Correzione degli esercizi

- Convertire i seguenti numeri binari in formato decimale:
  - **11,01** → 3 +1/4 = 13/4 = 3.25
  - **101,111** → 5 + 7/8 = 47/8 = 5.87
  - **10,1** → 2.5
- Esprimere i seguenti valori in notazione binaria:
  - **4.5** →100,1
  - **2.75** → 10,11
- Eseguire le seguenti somme binarie:
  - **1010,001 + 1,101** →1011,110
  - **111,11 + 0,01** →1000,00

Rappresentazione degli interi

Notazione in complemento a 2

- *n* bit per la notazione
  - Nella realta' n=32
  - Per comodita' noi supponiamo **n=4**
- Numeri positivi
  - **0** si rappresenta con n zeri **0000**
  - 1 → 0001, 2 → 0010 e cosi' come gia' visto fino al massimo positivo rappresentabile 0111 → **7**
- Numeri negativi
  - -1 si rappresenta con uni **1111 → -1**
  - -2 -> 1110, -3 → 1101 fino al minimo negativo rappresentabile **1000 → -8**
- Gli interi rappresentabili **$[-2^{n-1}, 2^{n-1}-1]$**
  - Nell'esempio $[-2^{4-1},2^{4-1}-1]=[-8,7]$

Complemento a due su 3 e 4 bit

a. Using patterns of length three

Bit pattern	Value represented
011	3
010	2
001	1
000	0
111	−1
110	−2
101	−3
100	−4

b. Using patterns of length four

Bit pattern	Value represented
0111	7
0110	6
0101	5
0100	4
0011	3
0010	2
0001	1
0000	0
1111	−1
1110	−2
1101	−3
1100	−4
1011	−5
1010	−6
1001	−7
1000	−8

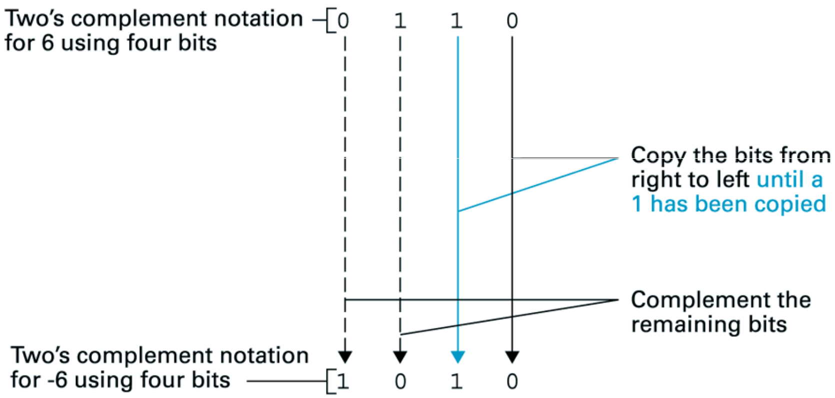
Complemento a due

- **Bit piu' a sinistra**: **segno** (**0** per positivi, **1** per negativi)
- Confrontiamo **k** e **–k**: da destra a sinistra, uguali fino **al primo 1** incluso, poi una il complemento dell'altra
- Esempio (4 bit): **2**=00**10**, **-2**=11**10**

Complemento a due: decodifica

- Se bit di segno =0 ➔ positivo, altrimenti negativo
- Se positivo, basta leggere gli altri bit
- Se negativo, scrivere gli stessi bit da destra a sinistra fino al primo 1, poi complementare, e poi leggere
- Es.: 1010 e' negativo, rappresenta 110 (6), quindi -6

Da k a -k



Metodo alternativo: codifica e decodifica

- **Intero positivo x → complemento a due su n bit**: se $\mathbf{x \le 2^{n-1}-1}$ scrivo $\mathbf{(x)_2}$, altrimenti non e' rappresentabile
  - Esempio: n=4, x=5, $\mathbf{(5)_2=0101}$, x=8>2^3-1=7
- **Intero negativo –x → complemento a due su n bit:** se $\mathbf{-x \ge -2^{n-1}}$ calcolo $\mathbf{2^n+(-x)=y}$ e scrivo $\mathbf{(y)_2}$
  - Esempio: n=4, **–x=-3** y=2^4-3=16-3=13 $\mathbf{(13)_2=1101}$
- **Compl. a due positivo** (0 = bit + significativo) →**decimale**: decodifica dal binario
  - Esempio: n=4, 0111=$(7)_2$
- **Compl. a due negativo** (1 = bit + significativo)→**decimale**: decodifica dal binario a decimale, ottengo y e poi sottraggo y-2^n
  - Esempio 1010 = $(10)_2$ 10-16=-6

Somma in complemento a due

- Come normale
- Anche per sottrazione ➔ basta avere i circuiti per somma e complemento
  - Es. (4 bit): **7-5** = **7 +(-5)** = 0111 + 1011 = 0010
  - 5 = 0101 ➔ -5 = 1011
  - L'eventuale n+1-simo **bit** generato **a sinistra** dal **riporto** deve essere **troncato**
  - Esempio 0111+1011=~~1~~0010

    7 -5 2

Esempi di somme

Problem in base ten		Problem in two's complement		Answer in base ten
3 + 2	→	0011 + 0010 = 0101	→	5
-3 + -2	→	1101 + 1110 = 1011	→	-5
7 + -5	→	0111 + 1011 = 0010	→	2

Overflow

- Si **sommano** due numeri positivi tali che **il risultato e' maggiore del massimo numero positivo rappresentabile con i bit fissati** (lo stesso per somma di due negativi)
- Si ha **overflow** se:
  - Sommando **due positivi** si ottiene **un numero che inizia per 1**: 0101+0100=**1**001, 5+4=-7
  - Sommando **due negativi** viene un **numero che inizia per 0:** 1011+1100= (1)**0**111, -5+(-4)= 7
- Nei computer c'e' overflow con **valori superiori a** 2.147.483.647= **2^{31}**

Notazione in eccesso

- **n bit** ➔ **2^n** possibili **configurazioni binarie** ordinate **da n zeri a n uni**
- Supponiamo per comodita' che **n=4**
- **0** e' rappresentato da un 1 seguito da n-1 zeri:

  0→1000
- n zeri codifica **-2^{n-1}**: -2^{4-1} = **-8→ 0000**
- n uni codifica **$2^{n-1} - 1$**: $2^{4-1}-1$= **7 →1111**
- n bit: **notazione in eccesso 2^{n-1}** rispetto al corrispondente binario
  - Es.: 4 bit, notazione in eccesso 8

Notazione in eccesso 8

Bit pattern	Value represented
1111	7
1110	6
1101	5
1100	4
1011	3
1010	2
1001	1
1000	0
0111	−1
0110	−2
0101	−3
0100	−4
0011	−5
0010	−6
0001	−7
0000	−8

Esercizi (1)

- Da complemento a 2 a base 10:
  - 00011, 01111, 11100, 11010, 00000, 10000
- Da base 10 a complemento a 2 su 8 bit:
  - 6, -6, 13, -1, 0
- Numero piu' grande e piu' piccolo per la notazione in complemento a 2 su 4, 6, 8 bit

Esercizi (2)

- Da eccesso 8 a decimale:
  - 1110, 0111, 1000,0010, 0000, 1001
- Da decimale a eccesso 8
  - 5, -5, 3, 0, 7, -8
- Numero piu' grande e piu' piccolo per la *notazione in eccesso* 8, 16, 32