Fondamenti di informatica

(introduzione)

Andrea De Lorenzo

Contatti

andrea.delorenzo@units.it

http://delorenzo.inginf.units.it

Ricevimento: (quasi) sempre, previo appuntamento

Obiettivi del corso

“Come è fatto e come funziona un computer?”

“Cosa posso fare se il computer/programma non funziona?”

“Come posso risolvere un problema usando il computer?”

Programma (sintetico)

- Rappresentazione di numeri
- Rappresentazione di testo
- Architettura degli elaboratori
- Periferiche
- Sistemi operativi
- Reti
- Sicurezza
- Fondamenti di programmazione
- Fondamenti di Java

Libri di testo

Tutorial online di Java:

https://docs.oracle.com/javase/tutorial/java/nutsandbolts/index.html

"Ma io queste cose le so già!"

Modalità di esame

- 6 appelli
- No appelli straordinari
- Possibile “pre-appello”

- Prova scritta
    - Domande a risposta chiusa
    - Domande a risposta aperta
    - Parte pratica di programmazione
- Orale facoltativo (solo se scritto sufficiente)
- Pre-appello: potrete usare il computer

ORARIO

- Lunedì
  - 14:00 - 15:30
- Martedì
  - 14:00 - 15:30; 15:45 - 16:30
- Giovedì
  - 10:00 - 11:30; 11:45 - 12:30

Socrative

https://www.socrative.com/apps.html

Rappresentazione dei numeri

Sistema posizionale

Ogni cifra va moltiplicata per la potenza di una costante detta **base**

$$\mathbf{1984}_{\mathbf{10}} = 1 \cdot 10^3 + 9 \cdot 10^2 + 8 \cdot 10^1 + 4 \cdot 10^0$$

Una **base b** ha un **alfabeto** di **b cifre**, da 0 a b-1. Se b>10, allora si usano le lettere

Conversione di base

Da base b a 10: moltiplico per le potenze di b e sommo

$$1011_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 11_{10}$$

1	**0**	**1**	**1**	**x**
$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$	=
8	**0**	**2**	**1**	**11**

Conversione di base

Da base b a base 10: moltiplico per le potenze di b e sommo

$$\mathbf{1AF_{16}} = 1 \cdot 16^2 + 10 \cdot 16^1 + 15 \cdot 16^0 = \mathbf{431_{10}}$$

1	**A(=10)**	**F(=15)**	**x**
16^2 = 256	16^1 = 16	16^0 = 1	=
256	**160**	**15**	**431**

Conversione di base

Da base 10 a base 2: divido per la base e tengo il resto fino al quoziente nullo, i resti (letti al contrario) danno il numero

$\mathbf{21}_{10}$ → 21:2 = 10(1)→ 10:2 = 5(0)→ 5:2 = 2(1)→

→ 2:2=1(0) → 1:2 = 0(1) → $\mathbf{10101}_{2}$

Conversione di base

Da base 10 a base 16: divido per la base e tengo il resto fino al quoziente nullo, i resti (letti al contrario) danno il numero

$\mathbf{672_{10}}$ → 672:16 = 42(0)→ 42:16 = 2(10)→ 2:16 = 0(2) → $\mathbf{2A0_{16}}$

Conversione di base

Da base 2 a base 16: raggruppo le cifre a gruppi di 4 e converto in esadecimale

$$\mathbf{10011011_2} \rightarrow 1001\ 1011$$

$$(1 \cdot 8 + 0 \cdot 4 + 0 \cdot 2 + 1 \cdot 1) \cdot 2^4 + (1 \cdot 8 + 0 \cdot 4 + 1 \cdot 2 + 1 \cdot 1) \cdot 2^0 = \mathbf{9B_{16}}$$

Numeri razionali

Da base 10 a base b: moltiplico per la base finchè la parte frazionaria non è nulla. La parte intera mi da il numero.

$\mathbf{0.125}_{\mathbf{10}} \rightarrow 0.125 \cdot 2 = \underline{\mathbf{0}}.250 \rightarrow 0.250 \cdot 2 = \underline{\mathbf{0}}.500 \rightarrow 0.500 \cdot 2 = \mathbf{1}.0 \rightarrow \mathbf{0.001}_{\mathbf{2}}$

	X 2	**Parte decimale**	**Parte intera**
0.125	**<u>0</u>**.250	.250	**0**
0.250	**<u>0</u>**.500	.500	**0**
0.500	**<u>1</u>**.000	.000	**1**

Numeri razionali

$\mathbf{0.1875_{10} = 0.0011_2}$

	X 2	**Parte decimale**	**Parte intera**
0.1875	<u>**0**</u>.375	.375	**0**
0.375	<u>**0**</u>.750	.750	**0**
0.75	<u>**1**</u>.500	.500	**1**
0.5	<u>**1**</u>.000	.000	**1**

0.1

Come lo rappresento in binario?

Virgola mobile

$$2^{\text{esponente}} \cdot \text{mantissa}$$

```
      8               24               lunghezza in bit
+----------+----------------------+
| Esponente|       Mantissa       |
+----------+----------------------+
31         22                     0    indice dei bit
```

Virgola mobile

120123456000000000_{10}

$1101010101100001110100101100101000001001010000000000000000_2$

(58 cifre)

00100001110101010110000111010010

$2^{33} \cdot 13984210$

Bit, byte, ...

Massimo intero rappresentabile: se ho n cifre in base b posso rappresentare al massimo il valore $\boldsymbol{b^n-1}$

Con n bit posso rappresentare al massimo 2^n valori

Bit, byte, ...

Grandezze fondamentali

b → bit (0 o 1)

B → Byte (8 bit)

kB → kilobyte (2^{10} byte)

MB → megabyte (2^{20} byte)

MB → megabyte (2^{20} byte)

GB → gigabyte (2^{30} byte)

TB → terabyte (2^{40} byte)

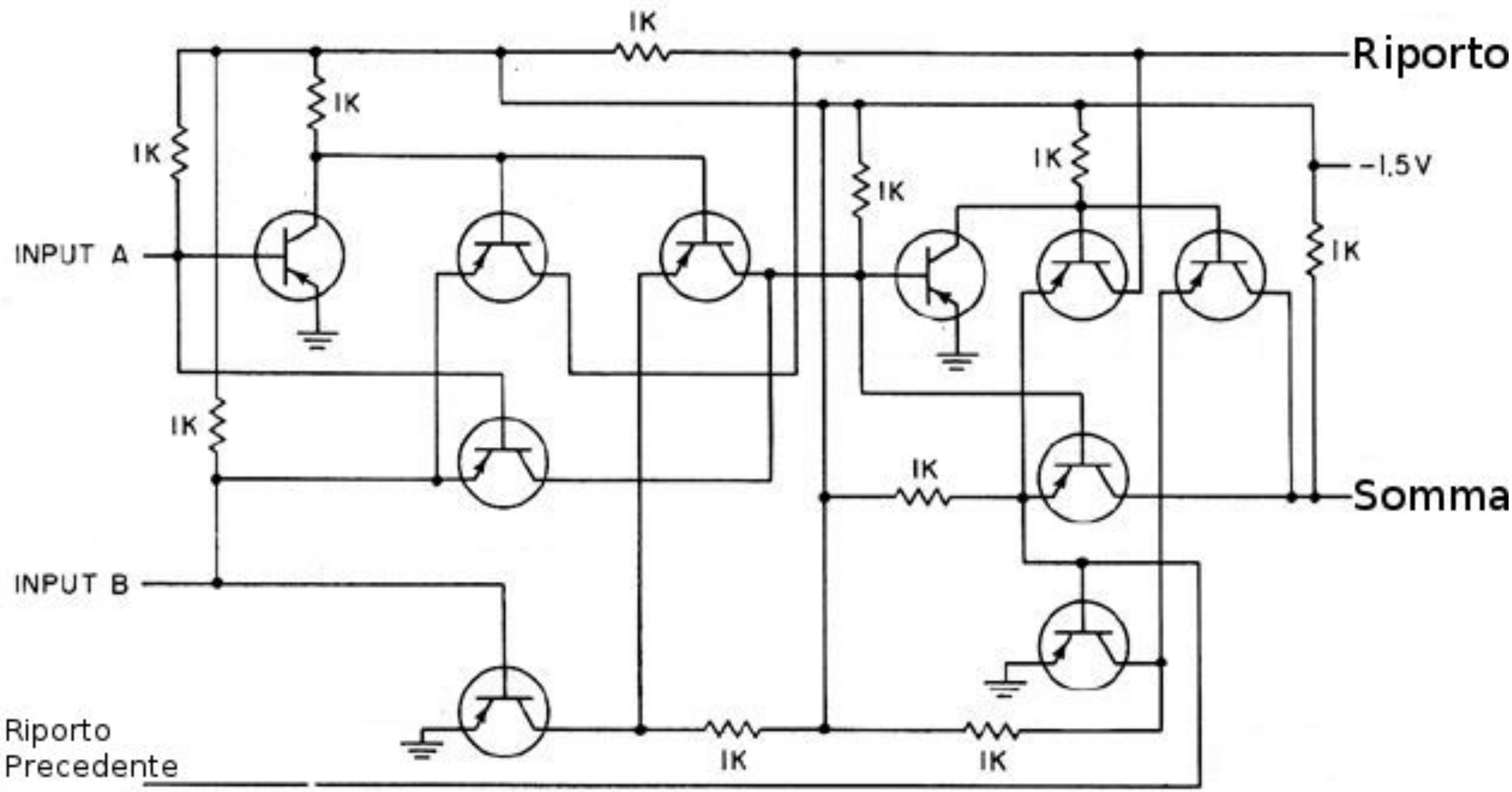
PB → petabyte (2^{50} byte)

SOMMA

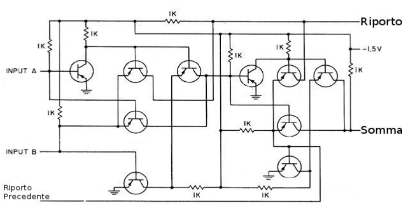
Come in base 10: si procede da **destra** a **sinistra**, riportando **l'eccedenza** sulla colonna successiva

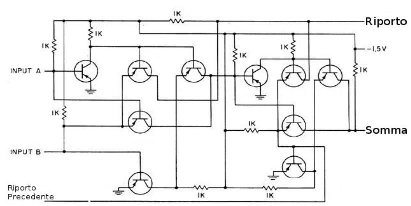
```
     11
  1000111+
  0010110=
 ---------
   1011101
```

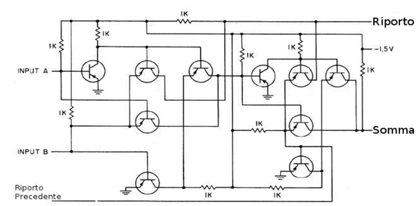
Transistor e somma

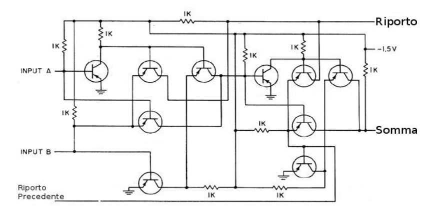


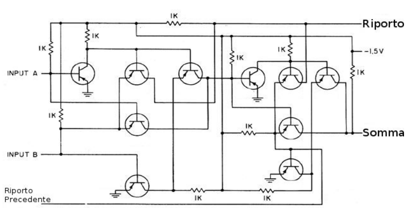
Transistor e somma

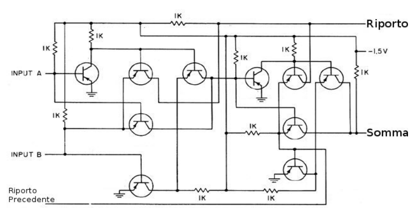


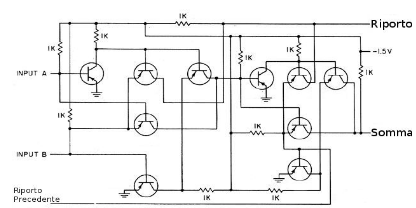


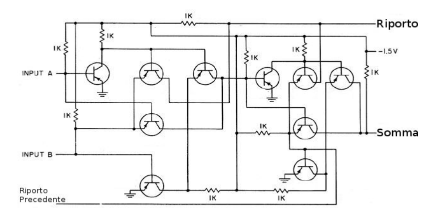












E SE LE CIFRE NON BASTANO PER RAPPRESENTARE IL NUMERO?

OVERFLOW

Numeri interi negativi

Segno: uso un bit per indicare il segno

$$0101_2 = 5_{10}$$
$$1101_2 = -5_{10}$$

Problemi:

- complicato fare operazioni
- Rappresento lo “0” in 2 modi

Numeri interi negativi

Complemento a uno: se il numero è negativo inverto i bit

$$0101_2 = 5_{10}$$
$$1010_2 = -5_{10}$$

Rappresento ancora lo “0” in 2 modi

Numeri interi negativi

Complemento a due: inverto i bit e poi sommo 1

$$0101_2 = 5_{10}$$
$$1011_2 = -5_{10}$$