Classe 3-I - Matricole 4,5
Lezione #3

Laboratorio di Informatica di Base

Ferdinando Cicalese

cicalese@dia.unisa.it

Rappresentazione dei Naturali

N = 0,1,2,...

La Notazione Posizionale (in base p)

$N_p \equiv a_n\, a_{n-1}\, a_{n-2} \ldots a_1\, a_0$

$N_p = a_n \times p^n + a_{n-1} \times p^{n-1} + \ldots + a_1 \times p + a_0$

Esempio (base 10)

$543 = 5 \times 10^2 + 4 \times 10^1 + 3 \times 10^0 = 500 + 40 + 3$

Rappresentazione dei Naturali

N = 0,1,2,…

La Notazione Additiva (numeri romani)

$N_p \equiv a_n\, a_{n-1}\, a_{n-2} \dots a_1\, a_0$

$N_p = a_n + a_{n-1} + \dots + a_1 + a_0$

Esempio I = 1, V = 5, X = 10, L = 50, C = 100, D = 500, M = 1000

DXXXII = D + X + X + X + I + I
= 500 + 10 + 10 + 10 + 1 + 1 = 532

Le Notazioni Usate in Informatica

Binaria (base 2) a_i = 0,1

Ottale (base 8) a_i = 0,1,2,3,4,5,6,7

Esadecimale (base 16) a_i = 0,1,2,…,9,A,B,C,D,E,F

Decimale	Binaria	Ottale	Esadec.
10	1010	12	A
124	1111100	174	7C

Codifica e decodifica
(Da binario a decimale e viceversa)

$N_2 = 101010$

$N_{10} = 1 \times \boxed{2^5} + 0 \times 2^4 + 1 \times \boxed{2^3} + 0 \times 2^2 + 1 \times \boxed{2^1} + 0 \times 2^0$
$= 32 + 8 + 2 = 42$

$N_2 = 11011$

$N_{10} = 2^0 + 2^1 + 2^3 + 2^4 = 1 + 2 + 8 + 16 = 27$

Codifica e decodifica
(Da decimale a binario)

$N_{10} = 51$ $N_2 = ???$

51 | 2
1 | 25 | 2
1 | 12 | 2
0 | 6 | 2
0 | 3 | 2
1 | 1 | 2
1 | 0

$N_2 = 110011$

$51 = 2x25+1 = 2x(2x12+1)+1 = 2x(2x(2x6+0)+1)+1 =$

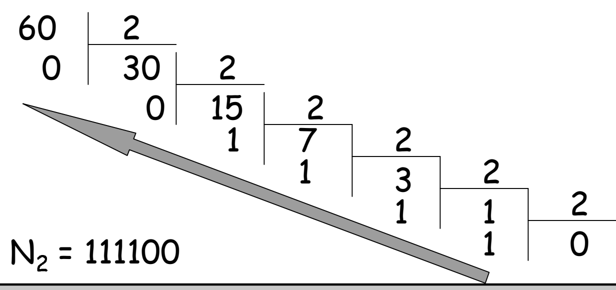
$= 2x(2x(2x(2x3+0)+0)+1)+1 =$

$= 2x(2x(2x(2x(2x1+1)+0)+0)+1)+1 = 2^5+2^4+2^1+2^0$

Codifica e decodifica

(Da decimale a binario)

$N_{10} = 60$ $N_2 = ???$



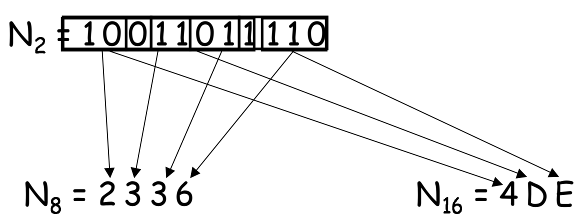
$N_2 = 111100$

$60 = 2x30+0 = 2x(2x15+0)+0 = 2x(2x(2x7+1)+0)+0 =$

$= 2x(2x(2x(2x3+1)+1)+0)+0 =$

$= 2x(2x(2x(2x(2x1+1)+1)+1)+0)+0 = 2^5+2^4+2^3+2^2$

Da Binario a Ottale/Decimale



Per trasformare un numero binario in ottale, operiamo la trasformazione considerando gruppi di 3 bit

Per trasformare un numero binario in esadecimale, operiamo la trasformazione considerando gruppi di 4 bit

Da Binario a Ottale/Decimale

$N_2 = 01101101010$

$N_8 = 1552$ $\qquad$ $N_{16} = 36A$

Per trasformare un numero binario in ottale, operiamo la trasformazione considerando gruppi di 3 bit

Per trasformare un numero binario in esadecimale, operiamo la trasformazione considerando gruppi di 4 bit

Rappresentazione degli Interi

N = 0,+1,-1,+2,-2,+3,-3,…

Come possiamo rappresentare il *segno* di un numero?

Aggiungiamo un ulteriore bit che poniamo a 1 se il numero è negativo!

Esempio

$N_{10} = +14$ $\qquad$ $N_2 = 01110$

$N_{10} = -14$ $\qquad$ $N_2 = 11110$

Rappresentazione degli Interi

$N = 0,+1,-1,+2,-2,+3,-3,\ldots$

Come possiamo rappresentare il *segno* di un numero?

Aggiungiamo u
poniamo a 1 se

Rappresentazione in Modulo e Segno

Con k bit si possono codificare tutti gli interi

$$-2^{k-1}+1 \leq N \leq 2^{k-1}-1$$

Esistono due codifiche dello 0

0000…0 1000…0

Esempio

$N_{10} = +14$ N_2

$N_{10} = -14$ N_2

Rappresentazione degli Interi

(Rappresentazione in complemento a 2)

Supponiamo di avere a disposizione k bit

La rappresentazione di –N si ottiene facendo la conversione in binario del numero 2^k-N

Esempio (con 5 bit)

$N_{10} = +14$ $N_2 = 01110$

$N_{10} = -14$ $2^k-14 = 32-14 = 18$ $N_2 = 10010$

Rappresentazione degli Interi

(Rappresentazione in complemento a 2)

Supponiamo di avere a disposizione k bit

La rappresentazione di –N
conversione in binario del

Con k bit si possono codificare tutti gli interi

$$-2^{k-1} \leq N \leq 2^{k-1}-1$$

$0000\ldots0_2 \equiv 0_{10}$

$1000\ldots0_2 \equiv -2^{k-1}{}_{10}$

Esempio (con 5 bit)

$N_{10} = +14$ $N_2 = 01110$

$N_{10} = -14$ $2^k-14 = 32-14 = 18$ $N_2 = 10010$

Rappresentazione degli Interi

(...in complemento a 2 - Operativamente)

Si complementa il numero bit a bit

Si somma 1 al al risultato così ottenuto

Esempio (con 5 bit)

$N_{10} = -14$ $N_2 = 10010$

1. Rappresentiamo 14 in binario 01110
2. Complementiamo bit a bit (0 -> 1 e 1 -> 0) 10001
3. Sommiamo 1 al numero ottenuto: 10010

Aritmetica Binaria

Somma e differenza di due numeri di k bit

Overflow

Riporto:	$\boxed{1}$			1	1	
		1	0	0	1	1	Addendo 1
		1	1	0	1	1	Addendo 2
Risultato:		0	1	1	1	0

Aritmetica Binaria

Somma e differenza *in complemento a 2* di due numeri di k bit

+6	0	0	1	1	0
+4	0	0	1	0	0
+10	0	1	0	1	0

+7	0	0	1	1	1
-10	1	0	1	1	0
-3	1	1	1	0	1

-2		1	1	1	1	0
+13		0	1	1	0	1
+11	**1**	0	1	0	1	1

-12		1	0	1	0	0
-7		1	1	0	0	1
+11	**1**	0	1	1	0	1

Aritmetica Binaria

Somma e differenza *in complemento a 2* di
due numeri di k bit

+6	0	0	1	1	0
+4	0	0	1	0	0
+10	0	1	0	1	0

+7	0	0	1	1	1
-10	1	0	1	1	0
-3	1	1	1	0	1

Non è Overflow

-2		1	1	1	1	0
+13		0	1	1	0	1
+11	**1**	0	1	0	1	1

Overflow

-12		1	0	1	0	0
-7		1	1	0	0	1
+11	**1**	0	1	1	0	1

Rappresentazione dei Numeri Reali

Numeri Razionali!!!

- Virgola fissa
- Virgola mobile (Floating Point)

Rappresentazione dei Numeri Reali

Virgola fissa - un Esempio

$2^{-1}\ 2^{-2} \dots 2^{-5}$

0 0 1 1 0 1 0 1 0 1.1 1 0 0 1

$2^{n}\ 2^{n-1} \dots \dots \dots \dots\ 2^{2}\ 2^{1}\ 2^{0}$

Per la parte decimale: $2^{-1} + 2^{-2} + 2^{-5} = 1/2 + 1/4 + 1/32$
$= .03125$

Rappresentazione dei Numeri Reali

Virgola Mobile - un Esempio

$-123.274 = -0.123274 \times 10^{3}$

mantissa caratteristica

In generale: mantissa **m** e caratteristica **n**

$$N = m \times b^{n}$$

b (la base) è fissato, e non e' (necessariamente) la base scelta per la rappresentazione dei numeri

Rappresentazione dei Numeri Reali

Virgola Mobile - Vantaggi

- Rappresentazione di numeri grandi con poche cifre
  Un miliardo : 0.1×10^{10}
- Rappresentazione di numeri piccoli con grande precisione (piu' cifre per la mantissa)

Esempio

Si provino a rappresentare in virgola mobile i numeri

$N_1 = -324563 \times 10^{20}$ $N_2 = -324323 \times 10^{-20}$

Rappresentazione dei Numeri Reali

Virgola Mobile - Caratteristiche

- densa vicino allo 0, più sparsa per numeri grandi
- se **m** è lungo l_1 e **n** è lungo l_2 allora
  $N < 2^{l_2-1} - 2^{-l_1+1}$
- In generale la cifra più significativa della mantissa è diversa da zero (numeri normalizzati)

$N = 0.0123 \times 10^3$ **$N = 0.123 \times 10^2$**

- è un po' più complesso eseguire operazioni

Operazioni Logiche

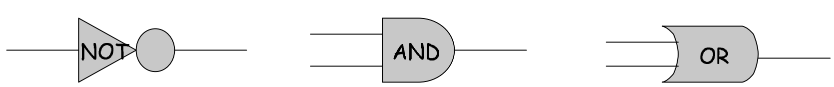
(Algebra di Boole)

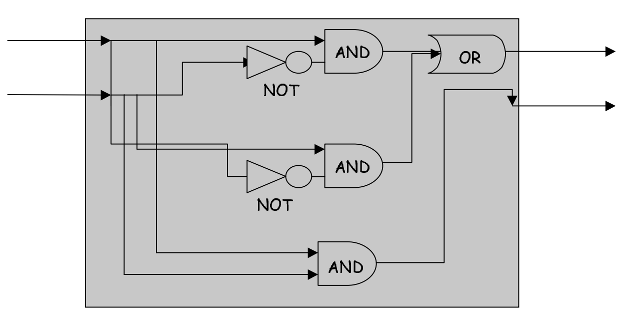
OR	0	1
0	0	1
1	1	1

AND	0	1
0	0	0
1	0	1

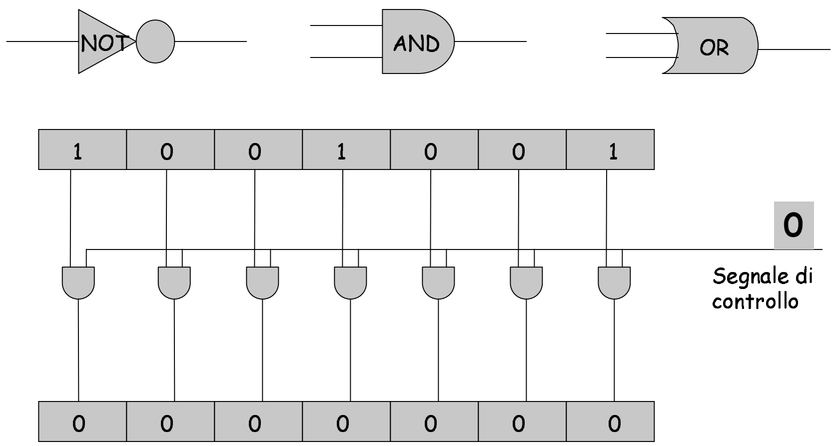
NOT	
0	1
1	0

Porte Logiche - Reti Logiche





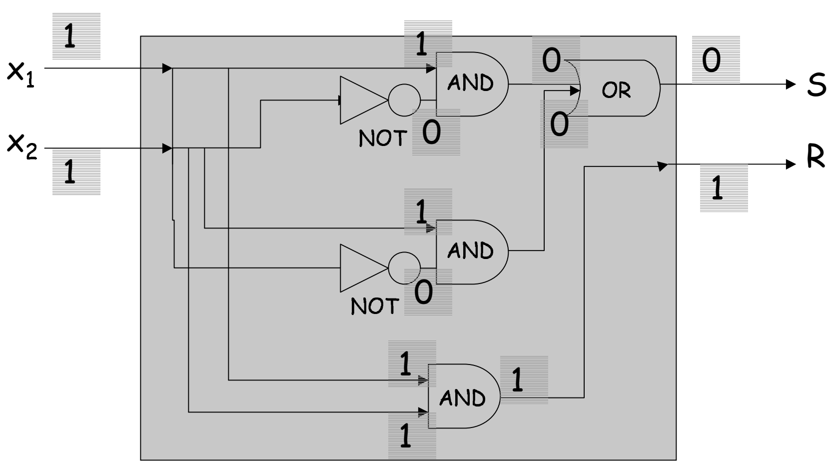
Porte Logiche - Reti Logiche



Trasferire sequenze di bit

Circuiti Logici

(Semiaddizionatore)

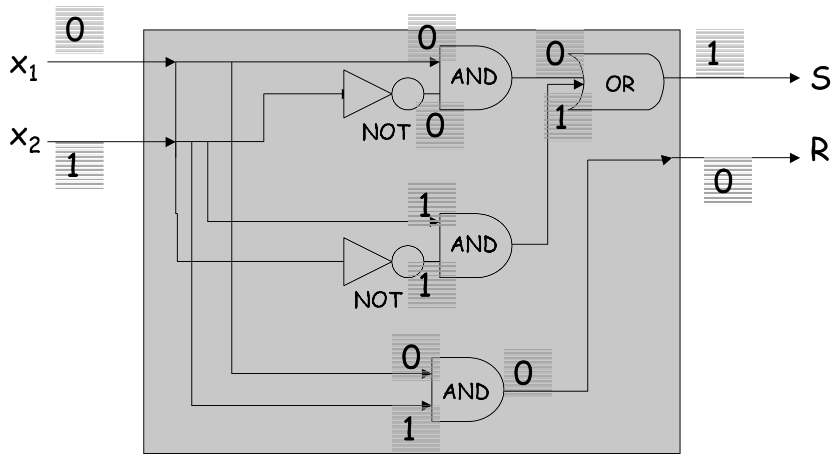


$S = (x_1 \text{ AND NOT}(x_2)) \text{ OR } (x_2 \text{ AND NOT}(x_1))$

$R = x_1 \text{ AND } x_2$

Circuiti Logici

(Semiaddizionatore)



S = (x_1 AND NOT(x_2)) OR (x_2 AND NOT(x_1))

R = x_1 AND x_2