Prof. Mariacarla Staffa
a.a. 2022/2023

Laboratorio di Architettura Degli Elaboratori

Rappresentazione dei Numeri

Numeri con parole di lunghezza fissa

- Supponiamo di avere una macchina che opera con parole di 16 bit, il numero 9 sarà quindi rappresentato come: 0000 0000 0000 1001
- Operazioni come l'addizione o la moltiplicazione possono produrre numeri troppo grandi per essere rappresentati. In questo caso parleremo di trabocco o ***overflow***
- Supponiamo di nuovo di avere una parola di 16 bit e supponiamo di voler elevare 1024 al quadrato. Questo produrrà un trabocco, infatti:

$$1024^2 = 1024 \cdot 1024 = 2^{10} \cdot 2^{10} = 2^{20} > 2^{16} - 1$$

Somma binaria

Nel sistema di numerazione in base 2 esistono due soli simboli: 0 e 1 e quindi quando si effettua l'operazione 1 + 1, il risultato è 0 con il riporto di 1, cioè 10 (da leggere uno, zero e non dieci).

Le regole per effettuare l'operazione di somma di due cifre binarie sono riassunte di seguito:

- 0 + 0 = 0
- 0 + 1 = 1
- 1 + 0 = 1
- 1 + 1 = 0 con riporto di 1

Somma in binario

L'operazione di somma in binario è algoritmicamente analoga a quella decimale con la differenza che il riporto si ha quando si eccede 1 (invece che 9)

Somma binaria

La somma dei due numeri interi 10001 e 11011 è pari a:

```
  10001 +
  11011 =
-----------
101100
```

Binary Addition Examples

- Add the following 4-bit binary numbers

$$\begin{array}{r} 1001 \\ +\ 0101 \\ \hline \end{array}$$

- Add the following 4-bit binary numbers

$$\begin{array}{r} 1011 \\ +\ 0110 \\ \hline \end{array}$$

Binary Addition Examples

- Add the following 4-bit binary numbers

```
   1
 1001
+0101
------
1110
```

- Add the following 4-bit binary numbers

```
  111
 1011
+0110
------
10001
```

Binary Addition Examples

- Add the following 4-bit binary numbers

```
   1
 1001
+0101
-----
 1110
```

- Add the following 4-bit binary numbers

```
  111
 1011
+0110
-----
10001
```

Overflow!

Moltiplicazione binaria

Le regole per la moltiplicazione sono:

- 0 * 0 = 0
- 0 * 1 = 0
- 1 * 0 = 0
- 1 * 1 = 1

Moltiplicazione in binario

Anche la moltiplicazione in binario è analoga a quella decimale

```
     0101 x
     0011 =
----------
     0101
    0101=
   0000==
  0000===
----------
  0001111
```

Esercizi

1. Moltiplicazione binaria 1001 * 0101
2. Moltiplicazione tra 63 e 30 convertiti in base 2, trasformare nuovamente il risultato poi in decimale

Soluzioni

1. Moltiplicazione binaria 1001 * 0101

```
   1001 *
   0101
--------
   1001
  0000-
 1001–
0000---
-----------
0101101
```

Soluzioni

2. Moltiplicazione tra 63 e 30 convertiti in base 2, trasformare nuovamente il risultato poi in decimale

$63 = 111111_2$ $30 = 11110_2$

111111 X
11110 =

1233332100 Riporti
1111110 +
11111100 +
111111000 +
1111110000 =

11101100010

$(11101100010)_2 = 2+32+64+256+512+1024 = 1890_{10}$

Rappresentazione di interi

- Occupiamoci ora della rappresentazione di numeri interi …, -2,-1,0,1,2,… mediante parole di lunghezza fissata
- Chiaramente dobbiamo codificarne non solo il valore assoluto ma anche il segno
- Le principali rappresentazioni di numeri interi sono:
  - Rappresentazione con modulo e segno
  - Complemento alla base (complemento a due)

Rappresentazione con Modulo e segno

- Si supponga che le parole siano di lunghezza m e la base sia b
- In tale rappresentazione la cifra più significativa di una parola rappresenta il segno. Per convenzione 0 rappresenta il segno più, 1 rappresenta il segno meno
- Le rimanenti m-1 cifre sono usate per rappresentare il valore assoluto di un intero
- Ad esempio si considerino parole binarie di lunghezza 4
  - 0100 → +100 → 4
  - 1011 → -011 →-3
  - Il più grande numero rappresentabile è 0111 (ovvero 7)
  - Il più piccolo numero rappresentabile è 1111 (ovvero -7)
  - Lo zero ha due possibili rappresentazioni: 0000 e 1000

Rappresentazione con Modulo e segno

ESEMPI:

- Es 1 su 4 bit

$1111 \rightarrow -111_2 \rightarrow -7_{10}$

- Es 2 su 4 bit

$0100 \rightarrow +100_2 \rightarrow 4_{10}$

- Es 3 su 4 bit

$1011 \rightarrow -011_2 \rightarrow -3_{10}$

Rappresentazione Modulo e segno

Poiché m-1 cifre sono utilizzate per rappresentare il valore assoluto, il range di numeri codificabili è

$$-(b^{m-1}-1),\ldots,0,\ldots,b^{m-1}-1$$

Svantaggio: nelle operazioni di addizione o sottrazione occorre controllare il segno e i valori assoluti dei due operandi per determinare il segno del risultato:

- 0010 + 0001 sono entrambi positivi quindi il segno sarà positivo → 0011
- 0101 + 1010 il primo è positivo mentre il secondo è negativo, il valore assoluto del primo è maggiore del valore assoluto del secondo quindi il segno sarà positivo → 0011
- 1100+0011 il primo è negativo mentre il secondo è positivo, il valore assoluto del primo è maggiore di quello del secondo, quindi il segno sarà negativo → 1001
- 1011-1010 Entrambi sono negativi ma il valore assoluto del secondo è minore di quello del primo. Quindi occorre sottrarre 010 da 011 cambiando il bit del segno → 1001

Rappresentazione Modulo e Segno

- Algoritmo per le operazioni di somma e sottrazione complesso
  1. Confrontare i bit di segno dei due numeri
  2. Se i bit di segno sono uguali
     - Il risultato ha lo stesso segno ed il valore è dato dalla somma dei valori assoluti dei due numeri
  3. Se i bit di segno sono uguali
     - Si confrontano i valori assoluti dei due numeri
     - Il risultato ha il segno del numero maggiore in valore assoluto ed ha valore pari alla differenza fra i due numeri

N=3bit
000 →+0
001 →+1
010 →+2
011 →+3
100 →-0
101 →-1
110 →-2
111 →-3

Somma (-2)+(-1) = -3
110+101=011 con riporto 1

Somma (-3)+(+2) = -1
111+010=001 con riporto 1

Rappresentazione in Complemento alla base

- Per semplificare le operazioni aritmetiche sui numeri interi si adotta una rappresentazione dei numeri detta ***in complemento alla base***.
- ***Definizione di Complemento alla Base:***
  Dato un numero x rappresentato con m cifre, si definisce il complemento alla ***base*** (b) di x, C(x), come segue:

$$C(x) = b^m - x$$

x	X_2	X_{10}
7	0111	7
6	0110	6
5	0101	5
4	0100	4
3	0011	3
2	0010	2
1	0001	1
0	0000	0
-1	1111	15
-2	1110	14
-3	1101	13
-4	1100	12
-5	1011	11
-6	1010	10
-7	1001	9
-8	1000	8

Esempio: con m=4 e $x=0011_2 = 3_{10}$

- In complemento alla base $C(x) = b^4-3=16-3 = 13_{10}$

Ma come si fa a rappresentare un numero negativo in complemento a due?

Complemento a 2

- La rappresentazione è analoga a quella unsigned con la differenza che il bit più significativo corrisponde a -2^{m-1} invece che 2^{m-1}

- Lo zero ha una unica rappresentazione 0…0
- Il range di rappresentazione è $[-2^{m-1}, 2^{m-1} -1]$
- Essendo -2^{m-1} in valore assoluto il «peso» più grande, se un numero inizia con 1 allora è negativo altrimenti è positivo

Confronti tra rappresentazioni un/signed

Unsigned

x	X_2	X_{10}
15	1111	15
14	1110	14
13	1101	13
12	1100	12
11	1011	11
10	1010	10
9	1001	9
8	1000	8
7	0111	7
6	0110	6
5	0101	5
4	0100	4
3	0011	3
2	0010	2
1	0001	1
0	0000	0

Segno e Modulo

x	X_2	X_{10}
7	0111	7
6	0110	6
5	0101	5
4	0100	4
3	0011	3
2	0010	2
1	0001	1
0	0000;1000	0;8
-1	1001	9
-2	1010	10
-3	1011	11
-4	1100	12
-5	1101	13
-6	1110	14
-7	1111	15

Complemento alla base

x	X_2	X_{10}
7	0111	7
6	0110	6
5	0101	5
4	0100	4
3	0011	3
2	0010	2
1	0001	1
0	0000	0
-1	1111	15
-2	1110	14
-3	1101	13
-4	1100	12
-5	1011	11
-6	1010	10
-7	1001	9
-8	1000	8

Complemento a 2

- Massimo valore rappresentabile: $2^{m-1} - 1$ → 01…1
- Minimo valore rappresentabile: -2^{m-1} → 10…0
- Il numero -1 è scritto come: 11…1

- Nota: i numeri positivi hanno lo 0 nella posizione più significativa , mentre i numeri negativi hanno 1 nella posizione più significativa

  → Quindi il bit più significativo può essere visto come il segno ☺

Interpretare i numeri espressi in Complemento a 2

- Nessun problema per i numeri positivi, ma per quelli negativi?

Il bit più significativo ad 1 ci suggerisce che il numero è negativo

Per Esempio: $1101_{c2} = -3$

L'1 ci dice che è negativo ma la restante parte 101 = 5 non ha senso!!!

La sequenza presa per intero 1101 = 13 non ha senso!!!

Così come per il sistema posizionale utilizziamo la rappresentazione per calcolare il valore nella sequenza espressa in complemento a 2 :

$$1101_{c2} = 1x-2^3 + 1x2^2 + 0x2^1 + 1x2^0 = -8+4+0+1 = -3$$

x	X_2	X_{10}
7	0111	7
6	0110	6
5	0101	5
4	0100	4
3	0011	3
2	0010	2
1	0001	1
0	0000	0
-1	1111	15
-2	1110	14
-3	1101	13
-4	1100	12
-5	1011	11
-6	1010	10
-7	1001	9
-8	1000	8

-2^{m-1}	2^{m-2}							2^1	2^0

"Taking the Two's Complement"

Per ottenere la sua rappresentazione negativa in complemento a due di un numero si utilizza la tecnica del

"**Taking the Two's complement**"

flips the sign of a two's complement number

- **Method:**
  1. Invert the bits
  2. Add 1
- Example: Flip the sign of $3_{10} = 0011_2$

"Taking the Two's Complement"

- "Taking the Two's complement" **flips the sign** of a two's complement number
- **Method:**
  1. Invert the bits
  2. Add 1
- **Example:** Flip the sign of $3_{10} = 0011_2$
  1. **1100**
  2. **+ 1**

  $1101_{c2} = -3_{10}$

Two's Complement Examples

- Take the two's complement of $6_{10} = 0110_2$

- What is the decimal value of the two's complement number 1001_2?

Two's Complement Examples (calcolo opposti)

- Take the two's complement of $6_{10} = 0110_2$
  1. 1001
  2. \+ 1

     1010_{c2}

  Infatti… = $-1x2^3+0x2^2+1x2^1+0x2^0 = -6_{10}$
- What is the decimal value of the two's complement number 1001_{c2}?

  $1001_{c2} = -1x2^3+0x2^2+0x2^1+1x2^0 = -8+1 = -7_{10}$

  Infatti…
  1. 0110
  2. \+ 1

     $0111_2 = 7_{10}$, so $1001_{c2} = -7_{10}$

Ricapitolando…

- Numeri positivi. Nel caso dei numeri binari positivi la rappresentazione in complemento a 2 è uguale alla rappresentazione segno-grandezza(segno modulo).
- Numeri negativi. Nel caso dei numeri binari negativi la rappresentazione in complemento a 2 è uguale all'inversione di ogni cifra (bit) del numero binario in valore assoluto (complemento a 1) a cui viene addizionato il numero binario uno (00000001)2
- Come si riconosce un numero positivo da uno negativo?
  - Positivo ⇒ bit più significativo 0
  - Negativo ⇒ bit più significativo 1
- Su N Bit si possono rappresentare numeri nella fascia [-2n-1 , 2n-1 -1]

IMPORTANTE!!!

- La rappresentazione in complemento a 2 consente di effettuare l'operazione aritmetica dell'addizione tra due numeri binari indipendentemente dal segno dei due numeri.
- E' sufficiente sommare i due numeri in complemento a 2 ignorando il riporto.

Complemento a 2

- Per complementare un numero occorre invertire ogni cifra e sommare 1
  - Rappresentare il numero -2 e -7 con 4 bit

    2=0010 → -2= 1101 +0001= 1110

    7=0111 → -7= 1000 +0001= 1001

  - Attenzione! questo non vale per -2^{m-1} il cui complemento non è rappresentabile con m bit (i numeri positivi arrivano a 2^{m-1} -1):
  - -2^3 =1000 → 0111 +0001= 1000

- La somma avviene come per i numeri unsigned

  $-2+1 = 1110_{c2} + 0001_{c2} = 1111_{c2} = 1x-2^3+1x2^2+1x2^1+1x2^0 = -8+4+2+1 = -1$

  -7+7 = 1001 + 0111= 0000 = 0 (con riporto 1) → il 5° bit è scartato perché in overflow, in questo caso il risultato è corretto

Two's Complement Addition

- Add 6 + (-6) using two's complement numbers

$$\begin{array}{r} 0110 \\ +\ 1010 \\ \hline \end{array}$$

- Add -2 + 3 using two's complement numbers

$$\begin{array}{r} 1110 \\ +\ 0011 \\ \hline \end{array}$$

Two's Complement Addition

- Add 6 + (-6) using two's complement numbers

```
  111
   0110
+  1010
------
 10000
```

- Add -2 + 3 using two's complement numbers

```
  111
   1110
+  0011
------
 10001
```

Subtracting Two's complement Numbers

- Calcolare le seguenti sottrazioni usando una rappresentazione in complemento a 2 su 4-bit :

1. 5_{10} -3_{10}

2. 3_{10} – 5_{10}

Subtracting Two's complement Numbers

- Calcolare le seguenti sottrazioni usando una rappresentazione in complemento a 2 su 4-bit :

1. **$5_{10} - 3_{10}$**

5_{10} è positivo = $0101_2 = 0101_{c2}$

-3 è negativo: rappresentazione in complemento a 2:

Se $3_{10} = 0011_2$ il suo complemento a 2 è (invertiamo i bit e aggiungiamo 1): $-3_{10} = 1101_{c2}$

→ **$5_{10} - 3_{10} = 0101_{c2} + 1101_{c2} = \cancel{1}0010_{c2} = 2_{10}$**

1. **$3_{10} - 5_{10}$**

3_{10} è positivo = $0011_2 = 0011_{c2}$

-5 è negativo: rappresentazione in complemento a 2:

Se $5_{10} = 0101_2$ il suo complemento a 2 è (invertiamo i bit e aggiungiamo 1): $-5_{10} = 1011_{c2}$

→ **$3_{10} - 5_{10} = 0011_{c2} + 1011_{c2} = 1110_{c2} = -2_{10}$**

Overflow nel complemento a 2

- Sommare un numero negativo e uno positivo non genera overflow
- L'esempio -7+7 mostra come l'overflow *non avviene come per gli unsigned* quando ho riporto finale di 1

- L'overflow avviene quando entrambi gli operandi sono negativi (primo bit=1) o entrambi positivi (primo bit=0) e il risultato ha segno opposto

  4+5=0100+0101 =1001 =-7

- Per estendere un numero in una rappresentazione con più bit basta riprodurre a sinistra il bit più significativo

  5=0101 → 00000101

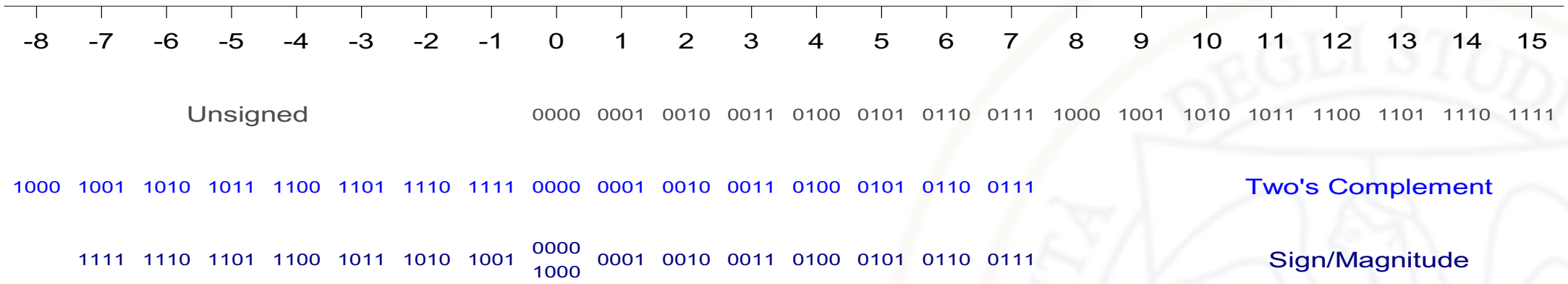
  -4=1100 → 11111100

Sign-Extension

- Sign bit copied to msb's
- Number value is same
- **Example 1:**
  - 4-bit representation of 3 = **0**011
  - 8-bit sign-extended value: **0000**0011
- **Example 2:**
  - 4-bit representation of -5 = **1**011
  - 8-bit sign-extended value: **1111**1011

Number System Comparison

For example, 4-bit representation:



Number System	Range
Unsigned	$[0, 2^N-1]$
Sign/Magnitude	$[-(2^{N-1}-1), 2^{N-1}-1]$
Two's Complement	$[-2^{N-1}, 2^{N-1}-1]$

Esercizi

- Fornire la rappresentazione binaria in complemento a 2 ad 8 bit del numero -15
- Fornire la rappresentazione binaria in complemento a 2 ad 8 bit del numero -109
- Eseguire la somma dei numeri 5 e -32 espressa in completamento a 2 ad 8 bit