

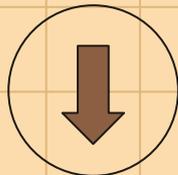
# La somma binaria

Come fa i conti una calcolatrice, ma  
scendendo proprio nel profondo



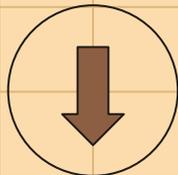
By Francesco Tucci - Linux Day 2022

# Io sono Francesco



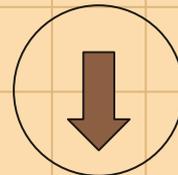
Professione

Lavoro nei datacenter di  
una grande  
multinazionale



Hobby e vita

Raspberry, Arduino,  
Fotografia, due gatti,  
giravo il mondo



Sistemi  
operativi

Tutti, scelgo quel che mi  
serve, per quel che devo  
fare



# Di cosa parliamo oggi

01

Numeri binari

Come si converte un decimale in binario e come si rappresenta

03

Porte logiche

Gli operatori che fanno le operazioni sui bit trasformati in tensioni

02

Elettronica digitale

Segnali, tensioni, transistor, quel che serve per maneggiare i bit nei chip

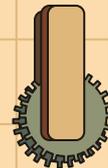
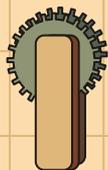
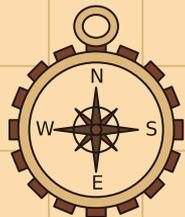
04

Somme in un chip

Come un circuito elettronico somma due numeri

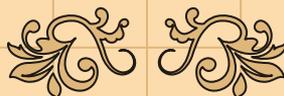


# Iniziamo!



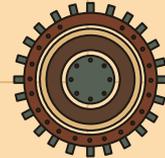
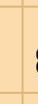
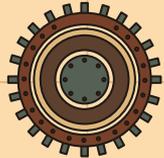
Il viaggio parte da due numeri decimali, convertiti in binario, salvati in un circuito elettronico che conosce solo livelli di tensione, che poi è in grado di sommarli e fornire il risultato.

E non è magia





01



# I numeri binari

Al mondo ci sono 10 persone che capiscono  
il binario, chi sì e chi no.



# Le basi

```
00111100011100
0001111100111
1111011110111
0111011000000
1000001110111
1000100100111
1100110010111
1111000010000
11000010011110
```

Siamo abituati a contare in decimale, con i numeri da **0** a **9**.  
Quando arriviamo a 9, mettiamo 0 e un 1 davanti

7, 8, 9, 10, ...

In binario le cifre sono solo due: **0** e **1**

0, 1, 10, 11, 100, ...



# La conversione

Valore del bit  
espresso in  
potenze di 2

128	64	32	16	8	4	2	1
$2^7$	$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
1	1	1	1	1	1	1	1

Valore del bit  
in decimale

bit

8 bit = 1 Byte



# Da binario a decimale

11010001 → 128+64+16+1 = 209

128	64	0	16	0	0	0	1
$2^7$	$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
1	1	0	1	0	0	0	1

bit

8 bit = 1 Byte



# Cose curiose



- In un byte il numero massimo è 255 (ma sono 256 valori, pari a  $2^8$ , 8 come i bit in un byte)
- Se il primo a dx è 1, è dispari
- Se voglio fare con segno sacrifico il primo bit a sx e i valori sono da -128 a 127



# Fare la somma

Decimale

```
156 +  
 89 =  
-----  
245
```

```
6+9=15    -> 5 r1  
r1+5+8=14 -> 4 r1  
r1+1=2     -> 2
```

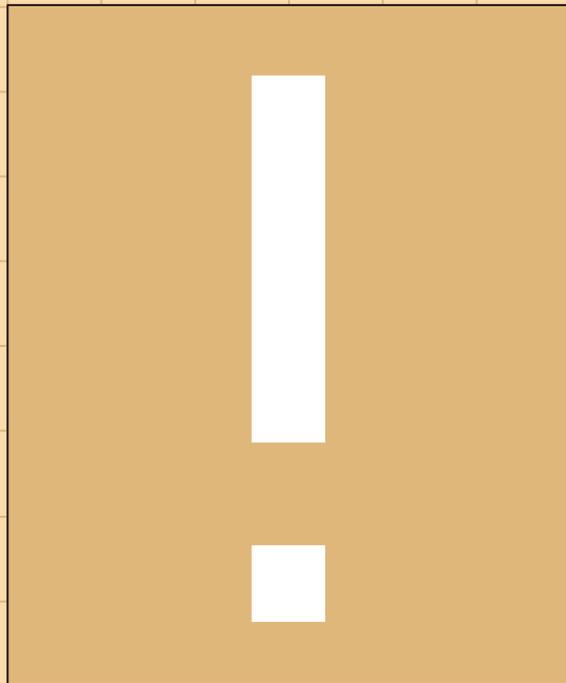
Binario

```
10011100 +  
01011001 =  
-----  
11110101
```

```
0+1=1    -> 1  
0+0=0    -> 0  
1+0=1    -> 1  
1+1=10   -> 0 r1  
r1+1+1=11 -> 1 r1  
r1+0+0=1 -> 1  
0+1=1    -> 1  
1+0=1    -> 1
```



# Overflow!



In decimale puoi fare

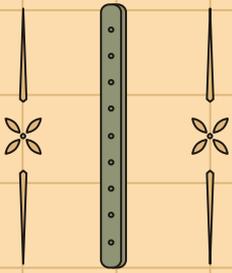
```
200 +  
200 =  
400
```

In binario

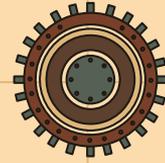
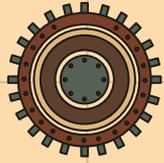
```
11001000 +  
11001000 =  
110010000
```

400 occupa 9 bit, non ci sta in 8 ed è un problema!





02



# Elettronica digitale

Rendiamo grazie sempre all'inventore del transistor!



# Tensioni, non carta

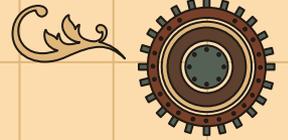
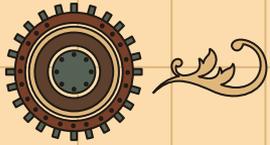


I dati che usiamo noi, appunti, conti, scritti, ... sono solitamente inchiostro su carta.

Nei calcolatori non c'è inchiostro e non c'è carta.

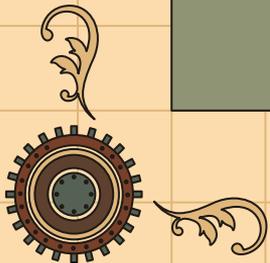
I dati possono essere gestiti solo con segnali elettrici

<b>C'è tensione:</b>	<b>valore binario 1</b>
<b>Non c'è tensione:</b>	<b>valore binario 0</b>



Pare che adesso le cose si facciano  
difficili.  
Come gestiamo delle tensioni al posto  
di numeri su carta?

L'ingegneria, nei tempi, ha fatto i miracoli!



# I numeri vanno memorizzati



Il modo più facile per memorizzare un valore di tensione è un condensatore.

Se è carico, c'è tensione - valore 1

Se è scarico, non c'è tensione - valore 0

8 condensatori fanno una cella di memoria da un Byte

Ho semplificato TANTISSIMO

# Quante celle di memoria



8 bit → massimo 255  
2x8 bit → massimo 65.535  
4x8 bit → massimo 4.294.967.295

Con celle di memoria da 32 bit iniziamo a ragionare (ci sarebbero anche i decimali, ma non sottilizziamo)

# E come le trovo?



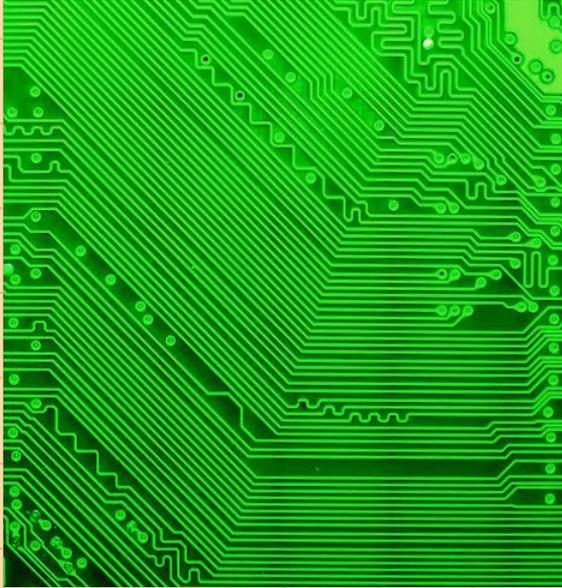
Ogni cella ha un suo indirizzo.  
Anche gli indirizzi sono espressi in binario.

Quando sentite “sistema a 16, 32 o 64bit” è la  
dimensione degli indirizzi delle celle di  
memoria.

64 bit vuol dire che può avere  
18.446.744.073.709.551.615 celle di memoria

Milioni di miliardi di miliardi

# Spostare i bit

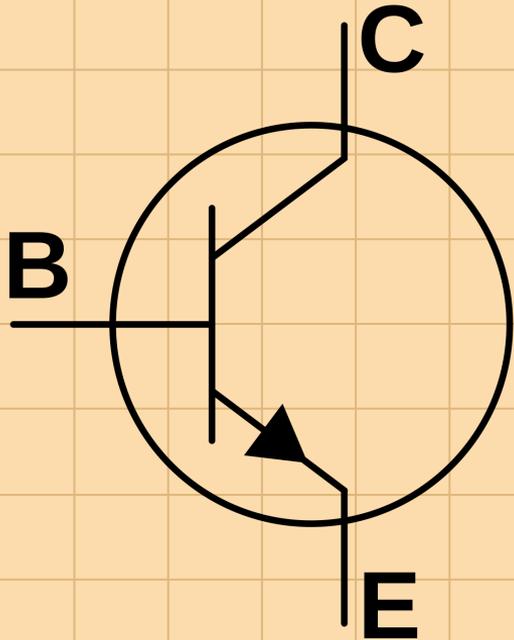


I segnali elettrici vengono trasmessi su migliaia di piste di rame sulle schede elettroniche. Immaginate milioni di segnali che viaggiano tutti sincronizzati.

Sincronia → clock

Più è bassa la differenza di tensione tra 0 e 1, più è veloce il cambio tra uno e l'altro

# Il transistor



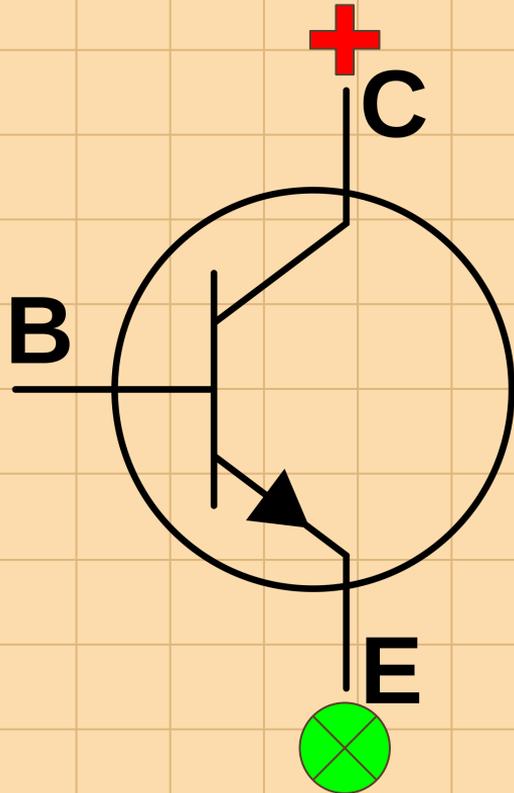
La più grande invenzione del nostro secolo.  
O una delle più grandi, fate voi.

Funziona in due modi

1. Amplifica i segnali
2. Funziona da interruttore

A noi serve la 2

# Interruttore digitale



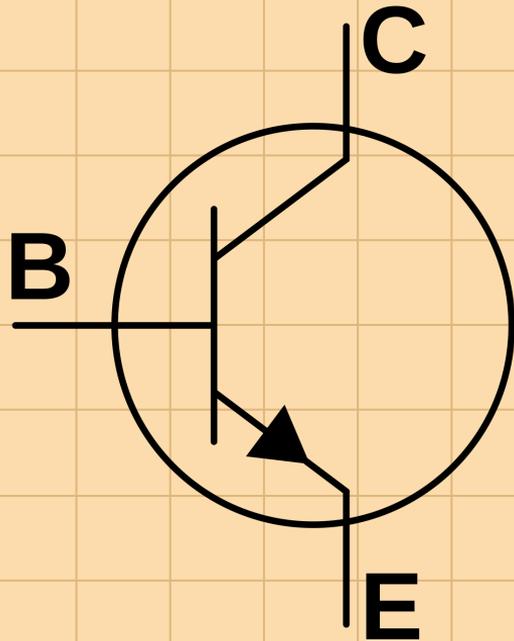
(ehm, semplifico, eh!)

Applico una tensione sul Collettore  
Metto una lampadina sull'Emettitore

Se collego la Base a **0V** (terra) la lampadina  
resta **spenta**

Se collego una tensione di almeno **0,7V** alla  
Base, la lampadina si **accende**

Ma a cosa serve questa cosa?



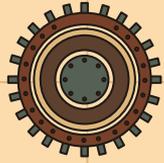
**A TUTTO!**

Con milioni e milioni di transistor, piccolissimi, messi nel modo giusto, con segnali che passano al loro interno, noi facciamo andare avanti il mondo.





03



# Le porte logiche



Se piove  nevicando prendi l'ombrello



# AND, OR, NOT



Spesso nella nostra vita ci troviamo nella  
posizione di dover scegliere

Fai questo **oppure** quello  
Fai questo **e** quello  
**Non** fare questo

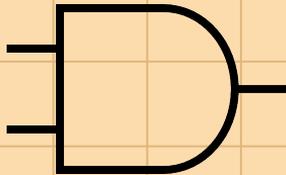
Chi programma lo fa spesso nelle **IF**



# Le tabelle della verità

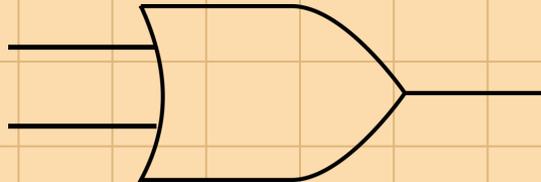
AND

È VERA solo se  
i due valori  
in ingresso  
sono entrambi  
veri



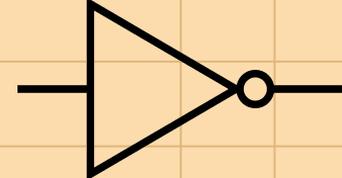
OR

È VERA quando  
almeno uno dei  
due valori in  
ingresso è  
VERO



NOT

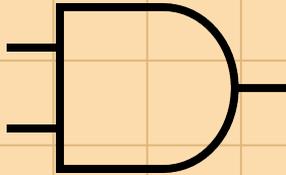
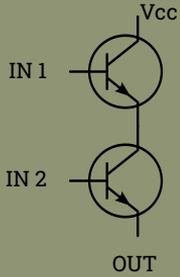
Inverte il  
valore in  
ingresso



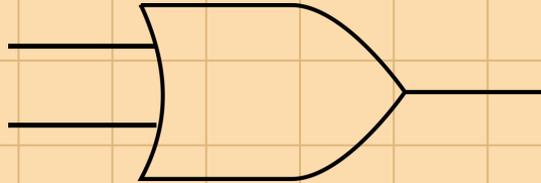
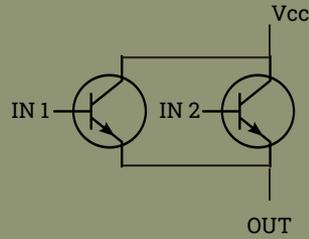
# Schemi interni

*Molto, molto semplificati*

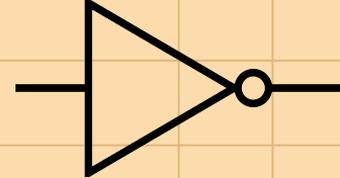
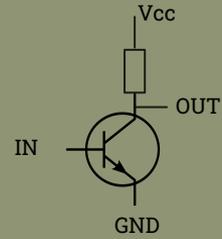
AND



OR



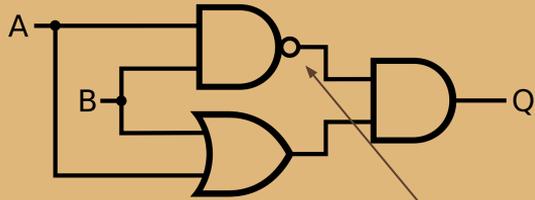
NOT



# Altre porte



Una XOR è  
composta da



Il pallino è  
una NOT

7 transistor

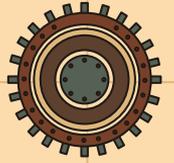
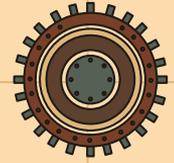
Per rispondere ad altre esigenze, spesso combinando, AND, OR e NOT, si possono creare porte con tabelle della verità diverse.

Ad esempio la XOR (*figlia di KMER*) che ci servirà dopo.

A	B	→	Q
0	0	→	0
0	1	→	1
1	0	→	1
1	1	→	0



04



Somma aritmetica

Solo la somma, che le altre operazioni sono difficili e il tempo è poco

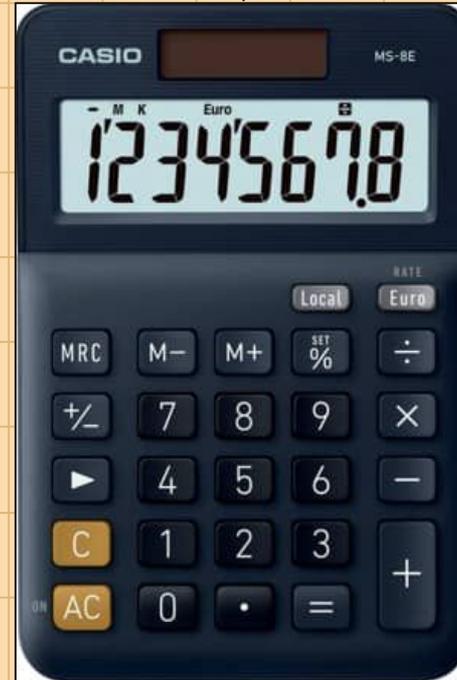


# Avete presente la calcolatrice da 2€?

- Costa poco
- Ha pochi tasti
- Ha un display LCD
- Fa poche operazioni

...ma...

È più complicata di quello  
che si possa immaginare



# Fare una somma

1 **Scrivo un addendo**

Lo vedo in decimale sul display  
Viene salvato in binario in una memoria  
.....

2 **Premo +**

Primo numero terminato  
So che operazione devo fare

4 **Premo =**

Primo numero terminato  
Avvio le operazioni di calcolo

3 **Scrivo l'altro addendo**

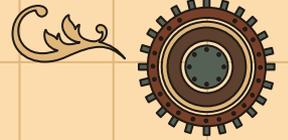
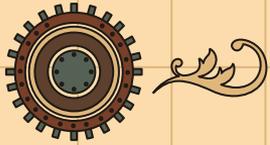
Lo vedo in decimale sul display  
Viene salvato in un'altra memoria  
.....

5 **Fa il conto**

Sposto le due celle di memoria nell'ALU  
Faccio il conto

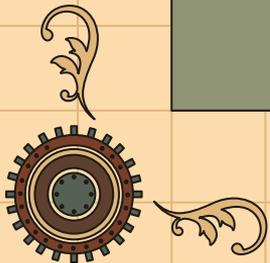
6 **Lo vedo sul display**

Il risultato sovrascrive la prima cella di memoria e viene visualizzato sul display

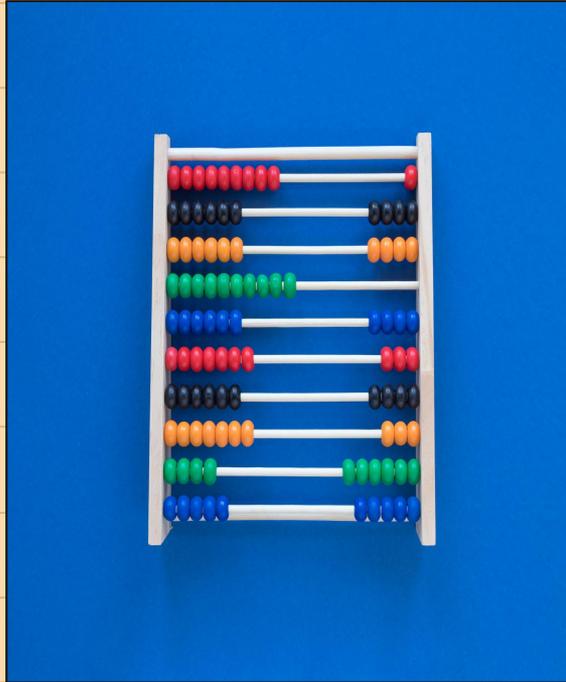


Dobbiamo sommare due numeri binari  
all'interno di un circuito elettrico in  
cui ci sono solo tensioni.

Non abbiamo foglio, penna e gomma



# Le basi



$$0 + 0 = 0$$

$$1 + 0 = 1$$

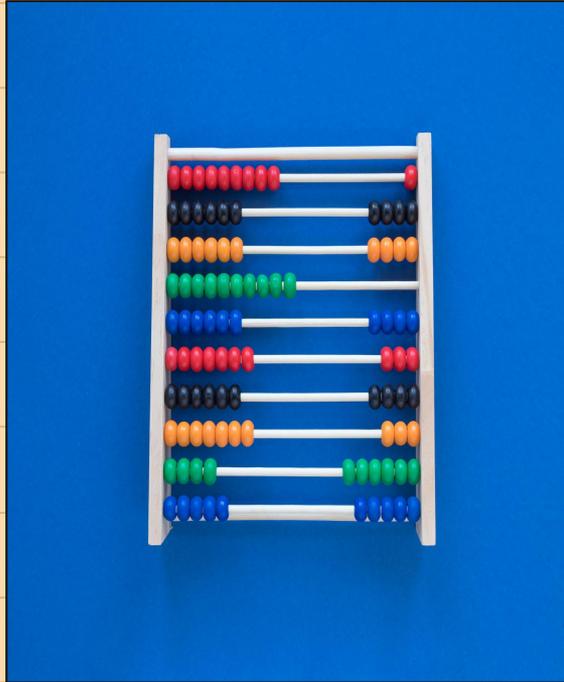
$$0 + 1 = 1$$

$$1 + 1 = \cancel{1}0 \text{ con riporto di } 1$$

**Devo gestire i riporti**

**Sia quello che genero che quello che mi arriva**

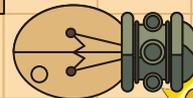
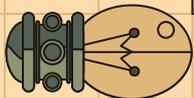
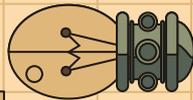
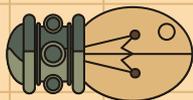
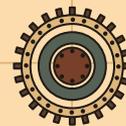
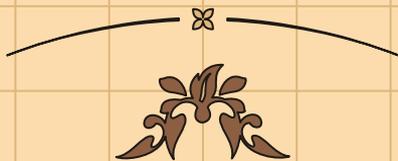
# La tabella della verità completa



$R0 + 0 + 0 = 0 \ R0$   
 $R0 + 1 + 0 = 1 \ R0$   
 $R0 + 0 + 1 = 1 \ R0$   
 $R0 + 1 + 1 = 0 \ R1$   
 $R1 + 0 + 0 = 1 \ R0$   
 $R1 + 1 + 0 = 0 \ R1$   
 $R1 + 0 + 1 = 0 \ R1$   
 $R1 + 1 + 1 = 1 \ R1$

Ho 3 ingressi, 2 uscite e 8 possibili combinazioni per ogni coppia di bit da sommare.

**Mi serve un  
circuito!**



# Il Full Adder somma due bit con i riporti



**A e B** sono i due bit in ingresso



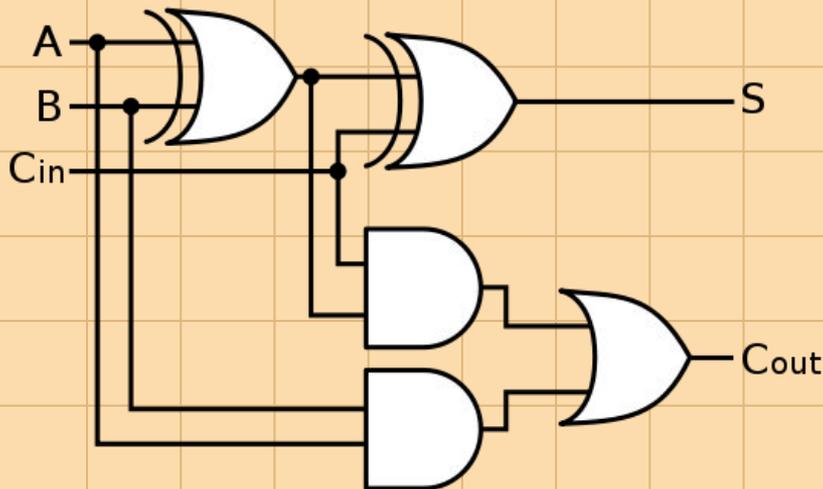
**C<sub>in</sub>** è il riporto in ingresso dall'adder precedente



**C<sub>out</sub>** è il riporto in uscita verso l'adder successivo



**S** è il risultato della somma



2 XOR (7+7 transistor)  
2 AND (2+2 transistor)  
1 OR (2 transistor)  
Totale → 20 transistor

# Per una somma a 4 Byte

32 full adder

Uno in cascata  
all'altro.

$5 \times 32 = 160$

Porte logiche

Il tutto solo  
per fare una  
somma.  
Per le altre è  
ancora più  
difficile

4.294.967.295

Numero massimo  
ottenibile

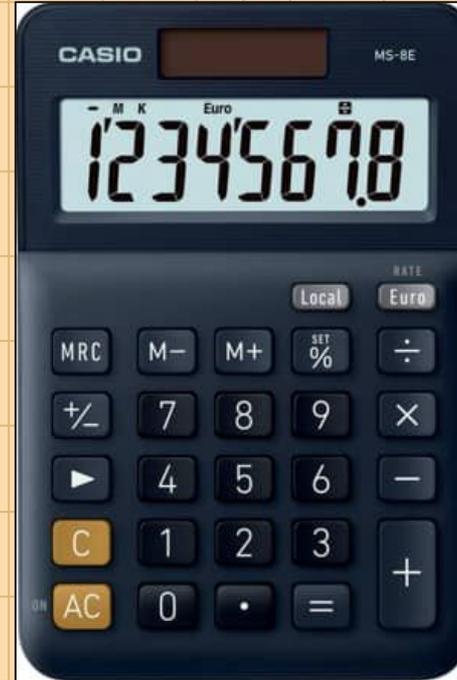
$20 \times 32 = 640$

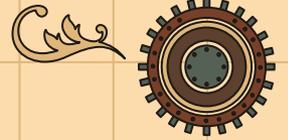
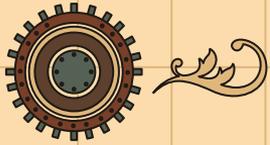
Transistor  
(solo per la somma)

# Nella calcolatrice c'è di più

- Altre operazioni
- Gestione della tastiera
- Memoria temporanea
- Controller del display
- Display LCD
- Circuito di alimentazione

Ed è un dispositivo che fa poco e costa pochissimo, ci sono alcune migliaia di transistor.

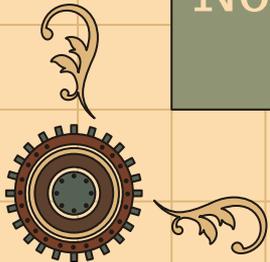


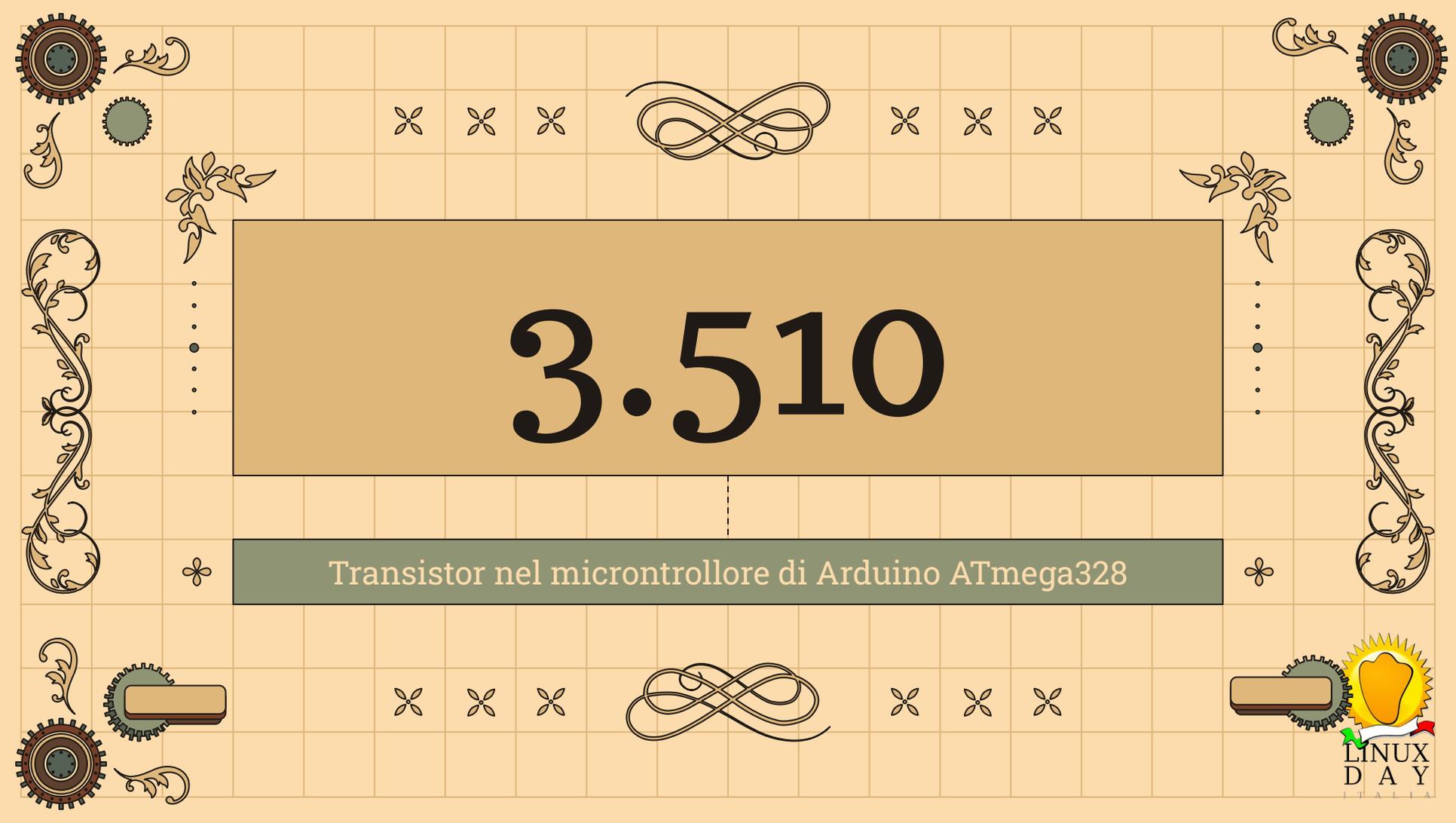


Provate a immaginare adesso la  
complessità di una moderna CPU.



No, non è possibile, la complessità va fuori dalla nostra  
comprensione.





3.510

Transistor nel microcontrollore di Arduino ATmega328

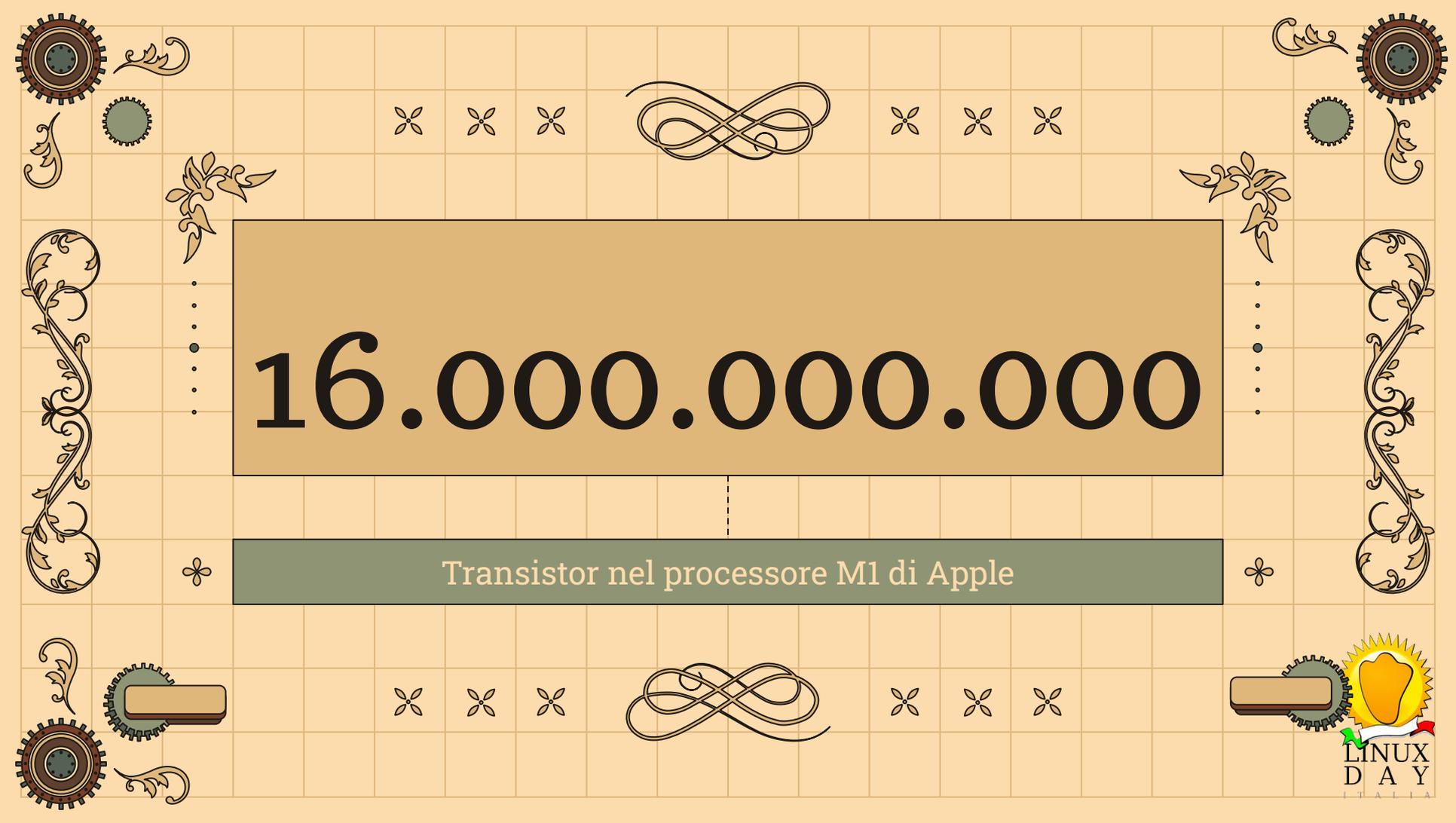




29.000

Transistor nella CPU Intel 8086





16.000.000.000

Transistor nel processore M1 di Apple





# Domande?

(spero di sapere le risposte)

# Grazie!

Francesco Tucci

 @cesco\_78

Blog: [www.iltucci.com](http://www.iltucci.com)

Podcast: <https://pilloledib.it>

CREDITS: This presentation template was created by  
**Slidesgo**, including icons by **Flaticon** and infographics &  
images by **Freepik**

Linux Day 2022

