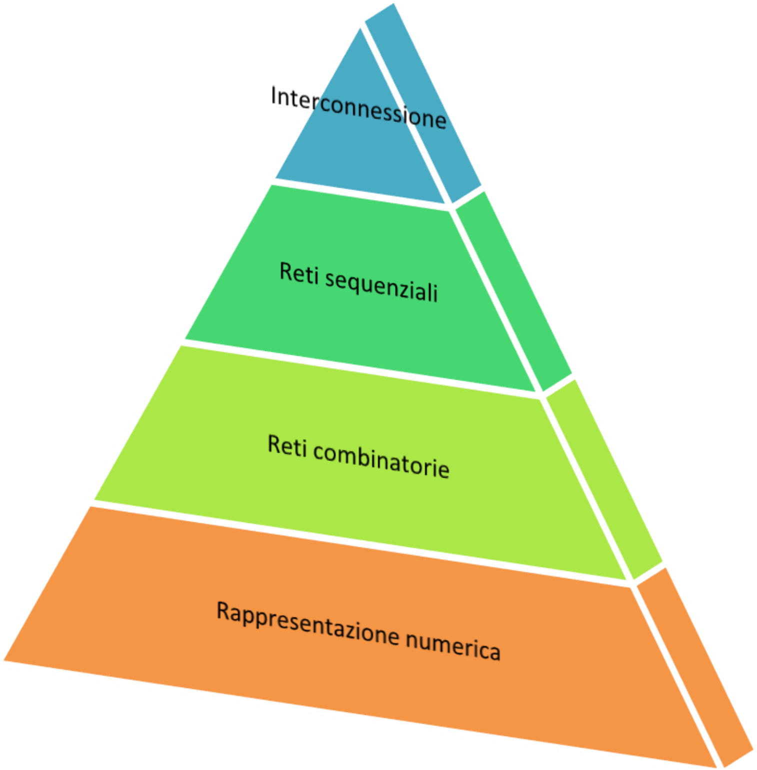
Architettura
degli Elaboratori Elettronici

Dott. Franco Liberati
liberati@di.uniroma1.it

ARGOMENTI DELLA LEZIONE

- Richiamo della rappresentazione numerica nei calcolatori: numeri interi e numeri reali
- Reti combinatorie
  - Codificatore e Decodificatore
  - Addizionatore e Sottrattore
  - Comparatore aritmetico e logico
- Reti sequenziali
  - Registro di memorizzazione
  - Registro Contatore
  - Registro a scorrimento
- Interconnessione
  - Multiplexer e demultiplexer, mesh, bus

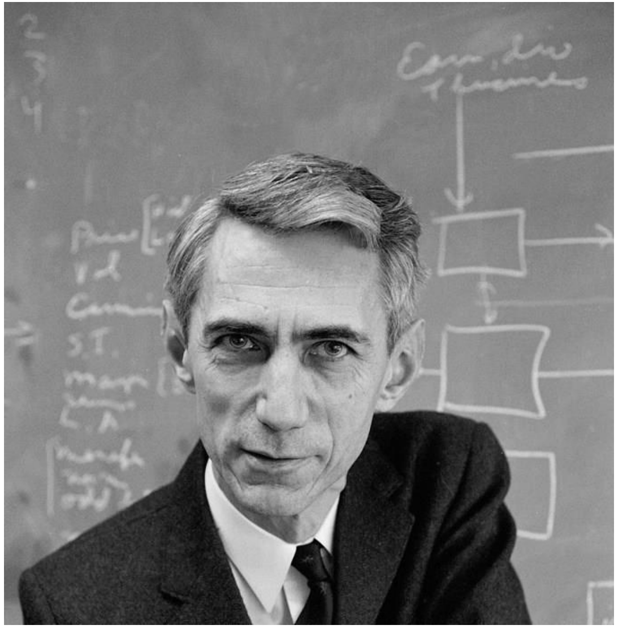


Principi generali della Teoria dell'Informazione

TEORIA DELL'INFORMAZIONE

Generalità

- *A Mathematical Theory of Communication* (1948)
  - Definizione delle componenti fondamentali delle comunicazioni digitali



CLAUDE SHANNON

TEORIA DELL'INFORMAZIONE

Un sistema di comunicazione

- La teoria enunciata da Shannon poneva l'attenzione su **come inviare un messaggio**, rappresentato da una concatenazione di parole, simboli (o segnali), e **ricostruirlo in modo esatto** (o con una buona approssimazione) quando ricevuto da una postazione remota
- Claude Shannon propose uno schema di **sistema di comunicazione** nel quale indicò gli elementi fondamentali costituenti (la sorgente, trasmettitore, canale, ricevitore, destinatario)

SISTEMA DI COMUNICAZIONE DEFINITO DA CLAUDE SHANNON

TEORIA DELL'INFORMAZIONE

Un sistema di comunicazione

- La **sorgente** indica l'insieme delle parole possibili
- Il **trasmettitore** è l'entità che codifica una parola in un segnale
- Il **canale** è il mezzo attraverso il quale si propaga il segnale codificato
- Il **Ricevente** è l'intermediario che decodifica il segnale nella corrispondente parola
- Il **destinatario**, colui al quale si indirizza la collezione di simboli

SISTEMA DI COMUNICAZIONE DEFINITO DA CLAUDE SHANNON

TEORIA DELL'INFORMAZIONE

Alfabeto

- Shannon non prese in considerazione il significato dei termini né del messaggio, ma nel definire la sorgente evidenziò che questa può essere specificata come un **insieme di simboli** possibili
- Lo scienziato statunitense comprese che in un sistema efficiente è importante garantire unicamente che siano i singoli segnali a giungere a destinazione in maniera corretta

Σ:={A, B, C, D, E, F, G, H, I, L, M, N, O, P, Q, R, S,T,U,V, Z}

Σ:={A, B, Γ, Δ, E, Z, H, Θ, I, K, Λ, M, N, Ξ, O, Π, P, Σ, T, Y, Φ, X, Ψ, Ω}

Σ:={A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z}

Σ:={0,1,2,3,4,5,6,7,8,9}

TEORIA DELL'INFORMAZIONE

Alfabeto binario

- Per una comunicazione affidabile Shannon scelse come **alfabeto** quello costituito da due soli simboli 1 e 0, i bit (o 'unità minima di informazione')
- Shannon dimostrò che sfruttando l'alfabeto binario era possibile costituire delle **parole** rappresentanti un messaggio

ESEMPIO

Per specificare, definire, riprodurre o memorizzare dieci parole differenti bisogna - in un sistema di comunicazione con alfabeto binario- associare almeno $\lceil \log_2 10 \rceil = \lceil 3.32 \rceil = 4$ bit

0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000 e 1001.

TEORIA DELL'INFORMAZIONE

Vantaggi dell'alfabeto binario

- Gli studi di Shannon si concentrarono anche sulla definizione dei criteri (cioè delle formule matematiche) grazie ai quali è possibile **ricevere le parole originarie trasmesse lungo un canale affetto da rumore**
- La base binaria si dimostrò quella più utile e più robusta nel caso di interferenze
  - Nel caso di errori, in un sistema di comunicazione a cifre binarie il messaggio originale può essere deducibile, cioè rimane 'visibile' anche oltre il disturbo. Inoltre la modifica di una cifra in un sistema non binario induce ad un numero di errori elevato
- Avere un alfabeto binario di riferimento inoltre è comodo perché molti dei componenti negli elettronici (porte logiche, commutatori elettronici, amplificatori) operano proprio con due livelli distinti di corrente elettrica (segnale alto, 1, e segnale basso, 0)

ESEMPIO

Una parola formata da tre simboli binari 000 può essere alterata e ricevuta, al più, come 111; con una distanza 7 dal valore reale (000,001,010,011,100,101,110,111)

Viceversa, nel caso si sfruttasse un alfabeto decimale il valore 000 potrebbe diventare 999, con una distanza mille (una differenza più marcata e ambigua) 000,001,002,003,...,997,998,999

TEORIA DELL'INFORMAZIONE

Alfabeto binario conseguenze

- Oltre a queste nozioni Shannon introdusse il concetto di **ridondanza**, cioè un numero di dati (o caratteri) rappresentanti una parola maggiore di quello richiesto, che sono aggiunti per garantire l'integrità dei messaggi nel caso di eventuali disturbi, e di **codifica economica**, cioè una riscrittura delle parole con un numero minimo di caratteri.

- Dal concetto di ridondanza si derivarono i **codici per il rilevamento e per la correzione degli errori (ECC)**; molto impiegati nelle trasmissioni lungo la rete internet o nell'archiviazione dei dati sui supporti digitali.

- Lo studio inerente ai codici economici, invece, portò alla **compressione dati**, strategia usata per accelerare i tempi di trasmissione dei segnali e per sfruttare al meglio lo spazio dei dispositivi di memorizzazione

ESEMPIO ECC

BIT DI PARITÀ

Si accoda alla parola il bit 1 se il numero di 1 è dispari altrimenti si accoda 0

10101000 1

11101000 1 ***ANOMALIA***

ESEMPIO COMPRESSIONE DATI

ALGORITMO RLE

0000000000000111111110000011000

0,13;1,8;0,5;1,2;0,3

31 caratteri contro 20 (guadagno 20/31 cioè 35.48%)

TEORIA DELL'INFORMAZIONE

Significato avulso dalla parola rappresentante

- Poiché Shannon non focalizzò l'attenzione sull'associazione tra **il significato e la parola rappresentante**, nella teoria dell'informazione questa relazione è di tipo non esclusivo ovvero il significato muta in accordo al dominio di utilizzo: la parola 0 può identificare il colore nero se si considera una immagine o il silenzio se si riproduce un suono o il valore zero se si fa riferimento a numeri

ASCII STANDARD	
Simbolo	Rappresentazione binaria
z	01111010

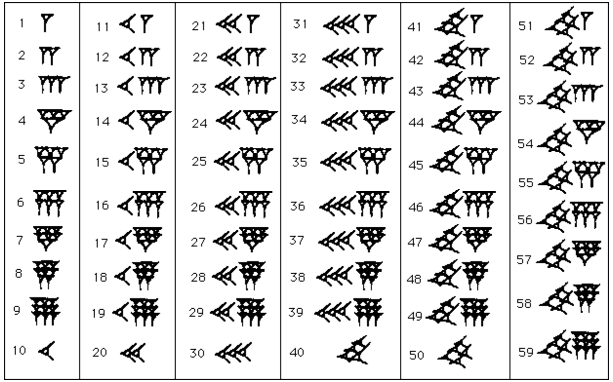
00010000 10110000 01010000

RAPPRESENTAZIONE NUMERICA

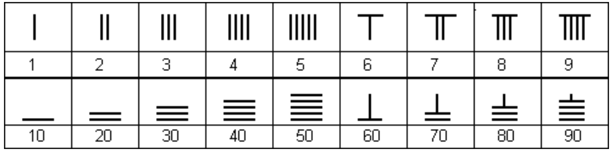
Numero: definizione e rappresentazione

- Un **numero** è un elemento astratto che rappresenta un insieme ben preciso di unità; serve a contare, e quindi a indicare la quantità di qualcosa o il posto che questo occupa all’interno di una serie. I numeri sono ordinati in una successione infinita, nella quale ogni elemento conta un’unità in più rispetto al precedente

- Un numero può essere **rappresentato** in vari modi impiegando simboli diversi



1	I	8	VIII
2	II	9	IX
3	III	10	X
4	IV	50	L
5	V	100	C
6	VI	500	D
7	VII	1000	M



RAPPRESENTAZIONE NUMERICA

Rappresentazione nel sistema posizionale

- Nei sistemi posizionali un numero è rappresentato considerando una base specifica
- In campo matematico si usa la base 10
- In campo informatico si fa riferimento al sistema binario, ottale e esadecimale

$$N=\sum_{i=0}^{m} a_i b^i \quad \text{con} \quad 0 \le a_i \le b-1 \ \text{e scelta} \ b>1$$

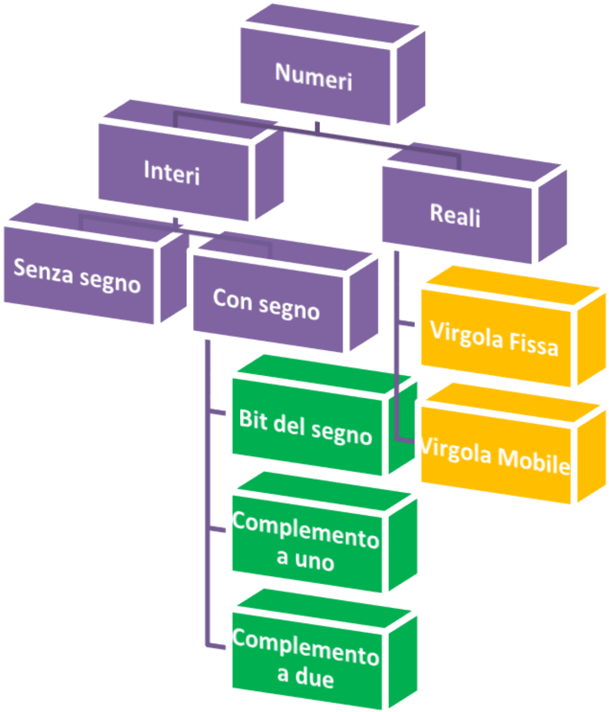
(2312) $=2\cdot10^3+3\cdot10^2+1\cdot10^1+2\cdot10^0$
$=2\cdot1000+3\cdot100+1\cdot10+2=2312$

Valore	Base
110582	**10**
11010111111110110	2
327766	8
1AFF6	16

RAPPRESENTAZIONE NUMERICA

I numeri nei sistemi di elaborazione automatica

- Gli **elaboratori elettronici** sono in grado di eseguire funzioni logico-aritmetiche su operandi rappresentanti mediante stringhe binarie di dimensione prestabilita.
- Gli operandi hanno il significato matematico di:
  - **Numeri interi**, per cui si ha una rappresentazione:
    - Senza segno
    - Con segno
      - Bit del segno
      - Complemento a uno
      - Complemento a due
  - **Numeri reali**, per cui si ha una rappresentazione:
    - Virgola Fissa
    - Virgola Mobile



RAPPRESENTAZIONE NUMERICA

Parola di rappresentazione

- Lo spazio a disposizione per immagazzinare le informazioni relative a un numero binario è limitato dalla **dimensione della parola di rappresentazione**
- La dimensione della parola è prefissata nel momento in cui si progetta l'elaboratore e ne condiziona ogni componente (dalla dimensione dei registri alla circuiteria necessaria per prelevare una parola in una sola fase dalla Memoria Centrale)
- Gli elaboratori elettronici attuali sono dotati di parole di 32 bit (dispositivi mobili); 64bit (computer) 128bit (supercalcolatori)

PAROLA DI 8 BIT

PAROLA DI 16 BIT

RAPPRESENTAZIONE NUMERICA

Intervallo di rappresentazione

- In una parola di dimensione *n* bit, tutte le informazioni relative al numero, segno compreso, devono occupare non più di *n* bit
- La dimensione della parola limita l'**intervallo di rappresentazione dei valori numerici**
- Dopo una operazione aritmetica il numero da rappresentare può eccedere la dimensione della parola provocando un **overflow** o un **trabocco**

PAROLA DI 16 BIT

PAROLA DI 16 BIT CON OVERFLOW/TRABOCCO

INTERO SENZA SEGNO

Sistema binario

- Un **intero senza segno** è un valore intero assunto positivo
- Gli *n* bit della parola sono usati <u>tutti</u> per rappresentare le cifre del numero
- Una parola a *n* bit permette di trattare un qualsiasi numero il cui **intervallo di rappresentazione** (o *range*) è $[0;2^n-1]$

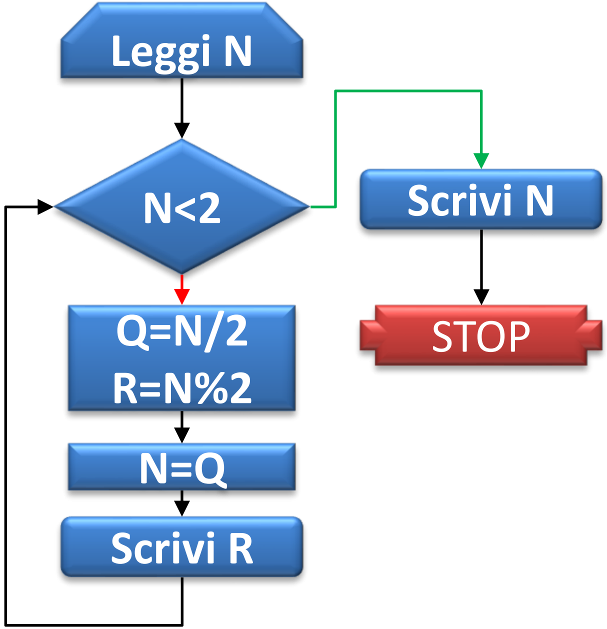
$$N = \sum_{i=0}^{\lceil \log_2 N \rceil} a_i 2^i \qquad 0 \le a_i \le 1$$

DECIMALE	BINARIO (dimensione della parola a 3bit)
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

INTERI SENZA SEGNO

Rappresentazione in base due

- Un metodo per convertire un valore da una base decimale ad una binaria è quello delle **divisione successive**



82	Quoziente	Resto
82:2	41	0 LSB
41:2	20	1
20:2	10	0
10:2	5	0
5:2	2	1
2:2	1	0
1MSB		
		1010010

$(82)_{10}=(01010010)_2$

INTERI SENZA SEGNO

Conversione da base binaria a decimale e viceversa

133	Quoziente	Resto
133:2	66	1 LSB
66:2	33	0
33:2	16	1
16:2	8	0
8:2	4	0
4:2	2	0
2:2	1	0
1MSB		
		10000101

$(133)_{10}=(10000101)_2$

Numero bin	1	0	0	0	0	1	0	1
Peso	7	6	5	4	3	2	1	0
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

$N=1\cdot 2^7+0\cdot 2^6+0\cdot 2^5+0\cdot 2^4+0\cdot 2^3+1\cdot 2^2+0\cdot 2^1+1\cdot 2^0$

$N=128+0+0+0+0+4+0+1=133$

INTERI SENZA SEGNO

Rappresentazioni con base non binaria

- Per rappresentazioni generiche con base maggiore a 10 si introducono, per convenzione, caratteri alfabetici in maiuscolo
  - Sistema **esadecimale** 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
  - Sistema in **base venti** 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F,G,H,I,L
- È importante usare simboli per evitare ambiguità

  ES: Nel sistema esadecimale 10 indica il valore 16 (cioè $1 \cdot 16^1 + 0 \cdot 16^0$); mentre il valore 10 è espresso con la lettera A

  $$(10)_{16} = 16 \text{ e } (A)_{16} = 10$$
- Superati i simboli alfabetici si ricorre a simboli speciali definibili dall’utente (le associazioni tra simboli e numero devono essere noti)

VALORE	BASE	RAPPRESENTAZIONE
2502		
	10	$(2502)_{10}$
	16	$(9C6)_{16}$
	8	$(4706)_8$
	2	$(000010011100011 0)_2$

INTERI SENZA SEGNO

Operazione di somma

- La **somma tra numeri interi senza segno**, comporta come risultato (considerando la somma di cifre ad uguale posizione) :
  - **1** se una delle due cifre è 1 e l'altra è zero e non c'è riporto
  - **0** altrimenti (con un riporto nel caso in cui entrambe le cifre sono 1)

Cifra addendo1	Cifra addendo2	Risultato	Riporto
0	**0**	**0**	**0**
0	**1**	**1**	**0**
1	**0**	**1**	**0**
1	**1**	**0**	**1**

INTERI SENZA SEGNO

Operazione di somma

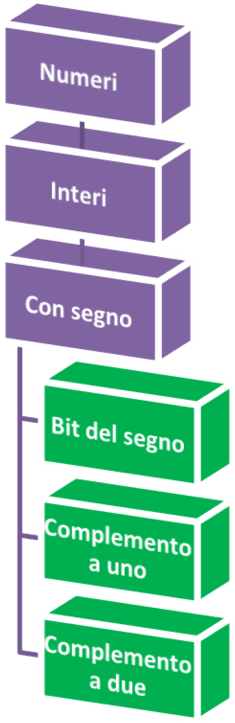
Esempio

0	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**1**	**0**	**0**	**0**				**Riporto**
0	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**1**	**0**	**0**	**44**	**+**	**Operando 1**
0	**0**	**0**	**1**	**1**	**0**	**1**	**1**	**0**	**1**	**0**	**1**	**0**	**874**	**=**	**Operando 2**
0	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**1**	**0**	**1**	**1**	**0**	**918**		**Risultato**
12	11	10	9	8	7	6	5	4	3	2	1	0			*posizione*

INTERI CON SEGNO

Modalità di rappresentazione

- La rappresentazione dei **numeri interi** richiede di riservare un bit relativo al segno (**+** oppure **-**) o di scegliere una codifica che consenta di discriminare in maniera chiara e precisa i valori negativi da quelli positivi
- I tre metodi usati sono:
  - bit del segno
  - complemento a uno
  - complemento a due (affermato in quasi tutti i calcolatori elettronici)



INTERI CON SEGNO

Bit del segno

- Il formato **bit del segno** ha il bit più significativo della parola riservato al segno del numero:
  - **1** per identificare il segno **-**
  - **0** per identificare il segno **+**

- Per la rappresentazione:
  - si codifica il numero in binario
  - si antepone il bit del segno
- Una parola di lunghezza *n* ha un intervallo di valori rappresentabili pari a:
  $[-2^{n-1}+1; 2^{n-1}-1]$

$(+15)_{10}$ =$(\mathbf{0}0001111)_2$

$(-12)_{10}$ =$(\mathbf{1}0001100)_2$

$(+0)_{10}$ =$(\mathbf{0}0000000)_2$

$(-0)_{10}$ =$(\mathbf{1}0000000)_2$

INTERI CON SEGNO

Complemento a uno

- I **numeri positivi nel complemento a uno** hanno uguale codifica del bit del segno (con bit più significativo settato a 0)
- I **numeri negativi nel complemento a uno** sono ottenuti complementando (ovvero ¬0=1 e ¬1=0) tutte le cifre che costituiscono la parola
- Il segno di un numero è individuato dal valore del primo bit, infatti tutti i numeri positivi iniziano con il bit 0, mentre quelli negativi iniziano con il bit 1

$(+15)_{10} = (00001111)_2$
$(-15)_{10} = (11110000)_2$

$(+0)_{10} = (00000000)_2$
$(-0)_{10} = (11111111)_2$

INTERI CON SEGNO

Complemento a due

- Nel formato **complemento a due** il bit più significativo ha peso negativo
- I **numeri positivi nel complemento a due** hanno uguale codifica del bit del segno (con bit più significativo settato a 0)
- I **numeri negativi nel complemento a due** si ottengono complementando l'intero positivo e sommando a questo ultimo il valore 1
- Si ha **una sola rappresentazione dello zero**
- L'intervallo di valori rappresentabile è
  $[-2^{n-1}; 2^{n-1}-1]$

$$N = -a_{n-1}2^{n-1} + \sum_{i=0}^{n-2} a_i 2^i$$

$(+15)_{10} = (00001111)_2$

$(-12)_{10} = (11110100)_2$

$(0)_{10} = (00000000)_2$

$(-12)_{10}$ Valore assoluto: $(00001100)_2$

Complementare: $(11110011)_2$

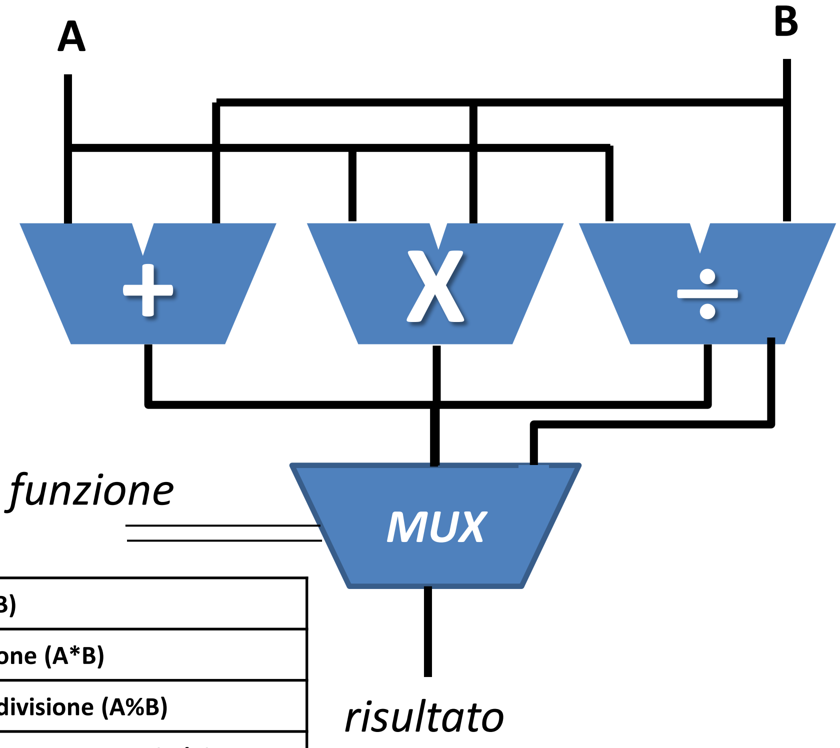
Somma di 1: $(11110100)_2$

INTERI CON SEGNO

Complemento a due: operazioni

- Sui numeri si interviene mediante **operazioni elementari** (somma, sottrazione, moltiplicazione, divisione, modulo) nella A.L.U. o **complesse** (integrazione, derivazione, calcolo di convoluzione,…) nei coprocessori matematici

UNITÀ LOGICO ARTIMETICA



00	Somma (A+B)
01	Moltiplicazione (A*B)
10	Resto della divisione (A%B)
11	Quoziente della divisione (A/B)

INTERI CON SEGNO

Complemento a due: somma

- La **somma tra numeri in complemento a due**, stabilisce come risultato (considerando la somma di cifre ad uguale posizione) :
  - **1** se una delle due cifre è 1 e l'altra è zero e non c'è riporto
  - **0** altrimenti (con un riporto nel caso in cui entrambe le cifre sono 1)

Cifra addendo1	Cifra addendo2	Risultato	Riporto
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

INTERI CON SEGNO

Complemento a due: somma

Esempio

0	0	1	1	0	1	1	0	1	1	0	1	0	0	0				Riporto
0	0	0	1	1	0	0	1	0	0	1	0	1	1	0	0	6444	+	Operando 1
0	0	0	1	1	0	1	1	0	1	1	0	1	0	1	0	7018	=	Operando 2
0	0	1	1	0	1	0	0	1	0	0	1	0	1	1	0	13462		Risultato
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			*posizione*

INTERI CON SEGNO

Complemento a due: somma

- La **somma tra numeri in complemento a due** può dar luogo ad un **trabocco** (condizione che non deve destrare preoccupazione) o un **overflow** (requisito che indica un valore risultante errato) che - in entrambi i casi - si evidenzia con un bit di riporto oltre alla dimensione della parola

INTERI CON SEGNO

Complemento a due: somma (trabocco)

- La somma di due numeri binari in complemento a due può comportare un **trabocco** (*carry*) non influente sul risultato finale
  - Può accadere quando gli operandi hanno segno opposto

1	1	1	1	1	1	1	1				**Riporto**
	0	0	0	0	1	1	1	1	15	+	**Sottraendo**
	1	1	1	1	1	0	1	1	-5		**Minuendo**
1	0	0	0	0	1	0	1	0	=10	10	**Risultato**
	7	6	5	4	3	2	1	0			***Posizione***

INTERI CON SEGNO

Complemento a due: somma (overflow)

- La somma di due numeri binari in complemento a due può comportare una **condizione di non rappresentabilità del risultato** (*overflow*) a causa della dimensione prefissata della parola
  - Accade quando i numeri hanno uguale segno e, oltre ad un bit di riporto oltre alla lunghezza della parola, c'è un cambio del segno nel risultato

1	0	0	0	0	0	0	1				**Riporto**
	1	0	0	0	1	0	0	1	-137	+	**Sottraendo**
	1	0	1	0	0	0	0	1	-95	=	**Minuendo**
1	0	0	1	0	1	0	1	0	42	(-232)	**Risultato**
	7	6	5	4	3	2	1	0			***posizione***

INTERI CON SEGNO

Complemento a due: sottrazione

- Laa **sottrazione tra numeri in complemento a due**, prevede come risultato (considerando le cifre ad uguale posizione) :
  - **1** se una delle due cifre è 1 e l’altra è zero; ma se la prima cifra è minore della seconda si deve richiedere un prestito alle cifre successive
  - **0** altrimenti

Cifra1	Cifra2	Risultato	Prestito
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

INTERI CON SEGNO

Complemento a due: sottrazione

Esempio 58-44

11101**0** -
10110**0**=
____
0

1110**10** -
1011**00**=
____
10

111**0**10 -
101**1**00=
____
10

110110 -
101100=
____
110

11**0**110 -
10**1**100=
____
110

101110 -
101100=
____
1110

1**0**1110 -
1**0**1100=
____
01110

101110 -
101100=
____
001110

14

Cifra1	Cifra2	Risultato	Prestito
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

INTERI CON SEGNO

Complemento a due: sottrazione (caso pratico)

- Gli elaboratori elettronici moderni risolvono la sottrazione per il complemento a due effettuando la somma di un numero con il suo opposto (si risparmia sulla circuiteria a scapito di un tempo di esecuzione - poco - più lungo)

Esempio 58 + (-44)

111010 -	0111010 +	0111010+
101100=	**1010100**=	1010100=
______	______	________
		*1*0001110

INTERI CON SEGNO

Complemento a due: moltiplicazione

- La **moltiplicazione tra numeri in base binaria** ha un procedimento analogo a quella che avviene nel calcolo decimale
- Si effettua il calcolo dei prodotti parziali, spostandoli di k posizioni in relazione alla k-esima posizione della cifra del moltiplicatore, per poi applicare la loro somma

			1	0	0	0	·
			1	0	0	1	=
			1	0	0	0	+
		0	0	0	0		+
	0	0	0	0			+
1	0	0	0				=
1	0	0	1	0	0	0	

NUMERI REALI

Definizione

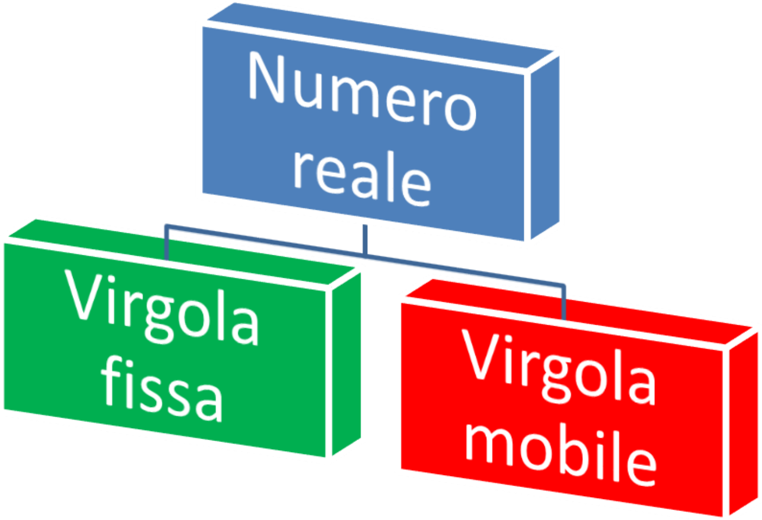
- Un **numero reale**, in generale, è definibile da una tripla $\Re$=<S;I;F> dove:
  - S è il segno
  - I è la parte intera
  - F la parte frazionaria
- La parte frazionaria è distinta da quella intera con una virgola o (nella notazione scientifica internazionale) con un punto

+3.75	**<+;3;75>**
-1.75	**<-;1;75>**
+11.504	**<+;11;504>**
-146.9	**<-;146;9>**

NUMERI REALI

Rappresentazione degli elaboratori elettronici

- Negli elaboratori elettronici i numeri reali possono essere rappresentati in due modi: **virgola fissa** (*fixed point*) o **virgola mobile** (*floating point*)
- Il motivo principale per l'introduzione dei numeri reali (o *floating point*) sta nella possibilità di utilizzare un **intervallo di numeri più ampio** per effettuare i calcoli riducendo però il numero di valori disponibili a parità di bit utilizzati per la codifica
- Non tutti i numeri reali possono essere rappresentati, ma solo un sottoinsieme limitato (aumenta il range, ma diminuisce la precisione)



NUMERI REALI

Virgola Fissa

- I numeri in **virgola Fissa** offrono una possibile rappresentazione dei valori reali
- Per operare con questo formato è necessario dapprima stabilire la lunghezza della parola e poi determinare la suddivisone in due campi: uno per la **parte intera** (rappresentabile ad esempio con il complemento a due) e un'altra per la **parte frazionaria**

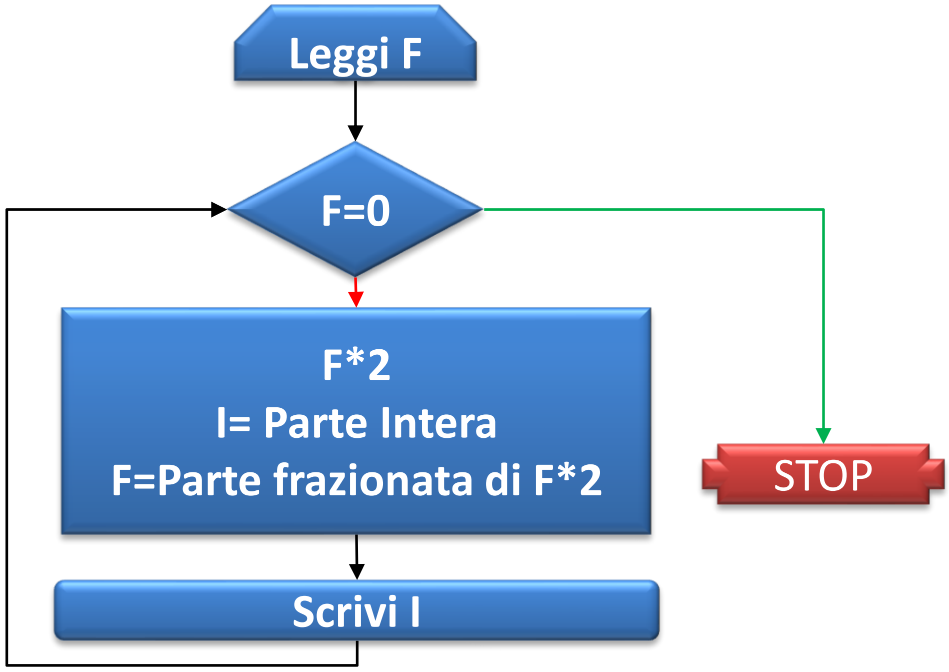
Parte Intera	Parte frazionaria
k	*n-k*

+3,75	**0011 | 1100**
-1,75	**1111 | 1100**

NUMERI REALI

Virgola fissa: moltiplicazioni successive

- Il calcolo della parte frazionaria può essere svolto con l'**algoritmo delle moltiplicazioni successive**



Valore	Parte Intera	Parte frazionaria
0.375·2=0.75	0	0,75
0.75·2=1.5	1	0.5
0.5·2=1.0	1	0
0		
		.011

Verifica:

$0 \cdot 2^{-1} + 1 \cdot 2^{-2} + 1 \cdot 2^{-3} = 0 + 0.25 + 0.125 = 0.375$

NUMERI REALI

Virgola Fissa

Parte Intera								Parte frazionaria							
S	2^{k-1}				2^{2}	2^{1}	2^{0}	2^{-1}	2^{-2}	2^{-3}	2^{-4}				2^{-n+k}

Parte Intera				Parte frazionaria			
0	1	0	1	1	1	0	0

+5,75

Parte Intera				Parte frazionaria			
1	1	1	0	1	0	1	0

-2,625

NUMERI REALI

Virgola Mobile

- La rappresentazione in **Virgola Mobile** è definita dalla tripla:

  < *S*; *M*; *E* > dove:
  - *S* è il *segno* del numero
  - *M* la sua *mantissa*
  - *E* il suo *esponente*
- È il metodo più usato nei moderni elaboratori elettronici ed è standardizzato come IEEE754

$$\Re = (-1)^S \cdot M \cdot 10^E$$

+3,75	**S=0 M=375 E=-2**
-1,75	**S=1 M=175 E=-2**
+11,504	**S=0 M=11504 E=-3**
-146,9	**S=1 M=1469 E=-1**

NUMERI REALI

Virgola Mobile: standard IEEE 754

- Nella rappresentazione in **Virgola Mobile in base 2** la prima decisione da prendere è quanti bit devono essere assegnati a esponente e mantissa
- La notazione scientifica, stabilita dall'IEEE754, definisce quattro formati:
  - Mezza precisione (16bit)
  - **Singola precisione (32bit)**
  - **Doppia precisione (64bit)**
  - Quadrupla precisione (128bit)
  - Ottupla precisione (256bit)

$$\Re = (-1)^S \cdot M \cdot 2^E$$

Formati standard **IEEE-754**

Singola precisione

Segno	Esponente	Mantissa
1 bit	8 bit	23 bit

Doppia precisione

Segno	Esponente	Mantissa
1 bit	11 bit	52 bit

NUMERI REALI

Virgola Mobile: standard IEEE 754

- La notazione scientifica adotta una **forma standard**
- La forma standard elimina gli zeri iniziali per avere esattamente una cifra diversa da zero a sinistra della virgola decimale (Es: il numero 34.56 ha forma standard: 3.456)
- Pertanto, un numero **in Virgola Mobile in base due normalizzato** ha una mantissa la cui cifra più a sinistra è diversa da zero
  - Per i numeri floating point normalizzati in base due, si ha una cifra uguale ad 1

+00101010,11001000100000000000000			
	S	**M**	**E**
NON normalizzata $+0,0101010110010001 0\ldots0 \cdot 2^{7}$	$(0)_2$	$(1010101100100001)_2$	00000111 (+7)
Normalizzata $+1,010101100100010\ldots0 \cdot 2^{5}$	$(0)_2$	$(1010101100100001)_2$	00000101 (+5)
		23bit	8bit

NUMERI REALI

Virgola Mobile: standard IEEE 754

Osservazione. Poiché il bit più a sinistra di una mantissa in base due normalizzata è sempre 1, non è necessario codificarlo (così si risparmia un bit); perciò il bit iniziale, in alcune architetture, diventa un **bit nascosto** (*hidden bit*)

+0101010,110010001 (42.783203)			
	S	**M**	**E**
Normalizzato con bit nascosto **+1,010101100100010…0·2⁵**	$(0)_2$	$(1010101100100010001)_2$ 23bit	00000101 (5) 8bit

NUMERI REALI

Virgola Mobile: standard IEEE 754

Osservazione. Lo standard usa anche la **polarizzazione dell'esponente** (o **notazione in eccesso**) cioè all'esponente si somma 127, singola precisione, o 1023, doppia precisione, e si sfruttano delle combinazioni speciali (es.:00000000 e 11111111) per rappresentare determinati valori

L'uso della polarizzazione dell'esponente semplifica l'ordinamento dei valori in Virgola Mobile attraverso il confronto tra interi

+0101010,110010001 (42.783203)			
	S	**$E^{POLARIZZATO}$**	**M**
Normalizzato con bit nascosto e polarizzato **$+1,01010110010001 0\ldots0 \cdot 2^{5+127}$**	0 1bit	$(10000100)_2$ 8 bit	$(1010101100100010\ldots0)_2$ 23 bit

NUMERI REALI

Virgola Mobile: standard IEEE 754 casi particolari

- La notazione scientifica sfruttando l'esponente polarizzato, inoltre, definisce dei **casi particolari** utili nel campo matematico

Singola Precisione		Doppia precisione		Significato
Esponente	**Mantissa**	**Esponente**	**Mantissa**	
0	**0**	**0**	**0**	**Zero**
0	**≠0**	**0**	**≠0**	**± valore denormalizzato**
1-254	**Qualsiasi**	**1-2046**	**qualsiasi**	**± valore floating point IEEE**
255	**0**	**2047**	**0**	**±∞**
255	**≠0**	**2047**	**≠0**	**NaN**

NUMERI REALI

Virgola Mobile: esempio singola precisione

- Il numero **-5,125** è rappresentabile in singola precisione come segue:

  Trasformazione delle parte intera e frazionata in binario:

  $-101{,}001 \cdot 2^{0}$

  Normalizzazione: $-01{,}01001 \cdot 2^{2}$

- Rappresentazione:

  $\ddot{E}=2+127=129=(10000001)_2$.

  $M=(101001000000000000000000)_2$

Esponente polarizzato	Mantissa con bit nascosto
110000001	0100100000000000000000

9bit 23bit

NUMERI REALI

Virgola Mobile: esempio singola precisione

Verifica

- Il segno è 1 quindi il numero è negativo
- La mantissa è 1*(bit nascosto)*$+2^{-2}+2^{-5}=1.28125$.
- Esponente è 10000001=129 quindi 129-127=2
- In sintesi: $(-1)\cdot 1.28125\cdot 2^2=-1.28125\cdot 4=-5.125$

Esponente polarizzato	Mantissa con bit nascosto
110000001	01001000000000000000000
9bit	23bit

NUMERI REALI

Virgola Mobile: esempio doppia precisione

- Il numero **+10,125** è rappresentabile in doppia precisione come segue:
  Trasformazione delle parte intera e frazionata in binario: $+1010{,}001 \cdot 2^0$
  Normalizzazione: $+01{,}010001 \cdot 2^3$
- Rappresentazione:
  S=segno positivo=$(0)_2$
  Ë=3+1023=1026=$(10000000010)_2$.
  M=$(1010001000000000000000000000000000000000000000000000)_2$

Esponente polarizzato	Mantissa con bit nascosto
010000000010	0100010000000000000000000000000000000000000000000000
12bit	52bit

NUMERI REALI

Virgola Mobile: esercizio proposto

Esercizio proposto

Dato il numero reale -**110,90625**

Rappresentarlo secondo lo standard IEEE754 singola precisione normalizzato e polarizzato e effettuare la verifica

NUMERI REALI

Virgola Mobile: operazioni somma e sottrazione

- L'algoritmo di **addizione** (e sottrazione) **in Virgola Mobile** si divide in quattro fasi fondamentali:
  1. Allineamento degli esponenti
  2. addizione (o sottrazione) delle mantisse
  3. normalizzazione
  4. arrotondamento delle mantisse (ed eventuale normalizzazione)

Esempio.

I valori sono 17,12 e 423,50:

Rappresentazione: $1{,}712 \cdot 10^1$ e $4{,}235 \cdot 10^2$

Allineamento esponenti: $1{,}712 \cdot 10^1$ e $42{,}35 \cdot 10^1$

Somma: $44{,}062 \cdot 10^1$

Normalizzazione: $4{,}4062 \cdot 10^2$

NUMERI REALI

Virgola Mobile: Overflow - Underflow

- Un elaboratore elettronico presenta un **overflow/underflow aritmetico** nel caso una operazione aritmetica che utilizza operandi reali genera un risultato più grande/piccolo di quelli rappresentabili con la parola a disposizione

$7{,}6252 = 76252 \cdot 10^{-4}$

7	6	2	5	2

$12345 = 12345 \cdot 10^{0}$

1	2	3	4	5

1	2	3	4	5	0	0	0	0
				7	6	2	5	2

1	2	3	5	2	6	2	5	2

NUMERI REALI

Virgola Mobile: operazioni moltiplicazione

- La **moltiplicazione** in virgola mobile si articola in:
  1. somma degli esponenti
  2. prodotto delle mantisse
  3. Normalizzazione (ed eventuale arrotondamento delle mantisse)

Esempio.
Dati i valori 980,12 e 50,00:

Rappresentazione: $9{,}8012 \cdot 10^2$ e $5 \cdot 10^1$

Somma esponenti: 2+1

Prodotto mantisse: 49,006

Normalizzazione mantissa:4, $9006 \cdot 10^1$

Rappresentazione: $4{,}9006 \cdot 10^{1+(3)}$

NUMERI REALI

Virgola Mobile: operazioni in binario

Dati i valori **5,703125** e **10,25**:

$(5{,}703125)_{10}=$ $(101{,}101101)_2=$ $1{,}011011010 \cdot 2^2$

$(10{,}25)_{10}=$ $(1010{,}010000)_2=$ $1{,}010010000 \cdot 2^3$

NUMERI REALI

Virgola Mobile: operazioni in binario

Somma

$(5{,}703125)_{10}=$ $(101{,}101101)_2=$ $1{,}011011010\cdot 2^2$
$(10{,}25)_{10}=$ $(1010{,}010000)_2=$ $1{,}010010000\cdot 2^3$

$1{,}011011010\cdot 2^2$ *e* $1{,}010010000\cdot 2^3$

Allineamento esponenti:

$1{,}011011010\cdot 2^2$ *e* $10{,}10010000\cdot 2^2$

Somma:

$1{,}011011010+10{,}100100000$

$11{,}11111101\cdot 2^2$

Normalizzazione:

$1{,}111111101\cdot 2^3$

Rappresentazione (normalizzata e polarizzata):

<0, 10000010, 1111111101000000…0> 15,953125

NUMERI REALI

Virgola Mobile: operazioni in binario

$(5{,}703125)_{10}=$ $(101{,}101101)_2=$ $1{,}011011010\cdot 2^2$

$(10{,}25)_{10}=$ $(1010{,}010000)_2=$ $1{,}010010000\cdot 2^3$

Moltiplicazione

$1{,}01101101\cdot 2^2$ *e* $1{,}01001\cdot 2^3$

Somma esponenti:

2+3=5 cioè 10+11=101

Moltiplicazione mantisse:

1,1101001110101

Risultato :

$1{,}1101001110101\cdot 2^5$

Normalizzazione mantissa:

$1{,}1101001110101\cdot 2^5$

Rappresentazione (normalizzata e polarizzata):

<0, 10000100, 11101001110101000000…0> 58,45703125

NUMERI REALI

Virgola Mobile: operazioni in binario

Esercizio proposto

Dati i numeri reali **0,5** e -**0,4375**

1. Rappresentarli in virgola mobile singola precisione in modalità normalizzata e polarizzata
2. Effettuare la somma
   A. Rappresentare il risultato della somma in modalità normalizzata e polarizzata
3. Effettuare il prodotto
   A. Rappresentare il risultato del prodotto in modalità normalizzata con il bit nascosto e polarizzata

NUMERI REALI

Virgola Mobile: operazioni in binario

Esercizio proposto (per casa)

Dati i numeri reali **12,0625** e -**28,1875**

1. Rappresentarli in virgola mobile singola precisione
2. Effettuare la somma
   A. Rappresentare il risultato della somma in modalità normalizzata e polarizzata
3. Effettuare il prodotto
   A. Rappresentare il risultato del prodotto in modalità normalizzata con il bit nascosto e polarizzata

Fine

Soluzione

-**110,90625**

Parte intera	**Parte frazionata**
110=55x2+**0**	0,90625x2=**1**,8125
55:2= 27x2+**1**	0,8125x2= **1**,625
27:2=13x2+**1**	0,625x2=**1**,25
13:2=6x2+**1**	0,25x2=**0**,5
6:2=3x2+**0**	0,5x2=**1**
3:2=1x2+**1**	
1:2=0x2+**1**	

-1101110,11101

S=1
M=11011101110100000000000
E=110

Rappresentazione normalizzata
<1,00000110, 11011101110100000000000>
Rappresentazione normalizzata e polarizzata
<110000101, 110111011101000000000000 >

NUMERI REALI

Virgola mobile: operazioni in binario

Esercizio proposto

Dati i numeri reali **0,5** e -**0,4375**

- ❑ Rappresentarli in virgola mobile singola precisione in modalità normalizzata e polarizzata

SEGNO=+

PARTE INTERA=0 = $(0)_2$

PARTE FRAZIONATA=0,5=$(0.1)_2$

VALORE=0,1 $\cdot 2^0$

VALORE=$1 \cdot 2^{-1}$

<0,11111111,1>

<0,01111110,00000000000000000000000>

SEGNO=-

PARTE INTERA=0 = $(0)_2$

PARTE FRAZIONATA=0,4375=$(0,0111)_2$

VALORE=0,0111 $\cdot 2^0$

VALORE=-1,110$\cdot 2^{-2}$

<1,11111110,111000>

<1,01111101,11000000000000000000000>

NUMERI REALI

Virgola mobile: operazioni in binario

Esercizio proposto

Dati i numeri reali **0,5** e -**0,4375**

- Effettuare la somma
  - Rappresentare il risultato della somma in modalità normalizzata e polarizzata

VALORE1=$+1 \cdot 2^{-1}$

VALORE2=$-1,110 \cdot 2^{-2}$

ALLINEAMENTO

VALORE1=$+1 \cdot 2^{-1}$

VALORE2=$-0,1110 \cdot 2^{-1}$

SOTTRAZIONE

	SOTT
1,000-	*0,110-*
0,111=	*0,111=*
______	______
0,001	*0,001*

NORMALIZZAZIONE

$0,001=1,0 \cdot 2^{-3}$ $(\cdot 2^{-1})$

Cioè:

$+1,0 \cdot 2^{-4}$

<0,01111011,00000000000000000000000>

NUMERI REALI

Virgola mobile: operazioni in binario

Esercizio proposto

Dati i numeri reali **0,5** e -**0,4375**

- ❑ Effettuare la somma (variante con il complemento a due

VALORE1=$+1 \cdot 2^{-1}$

VALORE2=$-1,11 \cdot 2^{-2}$

ALLINEAMENTO

VALORE1=$+1 \cdot 2^{-1}$

VALORE2=$-0,111 \cdot 2^{-1}$

SOTTRAZIONE COMPL 2

VALORE 1:1000$\cdot 2^{-4}$	01000	01000
VALORE 2:0111$\cdot 2^{-4}$	00111	11001
		*1*00001 $1 \cdot 2^{-4}$

NORMALIZZAZIONE

$1 \cdot 2^{-4}$

<0,01111011,00000000000000000000000>

NUMERI REALI

Virgola mobile: operazioni in binario

Esercizio proposto

Dati i numeri reali **0,5** e -**0,4375**

- ❑ Effettuare il prodotto
  - ❖ Rappresentare il risultato della somma in modalità normalizzata e polarizzata

VALORE=$+1\cdot 2^{-1}$
VALORE=$-1{,}110\cdot 2^{-2}$

SOMMA ESPONENTI
-3

MOLTIPLICAZIONEù1,1
1,110x1=1,11

NORMALIZZAZIONE
Non necessaria

Cioè:
$-1{,}11\cdot 2^{-3}$

<1,01111100,11000000000000000000000>