

Parte II

Indice

- ◆ Operazioni aritmetiche tra valori rappresentati in binario puro
 - somma
 - sottrazione
- ◆ Rappresentazione di numeri con segno
 - modulo e segno
 - complemento a 2
 - esercizi
- ◆ Operazioni aritmetiche tra numeri interi con segno
 - somma e sottrazione
 - shift, moltiplicazione e divisione

Operazioni aritmetiche

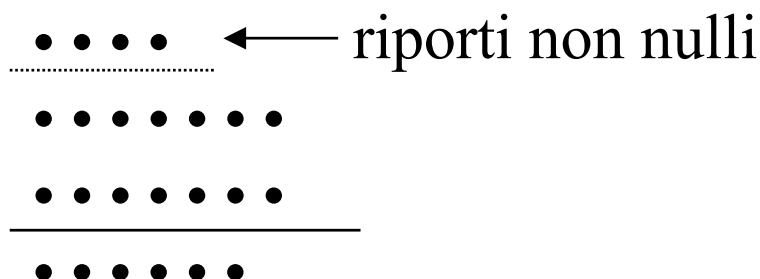
Valori in binario puro

Somma

◆ Si esegue la somma tra i bit di pari peso, ricordando che:

- $0 + 0 = 0$
- $0 + 1 = 1$
- $1 + 0 = 1$
- $1 + 1 = 0$ *con riporto 1*
- $1 + 1 + 1 = 1$ *con riporto 1*

◆ Esempio



Operazioni aritmetiche

Valori in binario puro

- ◆ In generale, la somma è definita su gruppi di 3 bit:
 - due addendi
 - il riporto (*carry*) generato dalla somma sulla colonna di peso inferiore
- ◆ In generale, la somma genera una coppia di bit:
 - una cifra di peso uguale a quella degli addendi
 - un riporto di peso immediatamente superiore

Operazioni aritmetiche

Valori in binario puro

Sottrazione

- ◆ Si esegue la sottrazione dei bit di uguale peso, ricordando che:
 - $0 - 0 = 0$
 - $1 - 0 = 1$
 - $1 - 1 = 0$
 - $0 - 1 = 1$ *con prestito dal bit di peso immediatamente superiore*

◆ Esempi

$$\begin{array}{r} \overset{\curvearrowright}{1011} - \overset{\curvearrowright}{1100} \\ \underline{0110} = \underline{0011} \\ 0101 \quad 1001 \end{array} \quad \leftarrow \begin{array}{l} \text{prestiti non} \\ \text{nulli} \end{array}$$

Operazioni aritmetiche

Valori in binario puro

- ◆ Anche la sottrazione opera su terne di bit:
 - le due cifre di peso uguale
 - il prestito (*borrow*) proveniente dalla cifra di peso immediatamente superiore
- ◆ In generale la sottrazione genera un bit:
 - una cifra di peso pari a quello dei bit interessati

Rappresentazione dei numeri con segno

- ◆ Si considerino le due operazioni seguenti su numeri in binario puro su 6 bit:

$$\begin{array}{r} 111111 + \\ \hline 1000000 \end{array} \quad \begin{array}{r} 000000 - \\ \hline 111111 \end{array}$$

non rappresentabile

$0 - 1 = 63 ???$

- ◆ Nel primo caso il numero ottenuto **non è rappresentabile** in quanto il risultato è su 7 bit
- ◆ Nel secondo caso il risultato dell'operazione è **errato** in quanto si è utilizzata la rappresentazione in binario puro e questa non contempla i numeri negativi

Rappresentazione dei numeri con segno

- ◆ Per rappresentare i numeri con segno la rappresentazione in binario puro è carente
- ◆ In generale le rappresentazioni adatte a rappresentare numeri con segno adottano la seguente convenzione:
 - il MSB è utilizzato per rappresentare il segno
 - il valore del MSB è il seguente:
 - 0 corrisponde al segno positivo
 - 1 corrisponde al segno negativo

Rappresentazione dei numeri con segno

Rappresentazione in Modulo e Segno (MS)

- ◆ Il MSB è utilizzato per rappresentare il segno
- ◆ I bit restanti della rappresentazione costituiscono il modulo (valore assoluto)
- ◆ Esempi (su 5 bit)

$$\underbrace{10101}_{5}_{\text{MS}} = -5_{10}$$

$$0 \underbrace{1101}_{13}_{\text{MS}} = +13_{10}$$

Modulo e segno (MS)

- ◆ Con n bit totali, si possono rappresentare i numeri interi nell'intervallo

$$[-(2^{n-1}-1) , + (2^{n-1}-1)]$$

- ◆ Nella rappresentazione in modulo e segno esistono **due diverse rappresentazioni dello 0**:

$$+ 0 = 00\dots0 \quad \text{e} \quad - 0 = 100\dots0$$

- ◆ Complicazione nei calcoli, spreco di una configurazione di bit
- ◆ Al fine di contrastare questo inconveniente vengono introdotte altre rappresentazioni per i numeri con segno

Complemento di un numero

- ◆ Dato un numero X in base r su n cifre, si definisce *complemento alla base* la quantità:

$$r^n - X$$

- ◆ Esempi

- Con $n = 2$ e $r = 10$

- il complemento alla base di 36 è:

$$10^2 - 36 = 64$$

- Con $n = 5$ e $r = 2$

- il complemento alla base di 00101 è:

$$2^5 - X = 100000 - 00101 = 11011$$

- ◆ Quando la base è 2 si ha il *complemento a 2 (CA2)*

Complemento di un numero

◆ Operativamente:

il complemento ad una base generica \mathbf{b} si ottiene:

- copiando (partendo dal LSB, ossia da destra) tutte le cifre pari a 0
- calcolando il *complemento a \mathbf{b}* della prima cifra (partendo sempre da destra) diversa da zero
- calcolando il *complemento a $\mathbf{b} - 1$* di tutte le cifre seguenti

Complemento di un numero

◆ Esempio

Con $n = 6$ e $r = 5$, il complemento alla base di 103200_5 si ottiene calcolando:

- i primi due zeri restano invariati
 - la prima cifra diversa da 0 a partire da destra è il 2 che complementato alla *base 5* produce 3
 - le cifre successive vengono complementate a $(base-1)$, ossia a 4
- si ottiene: 341300_5

Complemento di un numero

- ◆ Dato un valore X in base r su n cifre, si definisce *complemento alla base meno uno* la quantità:

$$(r^n - 1) - X$$

- ◆ Esempi

- Con $n = 2$, $r = 10$ e $X = 36$:

$$(10^2 - 1) - 36 = 63$$

- Con $n = 5$, $r = 2$ e $X = 00101$:

$$(2^5 - 1) - X = 11111 - 00101 = 11010$$

- ◆ Per la base 2 (secondo esempio) si parla di *complemento a 1*, il CA1 di un numero si ottiene complementando (invertendo) ogni singolo bit ($0 \rightarrow 1$ e $1 \rightarrow 0$)

Rappresentazione in complemento a 2

- ◆ E' la rappresentazione più utilizzata per i numeri interi con segno in quanto non ha il problema del doppio zero e facilita i calcoli
- ◆ **Numeri positivi:**

0	<i>modulo</i>
---	---------------
- ◆ **Numeri negativi:** si applica l'*operazione* di CA2 sulla *rappresentazione* CA2 del corrispondente valore positivo
- ◆ In CA2 esiste una sola rappresentazione dello zero
- ◆ Anche in CA2 i valori negativi hanno $MSB = 1$

Rappresentazione in Complemento a 2

- ◆ Con n bit totali, si possono rappresentare i numeri interi nell'intervallo

$$[-(2^{n-1}) , + (2^{n-1}-1)]$$

Esempio

con $n = 5$ si ha:

$$-16_{10} \leq X \leq +15_{10}$$

$$10000_2 \leq X \leq 01111_2$$

- ◆ La rappresentazione di un numero positivo in CA2 è equivalente a quelle in binario puro e in MS a parità di bit

Calcolo del CA2 di un numero

◆ Metodi per il calcolo del CA2 di un numero:

– si può applicare direttamente la definizione di complemento alla base

$$CA2(X) = 2^n - X$$

– si può partire dalla definizione di complemento alla base -1

$$CA1(X) = (2^n - 1) - X$$

e osservare che la definizione di CA2 può essere espressa in funzione di questa

$$CA2(X) = CA1(X) + 1$$

allora per complementare a 2 un numero è possibile complementarlo a 1 e poi sommare 1 al risultato

Calcolo del CA2 di un numero

- ◆ Metodi per il calcolo del CA2 di un numero (*seguito*):
 - si applica la seguente regola pratica:
 - si parte **da destra**, si trascrivono tutti gli 0 fino ad incontrare il primo 1 e si trascrive anch'esso
 - si complementano a 1 (0 → 1 e 1 → 0) tutti i bit restanti

Note

- Il CA2 di un numero negativo dà il corrispondente valore positivo
- $CA2(CA2(X)) = X$

Calcolo del CA2 di un numero

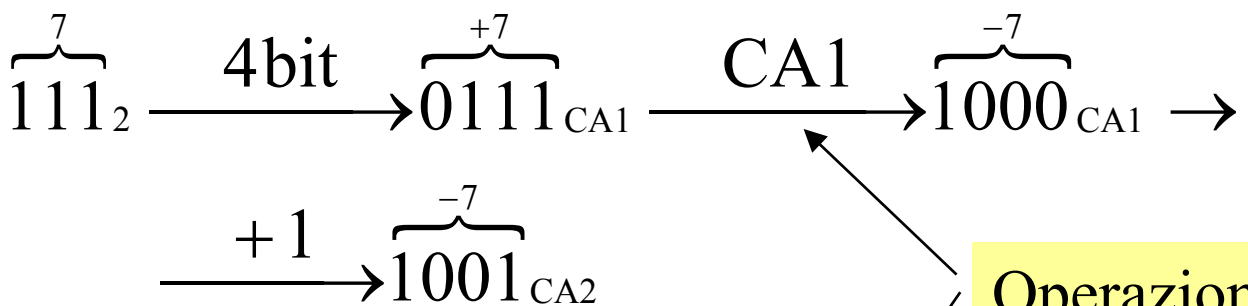
- ◆ Esempio Rappresentare in CA2 su 4 bit il numero -7

$$7_{10} = 111_2$$

- Applicando la definizione:

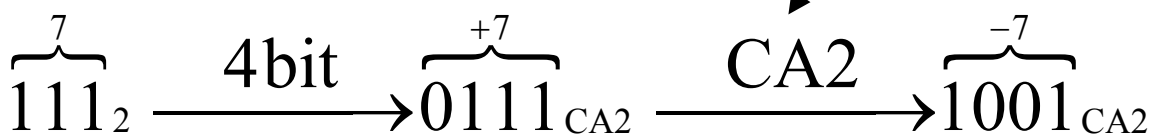
$$2^4 - 7 = 10000 - 111 = 1001_{CA2}$$

- Passando dal CA1:



Operazioni

- Con la regola pratica



Rappresentazione in CA2 e operazione di CA2

- ◆ Si distingue attentamente tra i concetti di *rappresentazione in CA2* e di *operazione di CA2*.

La *rappresentazione* dice come sono organizzati i bit, mentre il *calcolo* è una procedura di trasformazione dei bit

- In particolare:
 - per rappresentare un numero positivo in CA2 non serve applicare l'operazione di CA2
 - Per rappresentare un numero negativo in CA2 è necessario applicare l'operazione di CA2 alla rappresentazione del corrispondente valore positivo

Rappresentazione in CA2 e calcolo del CA2

Conversione da CA2 a decimale

- ◆ Se il numero è positivo (MSB = 0), si converte come numero binario puro
- ◆ Se il numero è negativo (MSB = 1), si applica l'operazione di CA2 a questo valore ottenendo la rappresentazione del corrispondente positivo, si converte il risultato come numero in binario puro (e si aggiunge il segno meno)
- ◆ Esempio

$$10110_{CA2} \rightarrow (?)_{10}$$

$$\overbrace{10110}_{-X}^{CA2} \rightarrow \overbrace{01010}^{+X}_{CA2} = 10_{10}$$

quindi $10110_{CA2} = -10_{10}$ ← coincidono

Operazioni aritmetiche

Complemento a 2

- ◆ La somma di due numeri nella rappresentazione in CA2 si esegue sommando tutti i bit degli addendi (segno compreso)
- ◆ Un eventuale riporto (*carry*) oltre il bit di segno (MSB) viene scartato
- ◆ Nel caso gli operandi siano di segno concorde (entrambi positivi o entrambi negativi) occorre verificare la presenza o meno di overflow (il segno del risultato non è concorde con quello dei due addendi)

Operazioni aritmetiche

Complemento a 2

- ◆ Le differenze tra due numeri in CA2 vengono trasformate in somme applicando la regola

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

ovvero

$$A - B = A + CA2(B)$$

- ◆ Per assicurarsi della correttezza del risultato di un'operazione di **somma** in CA2 bisogna verificare l'assenza di overflow:
 - non si ha overflow se gli operandi hanno segno discorde
 - si ha overflow se gli operandi hanno segno concorde e il segno del risultato è discorde con essi

Operazioni aritmetiche

Complemento a 2

◆ Esempi Si considerino le seguenti operazioni tra numeri rappresentati in CA2 su 4 bit:

$$\begin{array}{r} 1) \quad 0011 + (+3_{10}) \\ \quad 0101 = (+5_{10}) \\ \hline \quad 1000 \\ \quad \swarrow \\ \quad \text{overflow} \end{array}$$

Il segno del risultato non è lo stesso degli addendi

$$\begin{array}{r} 2) \quad 0101 + (+5_{10}) \\ \quad 1100 = (-4_{10}) \\ \hline \quad \textcircled{1}0001 \\ \quad \swarrow \\ \quad \text{carry : scartato} \end{array}$$

Segno degli addendi discorde: non ci può essere overflow, carry scartato

Operazioni aritmetiche

Complemento a 2

- ◆ Si ricordi che gli operandi devono sempre essere rappresentati con lo stesso numero di cifre
- ◆ Nell'ipotesi di avere un valore N in CA2 su n cifre (segno incluso) e di volerne ricavare la rappresentazione, sempre in CA2, su m bit ($m > n$), si attua l'*estensione del segno*:
 - si replica l'MSB negli $(m - n)$ bit più a sinistra
- ◆ Esempi (da 4 a 8 bit)
 - $0\ 010 \rightarrow 0\ 0000010$
 - $1\ 011 \rightarrow 1\ 1111101$

Operazioni aritmetiche

Complemento a 2

◆ Nell'ipotesi di sommare n valori in CA2 si possono trascurare eventuali overflow intermedi se si è sicuri che il risultato finale possa ancora essere rappresentato sul numero di bit assegnato per la rappresentazione

◆ Esempio Si esegua la somma dei seguenti valori rappresentati in

CA2 su 5 bit: $+9_{10}$ $+8_{10}$ -3_{10}

01001+ $(+9_{10})$

overflow 01000 = $(+8_{10})$

ignorato → 10001+

11101 = (-3_{10})

scarto il → 101110 $(+14_{10})$

riporto

← risultato corretto

Esercizi

◆ Esercizio 1 Si eseguano le seguenti operazioni in binario puro:

a) $01010 +$
 $10010 =$

b) $01111 +$
 $00001 =$

c) $11011 +$
 $01001 =$

d) $10000 -$
 $00010 =$

e) $11010 -$
 $10101 =$

f) $10010 -$
 $01111 =$

g) $10101 -$
 $10101 =$

h) $00011 +$
 $00011 =$

i) $01000 -$
 $00111 =$

Esercizi

$$\begin{array}{r} \text{a)} \quad 01010+ \\ \quad 10010= \\ \hline \quad 11100 \end{array}$$

$$\begin{array}{r} \text{b)} \quad 01111+ \\ \quad 00001= \\ \hline \quad 10000 \end{array}$$

$$\begin{array}{r} \text{c)} \quad 11011+ \\ \quad 01001= \\ \hline 100100 \end{array}$$

$$\begin{array}{r} \text{d)} \quad 10000- \\ \quad 00010= \\ \hline \quad 01110 \end{array}$$

$$\begin{array}{r} \text{e)} \quad 11010- \\ \quad 10101= \\ \hline \quad 00101 \end{array}$$

$$\begin{array}{r} \text{f)} \quad 10010- \\ \quad 01111= \\ \hline \quad 00011 \end{array}$$

$$\begin{array}{r} \text{g)} \quad 10101- \\ \quad 10101= \\ \hline \quad 00000 \end{array}$$

$$\begin{array}{r} \text{h)} \quad 00011+ \\ \quad 00011= \\ \hline \quad 00110 \end{array}$$

$$\begin{array}{r} \quad 01000- \\ \quad 00111= \\ \hline \quad 00001 \end{array}$$

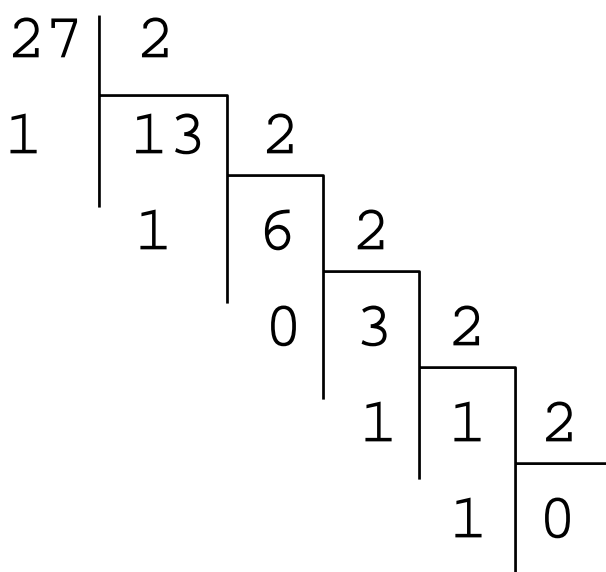
i)
II.27

Esercizi

◆ **Esercizio 2** Per le seguenti rappresentazioni in base 10 si determinino le rappresentazioni in MS e CA2 su 6 bit:

- a) 27 b) -10 c) -3
d) 0 e) 14 f) -17

a) Bisogna ricavare la rappresentazione binaria di 27_{10}



Esercizi

$$27_{10} = 11011_2$$

essendo il valore positivo le rappresentazioni in MS e CA2 coincidono e su 6 bit si ottiene:

$$+27_{10} = 011011_{\text{MS}} = 011011_{\text{CA2}}$$

b) Bisogna ricavare la rappresentazione binaria di 10_{10} che su 5 bit è 01010_2 ; dopodiché:

– in MS è sufficiente aggiungere il segno meno come $\text{MSB} = 1$

$$101010_{\text{MS}}$$

– in CA2 bisogna prima rappresentare $+10$: 001010_{CA2} e poi calcolare il CA2 di questo valore

$$110110_{\text{CA2}}$$

Esercizi

c) Bisogna ricavare la rappresentazione binaria di 3_{10} che su 5 bit è 00011_2 ; dopodiché:

- in MS è sufficiente aggiungere il segno meno come $MSB = 1$

$$100011_{MS}$$

- in CA2 bisogna prima rappresentare $+3$: 000011_{CA2} e poi calcolare il CA2 di questo valore

$$111101_{CA2}$$

d) In MS la rappresentazione dello 0 è doppia:

$$000000_{MS} \text{ e } 100000_{MS}$$

mentre in CA2 è unica: 000000_{CA2}

Esercizi

e) Bisogna ricavare la rappresentazione binaria di 14_{10} che su 6 bit è 001110_2 ; essendo il valore positivo, le rappresentazioni in MS e CA2 coincidono:

$$001110_{\text{MS}} = 001110_{\text{CA2}}$$

f) Bisogna ricavare la rappr. binaria di 17_{10} che su 5 bit è 10001_2 ; poi:

– in MS è sufficiente aggiungere il segno meno come $\text{MSB} = 1$

$$110001_{\text{MS}}$$

– in CA2 bisogna prima rappresentare $+17$: 010001_{CA2} e poi calcolare il CA2 di questo valore

$$101111_{\text{CA2}}$$

Esercizi

◆ **Esercizio 3** Si eseguano le seguenti operazioni rappresentando i numeri su 5 bit prima in MS e poi in CA2; si verifichi la correttezza dei risultati:

a) $5+15$

b) $4+7$

c) $7-10$

d) $-7-5$

e) $9-19$

f) $-10-15$

a) MS e CA2 coincidono e in entrambi i casi si verifica overflow

$$00101 + (+5)$$

$$\underline{01111} = (+15)$$

$$10100$$

↑
overflow

Esercizi

b) MS e CA2 coincidono e in entrambi i casi il risultato è valido e vale:

$$\begin{array}{r} 00100 + (+4) \\ \underline{00111} = (+7) \\ 01011 \end{array}$$

c) MS: l'operazione 7-10 viene vista come: $7+(-10)$

Addendi di segno discorde, modulo del secondo > modulo del primo \Rightarrow risultato negativo (MSB = 1)

modulo = differenza tra i moduli dei due addendi: $1010 - (10)$

$$\begin{array}{r} \underline{0111} = (7) \\ 0011 \end{array}$$

risultato finale: 10011_{MS}

Esercizi

c-*bis*) CA2: l'operazione $7-10$ diventa un'addizione $7+(-10)$, come nel caso precedente, salvo che -10 si calcola con il CA2:

$$\begin{array}{r} 00111 + \quad (+7) \\ 10110 = \quad (-10) \\ \hline 11101 \end{array}$$

d) MS: l'operazione da eseguire è la somma di due numeri negativi, ovviamente, anche il risultato sarà negativo:

$$\begin{array}{r} 10111 + \quad (-7) \\ 10101 = \quad (-5) \\ \hline 11100 \end{array}$$

Esercizi

d-bis) CA2: l'operazione da eseguire è una somma tra CA2(7)+CA2(5):

$$\begin{array}{r} 11001 + (-7) \\ 11011 = (-5) \\ \hline \textcircled{1}10100 \end{array}$$

↑
carry scartato

e) L'operazione non è eseguibile né in MS né in CA2 perché 19_{10} non è rappresentabile su 5 bit

Esercizi

f) MS: l'operazione da eseguire è la somma di due numeri negativi, quindi, anche il risultato deve essere negativo:

$$\begin{array}{r} 11010 + (-10) \\ 11111 = (-15) \\ \hline \text{overflow} \swarrow \textcircled{1}11001 \end{array}$$

in questo caso si è verificato overflow perché c'è stato un riporto dai bit che precedono il MSB

Esercizi

f-bis) CA2: l'operazione da eseguire è la somma tra CA2(-10) e CA2(-15):

$$\begin{array}{r} 10110 + (-10) \\ 10001 = (-15) \\ \hline \textcircled{1}00111 \\ \swarrow \\ \text{overflow} \end{array}$$

si è verificato overflow perché il segno del risultato non è concorde con quello dei due addendi

Esercizi

◆ Esercizio 4 Date le seguenti coppie di rappresentazioni considerarle prima in MS e poi in CA2; per ciascuna coppia si determini qual è il maggiore tra i valori rappresentati:

a) 01001 <?> 10001

b) 10110 <?> 11010

c) 101 <?> 11101

d) 11111 <?> 10001

a) In entrambi i casi è maggiore il valore di sinistra essendo positivo, mentre il valore di destra è negativo

Esercizi

b) MS: la rappresentazione di sinistra vale -6_{10} mentre quella di destra -10_{10} , quindi, il valore di sinistra è maggiore

b-bis) CA2: la rappresentazione di sinistra vale -10_{10} mentre quella di destra -6_{10} , quindi, il valore di destra è maggiore

c) MS: prima di eseguire il confronto bisogna rappresentare i due valori sullo stesso numero di cifre, quindi, si deve confrontare

$$10001_{MS} < ? > 11101_{MS}$$

da cui risulta che il valore di sinistra è maggiore

Esercizi

c-bis) CA2: prima di eseguire il confronto bisogna rappresentare i due valori sullo stesso numero di cifre, quindi, si deve confrontare

$$11101_{CA2} < ? > 11101_{CA2}$$

da cui risulta che i due valori rappresentati sono uguali

d) MS: la rappresentazione di sinistra vale -15_{10} mentre quella di destra -1_{10} , quindi, il valore di destra è maggiore

d) CA2: la rappresentazione di sinistra vale -1_{10} mentre quella di destra -15_{10} , quindi, il valore di sinistra è maggiore

Operazione di scalamento (shift)

- ◆ Dalla definizione di sistema numerico:

$$N = d_{n-1} \cdot r^{n-1} + \dots + d_0 \cdot r^0$$

- si vede che lo scalamento verso sinistra di tutte le cifre, con l'inserzione di uno zero nella posizione di peso più basso rimasta libera, equivale a moltiplicare il numero per la base
- ◆ Per contro, scalando tutte le cifre verso destra si esegue una divisione del numero per la base

Operazione di scalamento (shift)

◆ Dato il valore 26_{10} rappresentato in binario puro dalla sequenza di bit 11010_2 , si ha:

– *shift a sinistra* di una posizione:

$$11010\mathbf{0}_2 = 52_{10}$$

⇒ equivale a moltiplicare per 2

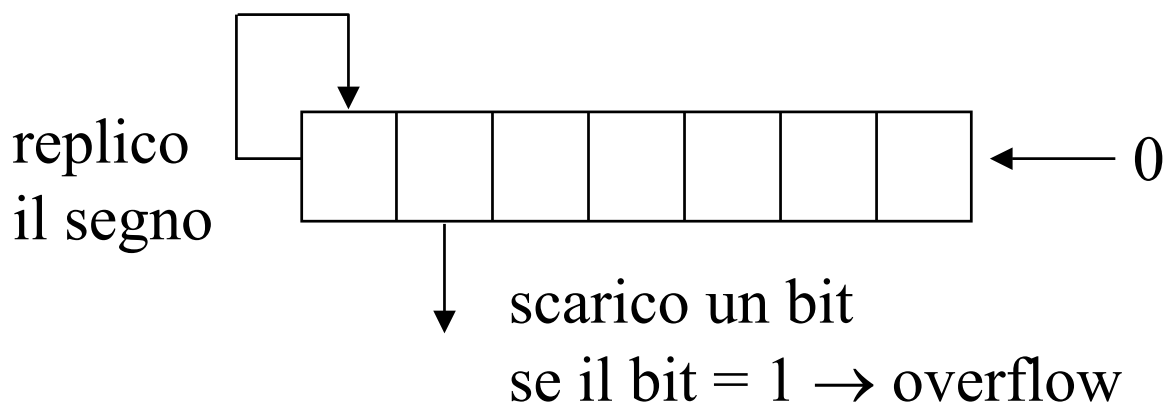
– *shift a destra* di una posizione:

$$1101_2 = 13_{10}$$

⇒ equivale a fare una divisione intera per 2

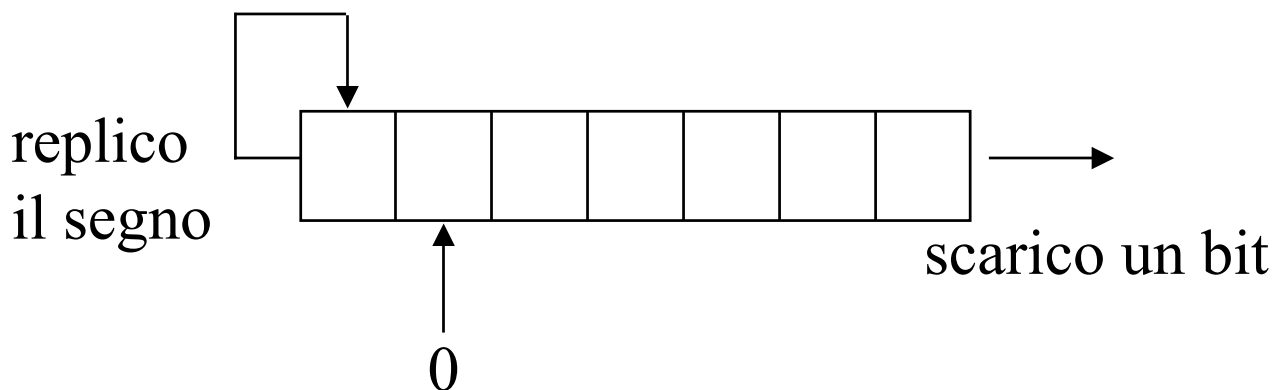
Moltiplicazioni e divisioni in MS

- ◆ La moltiplicazione per 2 in MS si esegue:
 - scalando a sinistra tutte le cifre del modulo
 - aggiungendo uno zero nella posizione di destra rimasta libera
 - replicando il segno
 - controllando che non ci sia overflow che si verifica quando scarico un 1
- ◆ Lo schema è il seguente:



Moltiplicazioni e divisioni in MS

- ◆ La divisione per 2 in MS si esegue:
 - scalando a destra tutte le cifre del modulo
 - aggiungendo uno zero nella posizione di sinistra rimasta libera
 - replicando il segno
 - il bit scaricato è il resto della divisione
- ◆ Lo schema è il seguente:



Moltiplicazioni e divisioni in MS

◆ Esempi

Si moltiplichino per 2 i seguenti valori rappresentati in MS:

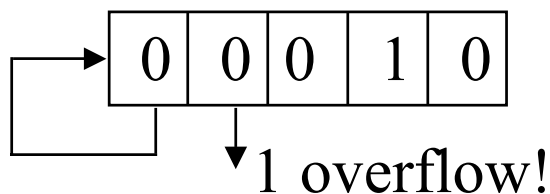
a) 001001 b) 01001 c) 10010

Per a) e c) non c'è overflow:

001001 (+9₁₀) → 010010 (+18₁₀)

10010 (-2₁₀) → 10100 (-4₁₀)

Per b) si verifica overflow in quanto verrebbe scaricato un 1:



Moltiplicazioni e divisioni in MS

◆ Esempi

Si dividano per 2 i seguenti valori rappresentati in MS:

a) 001001 b) 01000 c) 10010

Per a) si scarta il LSB

$$001001 (+9_{10}) \rightarrow 000100 (+4_{10})$$

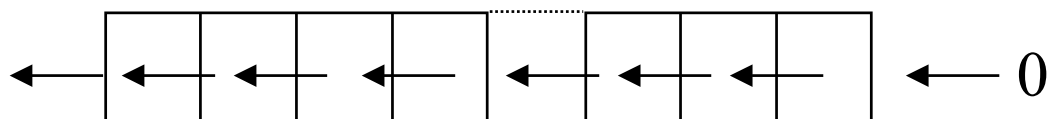
Per b) e c) il bit scaricato è 0

$$01000 (+8_{10}) \rightarrow 00100 (+4_{10})$$

$$10010 (-2_{10}) \rightarrow 10001 (-1_{10})$$

Moltiplicazioni e divisioni in CA2

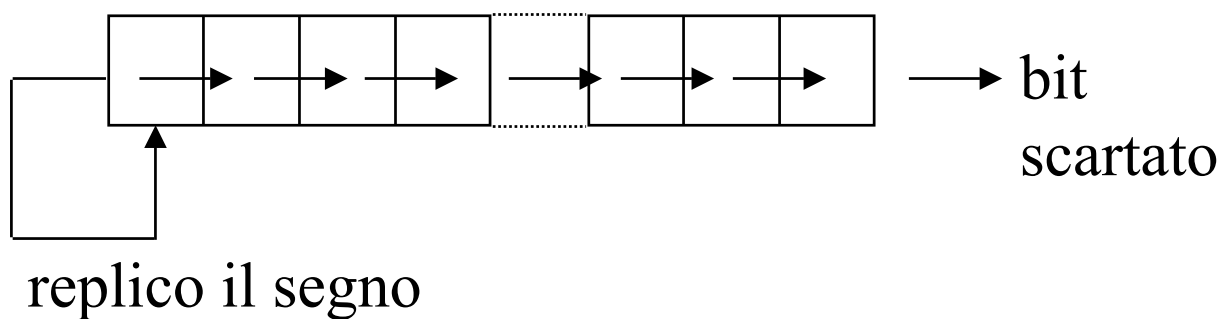
- ◆ La moltiplicazione per 2 in CA2 si esegue:
 - scalando a sinistra tutte le cifre, MSB compreso
 - aggiungendo uno zero nella posizione di destra rimasta libera
 - controllando che non ci sia overflow (che si verifica quando il bit più significativo cambia)
- ◆ Lo schema è il seguente:



se il nuovo MSB è diverso dal precedente c'è overflow

Moltiplicazioni e divisioni in CA2

- ◆ La divisione per 2 in CA2 si esegue:
 - scalando a destra tutte le cifre (MSB compreso)
 - replicando il segno
- ◆ Lo schema è il seguente:



Moltiplicazioni e divisioni in CA2

◆ Esempi

Si moltiplichino per due i seguenti valori rappresentati in CA2:

a) 001100 b) 01001 c) 10010

a) il bit di segno, scalando verso sinistra, rimane inalterato

$$001100 (+12_{10}) \rightarrow 011000 (+24_{10})$$

b) e c) si verifica overflow in quanto lo scalamiento a sinistra comporta un cambio di segno

$$01001 (+9_{10}) \rightarrow 10010 \text{ overflow!}$$

$$10010 (-14_{10}) \rightarrow 00100 \text{ overflow!}$$

Moltiplicazioni e divisioni in CA2

◆ Esempi

Si dividano per due i seguenti valori rappresentati in CA2:

a) 001100 b) 01001 c) 10010

Il bit di segno viene replicato:

a) 001100 ($+12_{10}$) \rightarrow 00110 ($+6_{10}$)

viene scartato lo 0 del LSB

b) 01001 ($+9_{10}$) \rightarrow 00100 ($+4_{10}$)

viene scartato l'1 del LSB

c) 10010 (-14_{10}) \rightarrow 11001 (-7_{10})

viene scartato lo 0 del LSB