

# Come funzionano i computer

**Capirli per usarli senza farsi usare**



# Come funzionano i computer

## *Capirli per usarli senza farsi usare.*

Materiale basato sull'omonima serie video di [Code.org](https://code.org)



Versione italiana a cura di Programma il Futuro ([www.programmailfuturo.it](http://www.programmailfuturo.it))

Direzione e coordinamento: Enrico Nardelli

Testo e layout: Francesco Lacchia

Grafica di copertina: Paolo Alberti e Francesco Lacchia

Distribuito sotto licenza Creative Commons: Attribution-NonCommercial-ShareAlike



Ultimo aggiornamento: 17/09/2018

# Indice

1	Introduzione.....	4
1.1	Svolgimento delle lezioni.....	4
2	Cosa rende un computer un computer?.....	5
2.1	Organizzazione della lezione (45 minuti).....	5
2.2	Introduzione.....	5
2.3	Ingressi (input).....	6
2.4	Memoria & Elaborazione.....	6
2.5	Uscite (output).....	6
2.5.1	Esempio.....	7
3	Dati e sistema binario.....	8
3.1	Organizzazione della lezione (2 ore).....	8
3.2	Introduzione.....	8
3.3	Sistema numerico decimale.....	9
3.4	Sistema numerico binario.....	9
3.5	Testo in binario.....	10
3.5.1	Esempio.....	11
3.6	Immagini in binario.....	11
3.7	Suoni in binario.....	12
4	Circuiti e logica.....	14
4.1	Organizzazione della lezione (2 ore).....	14
4.2	Introduzione.....	14
4.3	Porta NOT.....	14
4.4	Porta AND.....	15
4.5	Porta OR.....	15
4.5.1	Esercizio.....	16
4.6	Sommatore.....	17
5	Memoria, CPU, ingressi e uscite.....	21
5.1	Organizzazione della lezione (30 minuti).....	21
5.2	Introduzione.....	21
5.3	CPU e memoria.....	21
5.3.1	Esempio: visualizzazione di una lettera.....	21
5.3.2	Esempio: visualizzazione di una lettera in grassetto.....	22
6	Hardware e software.....	23
6.1	Organizzazione della lezione (30 minuti).....	23
6.2	Introduzione.....	23
6.3	Hardware, software.....	23
6.4	Sistema operativo.....	25
7	Mappatura con Proposta CINI.....	26

# 1 Introduzione

La semplicità d'uso dei dispositivi digitali ne ha fatto esplodere la diffusione. È stata una grande conquista, perché sta permettendo ad un numero sempre maggiore di persone di accedere ad ogni genere di servizio. C'è però da considerare un effetto collaterale: spesso questi dispositivi vengono usati senza avere consapevolezza di come funzionano. È su questo aspetto che occorre lavorare, per far crescere una generazione di persone capace di usare al meglio la tecnologia e guidarne la futura evoluzione. Occorre far assimilare agli studenti l'idea che **solo se si capisce la tecnologia, la si può dominare, altrimenti si rischia di esserne dominati!**

Questo corso è adatto **dai 9-10 anni in su** (senza alcun limite superiore). Considerato che gli insegnanti hanno una conoscenza più precisa delle loro classi, saranno in grado di graduare il livello di approfondimento in base all'età e alle conoscenze dei loro studenti.

Il materiale su cui si basa il corso è stato realizzato dall'organizzazione americana no profit Code.org ed è stato adattato alla situazione italiana da **Programma il Futuro** (<https://programmmailfuturo.it>), il progetto MIUR<sup>1</sup>-CINI<sup>2</sup> che ha l'obiettivo di diffondere nelle scuole la conoscenza dei concetti scientifici di base dell'informatica.

In questo corso viene spiegato quali sono le funzioni e le componenti che fanno sì che un computer sia un computer, come vengono rappresentati i dati al suo interno mediante semplici segnali elettrici e come dei circuiti integrati che fanno solo svolgere semplici operazioni matematiche riescano a realizzare tutto ciò a cui oggi siamo ormai abituati.

Vengono inoltre descritte le diverse parti di un computer, specializzate nella gestione di ingressi, uscite, elaborazione e memorizzazione dei dati.

Infine, viene spiegato cos'è il software e come esso interagisce con l'hardware.

## 1.1 Svolgimento delle lezioni

Al seguente indirizzo web

<https://programmmailfuturo.it/come/come-funzionano-i-computer>

è possibile trovare tutto il materiale necessario per condurre le lezioni sugli argomenti trattati, inclusi i video che ne costituiscono la prima presentazione.

In questo documento, realizzato per accompagnare i video disponibili per ogni lezione, vengono approfonditi e sviluppati i contenuti presenti nei video disponibili al precedente link. All'inizio di ogni capitolo viene inoltre indicata la metodologia suggerita agli insegnanti per condurre la lezione nel modo più efficace possibile ed il tempo medio stimato per la sua realizzazione (quello effettivo dipende, come sempre, dalla specifica situazione della classe).

Il breve [video introduttivo generale](#) in cui il fondatore della Microsoft, Bill Gates, presenta l'intera serie, potrà essere utilizzato come introduzione per presentare questa attività didattica qualche giorno prima del suo inizio, in modo da stimolare l'aspettativa degli studenti.

<sup>1</sup> Ministero dell'Istruzione dell'Università e della Ricerca

<sup>2</sup> Consorzio Interuniversitario Nazionale per l'Informatica

## 2 Cosa rende un computer un computer?

### 2.1 Organizzazione della lezione (45 minuti)

- Preparati guardando preventivamente il [video introduttivo generale](#) e il [video “Cosa rende un computer un computer”](#);
- preparati leggendo attentamente la descrizione presente più avanti in cui vengono esposti i contenuti del video;
- guarda con gli studenti il [video introduttivo](#) in cui il fondatore della Microsoft, Bill Gates, presenta l'intera serie di video;
- guarda con gli studenti il [video “Cosa rende un computer un computer”](#);
- riepiloga alla lavagna i concetti principali:
  - i **quattro concetti chiave** che contraddistinguono tutti i computer (sollecita gli studenti a suggerire il maggior numero possibile di esempi di computer che soddisfano questa definizione),
  - concentrati su ognuno dei suddetti quattro concetti chiave, riproducendo alla lavagna la Figura 3, ricorda in cosa consiste ognuno di essi e sollecita gli studenti a suggerire il maggior numero possibile di esempi;
  - proponi alla lavagna l'esempio sulla centralina meteo di pagina 7;
- guarda nuovamente con gli studenti il [video “Cosa rende un computer un computer”](#) per fissare definitivamente i concetti appresi.

### 2.2 Introduzione

Nel corso della storia, l'uomo ha sempre creato strumenti che lo hanno aiutato nello svolgimento del suo lavoro. Fino ad un secolo fa questi strumenti avevano solamente a che fare con il lavoro manuale, come, ad esempio: spostare carichi, scavare, manipolare e trasformare oggetti fisici. Poi ha cominciato ad interrogarsi sulla possibilità di progettare e costruire macchine che lo aiutassero a svolgere attività immateriali, come il ragionamento.

Gli “oggetti” “manipolati” da queste macchine sono i *dati* e la struttura che serve per realizzare una qualunque di queste macchine è caratterizzata dalle seguenti funzioni:

- acquisire dei dati in ingresso,
- memorizzare questi dati,
- elaborarli in vari modi,
- fornire dati in uscita come risultato dell'elaborazione svolta.

**Questa struttura è ciò che rende un computer un computer**, indipendentemente dalla tecnologia con cui viene realizzato. I primi computer erano fatti di legno e metallo con leve meccaniche e ingranaggi, poi durante il XX secolo, tutto ciò è stato sostituito da componenti elettrici ed elettronici. I primi computer erano molto grandi e molto lenti, potevano metterci ore a risolvere semplici problemi di matematica. Agli albori, queste macchine si limitavano ad elaborare e restituire numeri, mentre oggi è normale usarli per parlare tra noi, per immergersi in complessi scenari grafici tridimensionali, per controllare dei robot e fare quasi qualsiasi cosa che si possa immaginare. I computer moderni non assomigliano affatto a quelle vecchie goffe macchine, ma continuano a fare le stesse quattro cose indicate in Figura 1.



Figura 1

Per poter essere gestiti da un computer moderno, i dati devono essere rappresentati mediante **segnali elettrici**, mettendo così il computer in grado di fare calcoli, scrivere, sentire musica, vedere video, parlarsi, giocare e anche controllare dei robot.

## 2.3 Ingressi (input)



Gli ingressi (*input* in inglese) sono costituiti da tutti i dati che il mondo può inviare al computer, in altre parole tutto ciò che il mondo fa o che tu fai, per far sì che il computer faccia qualcosa.

Quali possono essere gli strumenti in grado di inviare dati al computer? La tastiera, il mouse, lo schermo tattile (touchscreen) oppure un microfono o una telecamera, ma anche ogni genere di sensore che rileva delle grandezze fisiche (termometro, rilevatore del battito cardiaco, sensore di luminosità ecc.).

## 2.4 Memoria & Elaborazione



Per poter raggiungere un qualsiasi risultato mediante un computer, tutti i dati ricevuti in ingresso devono essere in qualche modo **elaborati** e per poter svolgere queste elaborazioni è necessario che questi dati vengono archiviati in una **memoria**.

Il **processore** del computer legge i dati dalla memoria, li manipola e li trasforma (con delle operazioni logiche e matematiche) seguendo un **programma**, che è semplicemente costituito da una serie di istruzioni. Nel corso di queste elaborazioni, vengono salvati in memoria tutti i risultati parziali necessari all'ottenimento dell'obiettivo finale, che una volta raggiunto viene inviato al mondo esterno.

## 2.5 Uscite (output)



Il fine ultimo di un computer è quello di fornire al mondo esterno gli esiti delle sue elaborazioni: un elaboratore che tiene per sé i risultati raggiunti non è di molto interesse per le persone!

Quali possono essere le modalità con cui un computer fornisce i dati in uscita (*output* in inglese)? Dipende dallo scopo per il quale è stato progettato. Ecco alcuni possibili dispositivi di uscita: lo schermo su cui mostrare testo, foto, filmati o giochi interattivi, oppure una stampante, degli altoparlanti o dei segnali per controllare un robot. Quando un computer si connette ad Internet, l'output di un computer diventa l'input di un altro e viceversa.

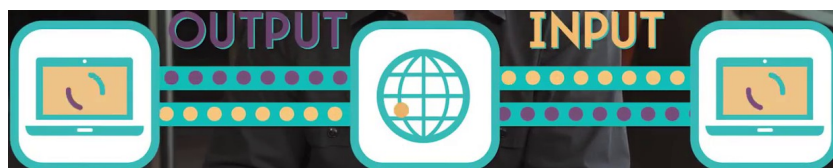


Figura 2



In Figura 3 viene evidenziato come sono legati tra loro i quattro concetti fondamentali che fanno sì che un computer sia un computer.

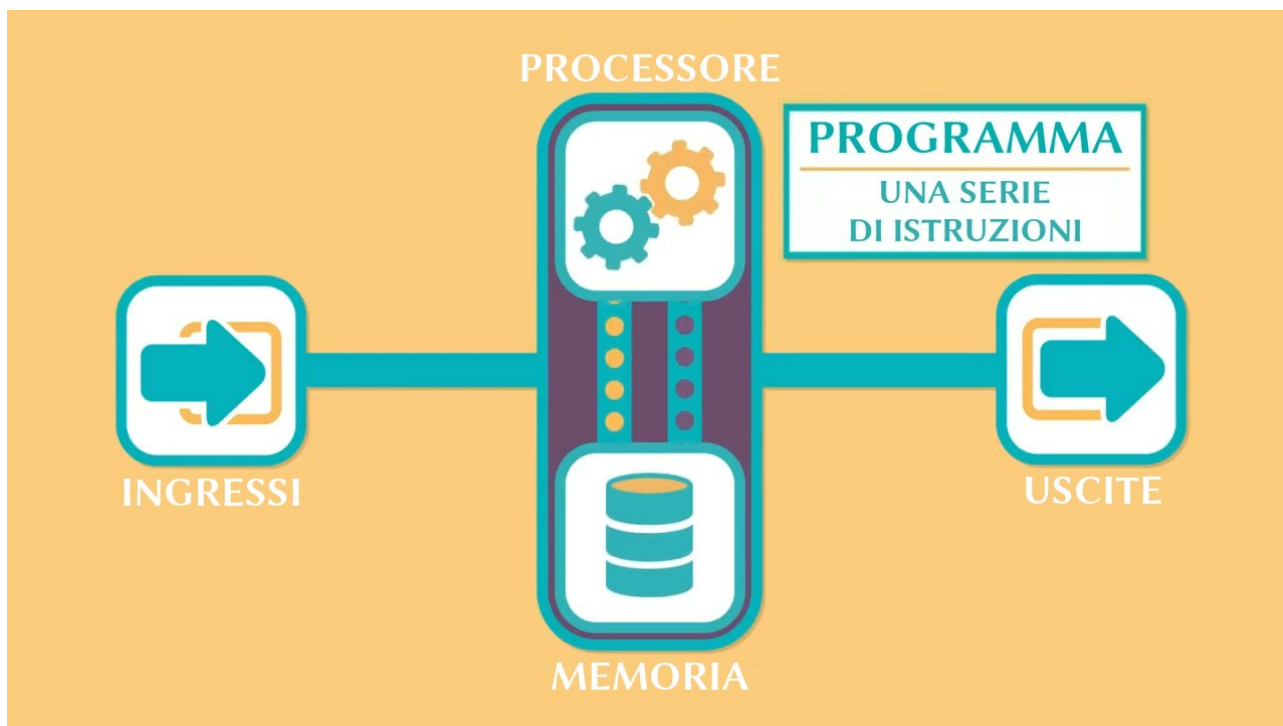


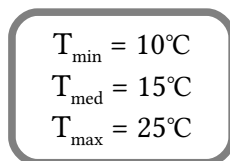
Figura 3

### 2.5.1 Esempio

Per contestualizzare su un caso pratico i precedenti concetti teorici, si propone qui la descrizione del principio di funzionamento di una centralina meteorologica che fornisce le temperature minima, media e massima della giornata.



La centralina meteorologica – che è a tutti gli effetti un computer – riceve continuamente in **INGRESSO** la misura della temperatura tramite un sensore, il **PROGRAMMA** ne rileva la lettura ogni 10 minuti e salva il dato in **MEMORIA**. Quindi, alla fine della giornata la memoria contiene 144 valori, pari a 24 (ore della giornata) x 6 (misurazioni per ogni ora). A questo punto il **PROCESSORE** confronta tutti questi valori per determinare qual è il minimo e qual è il massimo; dopo di che li somma e, dividendo il risultato per 144, ottiene la temperatura media della giornata. Infine, fornisce in **USCITA** questi tre risultati visualizzandoli sullo schermo, in modo testuale o grafico:



### 3 Dati e sistema binario

#### 3.1 Organizzazione della lezione (2 ore)

- Preparati guardando preventivamente il [video “Dati e sistema binario”](#);
- preparati leggendo attentamente la descrizione presente più avanti in cui vengono esposti i contenuti del video con l’aggiunta di esempi e approfondimenti;
- guarda con gli studenti il [video “Dati e sistema binario”](#);
- riepiloga alla lavagna i concetti principali ed esegui alcuni esempi ed esercizi con gli studenti:
  - filo percorso da corrente e **segnale binario** associato;
  - possibili combinazioni di bit ottenibili con uno, due, tre o più fili;
  - sistema numerico decimale: peso di ogni cifra in base alla sua posizione;
  - **sistema numerico binario**: peso di ogni cifra in base alla sua posizione, proponi esempi ed esercizi di conversione binario ⇒ decimale;
  - **testo** in binario, codice ASCII, proponi esempi ed esercizi simili quello di pagina 11;
  - **immagini** in binario: spiega il modello RGB come descritto a pagina 11, analizza gli esempi e sperimenta dinamicamente con gli studenti all’interno di un programma di grafica o anche semplicemente nelle impostazioni del colore di un elaboratore di testo;
  - **suoni** in binario: se hai a disposizione un programma di registrazione o elaborazione audio, potresti anche far notare agli studenti la forma d’onda che si genera durante la registrazione di un audio;
- guarda nuovamente con gli studenti il [video “Dati e sistema binario”](#) per fissare definitivamente i concetti appresi;
- vista la lunghezza e la complessità di questa lezione, è consigliabile dividerla in due sessioni.

#### 3.2 Introduzione

Come fa il computer a gestire i dati mediante segnali elettrici che circolano nei fili e nei circuiti di cui è costituito?

Un singolo filo elettrico può essere o meno percorso da elettricità, questi due possibili stati rappresentano la base di tutta l’informatica:

Filo percorso da corrente elettrica:



Filo NON percorso da corrente elettrica:



acceso	spento
ON	OFF
vero	falso
SI	NO
<b>1</b>	<b>0</b>

Questo elemento, che può assumere i due possibili valori “uno” e “zero”, rappresentati rispettivamente dalle cifre **1** e **0**, è chiamato **bit** ed è il più piccolo dato che un computer può memorizzare.

Se si usano più fili, si hanno a disposizione più bit, quindi più combinazioni di **1** o **0**. Ecco le prime combinazioni possibili:

- 1 filo ⇒ 1 bit: spento, acceso ⇒ **0**, **1**
- 2 fili ⇒ 2 bit: spento spento, spento acceso... ⇒ **00**, **01**, **10**, **11**
- 3 fili ⇒ 3 bit: **000**, **001**, **010**, **011**, **100**, **101**, **110**, **111**



Con questo approccio si può costruire un intero sistema di rappresentazione di numeri: il **sistema numerico binario**. Vediamo come, ricordando prima come funziona il sistema numerico decimale, quello che usiamo normalmente.

### 3.3 Sistema numerico decimale

Tutti noi abbiamo imparato a contare con il sistema numerico decimale che è basato sulle “dieci” cifre da 0 a 9 che rappresentano i valori da “zero” a “nove”. Sappiamo inoltre che la posizione di ogni cifra ha un peso diverso:

	Terza da destra	Seconda da destra	Prima da destra
Peso della cifra	<i>cento</i>	<i>dieci</i>	<i>uno</i>

Un cifra nella posizione meno significativa (la prima da destra) rappresenta direttamente uno dei *dieci* possibili valori da *zero* a *nove*.

Il valore rappresentato da una cifra nella seconda posizione da destra si ottiene moltiplicando il valore della cifra per *dieci*, ovvero per il numero di possibili valori rappresentato dalla cifra alla sua destra.

Il valore rappresentato da una cifra nella terza posizione da destra deve essere moltiplicato per *cento*, ovvero per il numero di possibili valori rappresentati dalle *due* cifre nelle *due* posizioni alla sua destra.

E così via...

Per esempio, il valore di un numero scritto nel sistema numerico decimale come 8 5 4 si calcola, dal momento che in questo sistema 8 rappresenta il valore *otto*, 5 il valore *cinque* e 4 *quattro*, in questo modo:

<i>otto</i> x <i>cento</i> = <i>ottocento</i>	<i>cinque</i> x <i>dieci</i> = <i>cinquanta</i>	<i>quattro</i> x <i>uno</i> = <i>quattro</i>
---	---	--

e sommando poi questi valori: *ottocento* + *cinquanta* + *quattro* = *ottocentocinquantaquattro*.

### 3.4 Sistema numerico binario

Il sistema numerico binario si basa invece solo su *due* cifre: **0** e **1**. Ma combinandole possiamo ugualmente rappresentare qualunque valore.

Anche nel sistema numerico binario la posizione di ogni cifra ha un peso diverso:

	Quarta da destra	Terza da destra	Seconda da destra	Prima da destra
Peso della cifra	<i>otto</i>	<i>quattro</i>	<i>due</i>	<i>uno</i>

Un cifra nella posizione meno significativa (la prima da destra) rappresenta direttamente uno dei *due* possibili valori *zero* o *uno*.

Il valore rappresentato da una cifra nella seconda posizione da destra si ottiene moltiplicando la cifra per *due*, ovvero per il numero di possibili valori rappresentati dalla cifra alla sua destra.

Il valore rappresentato da una cifra nella terza posizione da destra deve essere moltiplicato per *quattro*, ovvero per il numero di possibili valori rappresentati dalle *due* cifre nelle *due* posizioni alla sua destra.

Il valore rappresentato da una cifra nella quarta posizione da destra deve essere moltiplicato per *otto*, ovvero per il numero di possibili valori rappresentati dalle *tre* cifre nelle *tre* posizioni alla sua destra.

E così via...

Per esempio, il valore di un numero scritto in binario come **1101** si calcola così:

<i>uno x otto = otto</i>	<i>uno x quattro = quattro</i>	<i>zero x due = zero</i>	<i>uno x uno = uno</i>
--------------------------	--------------------------------	--------------------------	------------------------

e sommando poi questi valori: *otto + quattro + zero + uno = tredici*.

Ecco quindi che il valore *tredici*, scritto come **1101** nel sistema binario, corrisponde a scrivere 13 nel sistema decimale:

$$\mathbf{1101}_{\text{binario}} = 13_{\text{decimale}}$$

Quindi, grazie a questo sistema numerico che si basa solo sulle cifre **0** e **1**, ogni possibile valore può essere rappresentato all'interno di un computer mediante un gruppo di fili che sono percorsi da elettricità o meno.

Più fili si usano, maggiore è la quantità di numeri che si possono rappresentare e memorizzare. Ogni volta che si aggiunge un filo si sta moltiplicando per *due* la quantità di numeri che si possono rappresentare con i fili precedenti:

numero fili	quantità di numeri rappresentabile
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8	256
9	512
10	1024
...	...
32	più di quattro miliardi!

Ma finora si è pensato solo ai numeri, come fare con altri tipi di dati, come testo, immagini o suoni? Occorre rappresentare questi dati con dei numeri.

### 3.5 Testo in binario

Nei primi computer il primo dato diverso dai numeri preso in considerazione è stato il testo (caratteri alfanumerici ed un certo insieme di caratteri speciali). Le immagini e i suoni sono arrivati molto più avanti.

Si è quindi definita una tabella di codifica per associare ogni carattere ad un certo numero, in modo che, per esempio, il computer potesse visualizzare sullo schermo una lettera premuta sulla tastiera.

Agli inizi dell'informatica sono nate diverse tabelle di rappresentazione dei caratteri, ma una delle più famose, che ha permesso di far parlare tra loro diversi computer, è la tabella ASCII:

Byte	Cod	Char	Byte	Cod	Char	Byte	Cod	Char
00100000	32	Sp	01000000	64	@	01100000	96	`
00100001	33	!	01000001	65	A	01100001	97	a
00100010	34	"	01000010	66	B	01100010	98	b
00100011	35	#	01000011	67	C	01100011	99	c
00100100	36	\$	01000100	68	D	01100100	100	d
00100101	37	%	01000101	69	E	01100101	101	e
00100110	38	&	01000110	70	F	01100110	102	f
00100111	39	'	01000111	71	G	01100111	103	g
00101000	40	(	01001000	72	H	01101000	104	h
00101001	41	)	01001001	73	I	01101001	105	i
00101010	42	*	01001010	74	J	01101010	106	j
00101011	43	+	01001011	75	K	01101011	107	k
00101100	44	,	01001100	76	L	01101100	108	l
00101101	45	-	01001101	77	M	01101101	109	m
00101110	46	.	01001110	78	N	01101110	110	n
00101111	47	/	01001111	79	O	01101111	111	o
00110000	48	0	01010000	80	P	01110000	112	p
00110001	49	1	01010001	81	Q	01110001	113	q
00110010	50	2	01010010	82	R	01110010	114	r
00110011	51	3	01010011	83	S	01110011	115	s
00110100	52	4	01010100	84	T	01110100	116	t
00110101	53	5	01010101	85	U	01110101	117	u
00110110	54	6	01010110	86	V	01110110	118	v
00110111	55	7	01010111	87	W	01110111	119	w
00111000	56	8	01011000	88	X	01111000	120	x
00111001	57	9	01011001	89	Y	01111001	121	y
00111010	58	:	01011010	90	Z	01111010	122	z
00111011	59	;	01011011	91	[	01111011	123	{
00111100	60	<	01011100	92	\	01111100	124	
00111101	61	=	01011101	93	]	01111101	125	}
00111110	62	>	01011110	94	^	01111110	126	~
00111111	63	?	01011111	95	_	01111111	127	Del

I codici da 0 a 31 sono caratteri speciali spesso usati come segnali di controllo.

Ogni parola o frase gestita da un computer è in realtà una sequenza di numeri binari memorizzati e trasmessi come segnali elettrici accesi o spenti.

### 3.5.1 Esempio

Codifica della parola **Ciao**

- **c**: 67 ⇒ 10000011
- **i**: 105 ⇒ 01101001
- **a**: 97 ⇒ 01100001
- **o**: 111 ⇒ 01101111

Quindi all'interno del computer la parola **Ciao** è scritta così:

10000011 01101001 01100001 01101111

## 3.6 Immagini in binario

Però, come fanno i computer a rappresentare foto, video e altri elementi grafici che si trovano oggi su uno schermo? Un video è costituito da una serie di immagini (tipicamente 30 al secondo). Ogni immagine è composta da una matrice di minuscoli puntini denominati **pixel**; ogni pixel è caratterizzato da un certo colore.

Se si definisce un codice numerico da attribuire ai colori, il gioco è fatto: per rappresentare un'immagine, sarà sufficiente indicare il numero del colore di ogni pixel sullo schermo.

Ad esempio una possibile codifica dei colori mediante numeri è il modello RGB.

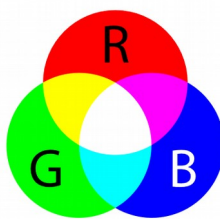













Figura 4

Il colore di ogni pixel è definito dalla combinazione dei tre colori di base rosso, verde e blu (da cui l'acronimo: Red, Green, Blue). Attenti a non confonderli con i colori primari che sono alla base di un altro modello usato in ambito di stampa, il modello CMYK (ciano, magenta, giallo, nero).

Il colore di ogni singolo pixel è definito dall'intensità dei tre colori che lo compongono. Ogni componente è definita da un numero di 8 bit, compreso quindi tra 0 e 255 (vedere tabella di pagina 10).

Qualche esempio:

										
<b>Red</b>	0	255	0	0	255	0	255	200	127	255
<b>Green</b>	0	0	255	0	255	255	0	150	127	255
<b>Blue</b>	0	0	0	255	0	255	255	10	127	255

Come si evince dalla precedente tabella, un pixel con questo colore  è rappresentato dalla tripletta RGB 0, 255, 255 e quindi in binario: 00000000 11111111 11111111. Se si applica questa notazione ad ogni pixel di una normale schermata da 1024x768 pixel, si ottengono più di 18 milioni di bit e se poi si considera un video con 30 immagini di questo tipo ogni secondo... appare lampante il motivo per cui un film occupa veramente molto spazio sui dischi dei nostri computer. Per ridurre un po' questo effetto, nel tempo gli informatici hanno messo a punto dei metodi per comprimere queste informazioni in modo che occupino meno memoria.

### 3.7 Suoni in binario

Il suono è costituito da vibrazioni dell'aria. Queste vibrazioni possono essere rilevate con un microfono che trasforma questo fenomeno fisico in un segnale elettrico che varia nel tempo. Se si rappresenta su un grafico l'andamento di questo segnale elettrico, si ottiene quella che viene chiamata la sua forma d'onda:

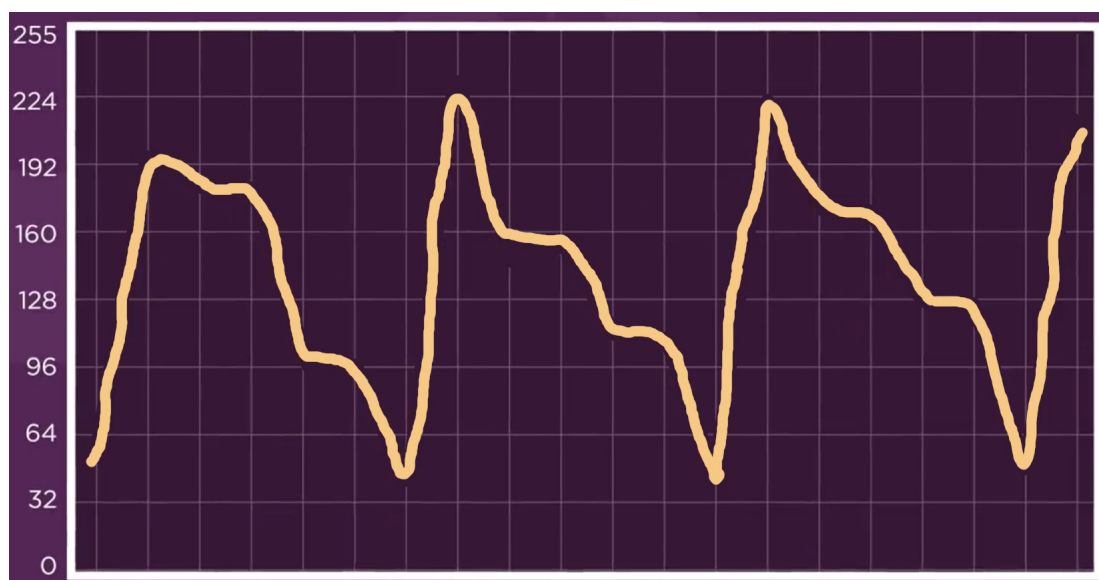


Figura 5

Con una certa approssimazione è possibile rappresentare in modo digitale questa forma d'onda misurandone la grandezza in istanti di tempo successivi. Si opera quindi un'azione di conversione dalla forma *analogica* del suono (la forma d'onda) alla forma *digitale* (la serie dei numeri che rappresenta la grandezza misurata nei vari istanti di tempo):

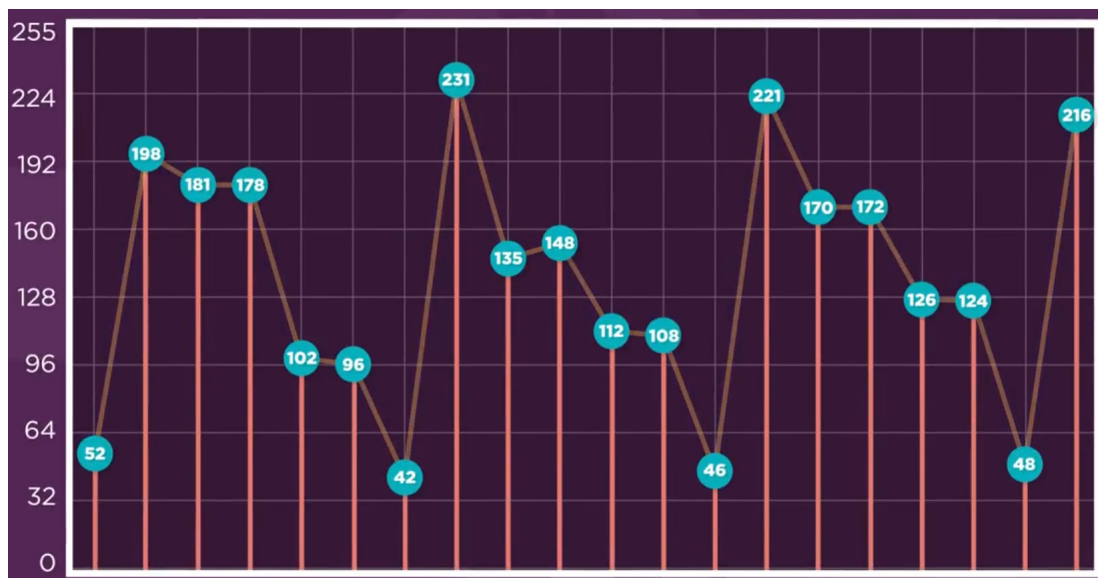


Figura 6

Per ridurre l'approssimazione, si può sia aumentare il numero di valori a disposizione per misurare la grandezza, ad esempio usando numeri a 16 bit (corrispondenti a 65.536 differenti valori) invece che a 8 bit (che consentono solo 256 differenti valori), sia aumentare la frequenza degli istanti di tempo in cui la forma d'onda viene misurata (cioè diminuire l'intervallo temporale tra una misurazione e la successiva). Entrambi questi approcci conducono ad avere una maggiore quantità di dati, sia per la memorizzazione che per la trasmissione.

## 4 Circuiti e logica

### 4.1 Organizzazione della lezione (2 ore)

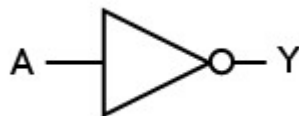
- Preparati guardando preventivamente il [video “Circuiti e logica”](#);
- preparati leggendo attentamente la descrizione presente più avanti in cui vengono esposti i contenuti del video con l’aggiunta di esempi e approfondimenti;
- guarda con gli studenti il [video “Circuiti e logica”](#);
- riepiloga alla lavagna i concetti principali ed esegui alcuni esempi ed esercizi con gli studenti:
  - disegna il simbolo della porta **NOT** e riepiloga il meccanismo di funzionamento e la relativa tabella di verità, fai esercitare anche qualche studente;
  - disegna il simbolo della porta **AND** e riepiloga il meccanismo di funzionamento e la relativa tabella di verità, fai esercitare anche qualche studente;
  - disegna il simbolo della porta **OR** e riepiloga il meccanismo di funzionamento e la relativa tabella di verità, fai esercitare anche qualche studente;
  - proponi esempi ed esercizi di analisi di circuiti, prima basati sull’unione di due porte e poi di tre, come nell’esercizio di pagina 16;
  - spiega il principio di funzionamento del circuito **sommatore** come descritto a pagina 17;
  - fai esercitare gli studenti a calcolare somme in colonna di numeri decimali come descritto a pagina 18;
  - fai esercitare gli studenti a calcolare somme in colonna di numeri binari come descritto a pagina 19;
- guarda nuovamente con gli studenti il [video “Circuiti e logica”](#) per fissare definitivamente i concetti appresi;
- vista la lunghezza e la complessità di questa lezione, è consigliabile dividerla in due sessioni e va valutato bene il livello di maturità della classe.

### 4.2 Introduzione

Si deve ora chiarire come faccia il computer a eseguire delle operazioni matematiche sui dati ricevuti in ingresso e archiviati in memoria, che si concretizzano in segnali elettrici accesi o spenti, ovvero i bit: uni e zeri. Grazie all’elaborazione elettronica di questi segnali è possibile produrre dei segnali di uscita che rappresentano il risultato dell’elaborazione richiesta. L’elaborazione di questi segnali elettrici avviene grazie a miliardi di minuscoli componenti elettronici, che insieme formano i circuiti di cui è costituito il computer.

La base dell’elaborazione elettronica è costituita da componenti denominati “porte logiche”: sono circuiti elementari che restituiscono in uscita un segnale che dipende in modo abbastanza semplice dai valori di uno o più segnali ricevuti in ingresso. In matematica si dice, in casi di questo genere, che l’uscita è “funzione” dei valori in ingresso.

### 4.3 Porta NOT



Si consideri la porta logica più semplice di tutte che ha un ingresso e un’uscita. Con un unico ingresso che può essere **1** o **0**, che possibili elaborazioni si possono immaginare?

1. un circuito che restituisce in uscita lo stesso segnale che riceve in ingresso, ma questo circuito è semplicemente... un filo!



- oppure un circuito che restituisce in uscita il segnale opposto a quello che riceve in ingresso, in altre parole il segnale di uscita **non** è mai uguale a quello di ingresso. Per questo motivo questa porta logica si chiama NOT, visto che “non” in inglese si dice “not”.

Questo circuito riceve in ingresso un segnale elettrico, ON o OFF, e lo inverte. Quindi, se il segnale che fornisci è **1**, il circuito ti risponde **0** e se dai al circuito **0**, questo ti restituisce **1**.

Se si chiama A il segnale di ingresso e Y quello di uscita, si può costruire la seguente tabella (denominata tabella di verità) che definisce il funzionamento della porta logica.

A	Y = NOT(A)
<b>0</b>	<b>1</b>
<b>1</b>	<b>0</b>

## 4.4 Porta AND

Se si considerano due segnali di ingresso, si possono definire molte porte logiche che danno in uscita un risultato che dipende dal valore dei due ingressi.



Il circuito rappresentato in figura riceve in ingresso due segnali, ognuno dei quali può essere **1** o **0**. Se almeno uno dei segnali in ingresso è **0**, allora anche il risultato in uscita è **0**. Se entrambi gli ingressi sono **1**, allora il risultato in uscita è **1**. In altre parole, il circuito restituisce **1** se e solo se il primo segnale e il secondo segnale sono entrambi **1**. Dal momento che la congiunzione “e” in inglese si dice “and”, allora questo circuito è chiamato AND.

Si può chiarire ulteriormente questo funzionamento, costruendo anche in questo caso la tabella di verità:

A	B	Y = A AND B
<b>0</b>	<b>0</b>	<b>0</b>
<b>0</b>	<b>1</b>	<b>0</b>
<b>1</b>	<b>0</b>	<b>0</b>
<b>1</b>	<b>1</b>	<b>1</b>

## 4.5 Porta OR

Ed ecco la terza porta logica fondamentale:



Anche questo circuito riceve in ingresso due segnali che possono essere **1** o **0**. Questa volta l'uscita è **1** se almeno uno dei segnali in ingresso è **1**. Se entrambi gli ingressi sono **0**, allora l'uscita è **0**. In altre parole, il circuito restituisce **1** se e solo se il primo segnale è **1** oppure il secondo segnale è **1**. Dal momento che la disgiunzione “oppure” in inglese si dice “or”, allora questo circuito è chiamato OR. Si osservi che, diversamente da come accade alle volte nel linguaggio parlato in cui “oppure” implica che le due alternative non possano essere entrambe vere, nel caso di questo circuito il risultato è **1** anche nel caso in cui **entrambi** gli ingressi siano **1**.

Questa è la tabella di verità della porta OR:

A	B	Y = A OR B
0	0	0
0	1	1
1	0	1
1	1	1

Esistono anche altre porte logiche, ma il meccanismo è sempre lo stesso: una regola che definisce come si comporta l'uscita in funzione degli ingressi. Le porte logiche sono quindi la concretizzazione di funzioni matematiche che acquisiscono in ingresso e restituiscono in uscita solo valori uno e zero. Queste porte logiche (o funzioni logiche) sono completamente definite dalle loro tabelle di verità.

È importante capire che una volta che abbiamo materializzato gli uni e zeri sotto forma di segnali elettrici, siamo in grado di costruire dei circuiti che realizzano queste porte logiche. E mettendo insieme tantissimi di questi circuiti, riusciamo a costruire i computer... non c'è altro!

### 4.5.1 Esercizio

Dopo aver chiarito al meglio il principio di funzionamento delle tre singole porte logiche presentate (NOT, AND e OR) è consigliabile far esercitare gli studenti con dei circuiti che uniscano tra loro diverse porte. È un esercizio di logica che permette di assimilare a fondo questi concetti.

Si inizia con l'analisi del funzionamento del circuito a partire dall'uscita Y:

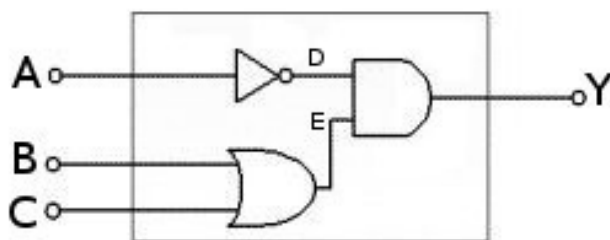


Figura 7

la porta logica più a destra è una porta AND, quindi l'uscita Y sarà **1** solo se entrambi i suoi ingressi (D ed E) saranno **1**; osserviamo quindi da cosa derivano i due ingressi della porta AND.

D è l'opposto di A (essendo l'uscita di una porta NOT che ha come ingresso il segnale A), quindi affinché Y sia **1**, D deve per forza essere **1** e quindi A può essere solo **0**.

Prendiamo ora in considerazione E (il secondo ingresso della porta AND) che è alimentato dall'uscita di una porta OR. Per fare in modo che anche E sia **1**, almeno uno dei due ingressi della porta OR dovrà essere **1**.

Ora arriviamo allo stesso risultato, analizzando il funzionamento a partire dagli ingressi A, B e C; per far questo, ci viene in aiuto la tabella di verità. Prima di tutto occorre elencare tutte le possibili combinazioni degli ingressi, che con tre bit sono otto:

A	B	C	Y
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Per ogni combinazione dei possibili valori di ingresso occorre ragionare sul circuito:

- ABC=000: se A=0 allora D=1, ma B e C sono entrambi 0, quindi E=0 e quindi Y=0,
- ABC=001: se A=0 allora D=1, in questo caso C=1 fa in modo che E=1 e quindi Y=1,
- ABC=010: se A=0 allora D=1, in questo caso B=1 fa in modo che E=1 e quindi Y=1,
- ABC=011: se A=0 allora D=1, in questo caso B=1 e C=1 implicano E=1 e quindi Y=1,
- ABC=100: se A=1 allora D=0, quindi *qualunque siano i valori di B e C*, Y sarà sempre 0.

E quindi ecco la tabella di verità completata:

A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Questo era solo uno dei tanti possibili esempi, combinando in modo arbitrario le tre porte logiche presentate, si possono creare tanti possibili esercizi su cui esercitarsi. **Si suggerisce di iniziare combinando solo due porte logiche** prima di passare a circuiti più complessi.

## 4.6 Sommatore

Come si è visto anche nel precedente esercizio, collegando insieme diverse porte logiche, è possibile creare circuiti più complessi che eseguono operazioni più complesse. Per esempio, è possibile costruire un circuito in grado di calcolare la somma di due numeri: un sommatore (in inglese ADDER).

In primo luogo, consideriamo ciò che si vuole realizzare, ragionando nel sistema decimale:

A	B	Y = A + B
0	0	0+0=0
0	1	0+1=1
1	0	1+0=1
1	1	1+1=2

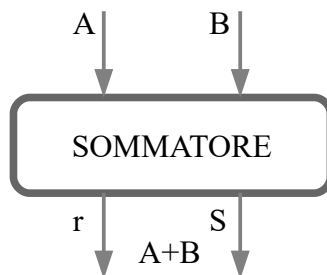
Poi aggiungiamo la conversione in binario:

A	B	Y = A + B DECIMALE	Y = A + B BINARIO
0	0	0	0
0	1	1	1
1	0	1	1
1	1	2	10

Attenzione! La rappresentazione binaria del numero decimale 2 necessita di due cifre, come quando si sommano due numeri decimali di un'unica cifra: si ottiene un risultato con due cifre. Infatti:  $5+5=10$ ! Dobbiamo quindi rappresentare ciò che stiamo facendo in questo modo:

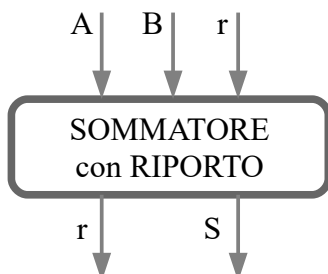
A	B	A+B	
		r	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Grazie a quanto visto finora, non dovrebbe essere difficile immaginare come si possano combinare alcune porte logiche e costruire un circuito che, a partire da due ingressi, fornisca il risultato con due uscite, cioè su due fili:



Finora si è analizzato il funzionamento di un sommatore che riceve in ingresso numeri costituiti da una sola cifra (un unico bit), ma con un ulteriore sforzo si può realizzare un blocco che permette di calcolare la somma di numeri in ingresso con quante cifre si vuole.

Per far questo, si deve tenere presente che, in generale, nella somma di due cifre in una certa posizione va tenuto in conto l'eventuale riporto derivante dalla somma delle cifre nella posizione precedente. Analogamente, il risultato in una certa posizione è costituito dalla cifra risultante per quella posizione più l'eventuale riporto per le due cifre da sommare nella posizione successiva.



Può sembrare complicato, ma rappresenta esattamente ciò che si fa normalmente eseguendo la somma in colonna. Ecco un esempio nel sistema decimale:

$$\begin{array}{r}
 1 \ 1 \ 1 \\
 3 \ 2 \ 5 \ + \\
 \hline
 7 \ 9 \ 8 \ = \\
 \hline
 1 \ 1 \ 2 \ 3
 \end{array}$$

Ovvero:

- $5+8=13$ , scrivo 3 e riporto 1;
- $1+2+9=12$ , scrivo 2 e riporto 1;
- $1+3+7=11$ , scrivo 1 e riporto 1;
- $1+0+0=1$ , scrivo 1 e ho così completato l'operazione.

A questo scopo è utile considerare le due cifre fornite in uscita dal sommatore, una come la somma e l'altra come il riporto che dovrà poi essere sommato alle cifre di peso maggiore. Di conseguenza, questo sommatore dovrà avere tre ingressi, per tenere conto dell'eventuale riporto proveniente dalle cifre di peso inferiore.

Mettendo poi insieme tanti di questi sommatori con riporto si ottiene il circuito mostrato in Figura 8, che permette di realizzare l'operazione desiderata.

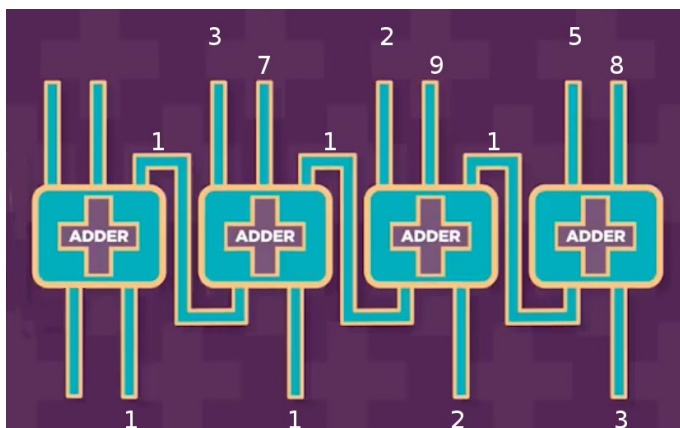


Figura 8

Ricordiamo che nei fili non possono circolare “cifre decimali”, ma solo segnali elettrici accesi o spenti, quindi uni o zeri. Vediamo quindi come funziona per davvero in binario... è la stessa cosa che accade nel sistema decimale!

Si consideri per esempio la somma  $5+7=12$ :

- convertendo il 5 in binario si ottiene **101** (come spiegato a pagina 9)
- convertendo il 7 in binario si ottiene **111**

e si svolga l'operazione in colonna:

$$\begin{array}{r}
 1\ 1\ 1 \\
 1\ 0\ 1\ + \\
 \hline
 1\ 1\ 1\ = \\
 \hline
 1\ 1\ 0\ 0
 \end{array}$$

ovvero:

- $1+1=10$  (che corrisponde a 2 in decimale), scrivo **0** e riporto **1**;
- $1+0+1=10$ , scrivo **0** e riporto **1**;
- $1+1+1=11$  (che corrisponde a 3 in decimale), scrivo **1** e riporto **1**;
- $1+0+0=1$ , scrivo **1** e ho così completato l'operazione;
- per chiudere il cerchio si deve verificare che **1100** convertito in decimale corrisponde proprio a 12!

Si può quindi finalmente visualizzare quest'operazione nel precedente schema di sommatori con riporto:

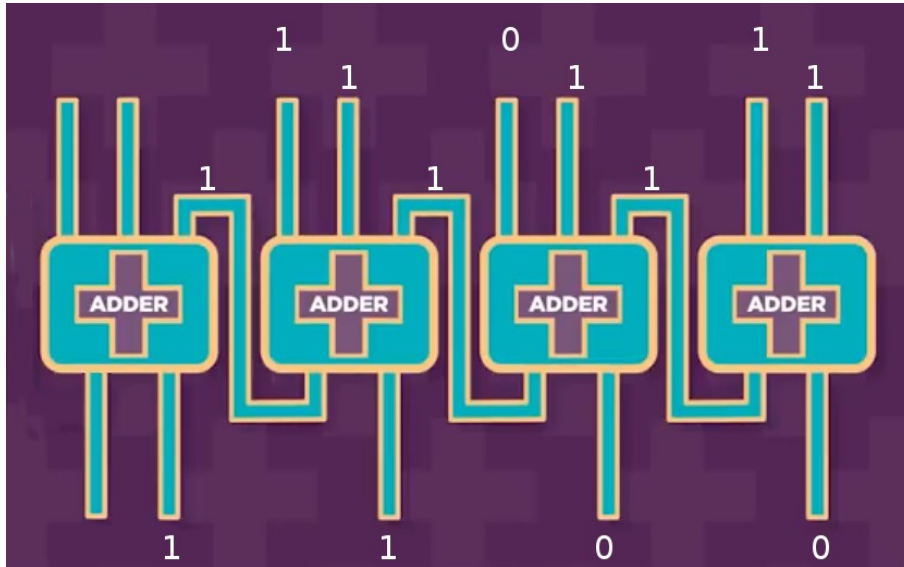


Figura 9

A questo punto si può immaginare che sia altrettanto possibile costruire circuiti in grado di svolgere tutte le operazioni di base (sottrazioni, moltiplicazioni ecc.) che, unite assieme ed eseguite a velocità spettacolari, permettono al computer di fare tutto ciò a cui siamo ormai abituati.

L'evoluzione tecnologica ha portato ad avere computer sempre più piccoli e sempre più veloci. E l'incremento della velocità è stato anche aiutato dall'aumento di miniaturizzazione. Va infatti tenuto presente che il passaggio della corrente elettrica non è istantaneo: l'elettricità si muove ad una velocità finita, prossima a quella della luce, ma pur sempre finita. Se, quindi, i circuiti interni al computer sono più piccoli, i segnali elettrici devono percorrere percorsi più brevi, ci impiegano quindi meno tempo e di conseguenza la velocità di elaborazione è più elevata.



## 5 Memoria, CPU, ingressi e uscite

### 5.1 Organizzazione della lezione (30 minuti)

- Preparati guardando preventivamente il [video “Memoria, CPU, ingressi e uscite”](#);
- preparati leggendo attentamente la descrizione presente più avanti in cui vengono esposti i contenuti del video;
- guarda con gli studenti il video [video “Memoria, CPU, ingressi e uscite”](#);
- riepiloga alla lavagna i concetti principali:
  - riepiloga i **quattro concetti chiave** che contraddistinguono tutti i computer e in cosa consistono ingressi e uscite;
  - disegna alla lavagna l'immagine di Figura 14 e spiega il processo che avviene dalla pressione di un tasto alla visualizzazione della lettera sullo schermo;
- guarda nuovamente con gli studenti il [video “Memoria, CPU, ingressi e uscite”](#) per fissare definitivamente i concetti appresi.

### 5.2 Introduzione

Una macchina, per essere considerata un computer, deve essere in grado di:

- acquisire dei dati in ingresso (INPUT),
- memorizzarli,
- elaborarli in vari modi,
- fornire dati in uscita come risultato dell'elaborazione svolta (OUTPUT).

Si è già accennato a ingressi e uscite:

- gli ingressi rilevano fenomeni nel mondo esterno e li convertono in dati binari, per esempio: la tastiera del computer, lo schermo tattile (touchscreen), un microfono, una telecamera o anche dei sensori come un termometro o un accelerometro.
- al contrario, le uscite sono la conversione in forma fisica dei risultati ottenuti dalle elaborazioni svolte: dei testi o delle immagini su uno schermo, della musica prodotta in delle casse, la stampa su un foglio di carta, un oggetto stampato da una stampante 3D, il movimento di un robot o anche un flusso di dati inviati ad un altro computer.

### 5.3 CPU e memoria

Presentiamo ora un semplice esempio per evidenziare come i dati in ingresso viaggiano tra l'unità centrale di elaborazione (la CPU – Central Processing Unit) e la memoria, per trasformarsi nelle uscite richieste.

#### 5.3.1 Esempio: visualizzazione di una lettera

Quando si preme un tasto sulla tastiera, per esempio la lettera “B”, la tastiera invia al computer un codice associato a questo tasto. Andando a consultare la tabella ASCII a pagina 11, si scopre che il codice associato alla lettera “B” è 66, che in binario corrisponde a: **01000010**.

Alla ricezione di questo codice, la CPU deve provvedere a visualizzare questa lettera sullo schermo, questi sono i passi necessari:

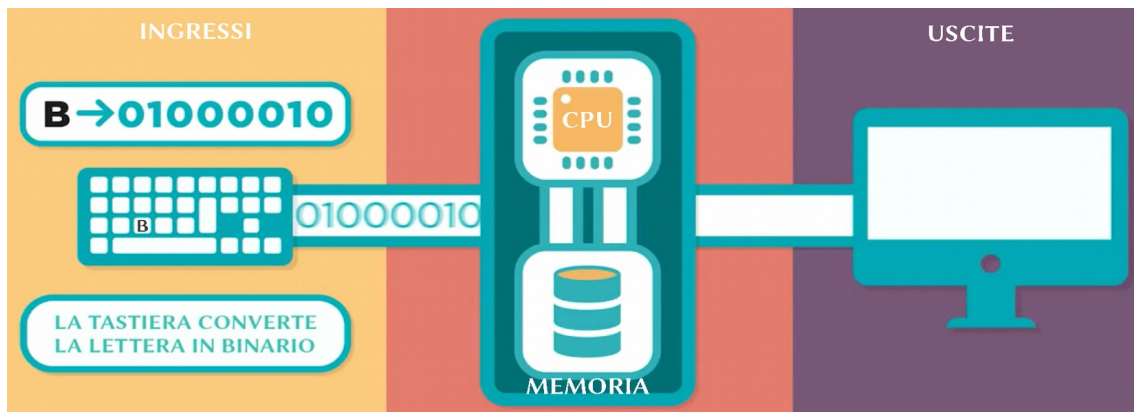


Figura 10

- la CPU chiede alla memoria come è fatta la lettera che corrisponde al codice ASCII **01000010**,
- la memoria ritrova i dati che descrivono come è fatta la lettera corrispondente a tale codice e li invia alla CPU,
- la CPU (o meglio, il programma in esecuzione nella CPU), a partire da questi dati ricava la sequenza di pixel che devono essere colorati sullo schermo,
- la CPU invia quindi allo schermo la sequenza di pixel da visualizzare,
- lo schermo, che in questo caso è il dispositivo di uscita, riceve la sequenza di segnali elettrici che corrispondono agli uni e zeri che indicano con quali colori accendere ogni suo pixel.

Tutto ciò avviene così velocemente che sembra istantaneo, ma per visualizzare ogni lettera, un computer esegue migliaia di istruzioni, iniziando dall'istante in cui il tuo dito preme il tasto.

### 5.3.2 Esempio: visualizzazione di una lettera in grassetto

Si ripercorra ora lo stesso esempio, ipotizzando però di volere cambiare la visualizzazione della lettera “B” in modo che sia visualizzata in **grassetto**:

- il modo specifico per indicare questa caratteristica del testo dipende dal programma che stiamo usando; solitamente è necessario – dopo aver selezionato la lettera “B” – cliccare su un pulsante grafico che indica questa caratteristica, ma per questo esempio immaginiamo di usare una combinazione di tasti sulla tastiera;
- come nell'esempio precedente, la tastiera invia al computer il codice ASCII associato alla particolare combinazione di tasti corrispondenti al grassetto;
- nell'esempio precedente, la CPU aveva recuperato dalla memoria le istruzioni necessarie per visualizzare la forma base della lettera “B”, in questo caso la CPU – che già sa che la lettera “B” è stata selezionata sullo schermo – deve invece richiedere le istruzioni grafiche della versione in grassetto di quella stessa lettera;
- dopodiché il procedimento prosegue esattamente come prima.

## 6 Hardware e software

### 6.1 Organizzazione della lezione (30 minuti)

- Preparati guardando preventivamente il [video “Hardware e software”](#);
- preparati leggendo attentamente la descrizione presente più avanti in cui vengono esposti i contenuti del video;
- guarda con gli studenti il [video “Hardware e software”](#);
- riepiloga alla lavagna i concetti principali:
  - definisci **hardware** e **software** e sollecita gli studenti a suggerire molti esempi;
  - definisci la **CPU** e sottolinea come sia costituita da componenti più piccoli e specializzati;
  - definisci il **codice binario** e spiega come questo comandi l’operatività della CPU;
  - chiarisci la relazione tra codice binario e **linguaggi di programmazione di alto livello** più comprensibili per noi umani;
  - definisci il **sistema operativo** e, prima di elencarle, verifica se gli studenti sanno suggerirne le funzioni principali;
- guarda nuovamente con gli studenti il [video “Hardware e software”](#) per fissare definitivamente i concetti appresi.

### 6.2 Introduzione

In questa lezione conclusiva, vengono presentati tre concetti molto importanti: l’hardware, il software e il sistema operativo che ne gestisce l’interazione.

### 6.3 Hardware, software

L’**hardware** è facile da definire (Figura 11): è costituito da tutti gli elementi fisici che si vedono dentro ai computer e da tutti i dispositivi ad esso connessi (circuiti integrati, fili, altoparlanti, connettori, dispositivi di ingresso e uscita e molto altro).



Figura 11



Figura 12

E il **software...** è tutto ciò che non si vede (Figura 12): l'insieme dei programmi eseguiti dal computer: giochi, pagine web visualizzate tramite un browser, musica, video e applicazioni di ogni tipo.

L'**unità centrale di elaborazione (CPU)** (Figura 13) è il principale circuito integrato che controlla il funzionamento di tutte le altre parti del computer.

All'interno, la CPU è suddivisa in componenti più piccoli, ognuno specializzato nell'esecuzione di specifiche attività.

Uno di questi è quello che esegue tutte le operazioni matematiche e logiche di cui abbiamo visto qualche semplice esempio nel capitolo Circuiti e logica. Ve ne sono altri per comunicare con le altre parti del computer. La CPU decide quali componenti usare e quando, in base alle istruzioni ricevute dal programma che sta eseguendo. Se la CPU riceve un'istruzione di SOMMA, sa che deve utilizzare il componente matematico-logico per eseguire un calcolo, mentre un'istruzione MEMORIZZA dice alla CPU di usare un componente diverso per archiviare un dato in memoria.



Figura 13

Anche le istruzioni che costituiscono il programma in esecuzione sono state precedentemente archiviate in memoria e vengono eseguite una dopo l'altra dalla CPU. I nomi usati (SOMMA e MEMORIZZA) sono solo simbolici, all'interno del computer viaggiano codificati come numeri binari, sono quindi anch'essi dei segnali elettrici accesi e spenti. Questo tipo di programma è costituito da istruzioni molto semplici e si chiama **codice binario**, che è la forma più semplice di software a cui viene ricondotto ogni programma scritto in qualunque linguaggio: anche i blocchi di Code.org!

Data l'enorme complessità dei sistemi attuali, non sarebbe assolutamente possibile scrivere i programmi direttamente in codice binario, quindi sono stati creati tanti diversi linguaggi di programmazione, ognuno adatto alla risoluzione di una certa tipologia di problemi. Questi linguaggi hanno istruzioni con nomi più facili da ricordare e si occupano di eseguire attività di più alto livello. Per esempio, per disegnare un rettangolo sullo schermo serve un'unica istruzione: ci pensa poi un programma apposito a scomporre quest'istruzione di alto livello in centinaia o migliaia di semplici istruzioni binarie che la CPU è in grado di capire.



Ma sui nostri computer non viene eseguito un unico programma alla volta, nascono quindi spontanee le seguenti domande:

- quando si accende il computer, “chi” si occupa di tutte le operazioni necessarie all’avvio?
- “chi” si occupa di caricare i vari programmi in memoria?
- “chi” decide che certi programmi possono interagire con alcuni dispositivi di ingresso e uscita e altri no?
- e ancora, oggi tutti sono abituati ad usare tanti programmi contemporaneamente, come si comporta la CPU quando si ascolta musica mentre si naviga sul web e si chiacchiera con i propri amici?

## 6.4 Sistema operativo

È il sistema operativo che si occupa di tutto ciò: il programma principale che “accende” il nostro dispositivo e lo predispose per l’operatività, svolgendo moltissime attività ordinarie per mantenere il



Figura 14

funzionamento corretto. È il sistema operativo che permette di copiare sui dischi interni dei nuovi programmi e caricarli in memoria ad ogni loro esecuzione e che permette o meno l’uso di certi dispositivi hardware da parte di certi programmi o anche solo la lettura o scrittura su alcuni archivi e non su altri.

E quando si pensa che il proprio computer stia eseguendo molti programmi contemporaneamente, in realtà, è il sistema operativo che velocemente fa eseguire alla CPU un programma dopo l’altro ognuno per qualche frazione di secondo.

Nella Figura 14 sono illustrate le quattro famiglie di Sistemi Operativi più diffusi nei computer moderni.

Grazie a tutte queste conoscenze, ci si rende conto di come un computer non sia altro che una macchina capace solo di svolgere semplicissime operazioni molto velocemente e in grado di ricordare benissimo enormi quantità di dati; affinché questa macchina possa realizzare qualcosa di utile, è necessaria la nostra creatività per scrivere i programmi che diano vita alle nostre idee.

## 7 Mappatura con Proposta CINI

Qui di seguito viene sinteticamente indicato come il materiale proposto in questa guida soddisfa quanto previsto dalla “Proposta di Indicazioni Nazionali per l’insegnamento dell’informatica nella scuola”, elaborata dal CINI<sup>3</sup> sul modello di analoghi documenti del MIUR<sup>4</sup> relativi ad indicazioni curriculari, per articolare la visione culturale e scientifica della comunità universitaria dell’informatica italiana relativa all’insegnamento dell’informatica nella scuola. Ricordiamo che, analogamente a quanto proposto in altri paesi europei (p.es. il Regno Unito), la proposta del CINI è strutturata su cinque ambiti, di cui tre (*algoritmi, programmazione, dati e informazione*) possono essere definiti “tecnici”, dal momento che fanno riferimento agli aspetti puramente scientifici e tecnologici della disciplina, mentre altri due considerano l’uso responsabile e l’uso come strumento di espressione personale (*consapevolezza digitale e creatività digitale*).

Il documento integrale è disponibile sul sito del Gruppo di Lavoro “Informatica e Scuola del CINI (<http://consorzio-cini.it/gdl-informatica-scuola>).

I contenuti di questa guida soddisfano i seguenti punti della **Proposta di Indicazioni Nazionali per l’insegnamento dell’Informatica nella Scuola**:

- **Traguardi scuola primaria:**
  - inizia a riconoscere la differenza tra l’informazione e i dati;
  - esplora la possibilità di rappresentare dati di varia natura (numeri, immagini, suoni, ...) mediante formati diversi, anche arbitrariamente scelti;
  - sa riconoscere la presenza dei computer nei dispositivi tecnologici della vita quotidiana.
- **Obiettivi 3<sup>^</sup> primaria:**
  - ambito consapevolezza digitale: riconoscere usi dell’informatica e delle sue tecnologie nella vita comune.
- **Obiettivi 5<sup>^</sup> primaria:**
  - ambito consapevolezza digitale: conoscere le principali componenti hardware e software dei dispositivi che usa.
- **Traguardi scuola secondaria di primo grado:**
  - classifica le tipologie di dati (es.: numerici, testuali, ...);
  - conosce l’architettura di principio (fisica e funzionale) di un sistema di elaborazione digitale;
  - riconosce le componenti hardware e software dei sistemi di elaborazione digitale.
- **Obiettivi secondaria di primo grado:**
  - ambito consapevolezza digitale: comprendere i principi fondamentali dell’architettura e del funzionamento (hardware e software) di sistemi e dispositivi informatici.

<sup>3</sup> Consorzio Interuniversitario Nazionale per l’Informatica

<sup>4</sup> Ministero dell’Istruzione dell’Università e della Ricerca