Informatica B - Esercitazione 0

Codifica binaria - Numeri naturali

Stefano Cereda
stefano1.cereda@mail.polimi.it

10/10/2017

Politecnico di Milano

Recap e Introduzione

Recap

- Combinazioni possibili e numero di bit richiesti:
  2^N (da 0 a $2^N - 1$) e $\lceil log_2 M \rceil$

Recap

- Combinazioni possibili e numero di bit richiesti:
  2^N (da 0 a $2^N - 1$) e $\lceil log_2 M \rceil$
- Codifica binaria pura:
  $1011_2 = 1 + 2 + 8 = 11_{10}$ e divisione ripetuta

Recap

- Combinazioni possibili e numero di bit richiesti:
  2^N (da 0 a $2^N - 1$) e $\lceil log_2 M \rceil$
- Codifica binaria pura:
  $1011_2 = 1 + 2 + 8 = 11_{10}$ e divisione ripetuta
- Somme in binario puro:
  $0100 + 111 = 1011$

Recap

- Combinazioni possibili e numero di bit richiesti:
  2^N (da 0 a $2^N - 1$) e $\lceil log_2 M \rceil$
- Codifica binaria pura:
  $1011_2 = 1 + 2 + 8 = 11_{10}$ e divisione ripetuta
- Somme in binario puro:
  $0100 + 111 = 1011$
- ASCII:
  'A'+1 = 'B'

Introduzione

- Numeri interi: codifica modulo e segno e codifica in complemento alla base

Introduzione

- Numeri interi: codifica modulo e segno e codifica in complemento alla base
- Non faremo i numeri reali

Numeri interi

Modulo e segno

Nella codifica in modulo e segno si utilizza il bit più significativo (*msb*) per indicare il segno dei numeri:

$0101_2 = +101_2 = +(1+4)_{10} = +5_{10}$
$1101_2 = -101_2 = -(1+4)_{10} = -5_{10}$

Modulo e segno

Nella codifica in modulo e segno si utilizza il bit più significativo (*msb*) per indicare il segno dei numeri:

$0101_2 = +101_2 = +(1+4)_{10} = +5_{10}$
$1101_2 = -101_2 = -(1+4)_{10} = -5_{10}$

Con N bit useremo un bit per il segno, i rimanenti per il modulo. Possiamo quindi rappresentare numeri nell'intervallo $-2^{N-1}+1 \leq x \leq +2^{N-1}-1$

Modulo e segno

Nella codifica in modulo e segno si utilizza il bit più significativo (*msb*) per indicare il segno dei numeri:

$0101_2 = +101_2 = +(1+4)_{10} = +5_{10}$
$1101_2 = -101_2 = -(1+4)_{10} = -5_{10}$

Con N bit useremo un bit per il segno, i rimanenti per il modulo. Possiamo quindi rappresentare numeri nell'intervallo
$-2^{N-1} + 1 \leq x \leq +2^{N-1} - 1$

Abbiamo infatti due possibili codifiche per il numero zero:
$+0 = +0000 = 00000$
$-0 = -0000 = 10000$
$(N = 5)$

Complemento alla base

La precedente codifica “spreca” un valore, con N bit vorremo infatti poter rappresentare 2^N valori, non $2^N - 1$.

La rappresentazione in complemento alla base rappresenta, con N bit, i valori da -2^{N-1} a $2^{N-1} - 1$

Complemento alla base

La precedente codifica “spreca” un valore, con N bit vorremo infatti poter rappresentare 2^N valori, non $2^N - 1$.

La rappresentazione in complemento alla base rappresenta, con N bit, i valori da -2^{N-1} a $2^{N-1} - 1$

La rappresentazione in complemento a 2, su N bit, di un numero x_{10} è definita come:

- x_2 se $x \geq 0$ (msb $= 0$)
- $(2^N - |x|)_2$ se $x < 0$ (msb $= 1$, dimostrare)

Complemento alla base

La precedente codifica “spreca” un valore, con N bit vorremo infatti poter rappresentare 2^N valori, non $2^N - 1$.

La rappresentazione in complemento alla base rappresenta, con N bit, i valori da -2^{N-1} a $2^{N-1} - 1$

La rappresentazione in complemento a 2, su N bit, di un numero x_{10} è definita come:

- x_2 se $x \geq 0$ (msb $= 0$)
- $(2^N - |x|)_2$ se $x < 0$ (msb $= 1$, dimostrare)

Abbiamo quindi una sola rappresentazione per il numero 0.

Attenzione: il *msb* indica il segno solo se siamo in base 2, non è un bit di segno in senso stretto.

Esempio cpl2 su 3 bit

$N = 3 \rightarrow -2^2 \leq x \leq 2^2 - 1 \rightarrow -4 \leq x \leq 3 \quad M = 2^3 = 8$

Esempio cpl2 su 3 bit

$$N = 3 \rightarrow -2^2 \leq x \leq 2^2 - 1 \rightarrow -4 \leq x \leq 3 \quad M = 2^3 = 8$$

10	cpl2	da 10 a cpl2
+3	011	1+2
+2	010	2
+1	001	1
0	000	0
-1	111	$2^3 - \lvert -1 \rvert = 8 - 1 = 7_{10} = 111_2$
-2	110	$8 - 2 = 6_{10} = 110_2$
-3	101	$8 - 3 = 5$
-4	100	$8 - 4 = 4$

Notare che msb non è il segno: cambiandolo non si ottiene il numero opposto.

Proprietà cpl2

Rappresentiamo $-x$ con $N = 4$ bit:
$-x \rightarrow (2^N - |-x|)_2 = 10000 - x = 1111 - x + 1$

Proprietà cpl2

Rappresentiamo $-x$ con $N = 4$ bit:

$-x \rightarrow (2^N - |-x|)_2 = 10000 - x = 1111 - x + 1$

$1111 - x$ possiamo ottenerlo semplicemente invertendo tutti i bit di x:

$$\begin{array}{r|c} 1111 & - \\ 0101 & = \\ \hline 1010 & \end{array}$$

Proprietà cpl2

Rappresentiamo $-x$ con $N = 4$ bit:

$-x \rightarrow (2^N - |-x|)_2 = 10000 - x = 1111 - x + 1$

$1111 - x$ possiamo ottenerlo semplicemente invertendo tutti i bit di x:

$$\begin{array}{r|c} 1111 & - \\ 0101 & = \\ \hline 1010 & \end{array}$$

Adottando la codifica in complemento a 2 il calcolatore necessita solo di un circuito di addizione (più quello banale di complementazione) per effettuare sia somme che sottrazioni!

Metodi di conversione da base 10 a cpl2

Per convertire un numero negativo da base 10 a cpl2 abbiamo tre metodi:

1. Utilizzare la definizione: $(2^N - |x|)_2$:
   $N = 3; -2 \rightarrow (2^N - 2)_2 = (6)_2 \rightarrow 110$

Metodi di conversione da base 10 a cpl2

Per convertire un numero negativo da base 10 a cpl2 abbiamo tre metodi:

1. Utilizzare la definizione: $(2^N - |x|)_2$:
   $N = 3; -2 \rightarrow (2^N - 2)_2 = (6)_2 \rightarrow 110$
2. Convertire l'opposto in base 2, invertire i bit e sommare 1:
   $N = 3; -2 \rightarrow +2_{10} = 010_2 \rightarrow 101 \rightarrow 110$

Metodi di conversione da base 10 a cpl2

Per convertire un numero negativo da base 10 a cpl2 abbiamo tre metodi:

1. Utilizzare la definizione: $(2^N - |x|)_2$:
   $N = 3; -2 \rightarrow (2^N - 2)_2 = (6)_2 \rightarrow 110$
2. Convertire l'opposto in base 2, invertire i bit e sommare 1:
   $N = 3; -2 \rightarrow +2_{10} = 010_2 \rightarrow 101 \rightarrow 110$
3. Convertire l'opposto in base due, ricopiarlo da *lsb* verso *msb* fino al primo 1 (compreso), copiare i restanti bit complementati:
   $N = 4; -6 \rightarrow +6_{10} = 0110_2 \rightarrow 1010$

Metodi di conversione da base 10 a cpl2

Per convertire un numero negativo da base 10 a cpl2 abbiamo tre metodi:

1. Utilizzare la definizione: $(2^N - |x|)_2$:
   $N = 3; -2 \rightarrow (2^N - 2)_2 = (6)_2 \rightarrow 110$
2. Convertire l'opposto in base 2, invertire i bit e sommare 1:
   $N = 3; -2 \rightarrow +2_{10} = 010_2 \rightarrow 101 \rightarrow 110$
3. Convertire l'opposto in base due, ricopiarlo da *lsb* verso *msb* fino al primo 1 (compreso), copiare i restanti bit complementati:
   $N = 4; -6 \rightarrow +6_{10} = 0110_2 \rightarrow 1010$

Metodi di conversione da base 10 a cpl2

Per convertire un numero negativo da base 10 a cpl2 abbiamo tre metodi:

1. Utilizzare la definizione: $(2^N - |x|)_2$:
   $N = 3; -2 \rightarrow (2^N - 2)_2 = (6)_2 \rightarrow 110$
2. Convertire l'opposto in base 2, invertire i bit e sommare 1:
   $N = 3; -2 \rightarrow +2_{10} = 010_2 \rightarrow 101 \rightarrow 110$
3. Convertire l'opposto in base due, ricopiarlo da *lsb* verso *msb* fino al primo 1 (compreso), copiare i restanti bit complementati:
   $N = 4; -6 \rightarrow +6_{10} = 0110_2 \rightarrow 1010$

Controlliamo l'ultimo risultato utilizzando la definizione di cpl2:
$1010_2 = 2^N - |x| = 2^N + x \rightarrow x = 1010_2 - 2^N = 10 - 16 = -6$

Operazioni algebriche in cpl2

Dati due numeri in base 10 da sommare algebricamente, dobbiamo innanzitutto controllare il numero di bit necessari:

- Se N è assegnato, dobbiamo verificare che sia sufficiente. $(-2^{N-1} \leq x \leq 2^{N-1} - 1)$
- Altrimenti dobbiamo calcolare il valore minimo capace di rappresentare entrambi i valori.

Somma senza riporto ne overflow

$(-7) + (+2) = -5$ con $N = 4$

Somma senza riporto ne overflow

$(-7) + (+2) = -5$ con $N = 4$

$-2^{N-1} = -8 \wedge 2^{N-1} - 1 = +7 \rightarrow$ i bit sono sufficienti

Somma senza riporto ne overflow

$(-7) + (+2) = -5$ con $N = 4$

$-2^{N-1} = -8 \wedge 2^{N-1} - 1 = +7 \rightarrow$ i bit sono sufficienti

$-7 \rightarrow 7_{10} = 0111_2 \rightarrow 1001_{cpl2}$
$+2 \rightarrow 2_{10} = 0010_2 \rightarrow 0010_{cpl2}$

Somma senza riporto ne overflow

$(-7) + (+2) = -5$ con $N = 4$

$-2^{N-1} = -8 \wedge 2^{N-1} - 1 = +7 \rightarrow$ i bit sono sufficienti

$-7 \rightarrow 7_{10} = 0111_2 \rightarrow 1001_{cpl2}$

$+2 \rightarrow 2_{10} = 0010_2 \rightarrow 0010_{cpl2}$

$$\begin{array}{c|c} 1001 & + \\ 0010 & = \\ \hline 1011 & \end{array}$$

Somma senza riporto ne overflow

$(-7) + (+2) = -5$ con $N = 4$

$-2^{N-1} = -8 \wedge 2^{N-1} - 1 = +7 \rightarrow$ i bit sono sufficienti

$-7 \rightarrow 7_{10} = 0111_2 \rightarrow 1001_{cpl2}$

$+2 \rightarrow 2_{10} = 0010_2 \rightarrow 0010_{cpl2}$

$$\begin{array}{c|c} 1001 & + \\ 0010 & = \\ \hline 1011 & \end{array}$$

$1011 < 0 \rightarrow 0101 = 5 \rightarrow -5$

Somma con riporto ma senza overflow

$(+7) + (-2) = +5$ con $N = 4$

$-2^{N-1} = -8 \wedge 2^{N-1} - 1 = +7 \rightarrow$ i bit sono sufficienti

$+7 \rightarrow 7_{10} = 0111_2 \rightarrow 0111_{cpl2}$
$-2 \rightarrow 2_{10} = 0010_2 \rightarrow 1110_{cpl2}$

$$\begin{array}{r|l} 0111 & + \\ 1110 & = \\ \hline 10101 & \end{array}$$

Ignoriamo il riporto:
$0101 > 0 \rightarrow 0101 = 5 \rightarrow +5$ Il risultato è corretto

Somma senza riporto ma con overflow

$(+7) + (+2) = +9$ con $N = 4$

$-2^{N-1} = -8 \wedge 2^{N-1} - 1 = +7 \rightarrow$ i bit sono sufficienti per gli operandi, ma avremo una situazione di overflow

$+7 \rightarrow 7_{10} = 0111_2 \rightarrow 0111_{cpl2}$
$+2 \rightarrow 2_{10} = 0010_2 \rightarrow 0010_{cpl2}$

$$\begin{array}{c|c} 0111 & + \\ 0010 & = \\ \hline 1001 & \end{array}$$

$1001 < 0 \rightarrow 0111 = 7 \rightarrow -7$

Non abbiamo riporto, cosa ci indica la presenza di overflow?

Somma con riporto e overflow

$(-7) + (-2) = -9$ con $N = 4$

$-2^{N-1} = -8 \wedge 2^{N-1} - 1 = +7 \rightarrow$ i bit sono sufficienti per gli operandi, ma avremo una situazione di overflow

$-7 \rightarrow 7_{10} = 0111_2 \rightarrow 1001_{cpl2}$
$-2 \rightarrow 2_{10} = 0010_2 \rightarrow 1110_{cpl2}$

$$\begin{array}{r|l} 1001 & + \\ 1110 & = \\ \hline 10111 & \end{array}$$

Ignoriamo il riporto: $0111 > 0 \rightarrow 0111 = 7 \rightarrow +7$

Operandi concordi ma risultato discorde indicano che siamo in una condizione di overflow.

(Operandi discordi non daranno mai overflow)