un bit (il primo a sx) per il segno
    - 0 positivo
    - 1 negativo
  - n-1 bit (i successivi) per il modulo
    - Intervallo di rappresentazione: $[-2^{n-1}+1, +2^{n-1}-1]$

- Esempio: n=5 → intervallo $[-15_{10}, 15_{10}]$
  - $14_{10} \rightarrow 01110_{ms}$
  - $-14_{10} \rightarrow 11110_{ms}$

Rappresentazione con modulo e segno

- Problemi
  - Doppia rappresentazione dello zero
    - Es, con 8 bit: 10000000_{ms} e 00000000_{ms}
  - Operazioni aritmetiche complicate dal segno
  - Limite sui numeri rappresentabili
    - Mentre in binario puro con n bit il max rappresentabile è 2^n-1, in modulo e segno è $2^{n-1}-1$
    - Esempio: con 5 bit in binario puro, il max è 31_{10}, mentre in modulo e segno è 15_{10}

Rappresentazione in complemento a 2

- Differenza con il sistema posizionale:
  - Il peso del bit più a sinistra/più significativo è negativo
- Il valore di $b_{n-1}b_{n-2}...b_0$ è dato dalla relazione

$$N = -2^{n-1} b_{n-1} + \sum_{i=0}^{n-2} 2^i b_i$$

- 00010010_{C2} = + 2^4+ 2^1= $+18_{10}$
- 10010010_{C2} = -2^7+ 2^4+ 2^1= −128+18 = -110_{10}
- 111111_{C2} = -2^5+ 2^4+ 2^3+ 2^2+ 2^1+ 2^0= − 2^5 + 2^5−1= -1_{10}
- 111101_{C2}=-2^5+ 2^4+ 2^3+ 2^2+ 2^0= − 32 + 29 = -3_{10}

$$\sum_{i=0}^{n-1} 2^i = 2^n - 1$$

Positivi e negativi

- $00010010_{C2} = +2^4+2 = +18_{10}$
- $10010010_{C2} = -2^7+2^4+2 = -128+18 = -110_{10}$
- $111111_{C2} = -2^5+2^4+2^3+2^2+2^1+2^0 = -2^5+2^5-1 = -1_{10}$
- $111101_{C2} = -2^5+2^4+2^3+2^2+2^0 = -32+29 = -3_{10}$

Osservazione:

Il bit più a sinistra ci indica il segno:

0 significa positivo

1 significa negativo

Perché si chiama "complemento a 2"?

Sia B= $(b_{n-1}\, b_{n-2}...b_0)_{C2}$

- Se B è positivo, o zero, allora
  - b_{n-1}=0 e
  - il valore di $b_{n-2}...b_0$ è uguale a |B| (valore assoluto di B)

- Se B è negativo allora
  - b_{n-1}=1 e
  - il valore di $b_{n-2}...b_0$ è uguale a 2^{n-1} -|B|
    (cioè il complemento a 2^{n-1} di |B|)

- Esempio: n=4; B=1011_{C2}=-5_{10}
  - Notare che 011_2 =3_{10} = 8_{10} - 5_{10} = 2^{4-1} -|B|

Esempio con n=4 bit

$0000_{C2} = 0_{10}$
$0001_{C2} = +1_{10}$
$0010_{C2} = +2_{10}$
$0011_{C2} = +3_{10}$
$0100_{C2} = +4_{10}$
$0101_{C2} = +5_{10}$
$0110_{C2} = +6_{10}$
$0111_{C2} = +7_{10}$

Massimo=$+7_{10}$

$1000_{C2} = -8_{10}$
$1001_{C2} = -8 +1 = -7_{10}$
$1010_{C2} = -8+2 = -6_{10}$
$1011_{C2} = -8+3 = -5_{10}$
$1100_{C2} = -8+4 = -4_{10}$
$1101_{C2} = -8+5 = -3_{10}$
$1110_{C2} = -8+6 = -2_{10}$
$1111_{C2} = -8+7 = -1_{10}$

minimo = -8_{10}

Intervallo di rappresentazione con 4 bit: [-8_{10}, $+7_{10}$]

Esempio con n=32 bit

$00\ldots00_{C2} = 0_{10}$

$00\ldots01_{C2} = 1_{10}$

$00\ldots10_{C2} = 2_{10}$

$00\ldots11_{C2} = 3_{10}$

....

$01\ldots10_{C2} = 2^{31}_{10} - 2_{10}$

$01\ldots11_{C2} = 2^{31}_{10} - 1_{10}$

Massimo= $2^{31}_{10} - 1_{10}$

minimo= -2^{31}_{10}

$10\ldots00_{C2} = -2^{31}_{10}$

$10\ldots01_{C2} = -2^{31}_{10} + 1_{10}$

$10\ldots10_{C2} = -2^{31}_{10} + 2_{10}$

$10\ldots11_{C2} = -2^{31}_{10} + 3_{10}$

.....

$11\ldots10_{C2} = -2^{31}_{10} + 2^{31}_{10} - 2_{10}$

$= -2_{10}$

$11\ldots11_{C2} = -1_{10}$

Intervallo di rappresentazione con 32 bit:

$[-2^{31}_{10}, 2^{31}_{10} - 1_{10}]$

Numeri Rappresentabili

- 8 bit in complemento a 2
  - $+127_{10} = 01111111_{C2} = 2^7 - 1$
  - $-128_{10} = 10000000_{C2} = -2^7$
- 16 bit in complemento a 2
  - $+32767_{10} = 0111111111111111_{C2} = 2^{15} - 1$
  - $-32768_{10} = 1000000000000000_{C2} = -2^{15}$
- 32 bit in complemento a 2
  - $+\ 2\ 147\ 483\ 647_{10} = 0111\ldots\ldots111_{C2} = 2^{31} - 1$
  - $-\ 2\ 147\ 483\ 648_{10} = 1000\ldots\ldots000_{C2} = -2^{31}$

Intervallo di rappresentabilità

Numeri rappresentabili con n bit
in complemento a 2

minimo: $1000\ldots\ldots000_{C2} = -2^{n-1}$

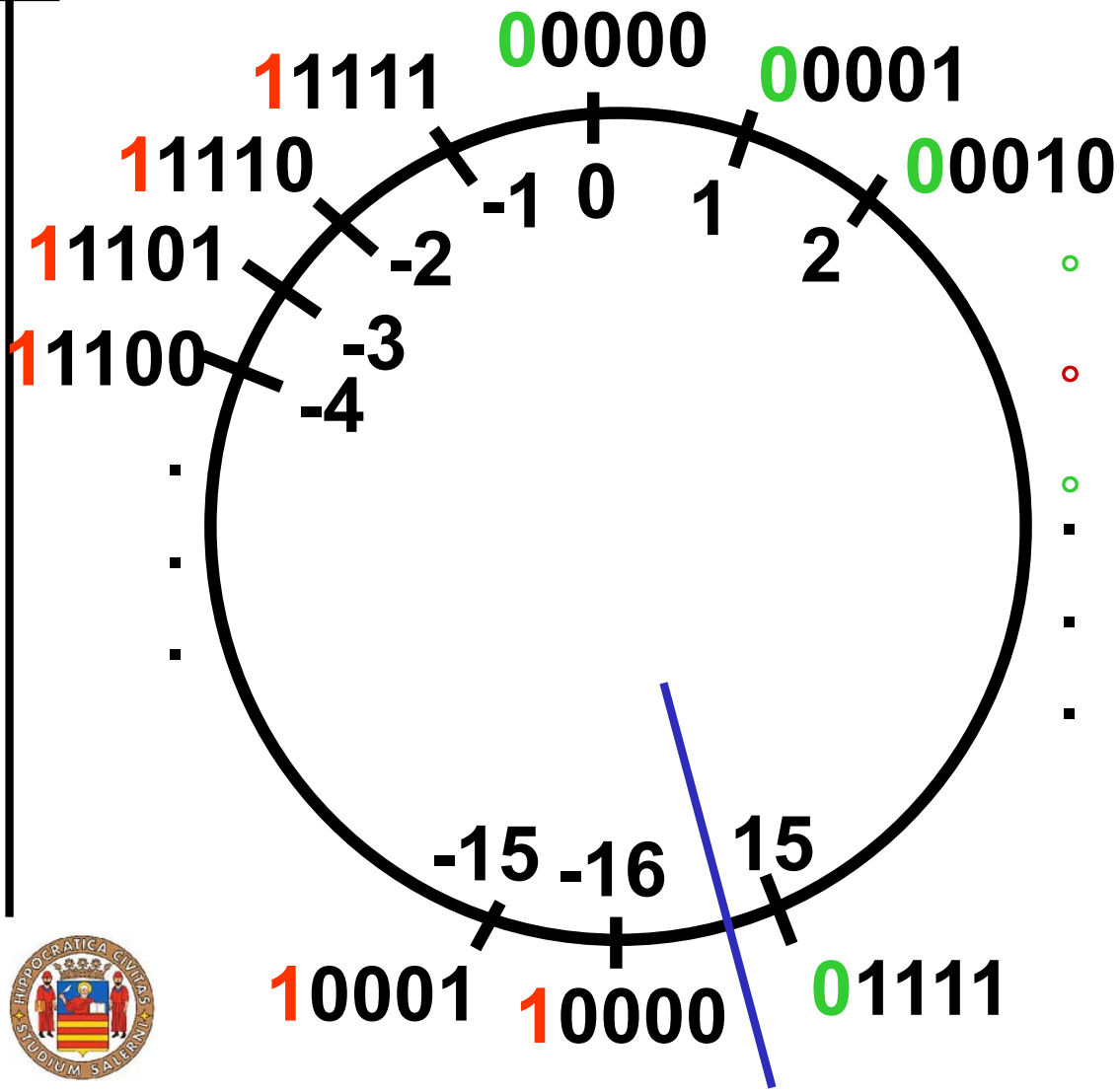
Massimo: $0111\ldots\ldots111_{C2} = 2^{n-1} - 1$

$$[-2^{n-1} , 2^{n-1} - 1]$$

Rappresentazione in complemento a 2

- Vantaggi
  - Una sola rappresentazione per lo zero
  - Facile ottenere l'opposto di un numero
  - Aritmetica semplice
  - Intervallo di rappresentazione: $[-2^{n-1}, 2^{n-1} - 1]$
    - Per es. con 4 bit $[-8_{10}, +7_{10}]$ anzichè $[-7_{10}, +7_{10}]$

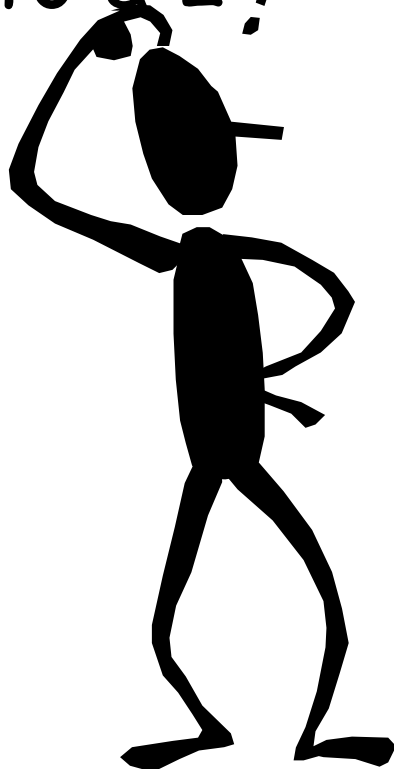
La linea dei numeri in complemento a 2 (su 5 bit)



- $2^{5-1}-1$ positivi : da 1 a 15
- 2^{5-1} negativi: da -16 a -1
- Un solo zero

Calcolo dell'opposto

Come calcolare l'opposto di un numero rappresentato in complemento a 2?



Calcolo dell'opposto

Cambiare di segno è semplice

Algoritmo 1:

- Si esegue il complemento bit a bit (negazione, cioè si trasforma ogni 1 in 0 e viceversa)
- Si somma 1

Esempio:

Con 6 bit:

Si esegue il complemento bit a bit

Si somma 1

$3_{10} = 000011_{C2}$

$111100_{C2} +$

$1 =$

111101_{C2}

$111101_{C2} = -2^5 + 2^4 + 2^3 + 2^2 + 2^0 = -32 + 29 = -3_{10}$

Calcolo dell'opposto
Perchè funziona?

$$N = -2^{n-1} b_{n-1} + \sum_{i=0}^{n-2} 2^i b_i$$ da cui

$$-N = 2^{n-1} b_{n-1} - \sum_{i=0}^{n-2} 2^i b_i$$

$$= -2^{n-1} + 2^{n-1} b_{n-1} + 2^{n-1} - \sum_{i=0}^{n-2} 2^i b_i$$

$$= -(1 - b_{n-1}) 2^{n-1} + (\sum_{i=0}^{n-2} 2^i + 1) - \sum_{i=0}^{n-2} 2^i b_i$$

$$= -(1 - b_{n-1}) 2^{n-1} + \sum_{i=0}^{n-2} 2^i (1 - b_i) + 1$$

$$\sum_{i=0}^{n-2} 2^i = 2^{n-1} - 1$$

1-b è il negato di b

Calcolo dell'opposto

Rivediamo:

$12_{10} = 00001100_{C2}$

Algoritmo 1

$$\begin{array}{r} 11110011_{C2} + \\ \qquad\qquad 1 = \\ \hline 11110100_{C2} \end{array}$$

C'è un altro metodo?

Algoritmo 2:

- Partendo da destra si lasciano invariati tutti i bit fino al primo 1 compreso, poi si invertono i rimanenti

Il caso dello zero

- Rappresentiamo lo zero con n=8 bit
- Cambiamo il segno con l'Algoritmo 1

$0_{10} = 00000000_{c2}$

Negazione: $11111111_{c2}+$

Somma di 1: $1 =$

Risultato: 100000000_{c2}

Non considerando l'ultimo riporto (il nono bit 1), si ha:

- 0 = 0

Il caso del minimo

- Consideriamo il minimo rappresentabile con n=8 bit
- Cambiamo il segno con l'Algoritmo 1

$-128_{10} = -2^7 = 10000000_{C2}$

Negazione: $01111111_{C2}+$

Somma di 1: $\underline{\qquad\qquad 1 =}$

Risultato: $10000000_{C2} = -128_{10}$!!!

In realtà sappiamo che con 8 bit l'intervallo di rappresentazione è $[-2^7, 2^7 -1] = [-128_{10}, 127_{10}]$

128_{10} non appartiene all'intervallo di rappresentabilità: c'è overflow!

Aumentare il numero di bit

- Per i numeri positivi si aggiungono 0 nella parte più significativa
  - $+18_{10}=$ $0010010_{c2} = 2^4+ 2= 18_{10}$
  - $+18_{10} =$ $00000000\ 00010010_{c2} = 2^4+ 2= 18_{10}$

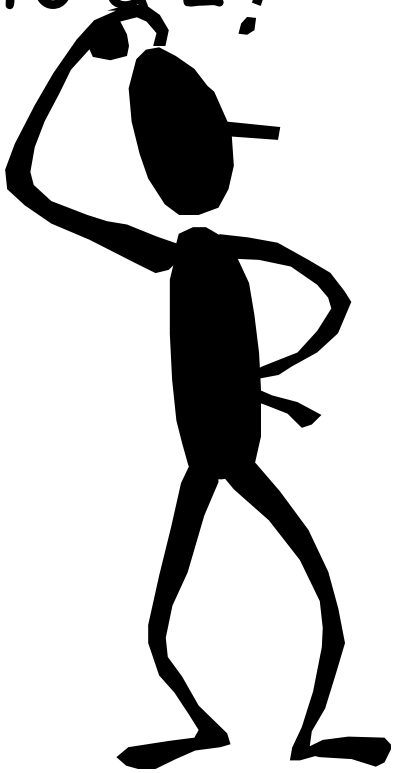
- Per i numeri negativi si aggiungono 1 nella parte più significativa
  - $-18_{10} =$ $101110 = -2^5+ 2^3+ 2^2+2= -32 +14=-18_{10}$
  - $-18_{10} =$ $1111\ 101110 = -2^9+(2^8 + \dots + 2^5) + 2^3+ 2^2+2$
    $= -2^9+(2^9 - 2^5) + 2^3+ 2^2+2$
    $= -2^5+ 2^3+ 2^2+2$
    $= -18_{10}$

$2^8+ 2^7+ 2^6 +2^5 = 2^9-2^5$

$$\sum_{i=h}^{k} 2^i = \sum_{i=0}^{k} 2^i - \sum_{i=0}^{h-1} 2^i = 2^{k+1} - 1 - (2^h - 1) = 2^{k+1} - 2^h$$

Addizione

Come calcolare la somma di due numeri rappresentati in complemento a 2?



Addizione

La somma in complemento a 2 si calcola come nel caso binario, ma:

- Si trascura l'eventuale bit di riporto finale (c_n)
- Si deve controllare la consistenza dei segni:
  - Se la somma di positivi è positiva o la somma di due negativi è negativa, il risultato è corretto
  - Altrimenti l'operazione non poteva essere fatta perché c'è traboccamento/overflow
    - Il risultato non è rappresentabile con quel numero di bit

Esempi

Esempio 1 (n=6 bit):
$12_{10} + 6_{10}$

$12_{10} = 001100_{C2}$
$6_{10} = 000110_{C2}$

001100
001100 +
000110 =
010010

Corretto!
$010010_{C2} = 2^4 + 2 = 18_{10}$
$12_{10} + 6_{10} = 18_{10}$

Esempio 2 (n=6 bit):
$25_{10} + (-13_{10})$

$25_{10} = 011001_{C2}$
$-13_{10} = 110011_{C2}$

110011
011001 +
110011 =
001100

Corretto!
$001100_{C2} = 2^3 + 2^2 = 12_{10}$
$25_{10} + (-13_{10}) = 12_{10}$

Esempi

n=6
intervallo rappresentabilità
$[-32_{10}, 31_{10}]$

Esempio 3 (n=6 bit):
$13_{10} + 23_{10}$

$13_{10} = 001101_{C2}$
$23_{10} = 010111_{C2}$

```
011111
 001101 +
 010111 =
 100100
```

Overflow!
$100100_{C2} = -2^5 + 2^2 = -28_{10}$
$13_{10}+23_{10}=36_{10}$
non rappresentabile con 6 bit

Esempio 4 (n=6 bit):
$-13_{10} + (-23_{10})$

$-13_{10} = 110011_{C2}$
$-23_{10} = 101001_{C2}$

```
100011
 110011 +
 101001 =
 011100
```

Overflow!
$011100_{C2} = 2^4 + 2^3 + 2^2 = 28_{10}$
$-13_{10} + (-23_{10}) = -36_{10}$ non rappresentabile con 6 bit

Overflow nell'addizione

- Si verifica se e solo se
  - Il risultato della somma di due interi positivi è un intero negativo
  - Il risultato della somma di due interi negativi è un intero positivo
- Esempio
  - Sommiamo 5 e 4 su 4 bit (max rappr. $2^3-1=7_{10}$)
  - $+5_{10} = 0101_{c2}$
  - $+4_{10} = 0100_{c2}$

$$\begin{array}{l} 0101+ \\ \underline{0100=} \\ 1001 \end{array}$$

Overflow!
La somma di due interi positivi non può essere un intero negativo

Overflow nell'addizione

- Una regola utile per determinare se c'è stato overflow
  - Se $c_{n-1} = c_n$ il risultato è corretto
  - Si ha invece overflow ogni qual volta $c_{n-1} \neq c_n$

c_n c_{n-1} c_i c_2 c_1

$a_{n-1}...a_i...a_2a_1a_0 +$

$b_{n-1}...b_i...b_2b_1b_0 =$

$s_n s_{n-1} \ ... \ s_i \ ... \ s_2 s_1 s_0$

Sottrazione

- Si calcola l'opposto del sottraendo e si somma al minuendo
- a - b = a + (-b)

Avremo quindi bisogno dei soli circuiti di somma e complemento

Riepilogo e riferimenti

- Rappresentazione in complemento a 2 e calcolo dell'opposto
  - [P] par. 6.1
- Addizione e sottrazione di interi nella rappresentazione in complemento a 2
  - [P] par. 6.2