

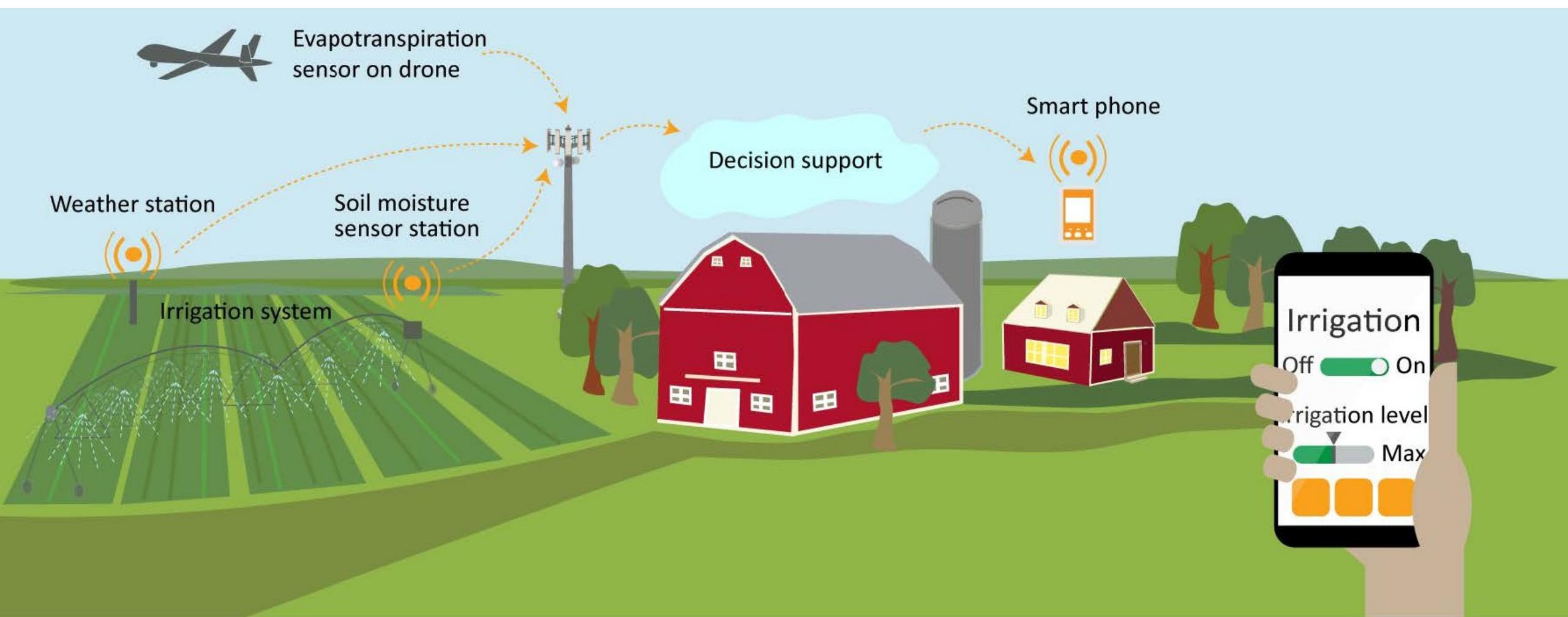
# **COMPUTER SCIENCE: hardware**

Piero Rivoira

Istituto Agrario Penna – Asti  
[piero.rivoira@yahoo.it](mailto:piero.rivoira@yahoo.it)

# Perché studiare computer science in un Istituto Agrario?

- Agricoltura di precisione



# Perché studiare computer science in un Istituto Agrario?

- Robot di mungitura



# Perché studiare computer science in un Istituto Agrario?

- Sistemi automatici di distribuzione degli alimenti



# Per chi vuol saperne di più

Andrew S. Tanenbaum  
Todd Austin

## ARCHITETTURA DEI CALCOLATORI

Un approccio strutturale

Sesta edizione



ALWAYS LEARNING

PEARSON

# Creiamo il nostro profilo su Ubuntu

```
# questo è un commento!
```

```
# lanciamo il terminale
```

```
ctrl+alt+t
```

```
$ sudo adduser nome_battesimo -uid 663
```

```
$ pw di labx
```

```
# per eliminare un profilo utente (in caso di errore)
```

```
$ sudo deluser --remove-home nome_utente
```

```
$ pw di labx
```

# Creiamo il nostro profilo su Ubuntu

```
# acquisiamo i privilegi dell'amministratore di sistema modificando il file di configurazione del Sistema Operativo (SO) /etc/sudoers
```

```
$ sudo visudo
```

```
# questo comando lancia l'editor di testo nano per creare ed aprire il file /etc/sudoers.tmp in modalità scrittura (per poterlo modificare); tale file è una copia di backup di /etc/sudoers
```

GNU nano 7.2 /etc/sudoers.tmp

# Creiamo il nostro profilo su Ubuntu

```
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults        env_reset
Defaults        mail_badpass
Defaults        secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin"

# This fixes CVE-2005-4890 and possibly breaks some versions of kdesu
# (#1011624, https://bugs.kde.org/show_bug.cgi?id=452532)
Defaults        use_pty

# This preserves proxy settings from user environments of root
# equivalent users (group sudo)
#Defaults:%sudo env_keep += "http_proxy https_proxy ftp_proxy all_proxy no_proxy"

# This allows running arbitrary commands, but so does ALL, and it means
# different sudoers have their choice of editor respected.
#Defaults:%sudo env_keep += "EDITOR"

# Completely harmless preservation of a user preference.
#Defaults:%sudo env_keep += "GREP_COLOR"

# While you shouldn't normally run git as root, you need to with etckeeper
#Defaults:%sudo env_keep += "GIT_AUTHOR_* GIT_COMMITTER_"

# Per-user preferences; root won't have sensible values for them.
#Defaults:%sudo env_keep += "EMAIL DEBEMAIL DEBFULLNAME"

# "sudo scp" or "sudo rsync" should be able to use your SSH agent.
#Defaults:%sudo env_keep += "SSH_AGENT_PID SSH_AUTH_SOCK"

# Ditto for GPG agent
#Defaults:%sudo env_keep += "GPG_AGENT_INFO"

# Host alias specification

# User alias specification
```

# Creiamo il nostro profilo su Ubuntu

GNU nano 7.2 /etc/sudoers.tmp

```

# This file is only used by the 'visudo' command to merge multiple
# equivalent users (group sudo)
Defaults:%sudo env_keep += "http_proxy https_proxy ftp_proxy all_proxy no_proxy"

# This allows running arbitrary commands, but so does ALL, and it means
# different sudoers have their choice of editor respected.
Defaults:%sudo env_keep += "EDITOR"

# Completely harmless preservation of a user preference.
Defaults:%sudo env_keep += "GREP_COLOR"

# While you shouldn't normally run git as root, you need to with etckeeper
Defaults:%sudo env_keep += "GIT_AUTHOR_* GIT_COMMITTER_"

# Per-user preferences; root won't have sensible values for them.
Defaults:%sudo env_keep += "EMAIL DEBEMAIL DEBFULLNAME"

# "sudo scp" or "sudo rsync" should be able to use your SSH agent.
Defaults:%sudo env_keep += "SSH_AGENT_PID SSH_AUTH_SOCK"

# Ditto for GPG agent
Defaults:%sudo env_keep += "GPG_AGENT_INFO"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL
piero   ALL=(ALL:ALL) ALL

# Members of the admin group may gain root privileges
%admin  ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "@include" directives:

@include /etc/sudoers.d

```

**# Elenco degli amministratori di sistema**

**# portarsi con il cursore sull'ultima riga della lista**

**alt-6 # copia**

**ctrl-u # incolla l'intera riga**

**# inserire il proprio nome utente**

**ctrl-o # per salvare**

**# cancellare l'estensione .tmp**

**ctrl-x # per chiudere nano**

**alt-u # in caso di errore**  
(per annullare l'ultimo comando inserito)

**Elenco degli amministratori di sistema**

▲G Guida ▲O Salva ▲W Cerca ▲K Taglia ▲T Esegui ▲C Posizione  
▲X Esci ▲R Inserisci ▲\ Sostituisce ▲U Incolla ▲J Giustifica ▲/ Vai a riga ▲- Annulla  
M-U Annulla M-E Ripeti M-A Contrassegna M-] Parentesi  
M-6 Copia M-Q Precedente M-W Successiva M-B Indietro  
M-Q Cerca ind. M-F Avanti

# Diamo un'occhiata all'hardware del nostro pc



Main   Security   Configuration   Boot Options   Exit

System Time	[08:14:47]
System Date	02/02/2024
Product Name	HP 250 15.6 inch G9 Notebook PC
System Family	HP 200
Product Number	6F206EA#ABZ
System Board ID	8A1D
Asset Tag	
Processor Type	12th Gen Intel(R) Core(TM) i3-1215U
Total Memory	8 GB
BIOS Vendor	Insyde
BIOS Revision	F.59
► Device Firmware Revision	
Serial Number	CND3213CW3
UUID	24E1FD44-72FB-ED11-BF94-E073E7F3AB40
System Board CT Number	PQWDFG31VI4D5H
Primary Battery SN	05454 05/20/2023
► System Log	
Build ID	22WW2V6Z601#SABZ#DABZ
Feature Byte	3K47 6b7J 7MaB apaq awbD bhcb fPm9 pnpq pr.a8

Item Specific Help

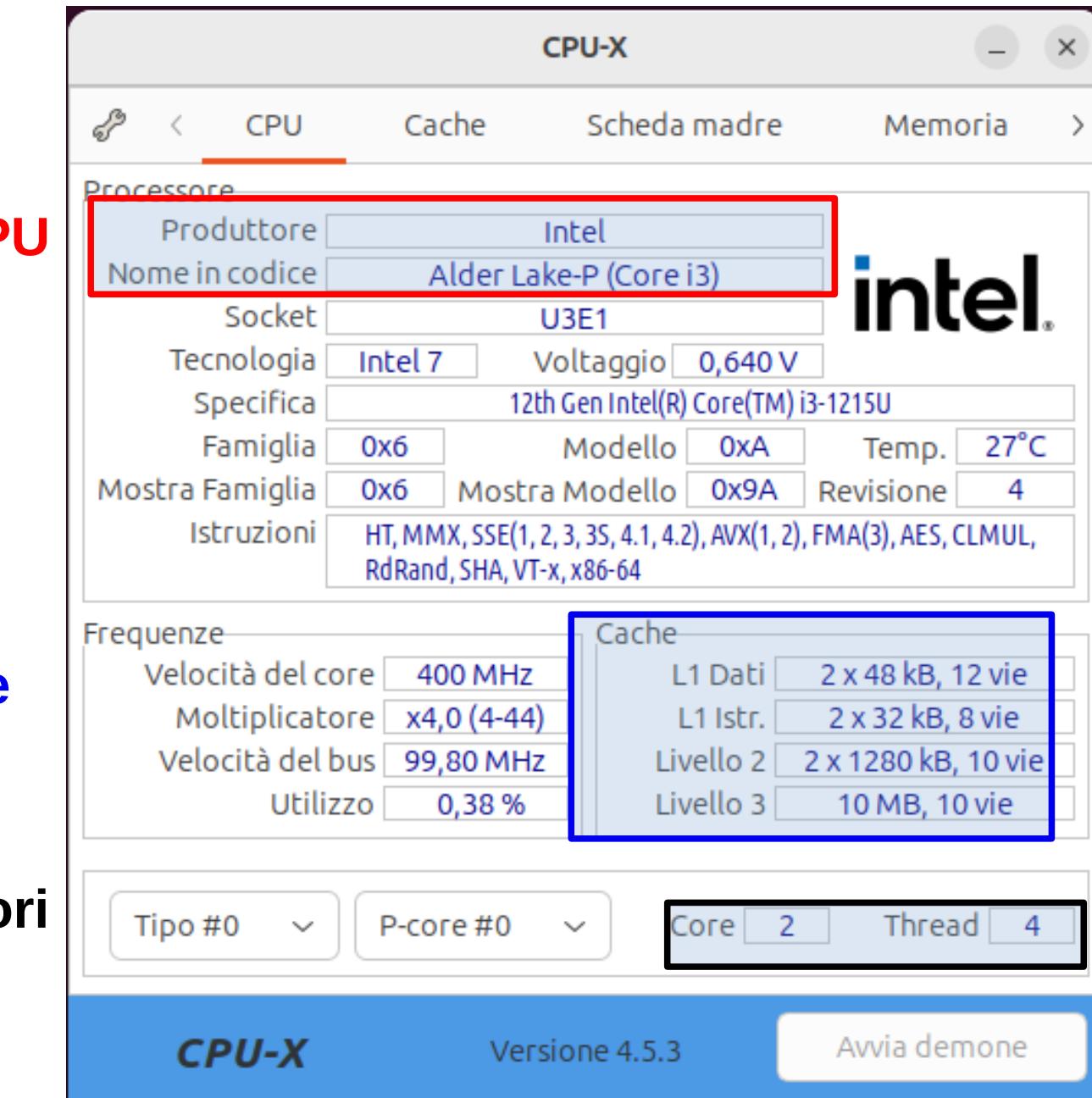
Provides firmware revision information of devices built in the system.

# Diamo un'occhiata all'hardware del nostro pc

ctrl-alt-t

```
$ sudo apt install cpu-x  
$ cpu-x
```

CPU



Cache

N° di processori

# Il processore (Central Processing Unit)

ctrl-alt-t

\$ lscpu

**4.4 GHz**

( $4.4 \times 10^9$  cicli / sec)

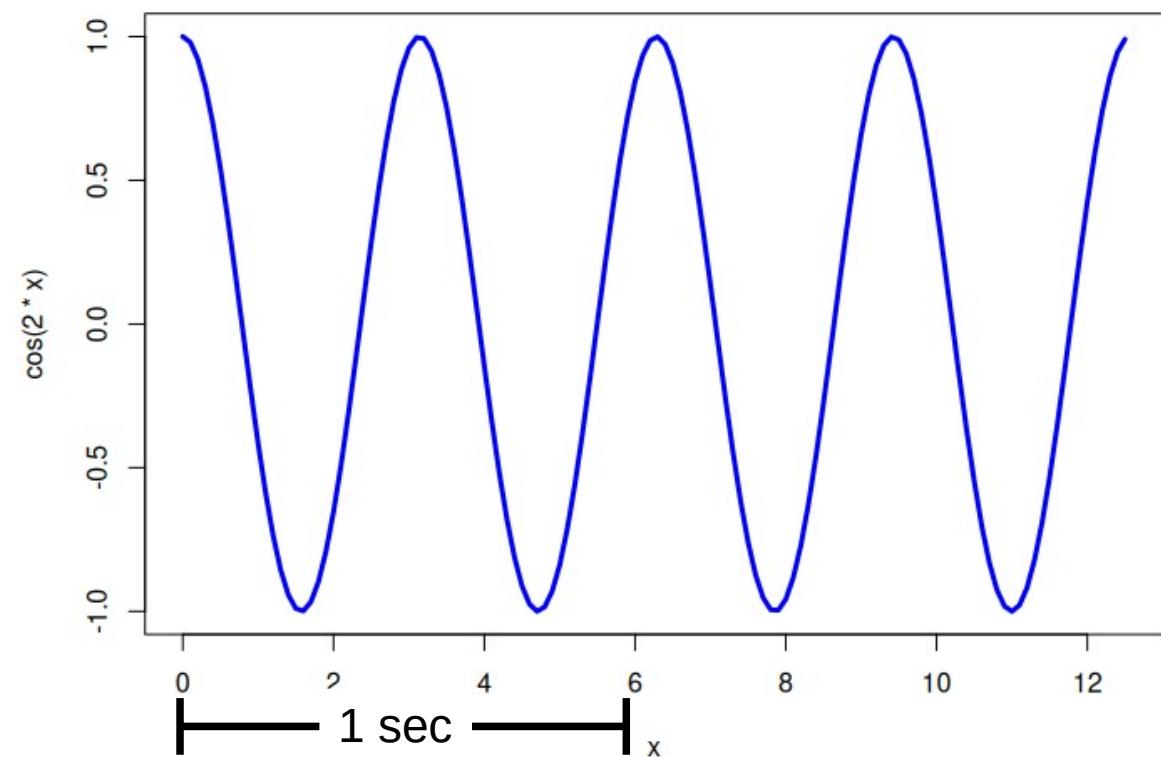
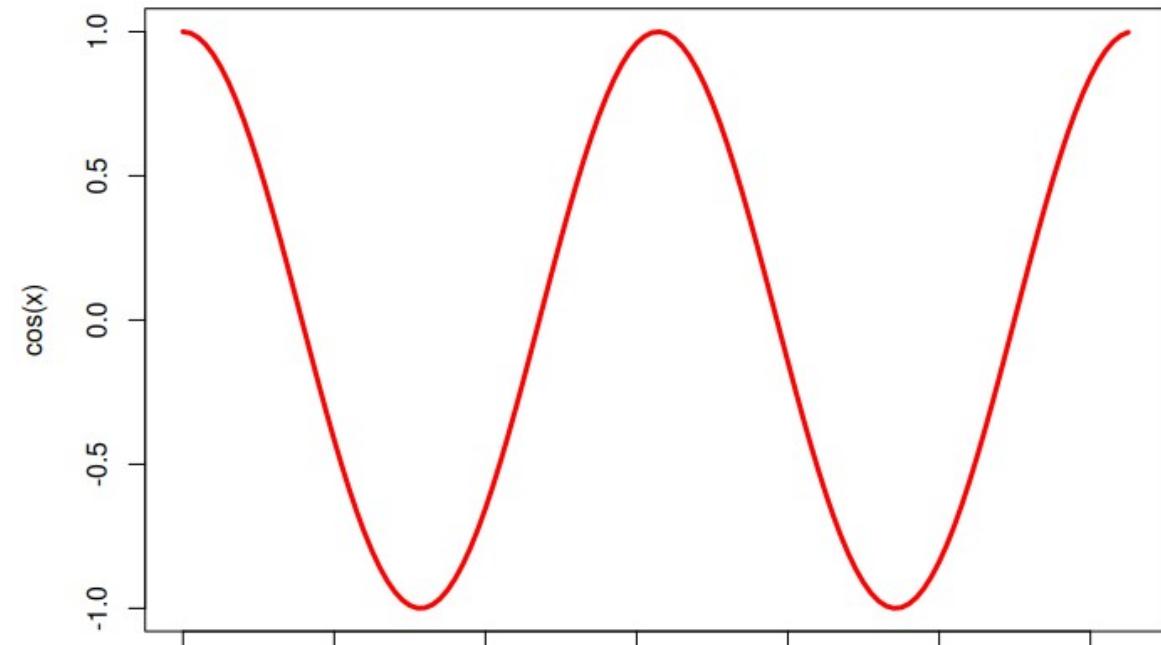
```
lab15@lab15:~$ lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Address sizes:         39 bits physical, 48 bits virtual
Byte Order:            Little Endian
CPU(s):                8
On-line CPU(s) list:  0-7
Vendor ID:             GenuineIntel
Model name:            12th Gen Intel(R) Core(TM) i3-1215U
CPU family:             6
Model:                 154
Thread(s) per core:   2
Core(s) per socket:   6
Socket(s):             1
Stepping:              4
CPU(s) scaling MHz:  18%
CPU max MHz:          4400,0000
CPU min MHz:          400,0000
BogoMIPS:              1992,00
Flags:                 fpu vme de pse tsc msr pae mce cx8 apic
                      tsc art arch_perfmon pebs bts rep_good no
                      ma cx16 xtpr pdcm sse4_1 sse4_2 x2apic m
                      p ibrs_enhanced tpr_shadow flexpriority e
                      saveopt xsavec xgetbv1 xsaves split_lock
                      tpkg gfni vaes vpclmulqdq rdpid movdiri m
Virtualization features:
Virtualization:        VT-x
Caches (sum of all):
L1d:                  224 KiB (6 instances)
L1i:                  320 KiB (6 instances)
L2:                   4,5 MiB (3 instances)
L3:                   10 MiB (1 instance)
```

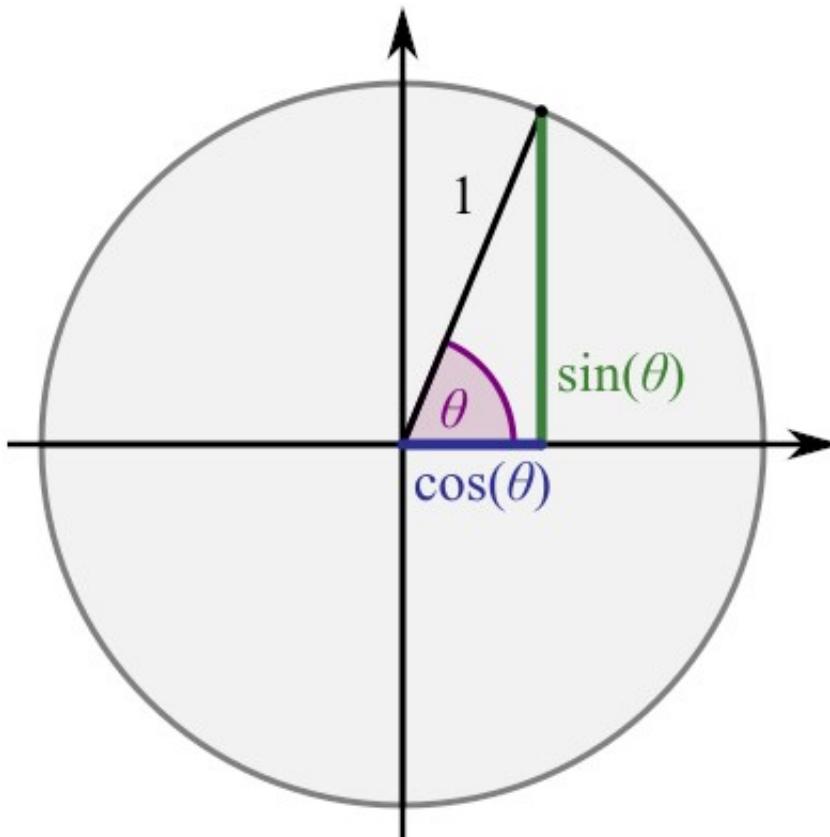
# Cicli di clock

Frequenza della vibrazione di un cristallo  
di quarzo percorso da una corrente  
elettrica

ctrl-alt-t

```
$ R
> x = seq(0, 4*pi, 0.1)
> plot(x, cos(x), type = 'l', col = 'red',
lwd = 3)
> plot(x, cos(2*x), type = 'l', col =
'blue', lwd = 3)
>
```





# Cicli di clock

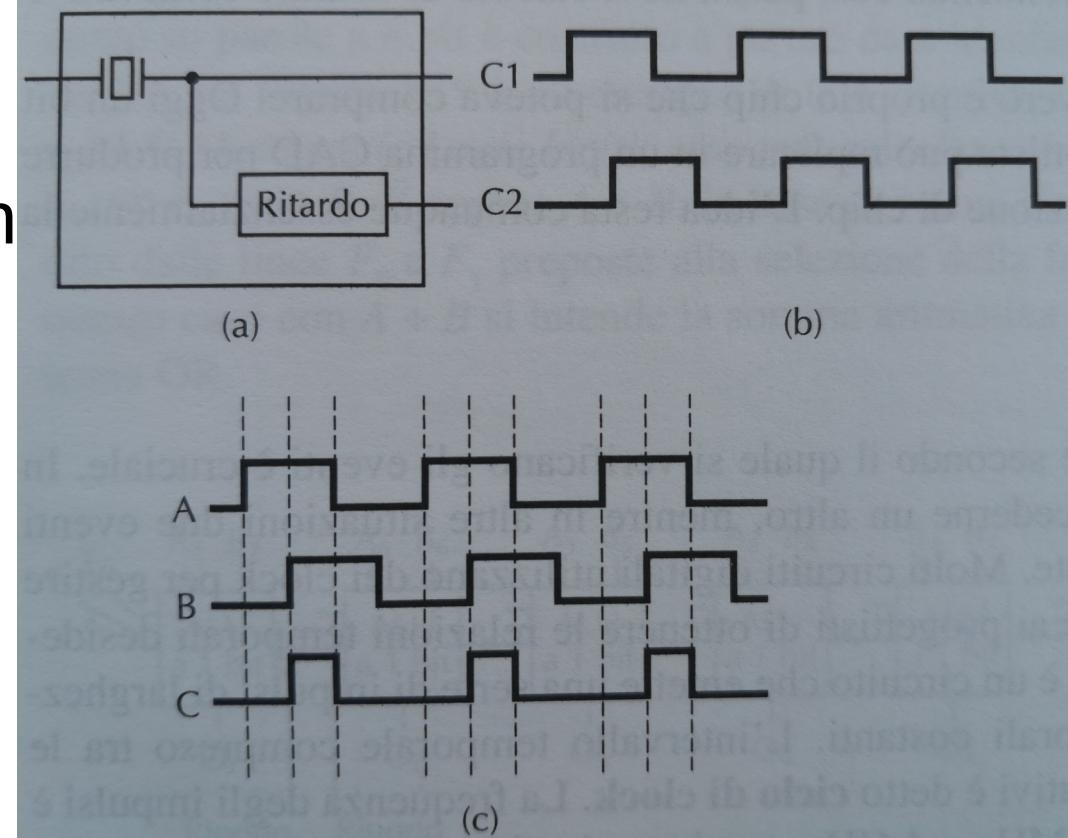
- Nei circuiti digitali è importante l'ordine cronologico con cui si verificano gli eventi:
  - successivi
  - simultanei
- **Ciclo di clock:** intervallo di tempo compreso fra le estremità di due impulsi consecutivi
- Frequenza: 100 MHz ÷ 4 GHz
- Cicli di:  
In R:  
 $> 1/10^8$   
[1]  $1e-08 = 10^{-8} = 10 * 10^{-9} = 10 \text{ ns}$   
 $> 1/4 * 10^{-9}$   
[1]  $2.5e-10 = 250 * 10^{-12} = 250 \text{ ps}$

# Cicli di clock

- Ad ogni ciclo il processore esegue una o più operazioni molto semplici
- Per farlo deve accedere ai dati contenuti in memoria
- Se più eventi si verificano in uno stesso ciclo in un determinato ordine, il ciclo dev'essere suddiviso in sottocicli → necessaria **risoluzione più fine**

# Cicli di clock

- Si usa un segnale (**C2**) caratterizzato da un ritardo noto rispetto a quello del clock principale (**C1**)
- 4 riferimenti temporali utili, ossia fronte di:
  - Salita di C1
  - Salita di C2
  - Discesa di C1
  - Discesa di C2
- Ad ogni fronte può essere associato un diverso evento



# La struttura microscopica di un conduttore in rame

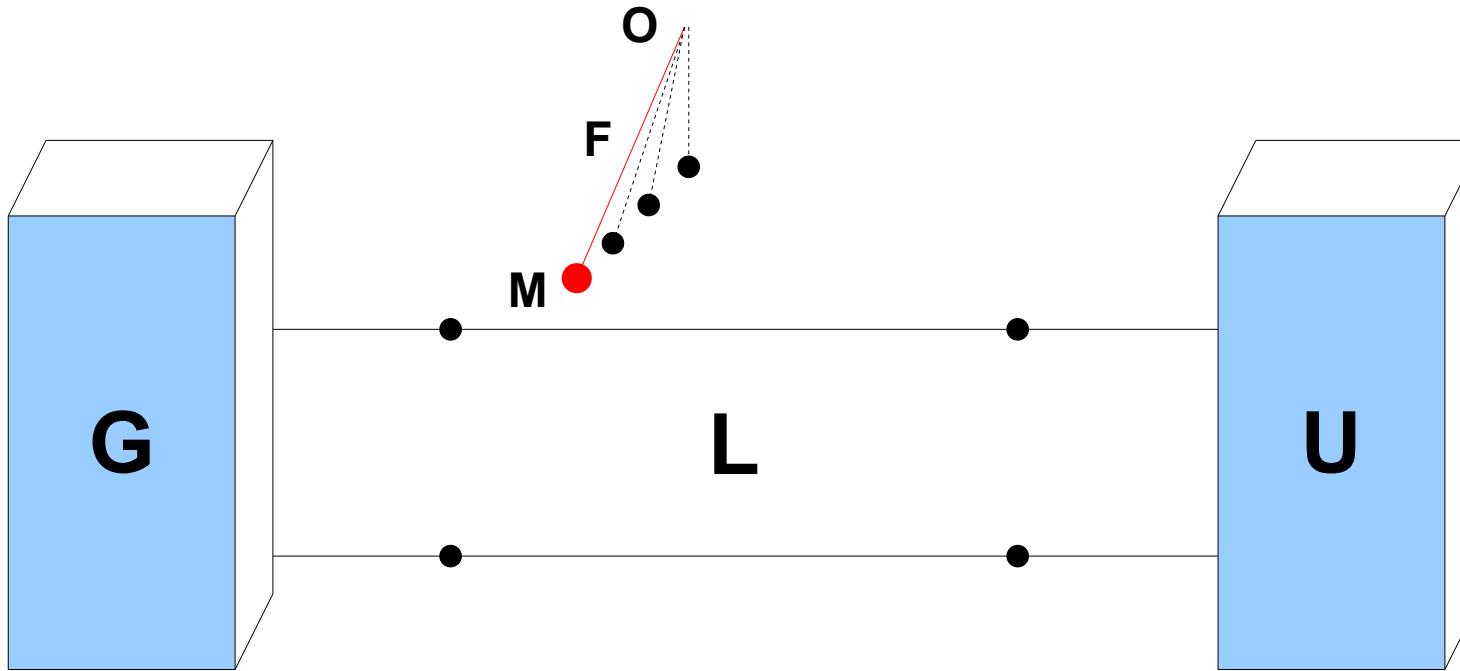
- $8.5 \times 10^{28}$  e<sup>-</sup> liberi / m<sup>3</sup> di Cu
- Si muovono in modo casuale a ≈ 1570 km/s (velocità termica o di Fermi)
- Distanza percorsa prima di collidere con qualcosa =  $3.9 \times 10^{-8}$  m
- Una collisione ogni  $2.4 \times 10^{-14}$  s
- Energia di legame che trattiene gli e<sup>-</sup> sulla superficie del metallo =  
= 4.7 eV (Elettronvolt)  
(1 eV =  $1,60218 \times 10^{-19}$  J; 1 Joule = Newton \* m)
- Secondo la meccanica quantistica gli e<sup>-</sup> sono onde

# Il circuito elettrico elementare

È costituito da:

- un **generatore G**, che converte una potenza di natura qualsiasi in potenza elettrica;
- un **utilizzatore U**, che converte la potenza elettrica in potenza di natura qualsiasi;
- una **linea elettrica L** formata da una coppia di fili metallici – generalmente in rame – le estremità dei quali sono fissate alle terminazioni, anch'esse metalliche, che sporgono dai due apparecchi: trasporta la potenza elettrica dal generatore all'utilizzatore.

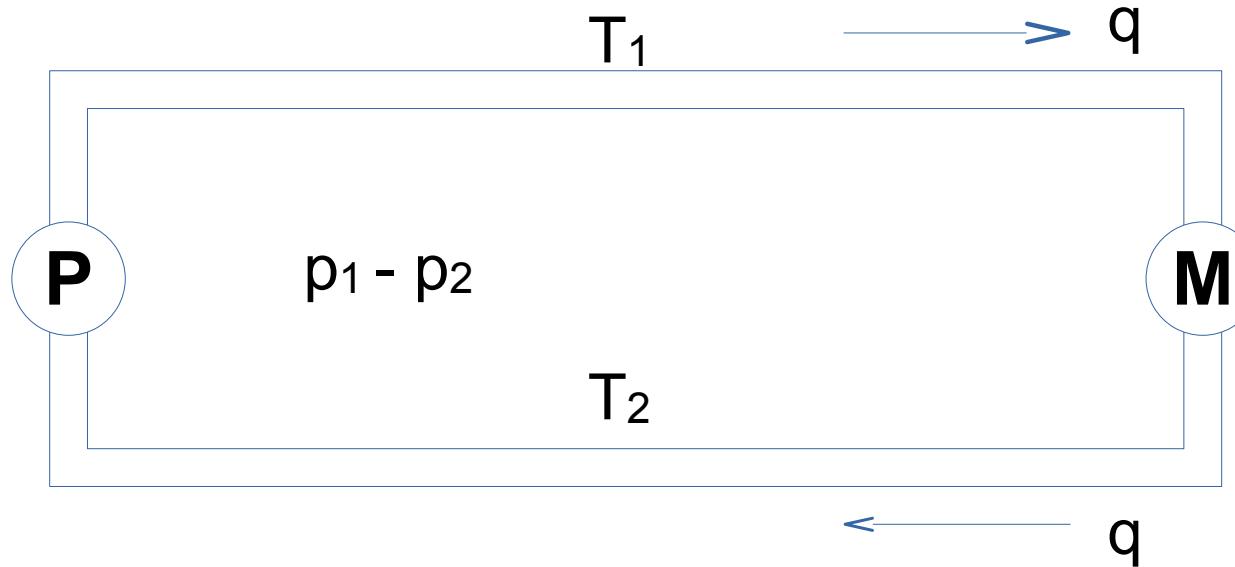
Un pendolino elettrico è formato da una sferetta metallica **M** sospesa mediante un filo **F** isolante ad un punto fisso **O**; si orienta  $\perp$  ai fili della linea



Un ago magnetico posto in prossimità di L si orienta  $\perp$  ad essa (*effetto magnetico*)

# L'analogia con il circuito idraulico

La pompa mantiene fra i due tubi una *differenza di pressione* ( $p_1 - p_2$ ) – corrispondente alla **tensione** – la quale fa circolare nel circuito una *portata volumetrica di fluido*  $q$ , analoga alla **corrente**



Il circuito idraulico è costituito da una pompa **P**, analoga al generatore, da un motore idraulico **M**, analogo all'utilizzatore e da una coppia di tubi **T<sub>1</sub>** e **T<sub>2</sub>** analoghi alla linea

## L'analogia con il circuito idraulico

La potenza idraulica trasferita dalla pompa al motore è data dal prodotto  $(p_1 - p_2) * q$  (tralasciando le differenze di quota e di energia cinetica del fluido nei due tubi); analogamente, la potenza elettrica può esprimersi come prodotto della tensione per la corrente

La pompa mantiene fra i due tubi una *differenza di pressione* ( $p_1 - p_2$ ) – corrispondente alla **tensione** – la quale fa circolare nel circuito una *portata volumetrica di fluido*  $q$ , analoga alla **corrente**.

La potenza idraulica trasferita dalla pompa al motore è data dal prodotto  $(p_1 - p_2) * q$  (tralasciando le differenze di quota e di energia cinetica del fluido nei due tubi); analogamente, la potenza elettrica può esprimersi come prodotto della tensione per la corrente

# Un generatore di tensione: la cella galvanica

- Se si collega un **generatore** ad un filo in rame un voltmetro collegato in serie al conduttore rileverà una tensione elettrica
- Una batteria è una **cella galvanica** che sfrutta una reazione redox per produrre corrente elettrica
- La semireazione di ossidazione e quella di riduzione avvengono in comparti separati e gli  $e^-$  devono essere trasferiti dal riducente all'ossidante attraverso un **conduttore esterno**
- Una cella utilizzata come sorgente di energia elettrica viene chiamata cella, o pila, voltaica o galvanica in onore di Alessandro Volta e Luigi Galvani, gli scienziati che sperimentarono per primi la trasformazione dell'energia chimica in energia elettrica

# Un generatore di tensione: la cella galvanica

- Lo zinco metallico ( $Zn$ ) e lo ione rame ( $Cu^{2+}$ ) in soluzione reagiscono fra loro con trasferimento di  $e^-$ :

possono passare  
solo gli ioni

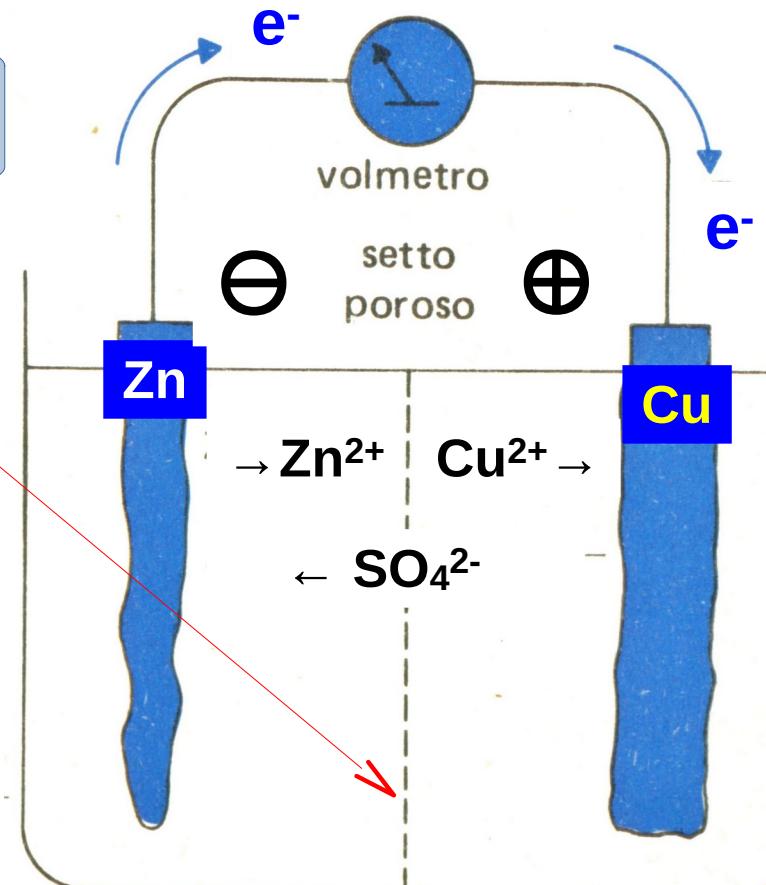
Elettrodo di Zn immerso  
in una soluzione di  $ZnSO_4$

Anodo



Consumo di zinco  
metallico

OSSIDAZIONE



Elettrodo di Cu immerso  
in una soluzione di  $CuSO_4$

Catodo



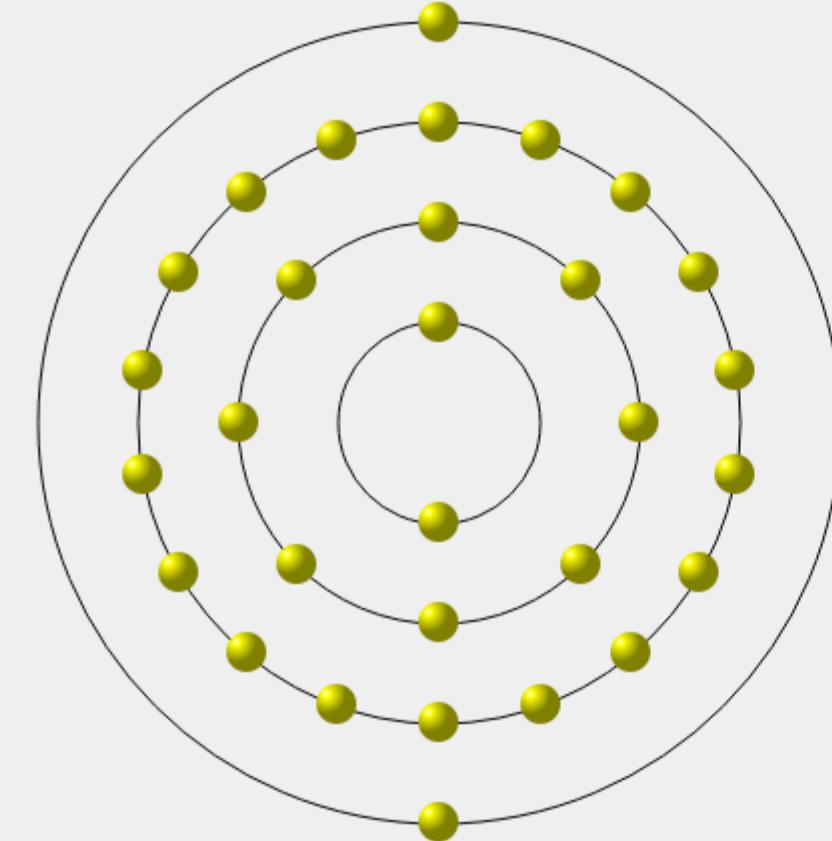
Deposizione di rame  
metallico

RIDUZIONE

## [Zn] Zinco (30 - Blocco d)

- □ ×

Modello atomico

 $[Ar] 3d^{10} 4s^2$ 

Help

&lt; Precedente

&gt; Successivo

Close

Fr

Ra

Ac

Rf

Db

Sg

Bh

Hs

Mt

Ds

Rg

Cn

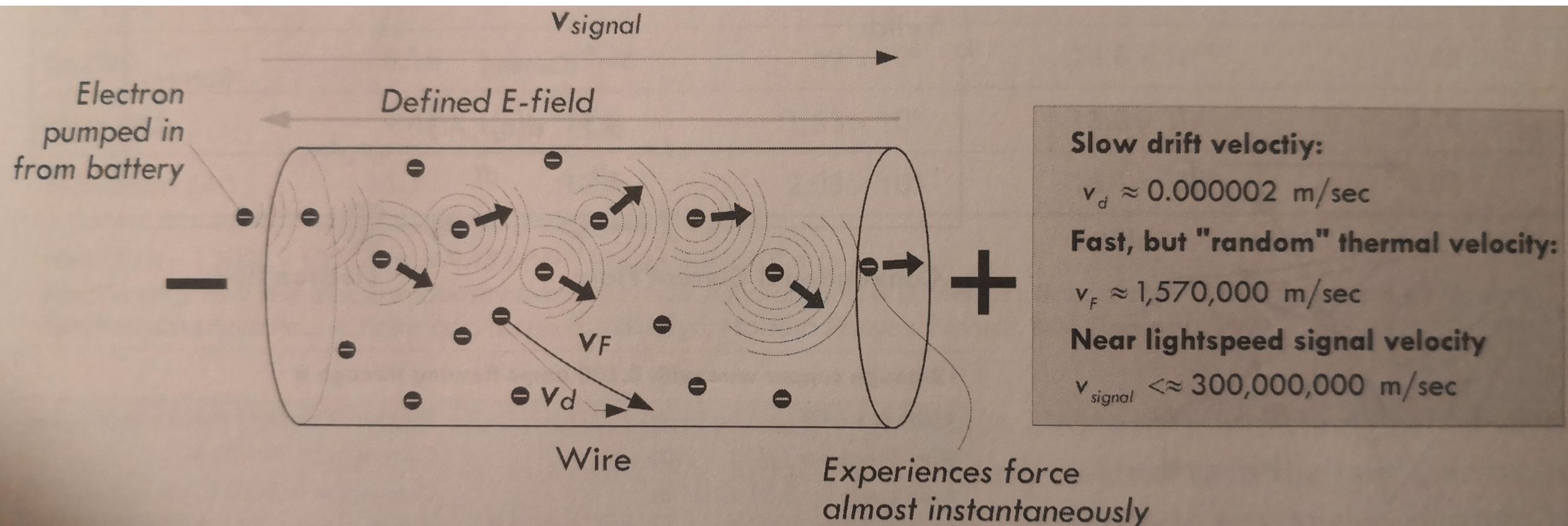
26 Fe	27 Co	28 Ni	29 Cu	30 Zn
44 Ru	45 Rh	46 Pd	47 Ag	48 Cd
76 Os	77 Ir	78 Pt	79 Au	80 Hg
108 Hs	109 Mt	110 Ds	111 Rg	112 Cn

# Cosa succede in un conduttore se vi si applica una tensione elettrica?

- All'Anodo (polo -) lo Zn perde i due e<sup>-</sup> più esterni ( $4s^2$ ) e passa in soluzione come catione Zn<sup>2+</sup>
- Al catodo (polo +) gli e<sup>-</sup> reagiscono con il catione Cu<sup>2+</sup> formando Cu metallico
- La concentrazione di e<sup>-</sup> all'anodo provoca un aumento delle collisioni fra gli stessi senza aumentarne la velocità (la velocità termica degli e<sup>-</sup> è già molto elevata) ma modificandone lievemente la traiettoria
- L'effetto è una velocità di spostamento degli e<sup>-</sup> verso il catodo di alcuni  $\mu\text{m} / \text{s}$  detta *drift velocity*

# Cosa succede in un conduttore se vi si applica una tensione elettrica?

- Il fenomeno responsabile del flusso di corrente elettrica, quindi, non è lo spostamento degli e<sup>-</sup> ma la catena di urti fra gli stessi, una sorta di «onda», che si propaga dall'anodo al catodo a velocità  $\approx c$



## La legge di Ohm

Collisioni degli e<sup>-</sup> contro altri e<sup>-</sup>, ioni metallici e impurità = **resistenza** alla corrente elettrica

1826: Georg Simon Ohm scopre una relazione lineare fra la tensione e la corrente

$$I = V/R$$

Dove:

$I$  = corrente (Ampere)

$V$  = tensione (Volt)

$R$  = resistenza (Ohm,  $\Omega$ ), dipende anche dalle dimensioni del materiale

## Resistività e conduttanza

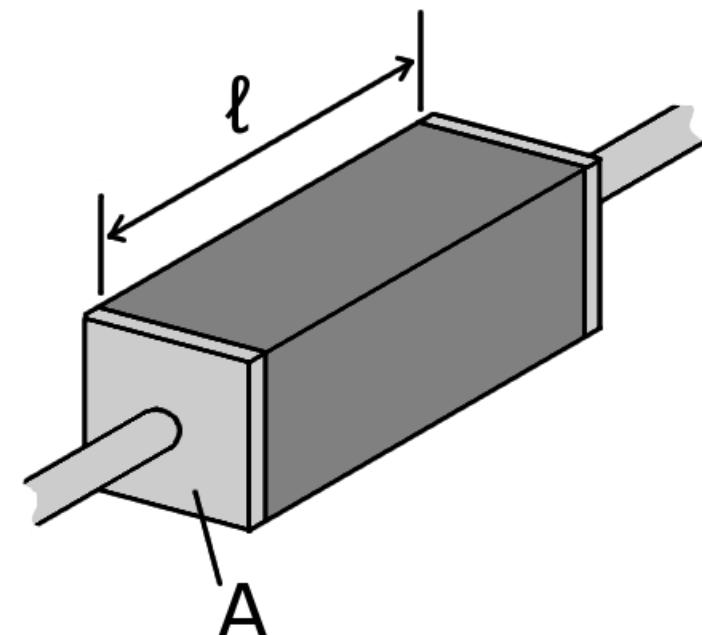
- La resistività (rho) non dipende dalle dimensioni del materiale ma solo dalle sue proprietà chimiche

$$\rho \approx R * A / L = \Omega * \text{m}$$

$A$ : area trasversale del materiale,  $L$ : lunghezza,  $R$ : resistenza

La conduttanza (sigma) è l'inverso della resistività

$$\sigma \approx 1/\rho = 1 / \Omega * \text{m}$$



# Resistività e conduttanza

MATERIAL	RESISTIVITY $\rho$ ( $\Omega\text{m}$ )	CONDUCTIVITY $\sigma$ ( $\Omega\text{m}$ ) $^{-1}$	TEMPERATURE COEFFICIENT $\alpha$ ( $^{\circ}\text{C}^{-1}$ )	THERMAL RESISTIVITY (W/cm $^{\circ}\text{C}$ ) $^{-1}$	THERMAL CONDUCTIVITY $K$ (W/cm $^{\circ}\text{C}$ )
<b>Conductors</b>					
Aluminum	$2.82 \times 10^{-8}$	$3.55 \times 10^7$	0.0039	0.462	2.165
Gold	$2.44 \times 10^{-8}$	$4.10 \times 10^7$		0.343	2.913
Silver	$1.59 \times 10^{-8}$	$6.29 \times 10^7$	0.0038	0.240	4.173
Copper	$1.72 \times 10^{-8}$	$5.81 \times 10^7$	0.0039	0.254	3.937
Iron	$10.0 \times 10^{-8}$	$1.0 \times 10^7$	0.0050	1.495	0.669
Tungsten	$5.6 \times 10^{-8}$	$1.8 \times 10^7$	0.0045	0.508	1.969
Platinum	$10.6 \times 10^{-8}$	$1.0 \times 10^7$	0.003927		
Lead	$0.22 \times 10^{-6}$	$4.54 \times 10^6$		2.915	0.343
Steel (stainless)	$0.72 \times 10^{-6}$	$1.39 \times 10^6$		6.757	0.148 (312)
Nichrome	$100 \times 10^{-8}$	$0.1 \times 10^7$	0.0004		
Manganin	$44 \times 10^{-8}$	$0.23 \times 10^7$	0.00001		
Brass	$7 \times 10^{-8}$	$1.4 \times 10^7$	0.002	0.820	1.22
<b>Semiconductors</b>					
Carbon (Graphite)	$3.5 \times 10^{-5}$	$2.9 \times 10^4$	-0.0005		
Germanium	0.46	2.2	-0.048		
Silicon	640	$3.5 \times 10^{-3}$	-0.075	0.686	1.457 (pure)
Gallium arsenide				1.692	0.591
<b>Insulators</b>					
Glass	$10^{10}-10^{14}$	$10^{-14}-10^{-10}$			
Neoprene rubber	$10^9$	$10^{-9}$			
Quartz (fused)	$75 \times 10^{16}$	$10^{-16}$			
Sulfur	$10^{15}$	$10^{-15}$			
Teflon	$10^{14}$	$10^{-14}$			

# Conduttori, semiconduttori e resistori

Rame (Cu): 29 protoni, 29 e- e 34 neutroni

&Informazion  
Panoramica

## Rame

Panoramica dei dati

Icona	Dati	Valore
	Punto di fusione	1357 K
	Punto di ebollizione	2840 K
	Affinità elettronica	1.236 eV
	Configurazione elettronica	[Ar] 3d <sup>10</sup> 4s <sup>1</sup>
	Raggio covalente	138 pm
	Raggio di van der Waals	200 pm
	Massa atomica	63,546 u
	Energia di prima ionizzazione	7.726 eV
	Elettronegatività	1,9
	Stati di ossidazione	2, 1

Modello atomico

Isotopi

Varie

Spettro

Informazioni aggiuntive

[Cu] Rame (29 - Blocco d)

- □ ×

8 9 10 11

6 Fe	27 Co	28 Ni	29 Cu
44 Ru	45 Rh	46 Pd	47 Ag
6 Os	77 Ir	78 Pt	79 Au

# Conduttori, semiconduttori e resistori

Rame (Cu): 29 protoni, 29 e- e 34 neutroni

&Informazion  
Panoramica

**Rame**

Panoramica dei dati

Modello atomico

[Ar]  $3d^{10} 4s^1$

Isotopi

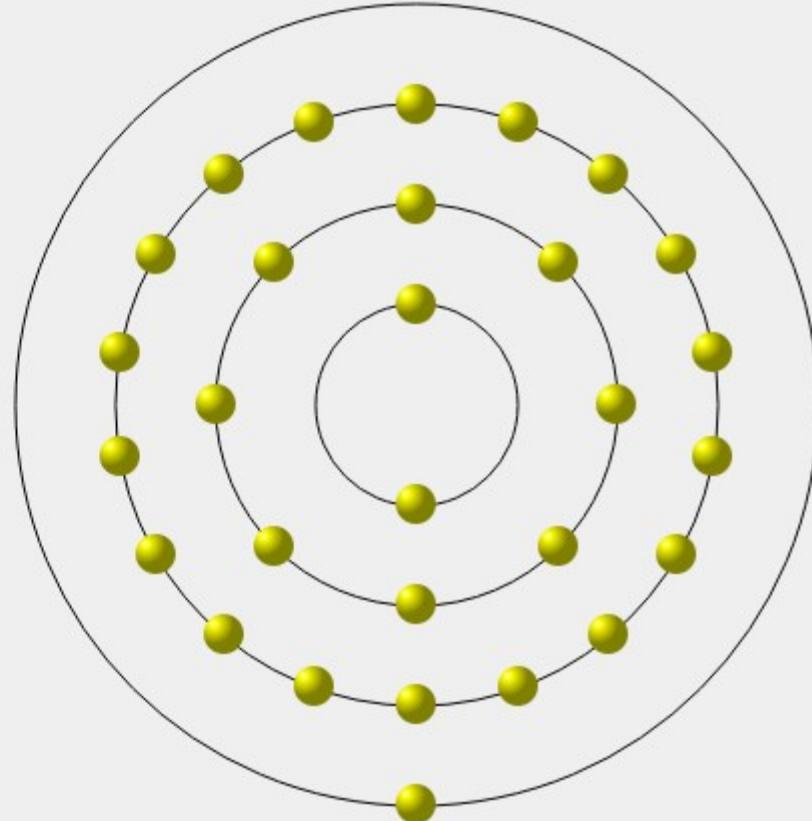
Varie

Spettro

Informazioni aggiuntive

[Cu] Rame (29 - Blocco d)

Modello atomico



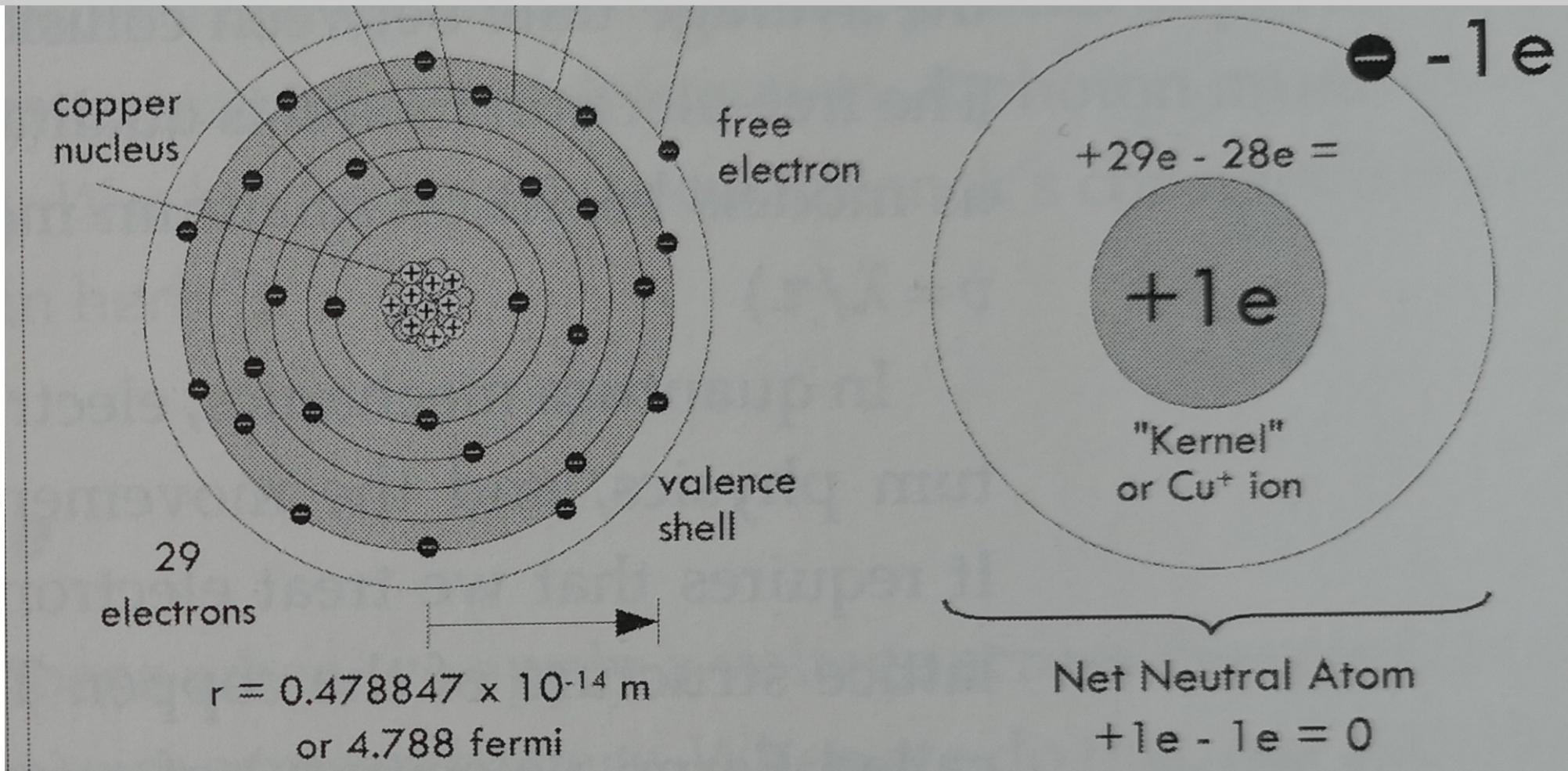
8 9 10 11

6 Fe	27 Co	28 Ni	29 Cu
44 Ru	45 Rh	46 Pd	47 Ag
6 Os	77 Ir	78 Pt	79 Au

# Copper atom Planetary Electron Model

# Ionic Model

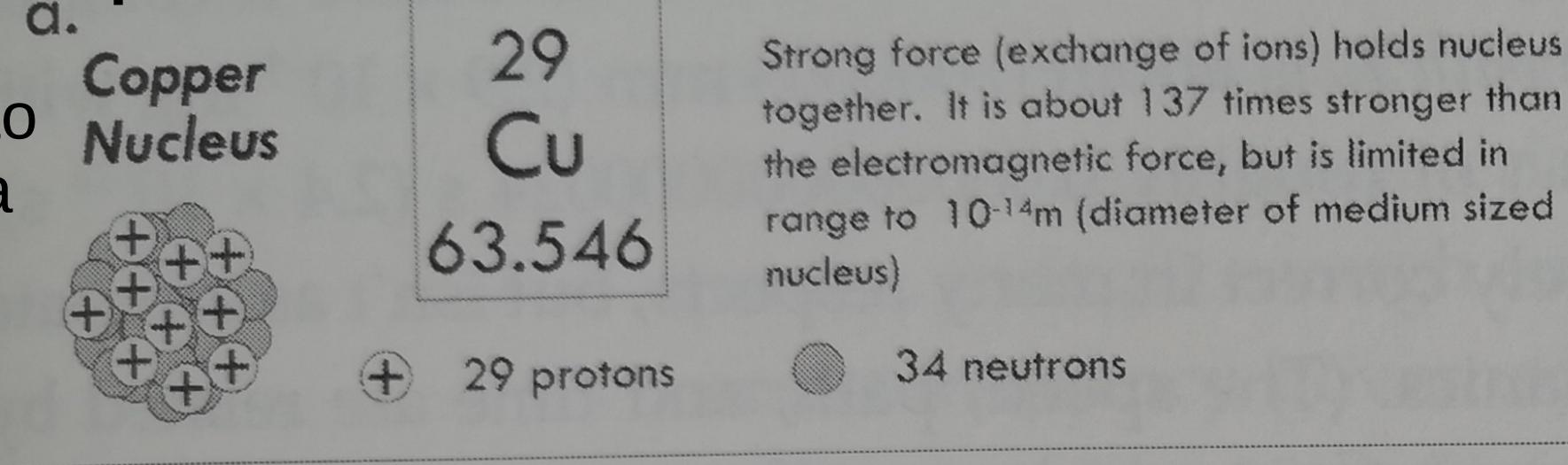
$1s^2\ 2s^2\ 2p^6\ 3s^2\ 3p^63d^{10}\ 4s^1$



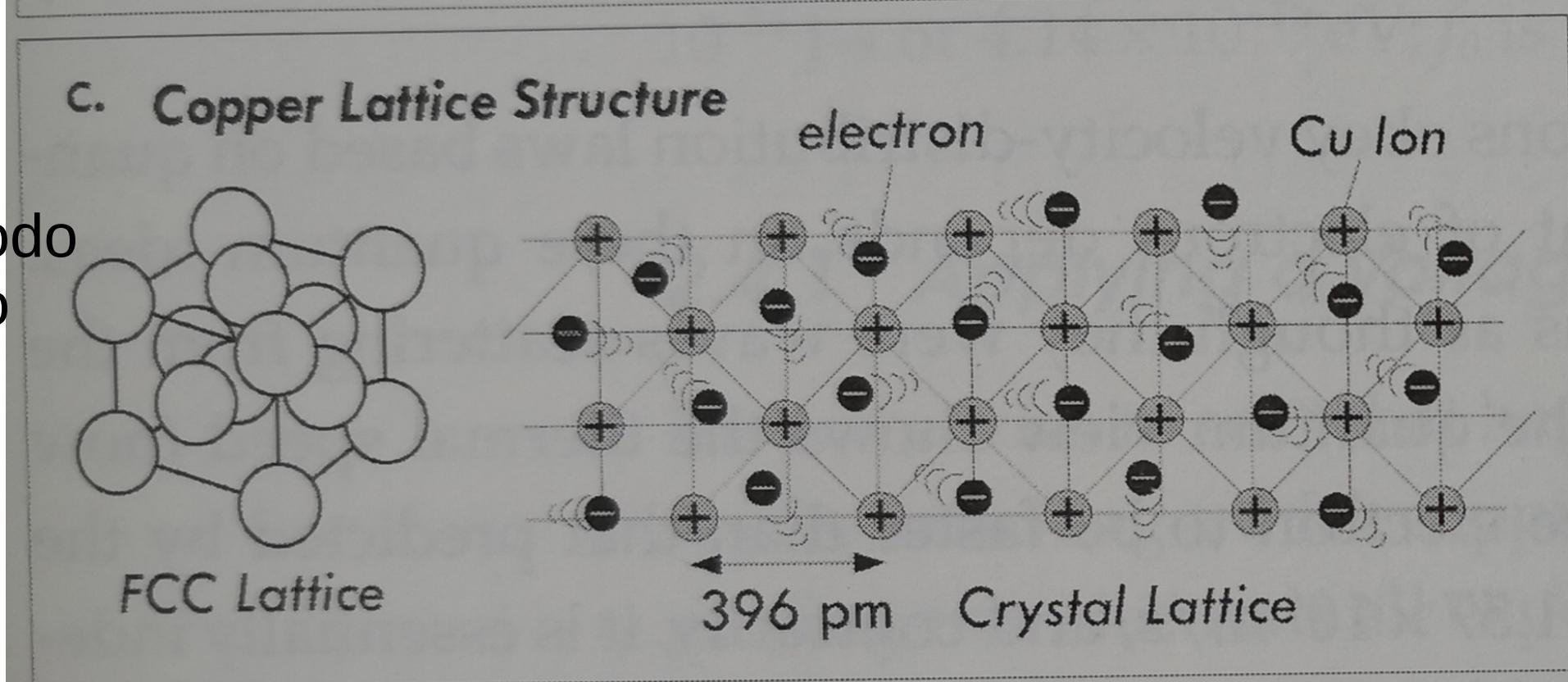
Planetary model depicted here is oversimplistic.  
Quantum mechanics paints a different picture...

# La struttura microscopica di un conduttore in rame

- Reticolo 3D formato da cationi Cu<sup>+</sup> e da una nuvola elettronica

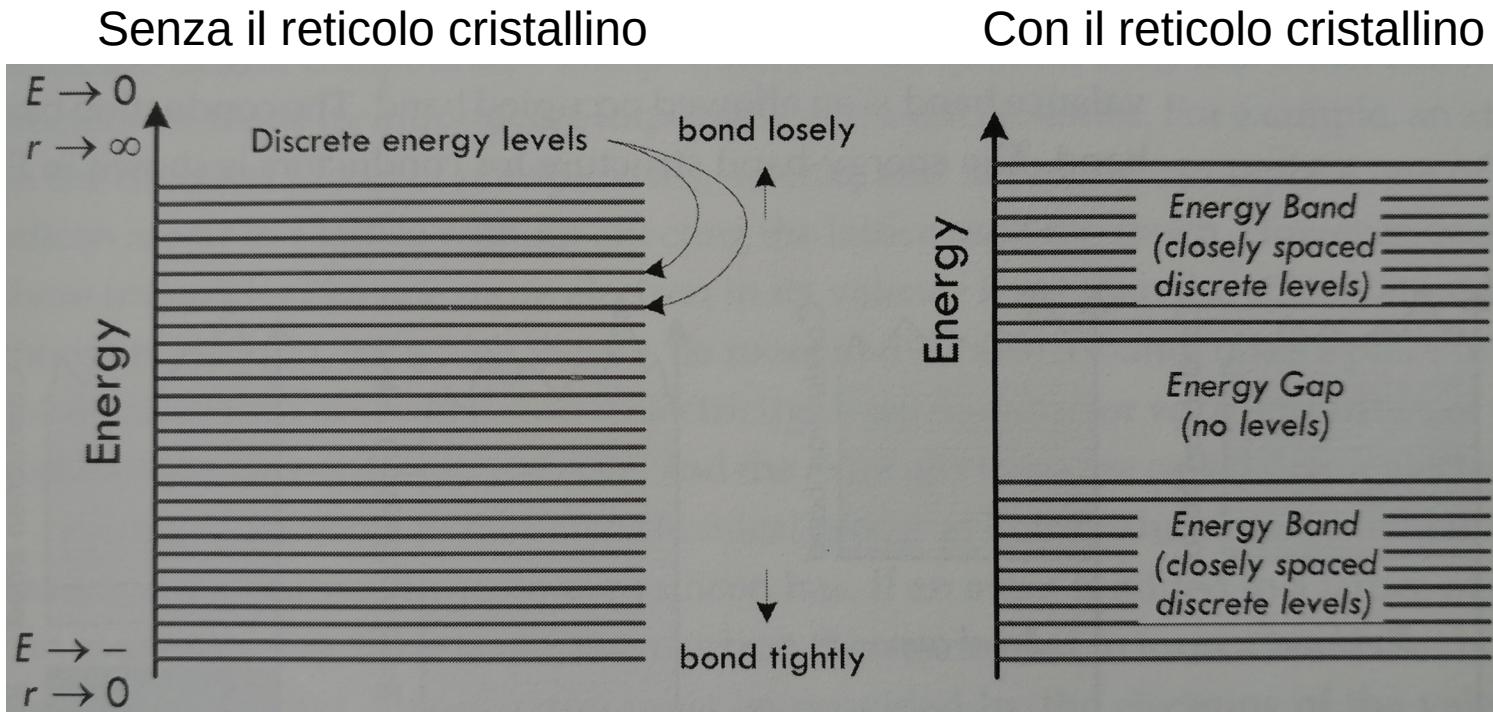


- Senza applicare una tensione elettrica, gli e- si muovono in modo casuale, ma sono attratti dagli ioni Cu<sup>+</sup>



# Resistività e conduttanza

## Diagramma energetico degli elettroni



- **Principio di esclusione di Pauli:**  
2 e<sup>-</sup> non possono avere lo stesso stato quantico  
nello stesso livello energetico non ci possono essere > 2 e<sup>-</sup> con spin opposti

# Lo spin

Il momento angolare tridimensionale di una particella o **spin** è il prodotto del vettore  $r$  di posizione (rispetto ad un punto detto “origine”) e della sua quantità di moto:

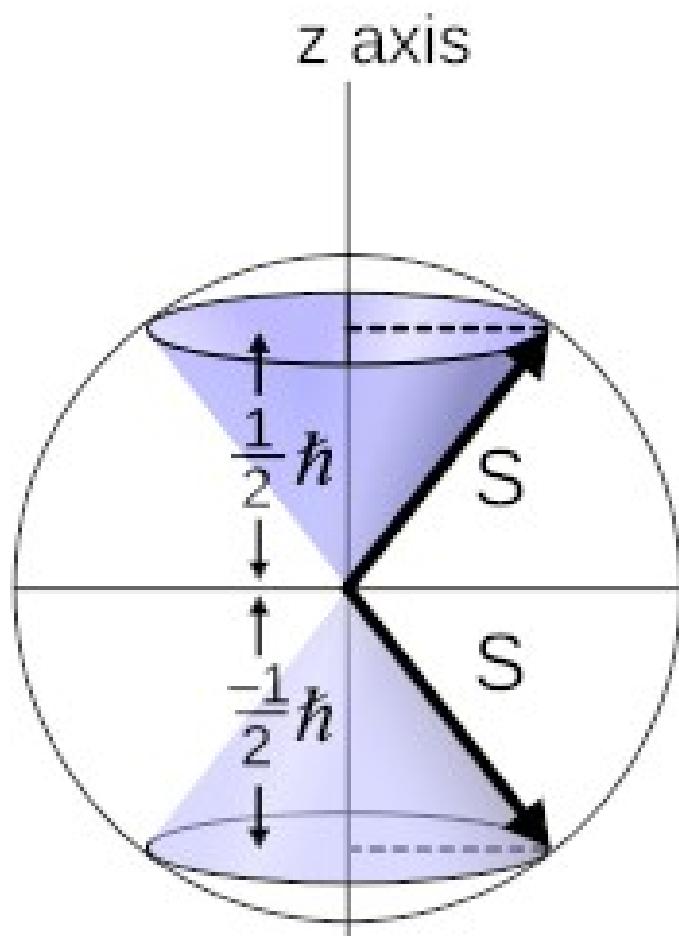
$$\begin{aligned}\text{Memento Angolare} &= \text{vettore di posizione} \times \text{quantità di moto} \\ \text{MA} &= r \times p\end{aligned}$$

$$\text{Quantità di moto } p = \text{massa} \times \text{velocità}$$

# Lo spin

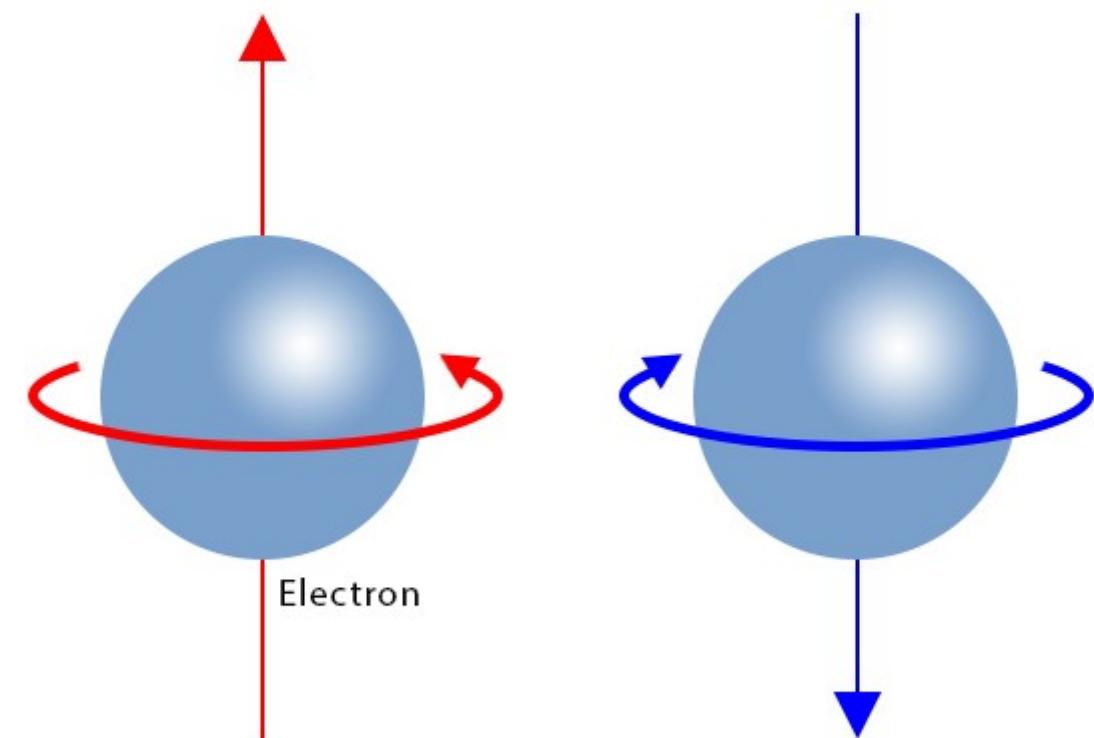
$s$  = numero quantico di spin

$m_s$  = N° quantico magnetico di spin  
(componente dello spin sull'asse z)



## Spin Quantum Number ( $m_s$ )

$m_s$  indicates the orientation of the electron spin

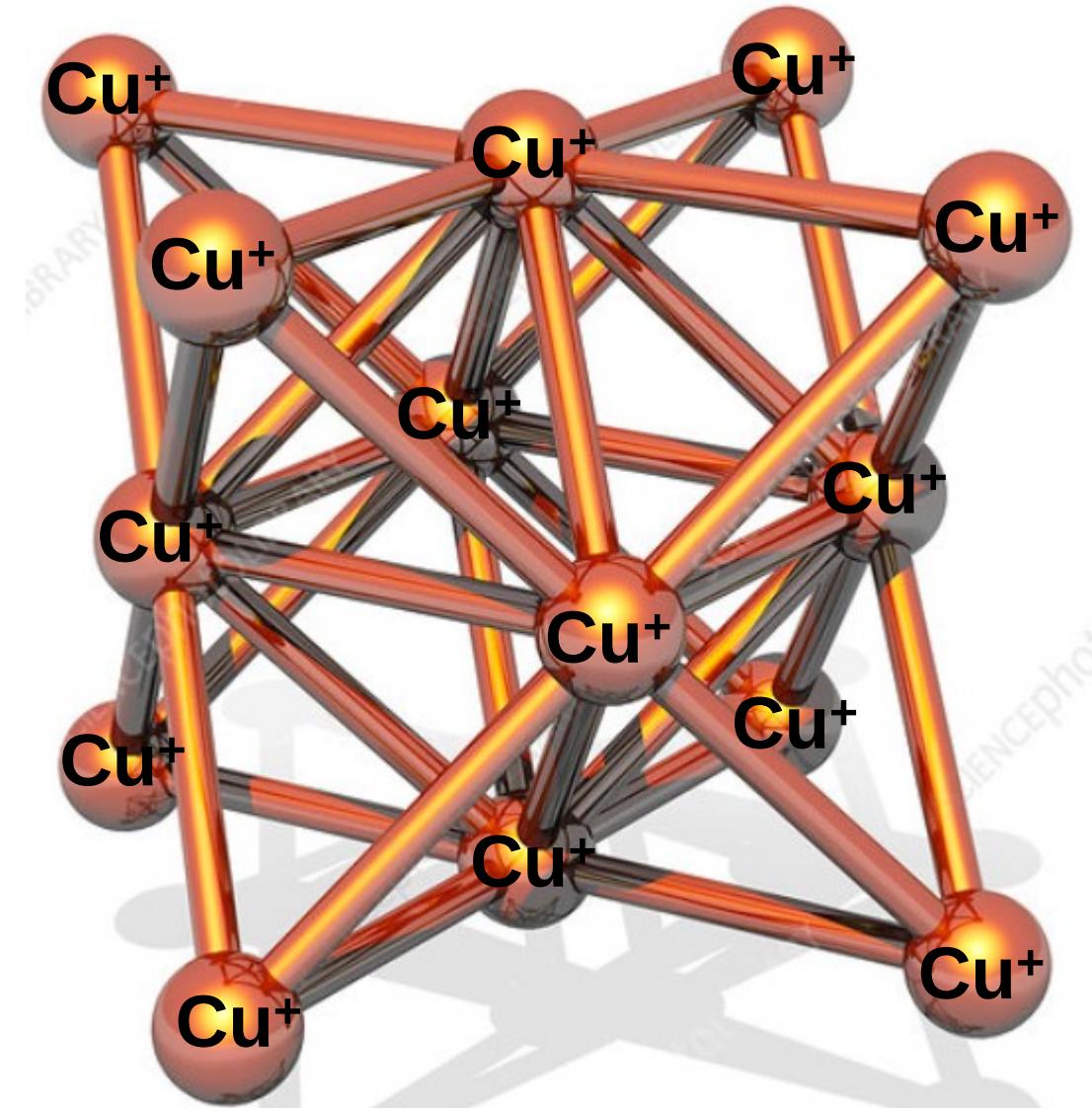


$$m_s = +\frac{1}{2} \Rightarrow \text{"Spin-up"}$$

$$m_s = -\frac{1}{2} \Rightarrow \text{"Spin-down"}$$

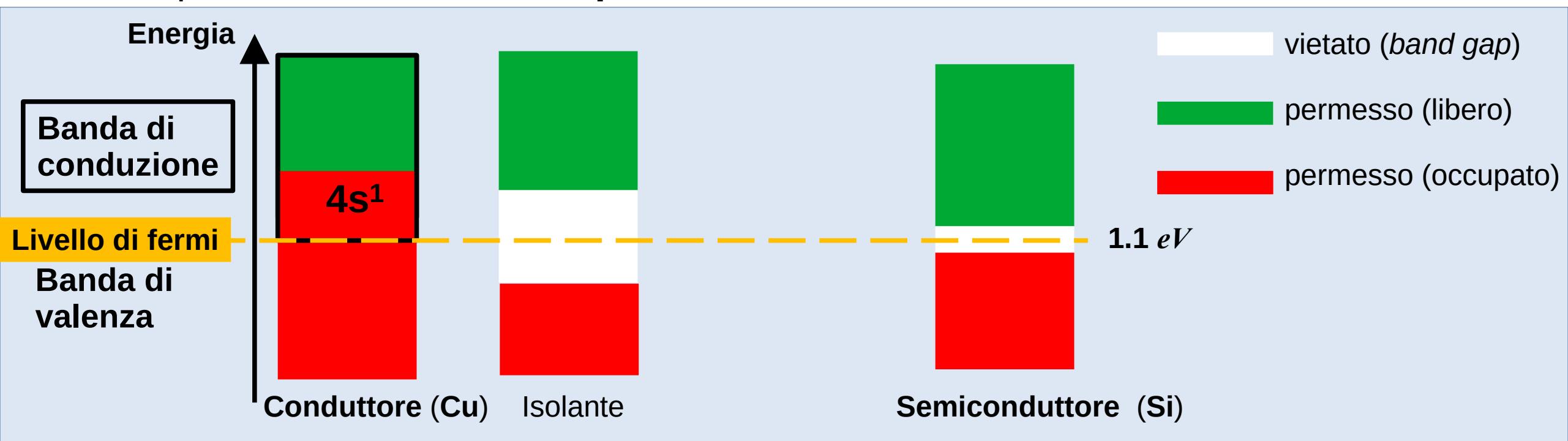
# Resistività e conduttanza

- Il Cu ha molti e<sup>-</sup> liberi ( $4s^1$ )
- Gli e<sup>-</sup> riempiono i livelli energetici liberi più bassi della banda consentita, fino ad un max di 2 per livello
- Gli e<sup>-</sup> dei livelli energetici più bassi sono legati più strettamente al nucleo (*innermost electrons*)



# Resistività e conduttanza

- Se si applica un campo elettrico l'energia (**E**) degli e<sup>-</sup> aumenta ma solo gli e<sup>-</sup> che occupano i livelli energetici più alti possono aumentare la propria **E** passando al livello energetico successivo, libero, della banda di valenza = **conduttori**
- Negli **isolanti** la banda di valenza è completamente occupata e gli e<sup>-</sup> non riescono a saltare il gap
- Nei **semiconduttori** il gap è meno ampio e può essere saltato dagli e<sup>-</sup> se questi ultimi sono esposti ad un debole campo elettrico



# Materiali isolanti, conduttori e semiconduttori

Riempimento degli stati elettronici in differenti tipi di materiali.

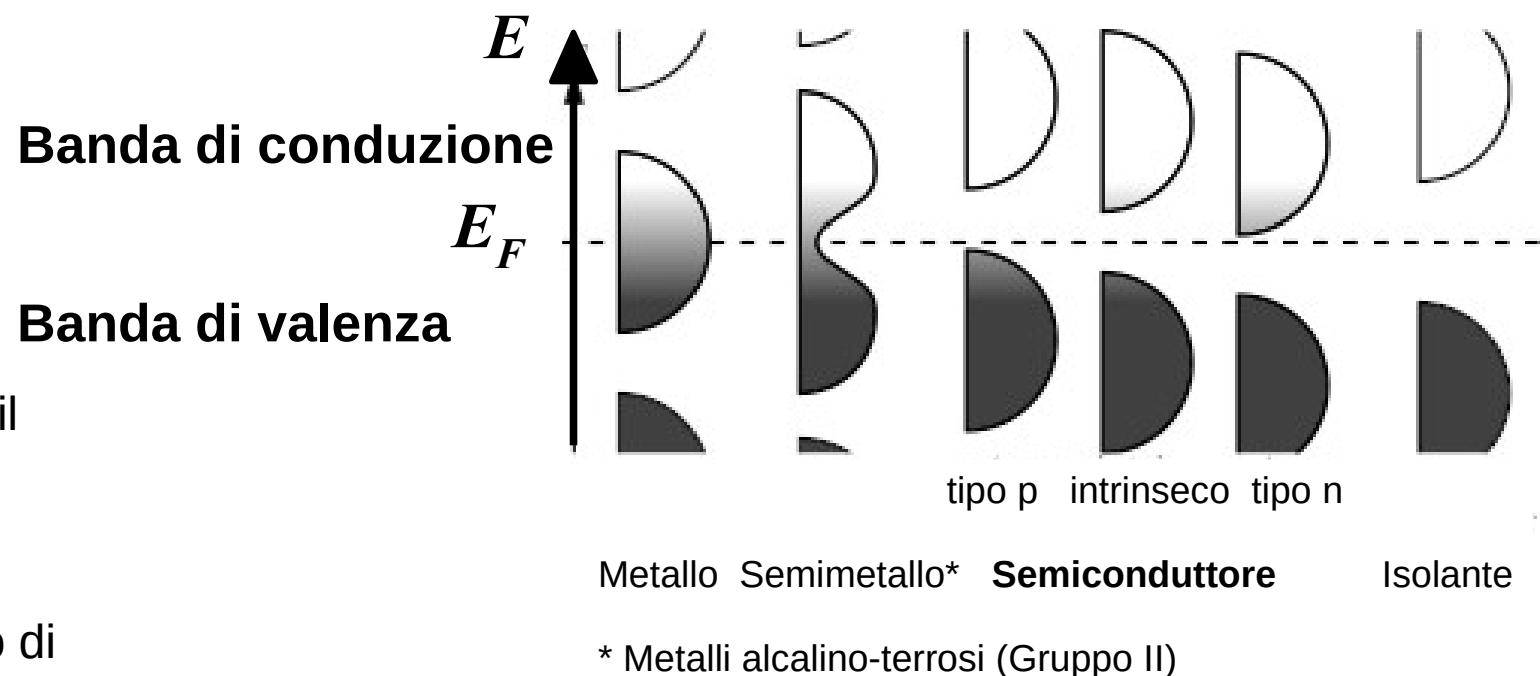
Asse verticale: energia

Asse orizzontale: (densità di) probabilità del livello energetico corrispondente  
(distribuzione di Fermi-Dirac)

- Tutti gli stati sono occupati
- Tutti gli stati sono liberi

$E_F$  : livello di Fermi = livello energetico che ha il 50% di probabilità di essere occupato in un momento qualsiasi

Sia negli isolanti sia nei semiconduttori il livello di Fermi si trova nel gap che separa la banda di valenza da quella di conduzione, ma nei secondi le due bande sono così vicine da permettere agli e<sup>-</sup> più esterni di passare alla banda di conduzione se esposti ad un debole campo elettrico



## Semiconduttori naturali (Si e Ge) e composti ibridi

- Quando un  $e^-$  nella banda di valenza di un semiconduttore attraversa il gap energetico e salta nella banda di conduzione, lascia dietro di sé un «buco», ossia uno stato energetico vuoto. Quest'ultimo può essere occupato da un altro  $e^-$  che si trovava in un livello di energia solo leggermente inferiore, il quale lascerà dietro di sé un buco e così via.
- Il buco si comporta come una carica positiva (carenza elettronica) che contribuisce a condurre elettricità
- Se nel reticolo cristallino del Si si sostituisce un atomo di Si ( $[Ne]3s^23p^2$ ) con un atomo di **P** ( $[Ne]3s^23p^3$ ), arsenico **As** ( $[Ar]3d^{10}4s^24p^3$ ) o antimonio **Sb** ( $[Kr]4d^{10}5s^25p^3$ ), ciascuno dei quali ha un  $e^-$  in più del Si nella banda di valenza, l'assenza, in quest'ultima, di stati energetici liberi costringe l' $e^-$  a saltare nella banda di conduzione (**semiconduttori di tipo n**). Le impurità di questo tipo si chiamano donatori di  $e^-$ .

## Semiconduttori naturali (Si e Ge) e composti ibridi

Se, invece, nel reticolo cristallino del Si si sostituisce un atomo di Si ( $[Ne]3s^23p^2$ ) con un atomo di boro **B** ( $1s^22s^22p^1$ ), **Al** ( $[Ne]3s^23p^1$ ) o **Ga** ( $[Ar]3d^{10}4s^24p^1$ ), ciascuno dei quali ha un  $e^-$  in meno del Si nella banda di valenza, questo  $e^-$  dev'essere fornito dalla banda di valenza del Si, in cui si forma un buco. I buchi si comportano come portatori di cariche positive mentre le impurità di questo tipo sono accettori di  $e^-$  (**semiconduttori di tipo p**).

## La corrente elettrica

- L'unità di misura della corrente nel SI è l'*ampere* (*A*), definito come la quantità di corrente che – circolando in due conduttori indefiniti, rettilinei, paralleli, di sezione circolare trascurabile, posti nel vuoto alla distanza di 1 m – determina una forza mutua fra i conduttori per unità di lunghezza pari a  $2 \cdot 10^{-7}$  N/m
- Ciò è dovuto alla proprietà di una carica elettrica di attrarre o respingere altre cariche

## La carica dell'elettrone

Tutti gli  $e^-$  dell'Universo hanno la stessa carica elettrica (negativa)

$$-e = -1,6022 * 10^{-19} C$$

Quindi, si può calcolare il numero di  $e^-$  che passa attraverso la linea quando la corrente è di 1 *Ampere*:

$$Ampere = Coulomb / s$$

$$\frac{-1 C}{-1,6022 * 10^{-19} C / e^-} \rightarrow e^- = \frac{1}{1,6022} 10^{19} = \mathbf{6,24141805 * 10^{18}}$$

## Bipolo

Apparecchio elettrico costituito da un involucro dal quale sporgono, su di una piastrina o su di un supporto in materiale isolante, due *terminali* o *morsetti* o *poli*; questi ultimi possono presentarsi come viti, bulloni, ghiere, dovendo avere una forma adatta a fissarvi i fili della linea

## Potenza elettrica

Lo stato elettrico ai morsetti di un bipolo è completamente conosciuto quando siano note la tensione  $u$  e la corrente  $i$ . È allora possibile definire la **potenza elettrica  $p$**  scambiata (assorbita o ceduta) ai morsetti di un bipolo:

$$p = u * i$$

L'unità di misura della potenza nel SI è il **watt (W)**:

$$1 \text{ W} = 1 \text{ V} * 1 \text{ A}$$

## Lavoro elettrico (energia)

Il **lavoro elettrico**  $w$  scambiato ai morsetti di un bipolo nell'intervallo di tempo  $(t_0, t)$  è, approssimativamente, definito dal prodotto **della potenza** per il tempo:

$$w = p * t$$

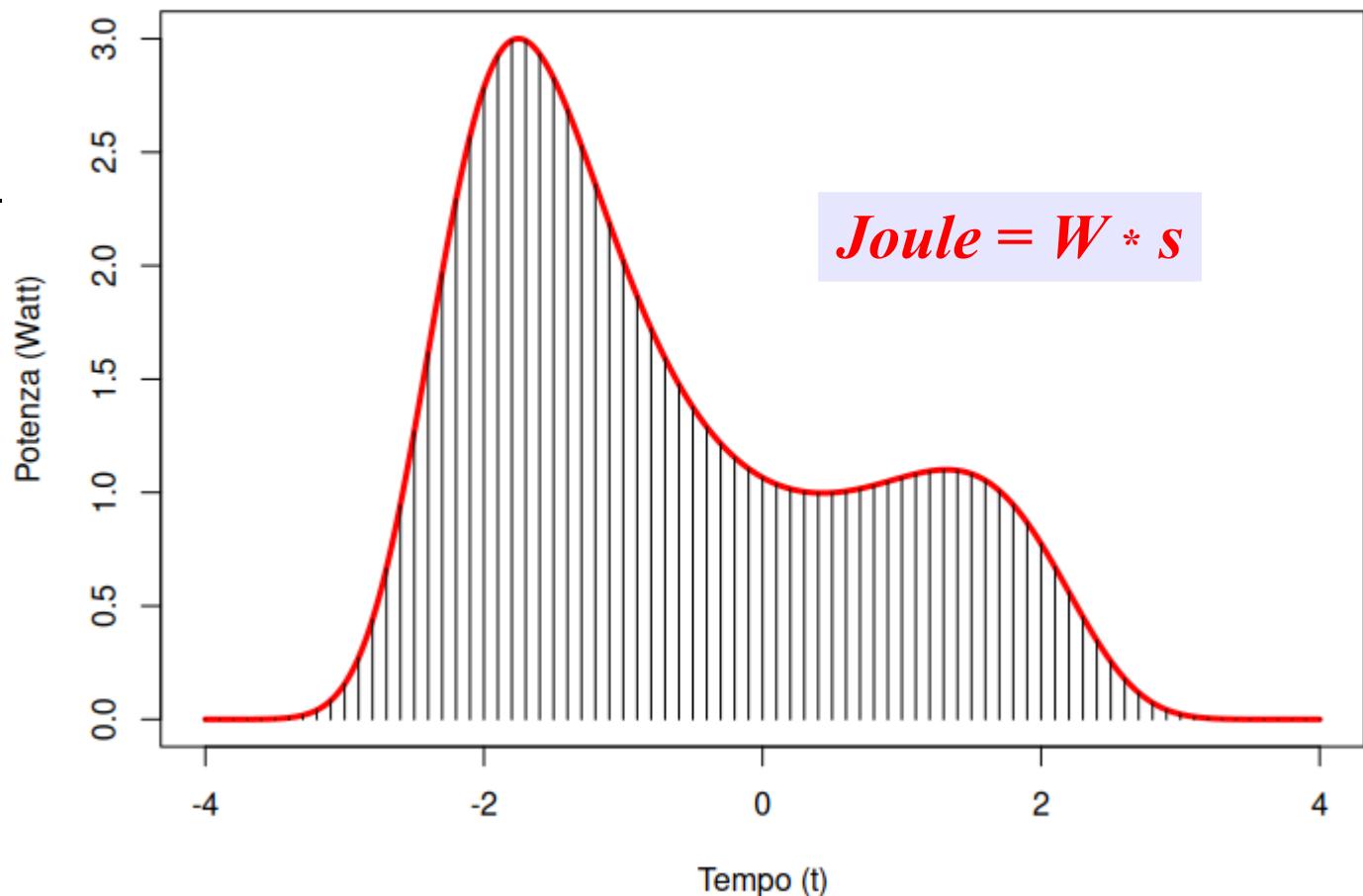
L'unità di misura di  $w$  è il *joule* (J):

$$1 \text{ J} = 1 \text{ W} * 1 \text{ s}$$

# Lavoro elettrico (energia)

```
> x = seq(-4, 4, 0.01)
> y = exp(0.4 * (x-0.4)^2 - 0.08*x^4)
> plot(x, y, type = 'l', col="red",
lwd="3", xlab = 'Tempo (t)', ylab =
'Potenza (Watt)')
> z = seq(-4, 4, 0.1)
> for (i in z)
{
  segments(i, 0, i, (exp(0.4 * (i-0.4)^2 -
0.08*i^4)))
}
```

1 **Joule** = quantità di energia necessaria a fornire una potenza di 1 **Watt** per 1 secondo



## Campo e potenziale elettrico

La regione dello spazio nella quale è definita una determinata grandezza fisica il cui valore in un punto è funzione della **posizione del punto** nello spazio, ossia delle coordinate  $x, y, z$  del punto, è sede di un **campo elettrico**

Il **vettore campo elettrico** è una grandezza che descrive la forza elettrica (forza di attrazione o di repulsione fra due cariche di segno opposto o, rispettivamente, dello stesso segno) in ogni punto dello spazio

$$\vec{E} = \frac{\vec{F}}{q^+}$$

## Campo elettrico $V$

Campo creato da una carica elettrica  $q$  nello spazio circostante; se all'interno del campo di  $q$  si trova un'altra carica  $q'$ ,  $q$  eserciterà una **forza repulsiva o attrattiva  $F$**  su di essa a seconda che  $q$  e  $q'$  abbiano, rispettivamente, lo stesso segno o segno opposto

Il valore di  $V$  è dato dalla **Legge di Coulomb**:

$$V = \frac{1}{4\pi\epsilon} \frac{qq'}{r^2}$$

in cui

$r$  = distanza fra  $q$  e  $q'$

$\epsilon$  = costante dielettrica, legata alle proprietà del mezzo

## Potenziale elettrico $E$

Il potenziale in un punto del campo elettrico è l'**energia potenziale posseduta dalla carica unitaria** posta in quel punto;  
la sua unità di misura è il volt:

$$E = \frac{U}{q} = \frac{\text{joule}}{\text{coulomb}} = \frac{\text{watt} * \text{s}}{\text{ampere} * \text{s}} = \frac{\text{volt} * \text{ampere}}{\text{ampere}} = \text{volt}$$

$E$  = potenziale elettrico del punto considerato;

$U$  = energia potenziale posseduta dalla carica  $q$

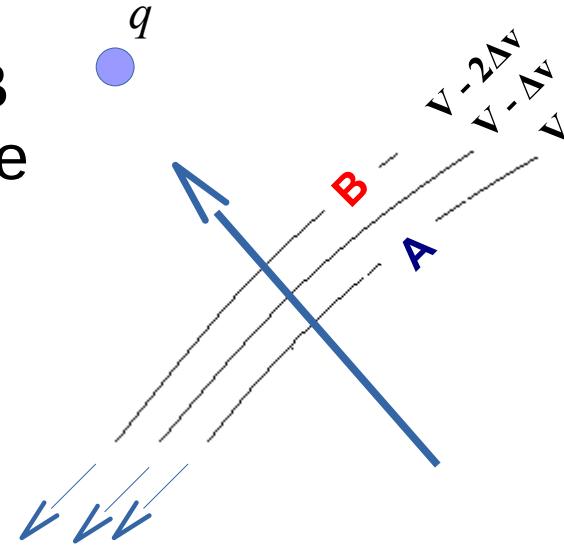
In fisica, l'energia potenziale di un oggetto è l'energia che esso possiede a causa della sua posizione o del suo orientamento rispetto ad un campo di forze

# Campo e Potenziale elettrico

Se due punti **A** e **B** di un campo elettrico  $\notin$  ad una superficie equipotenziale, fra di essi sussiste una **differenza di potenziale  $\Delta E$**

L'intensità del campo elettrico  $V$  è un vettore di componenti  $x, y$ , ortogonale alle superfici  $E = \text{cost.}$  (superficie EQUIPOTENZIALI) e diretto nel verso in cui il potenziale decresce

Se il sistema passa dal punto **A** al punto **B** il potenziale diminuisce di  $\Delta E = E_A - E_B$  compiendo un lavoro  $w \propto \Delta E$ .

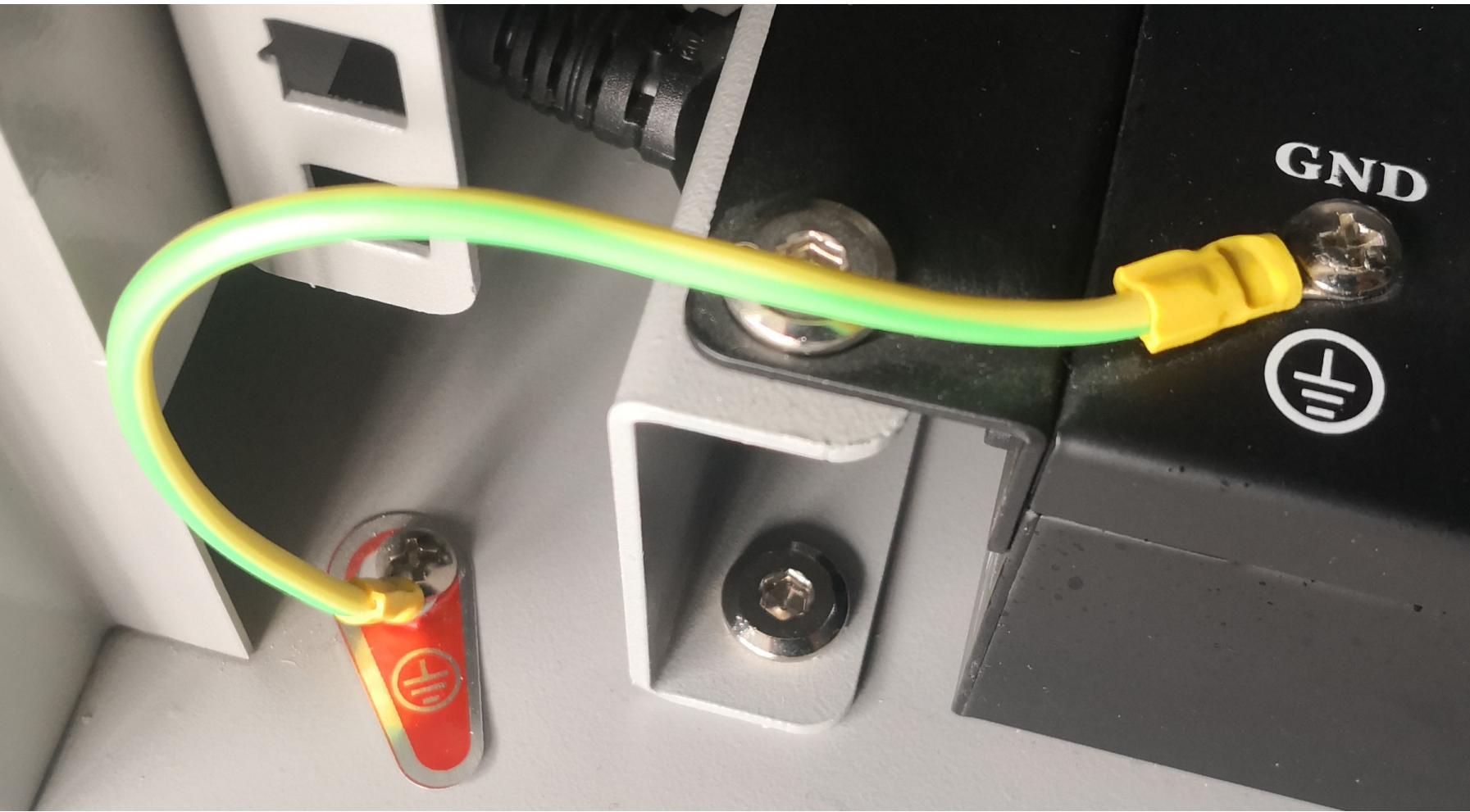


## La messa a terra (*Earth Ground*)

- Connessione che termina con una barra metallica affondata nel suolo per un paio di metri
- Cavo in rame nudo o rivestito da una guaina verde
- La Terra rappresenta una enorme massa elettricamente neutra con  $\approx N^\circ$  di cariche  $\oplus$  e  $\ominus$   
→ potenziale 0 (zero), che non può essere alterato da generatori, batterie e fenomeni elettrostatici (fulmini)
- Potenziale di riferimento per tutti i segnali ed i dispositivi elettronici

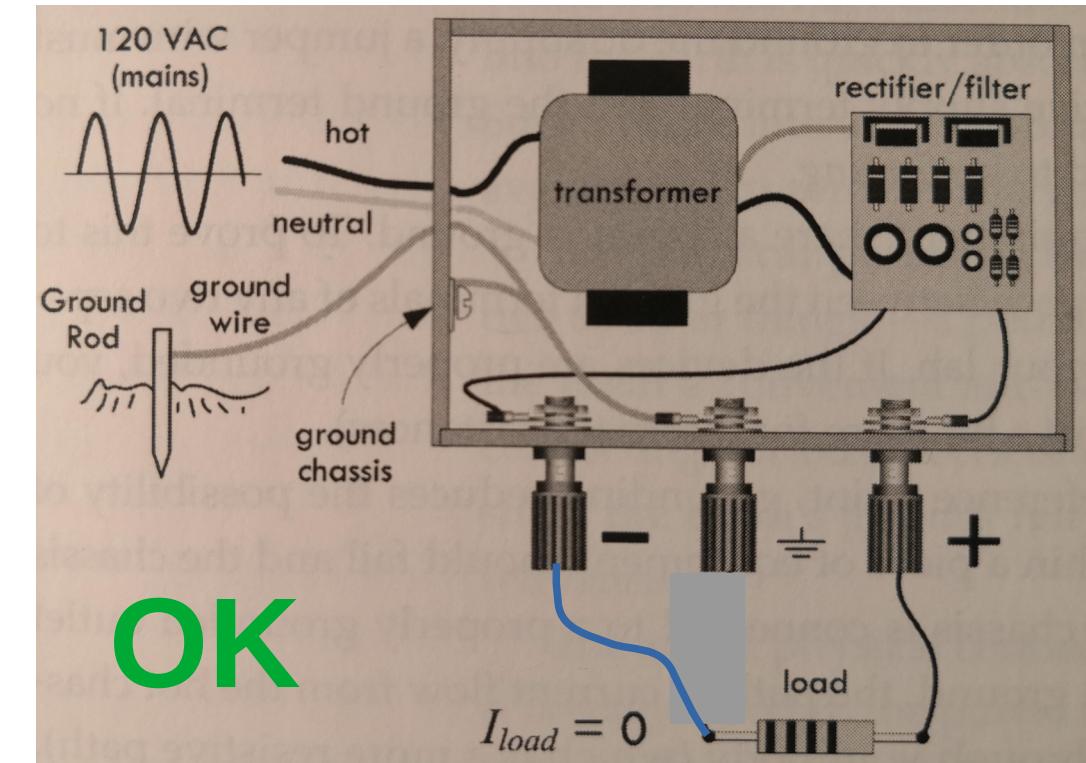
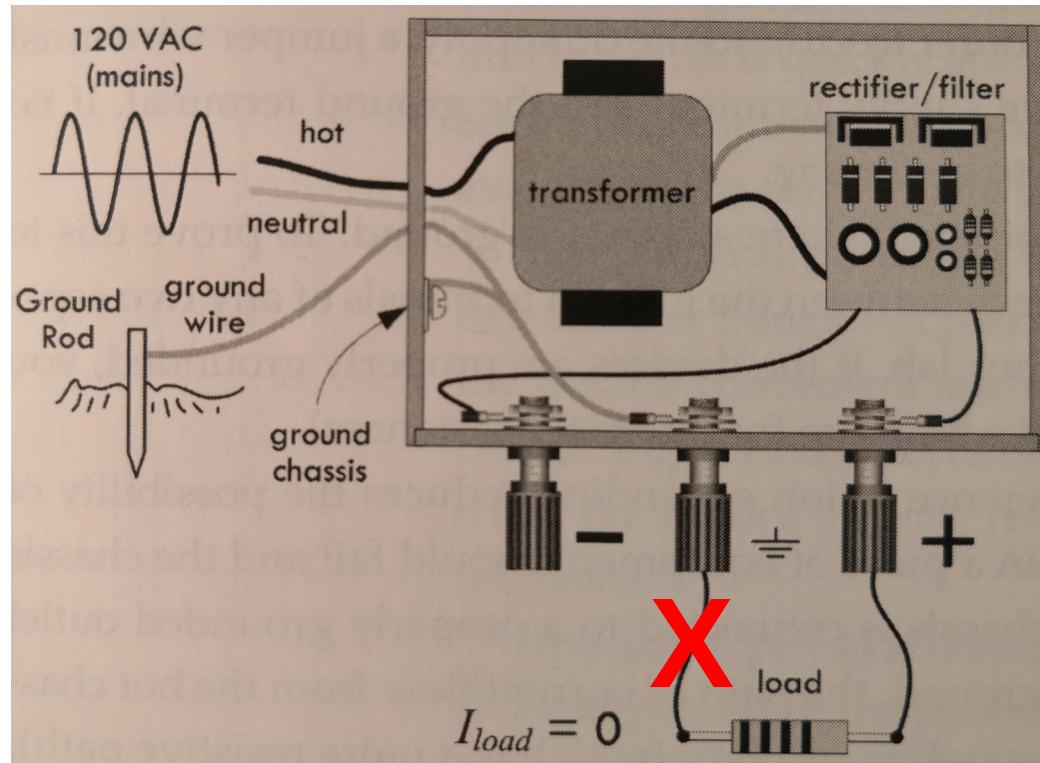


# La messa a terra (*Earth Ground*)



# La messa a terra (Earth Ground)

- Il cavo della messa a terra dev'essere collegato al contenitore esterno dell'apparecchio



## Le porte logiche (*logic gates*)

Circuiti dotati di uno o più ingressi e di un'uscita

Il segnale può essere:

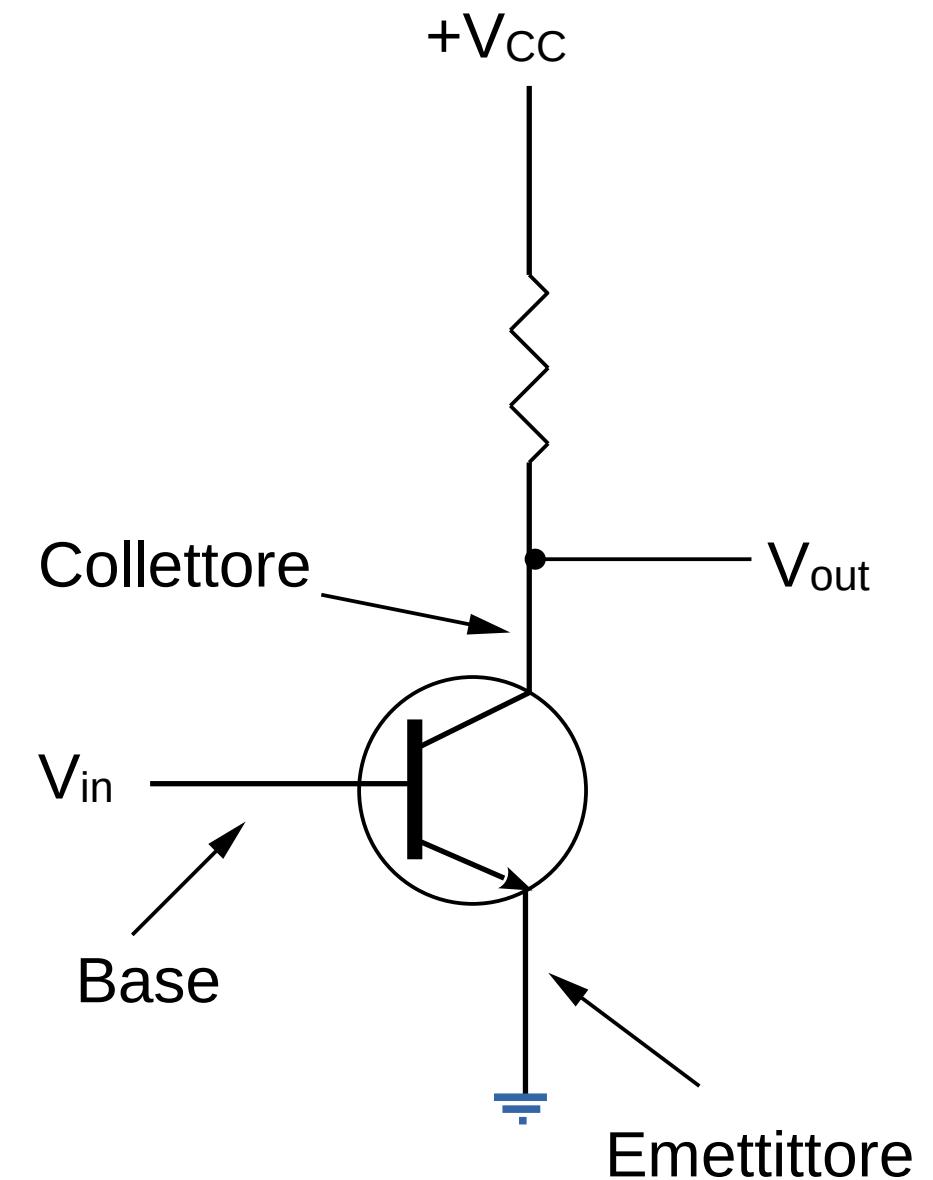
- tensione elettrica  $0 \leq V \leq 0.5 \rightarrow$  numero binario 0 (zero)
- tensione elettrica  $1 \leq V \leq 1.5 \rightarrow$  numero binario 1 (uno)

Il segnale in uscita dipende dal/i segnale/i in ingresso

# Il transistor

Velocissimo interruttore binario formato da:

- **Collettore**
- **Base** → segnale in ingresso (*input*)
- **Emettitore** → messa a terra



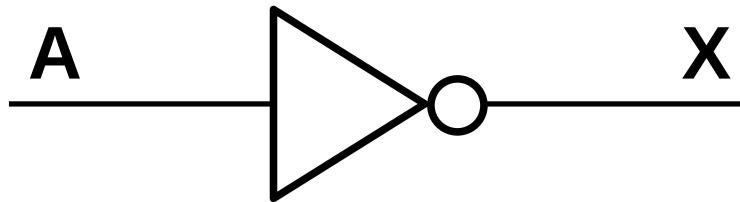
# Invertitore

2 casi possibili:

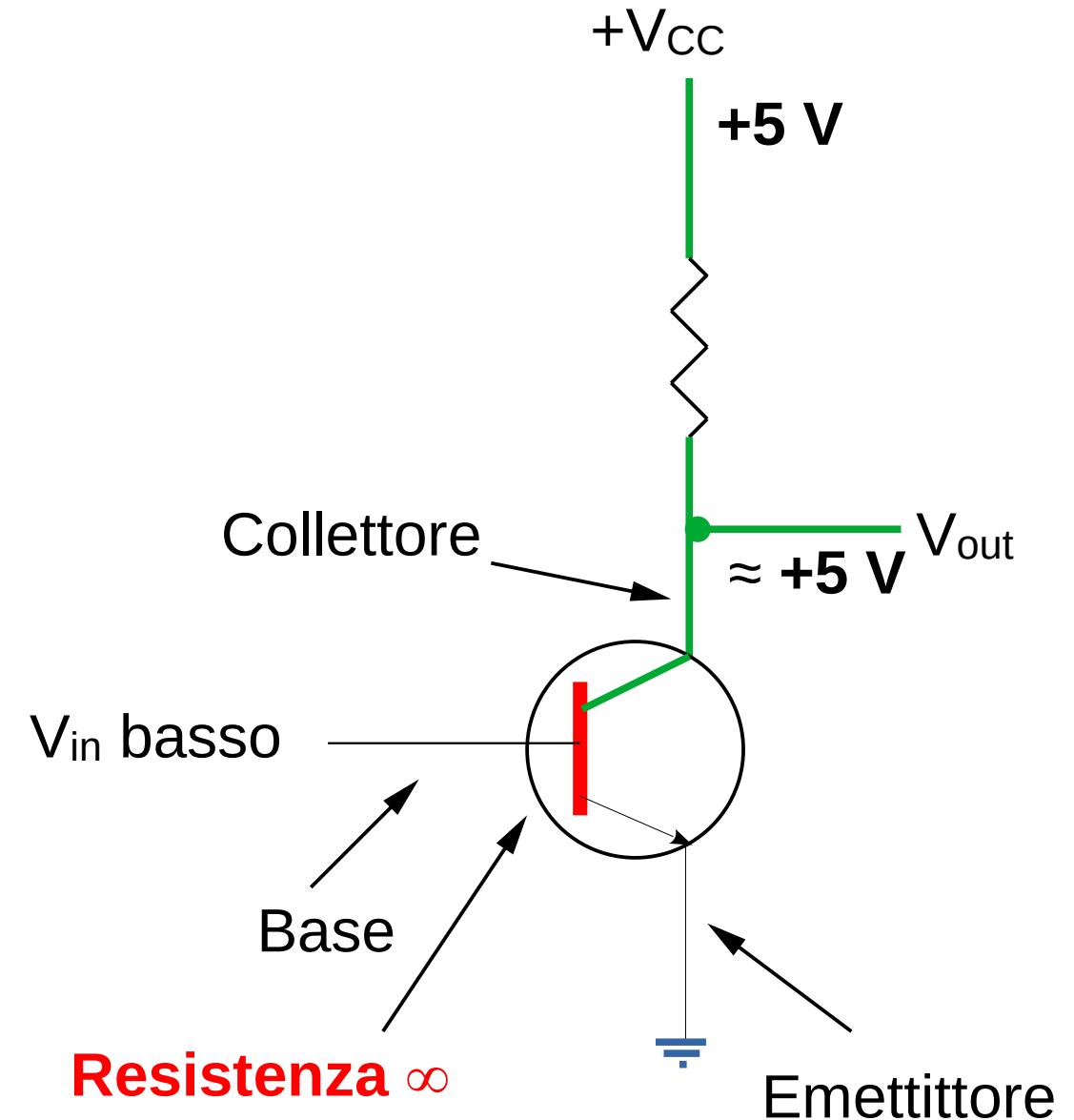
$V_{in}$  è basso  $\rightarrow V_{out} \approx V_{cc}$

(tensione regolata esternamente = +5 V)

NOT



A	X
0	1
1	0

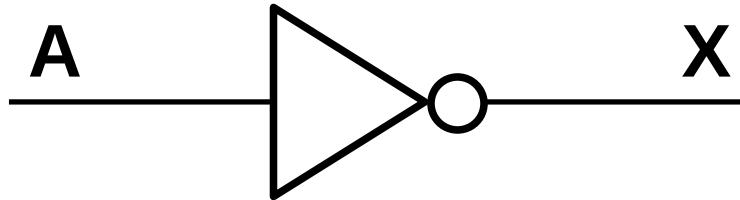


# Invertitore

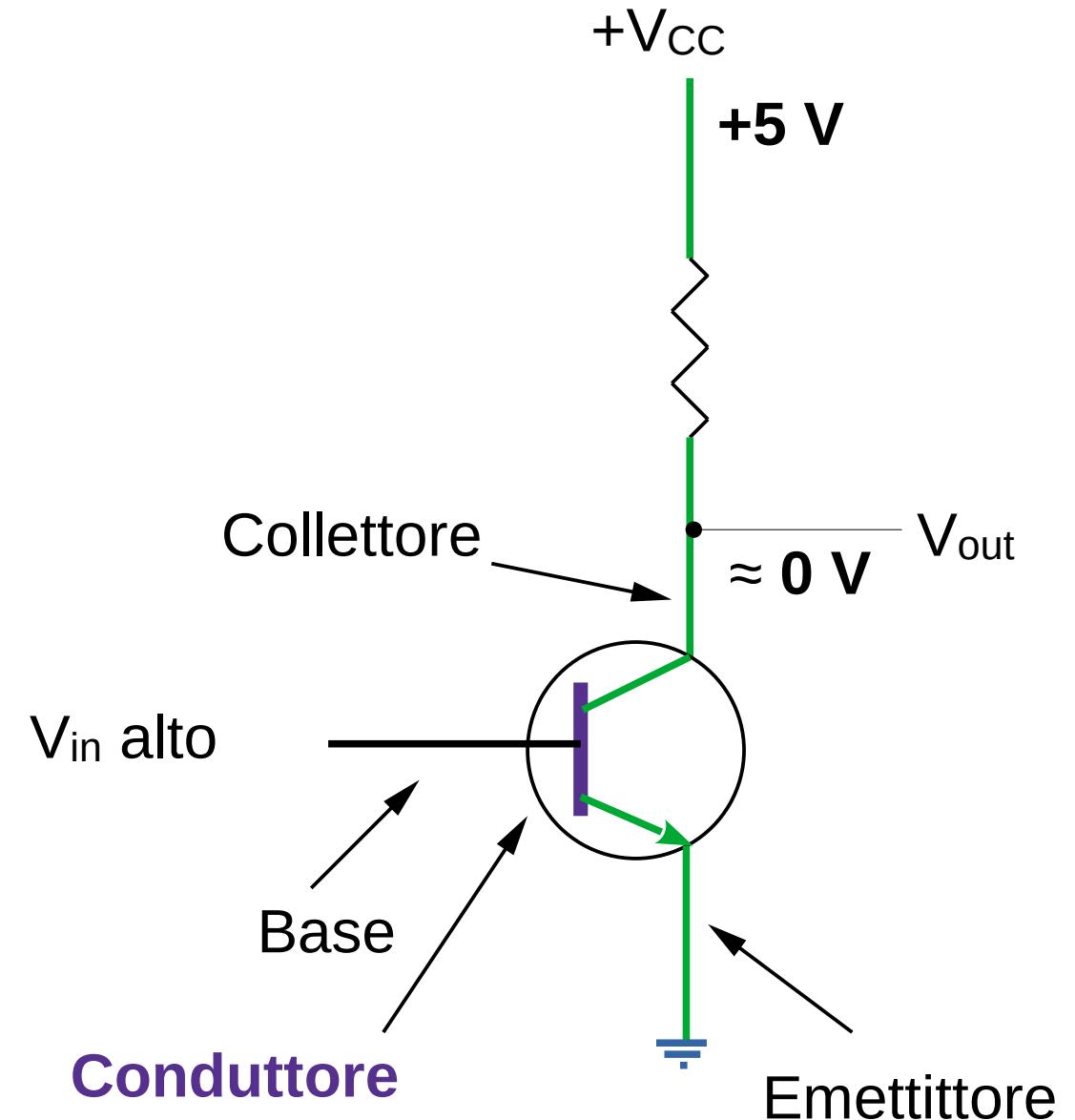
2 casi possibili:

$V_{in}$  è alto  $\rightarrow V_{out} \approx 0 \text{ V}$

NOT



A	X
0	1
1	0



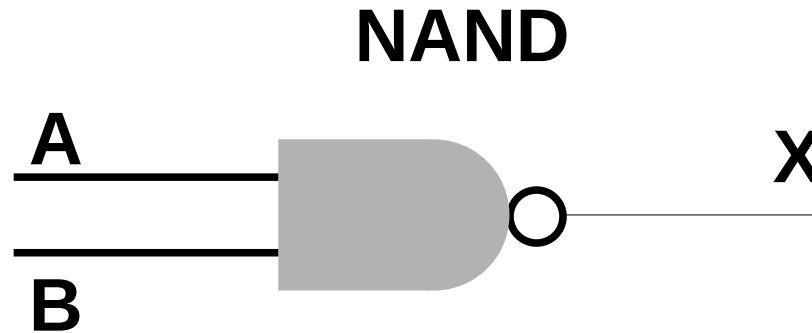
# Porta NAND

4 casi possibili:

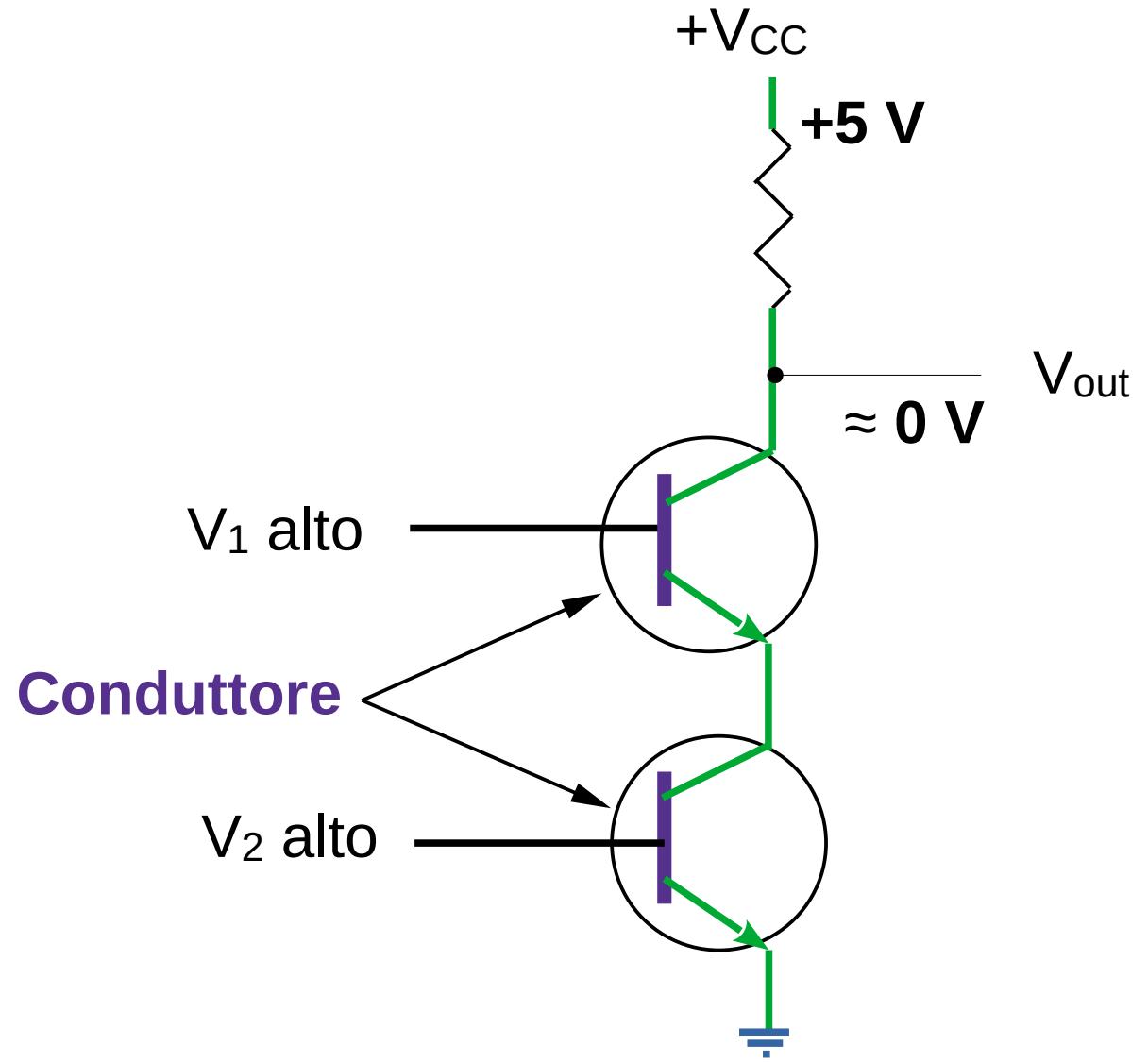
$V_1$  è alto

$V_2$  è alto

$$\rightarrow V_{\text{out}} \approx 0 \text{ V}$$



A	B	X
1	1	0
1	0	1
0	1	1
0	0	1

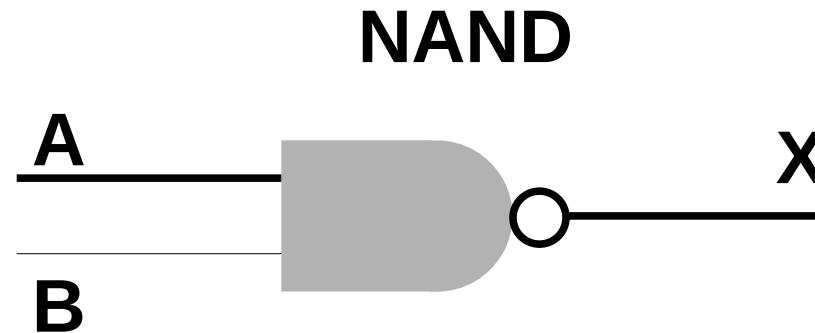


# Porta NAND

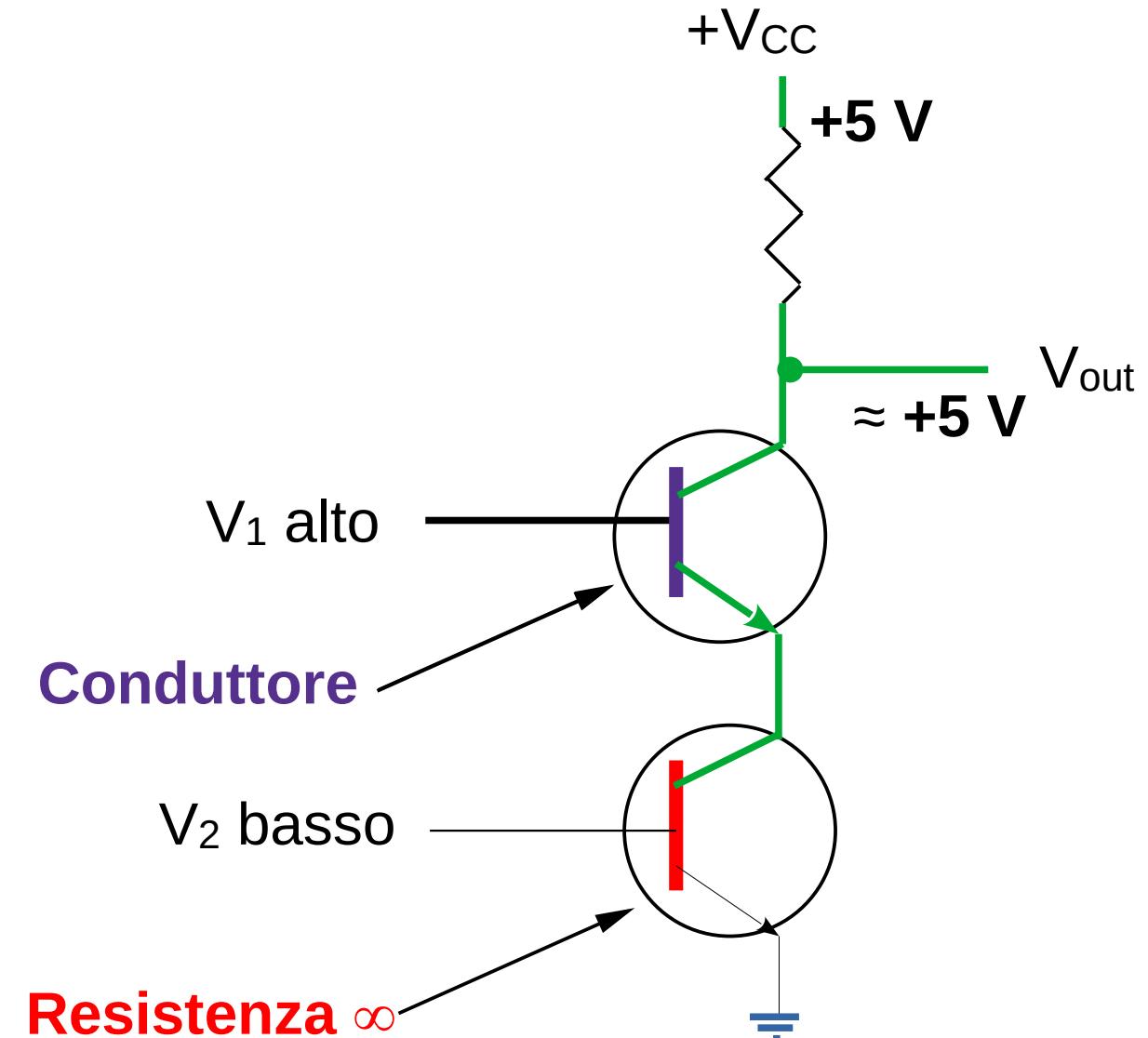
4 casi possibili:

$V_1$  è alto

$V_2$  è basso  $\rightarrow V_{out} \approx +5 \text{ V}$



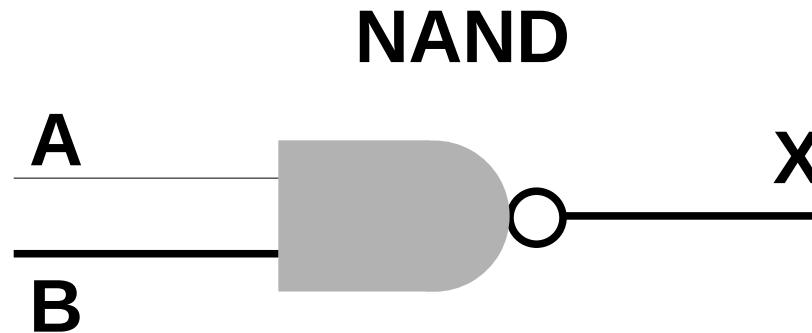
A	B	X
1	1	0
1	0	1
0	1	1
0	0	1



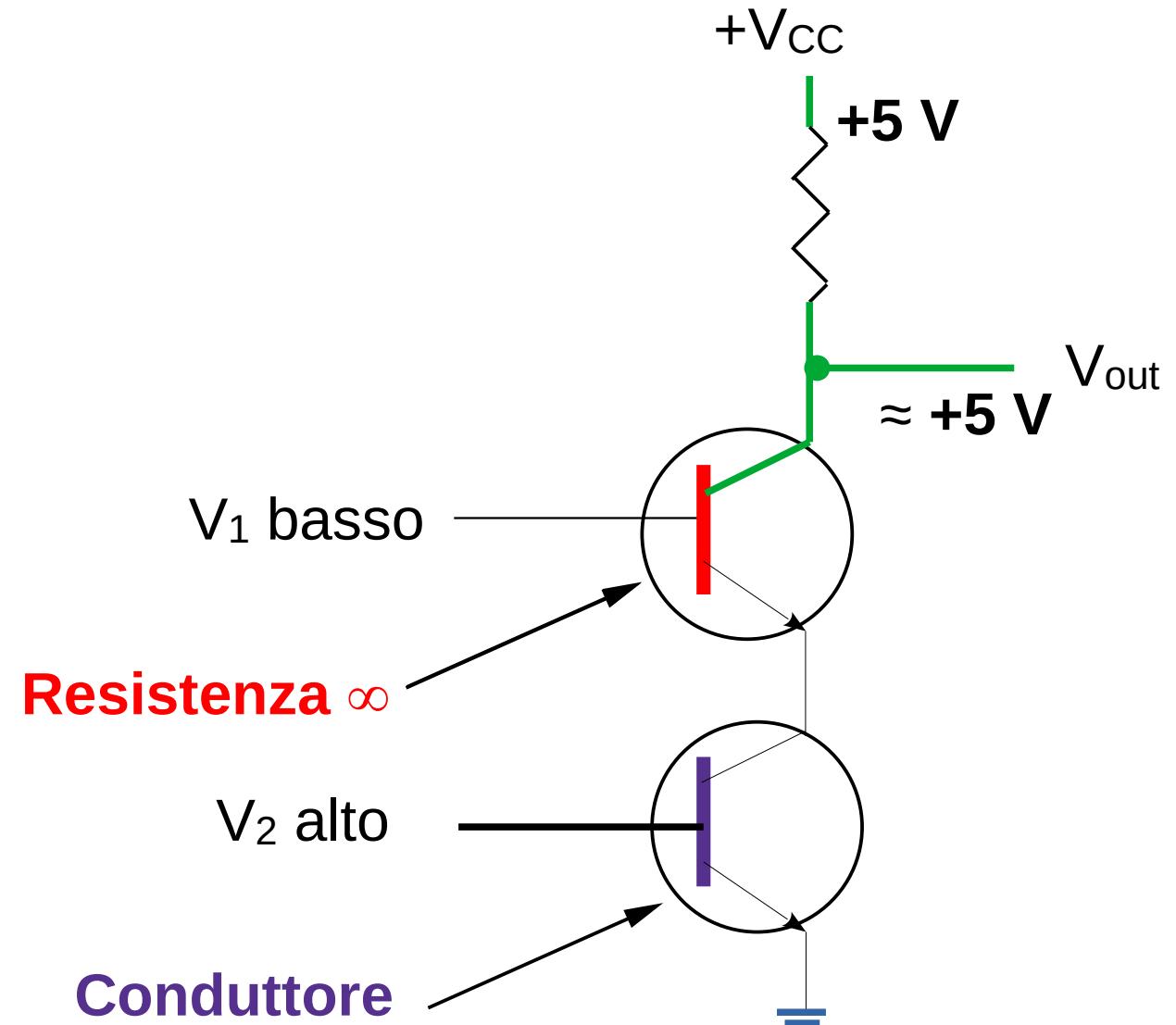
# Porta NAND

4 casi possibili:

$V_1$  è basso  $\rightarrow V_{out} \approx +5 \text{ V}$   
 $V_2$  è alto



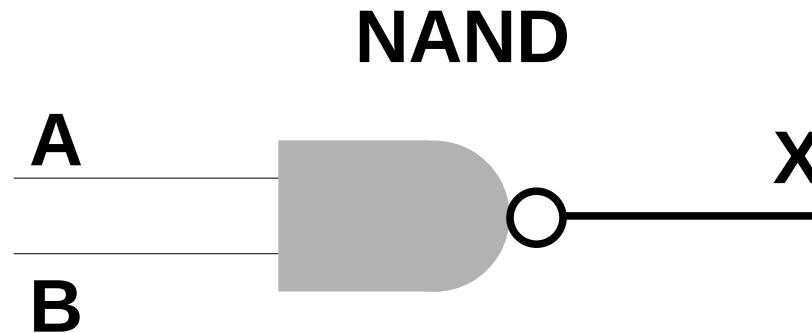
A	B	X
1	1	0
1	0	1
0	1	1
0	0	1



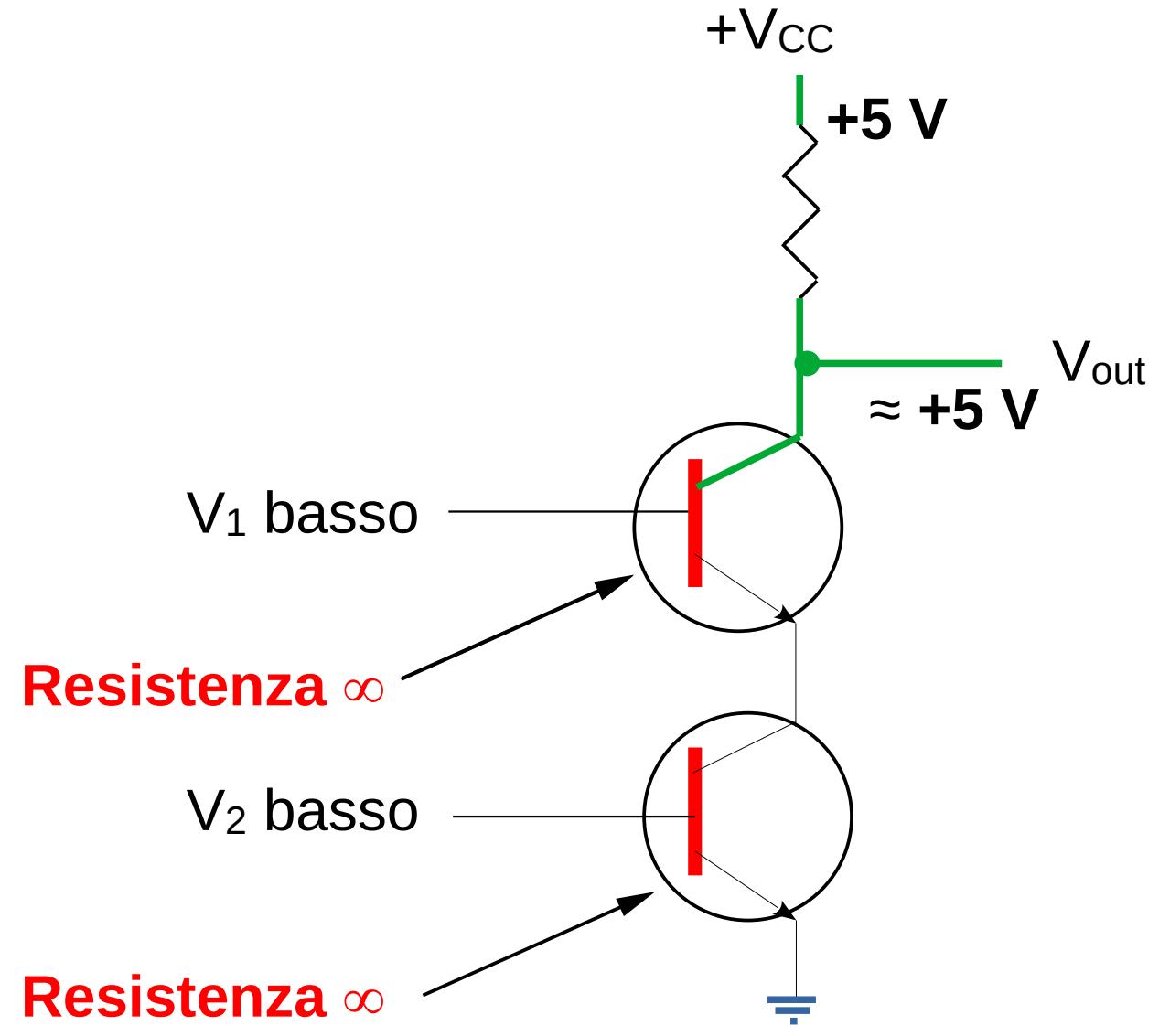
# Porta NAND

4 casi possibili:

$V_1$  è basso  $\rightarrow V_{out} \approx +5 \text{ V}$   
 $V_2$  è basso



A	B	X
1	1	0
1	0	1
0	1	1
0	0	1



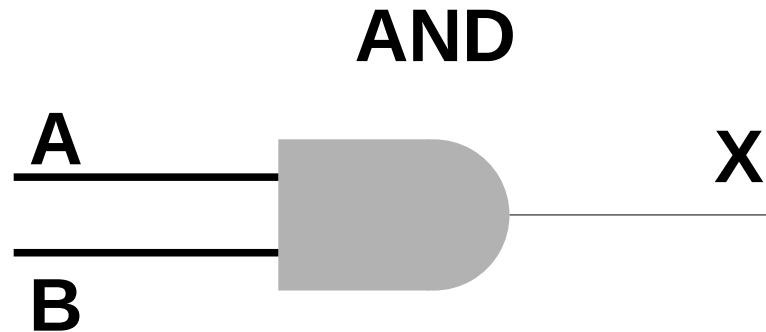
# Porta AND

4 casi possibili:

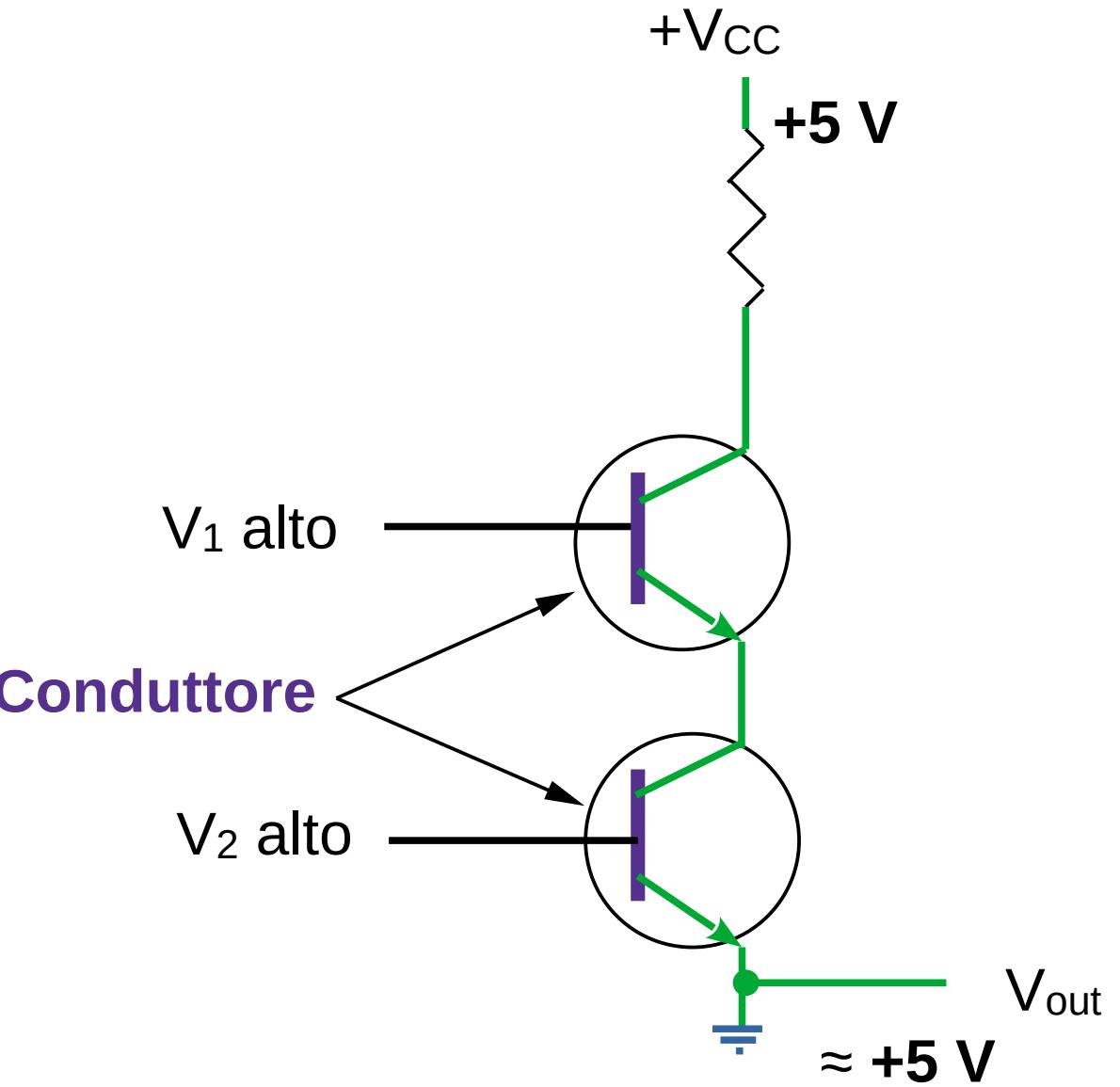
$V_1$  è alto

$V_2$  è alto

$$\rightarrow V_{\text{out}} \approx +5 \text{ V}$$



A	B	X
1	1	1
1	0	0
0	1	0
0	0	0

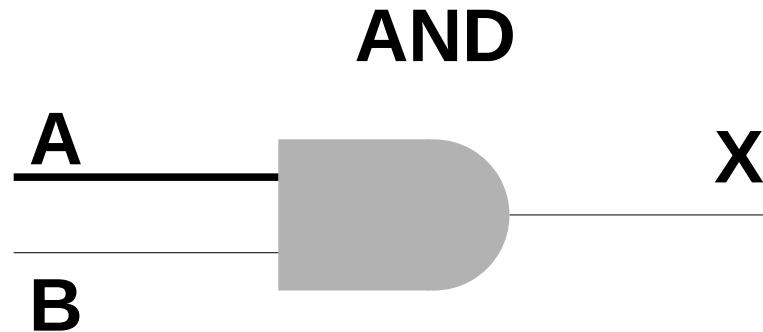


# Porta AND

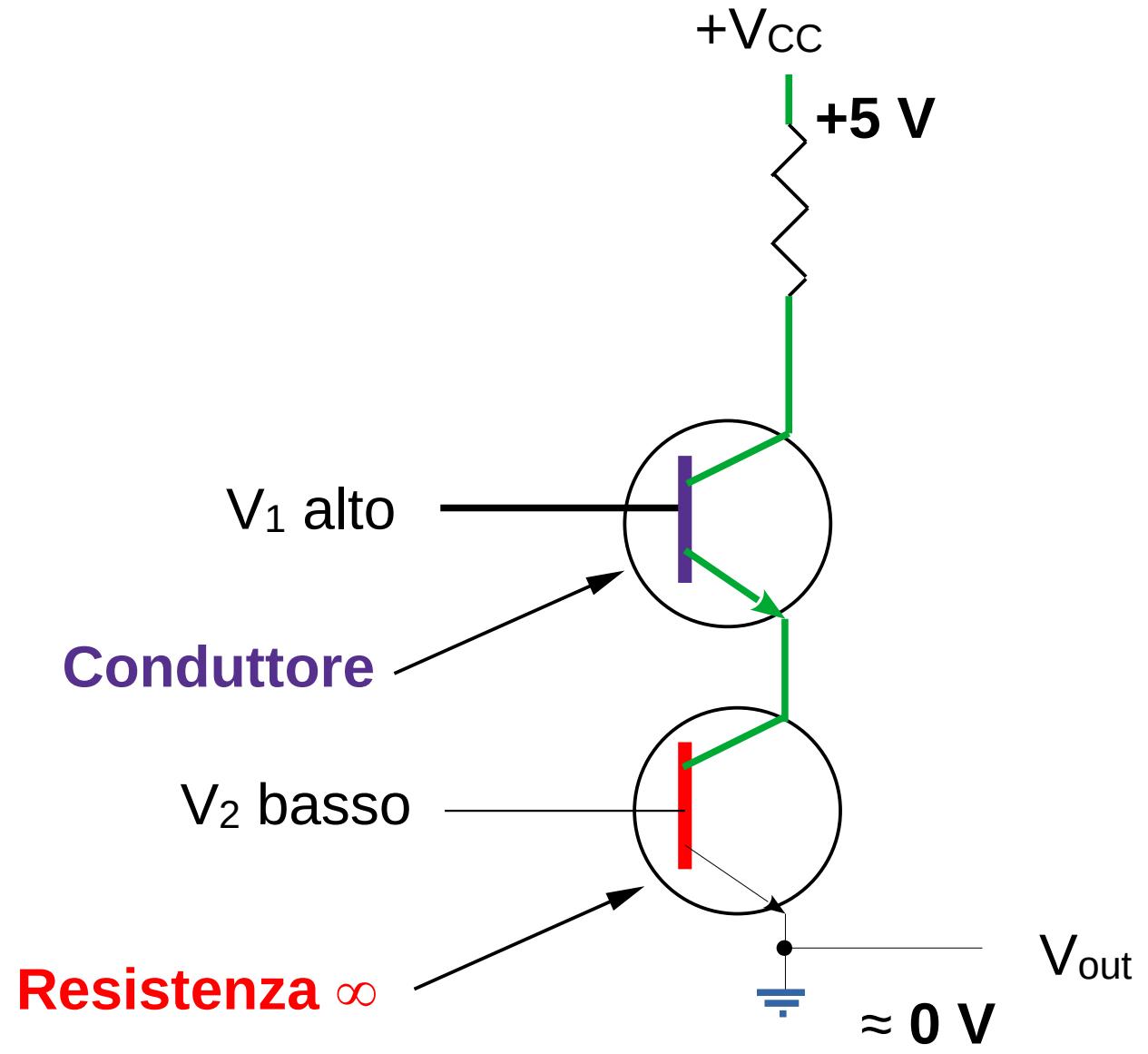
4 casi possibili:

$V_1$  è alto

$V_2$  è basso  $\rightarrow V_{out} \approx 0 \text{ V}$



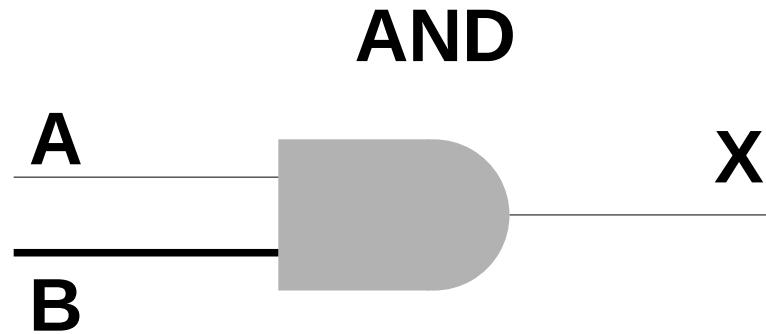
A	B	X
1	1	1
1	0	0
0	1	0
0	0	0



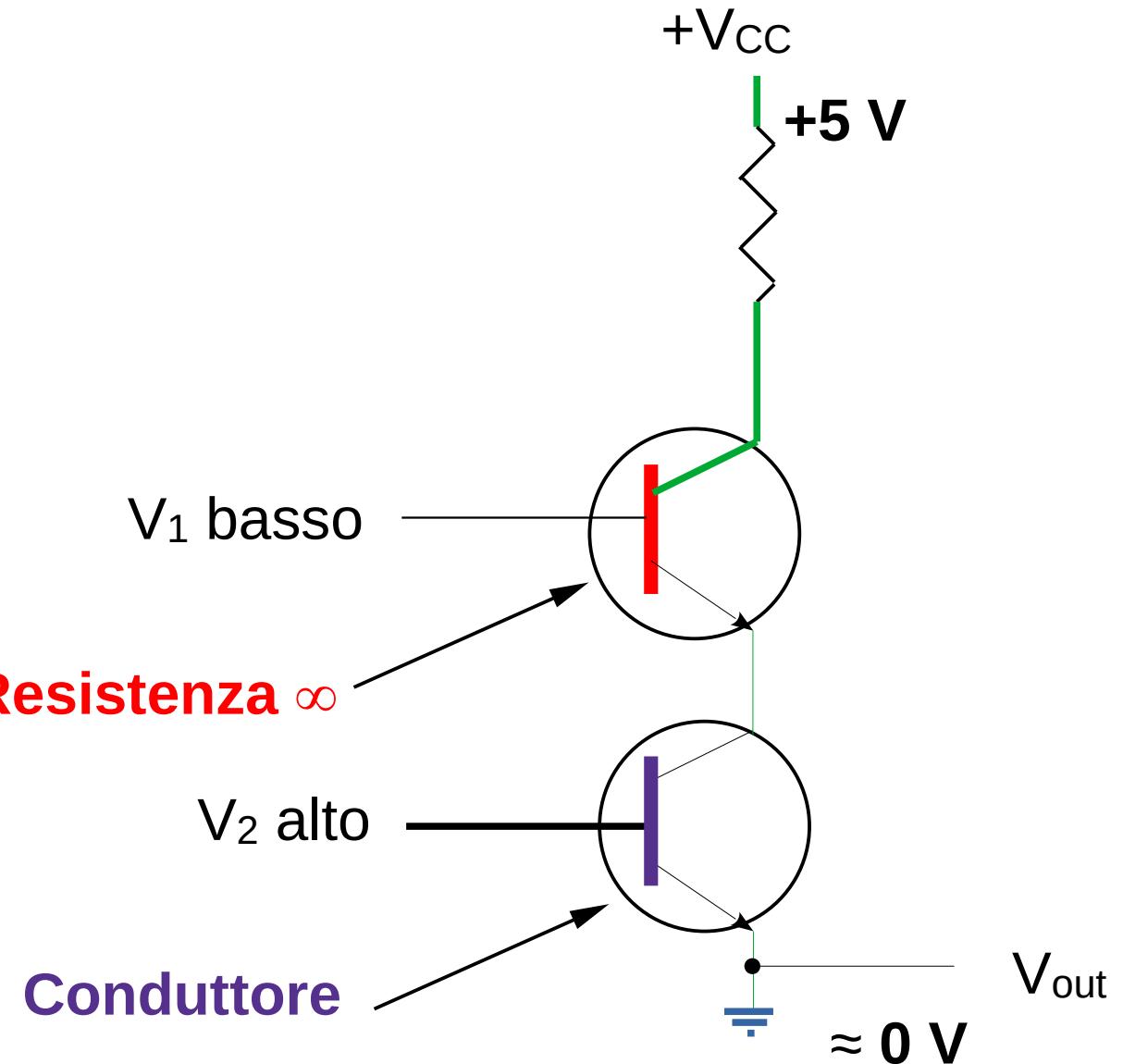
# Porta AND

4 casi possibili:

$V_1$  è basso  $\rightarrow V_{out} \approx 0 \text{ V}$   
 $V_2$  è alto



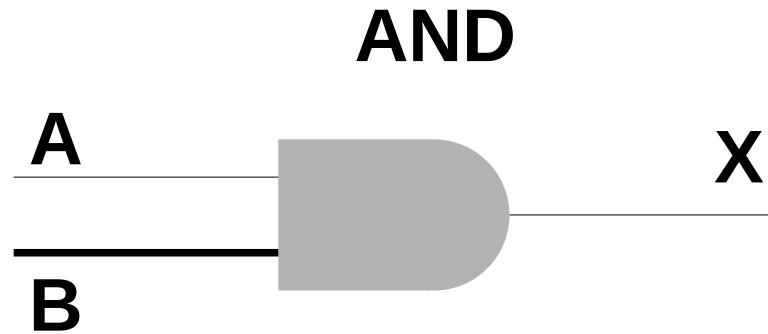
A	B	X
1	1	1
1	0	0
0	1	0
0	0	0



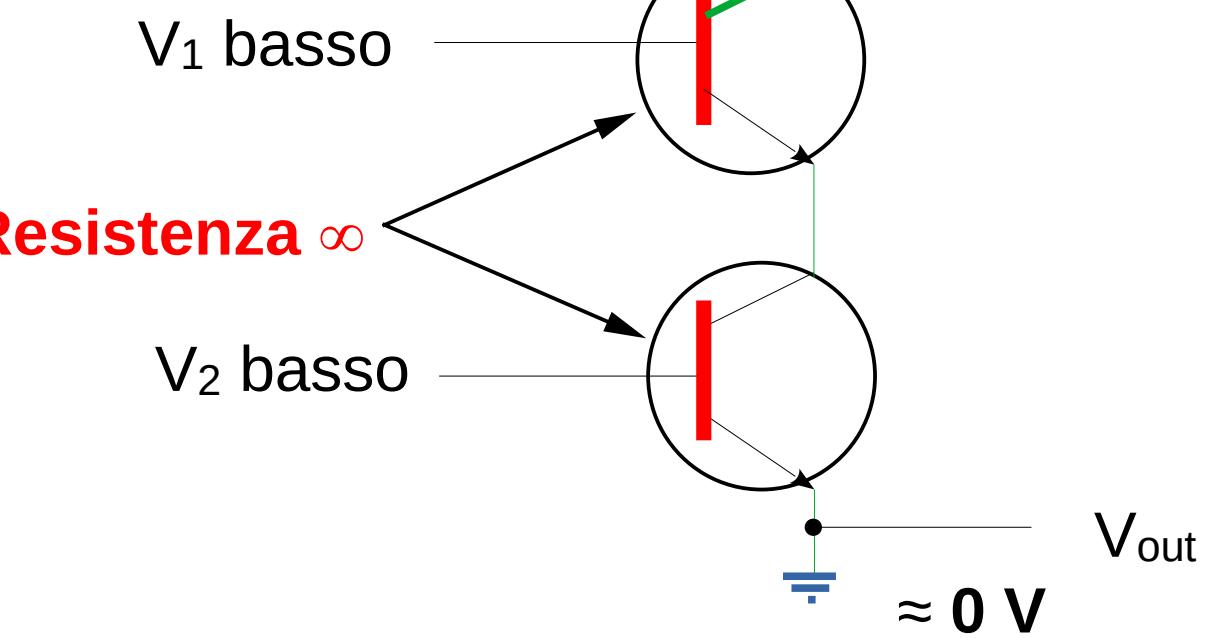
# Porta AND

4 casi possibili:

$V_1$  è basso  $\rightarrow V_{out} \approx 0 \text{ V}$   
 $V_2$  è basso



A	B	X
1	1	1
1	0	0
0	1	0
0	0	0



# Porta NOR

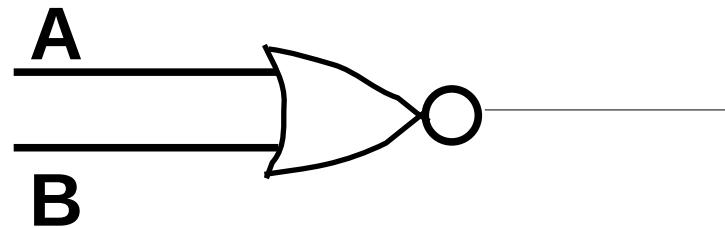
4 casi possibili:

$V_1$  è alto

$V_2$  è alto

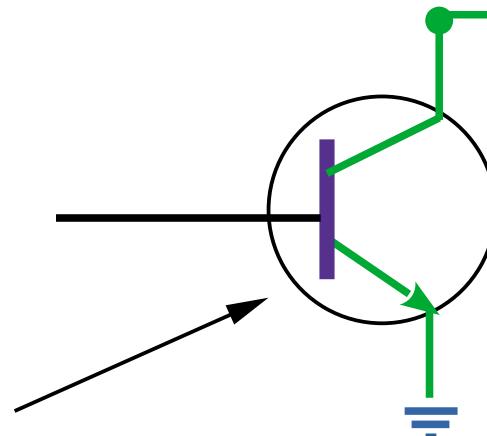
$$\rightarrow V_{\text{out}} \approx 0 \text{ V}$$

NOR



$V_1$  alto

Conduttore

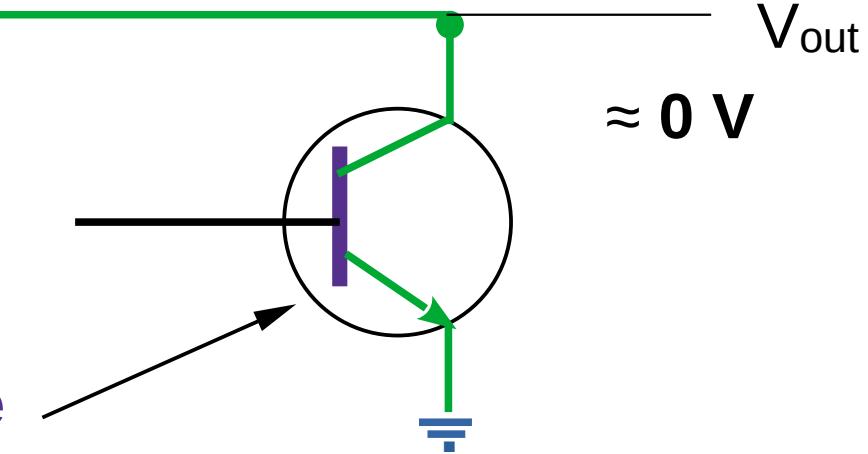


$+V_{\text{CC}}$

$+5 \text{ V}$

$V_2$  alto

Conduttore



$V_{\text{out}}$   
 $\approx 0 \text{ V}$

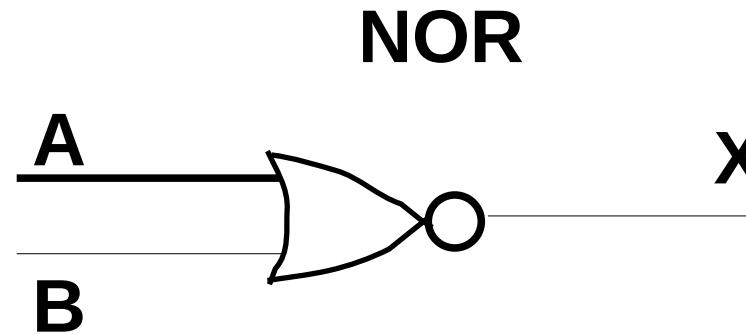
A	B	X
1	1	0
1	0	0
0	1	0
0	0	1

# Porta NOR

4 casi possibili:

$V_1$  è alto

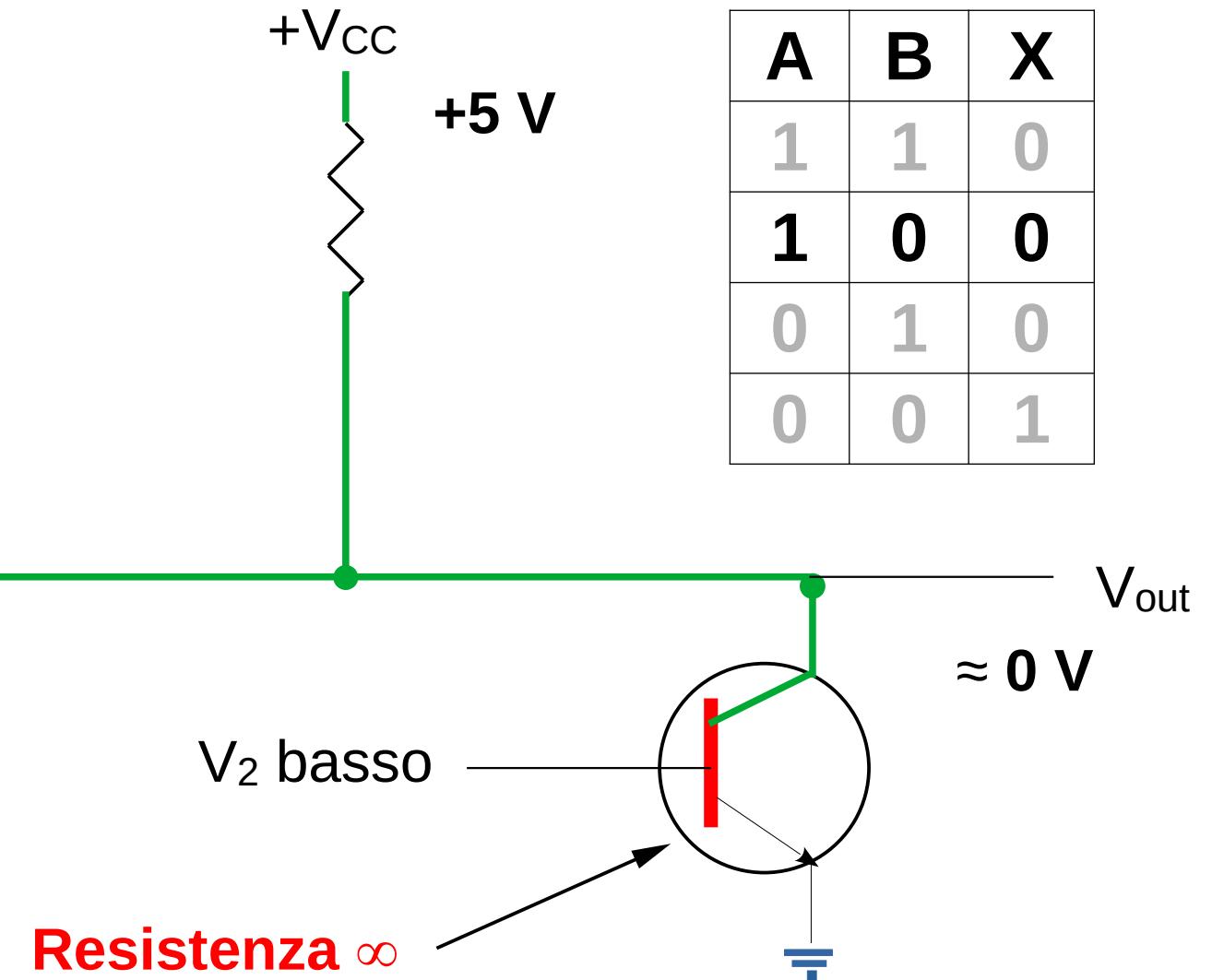
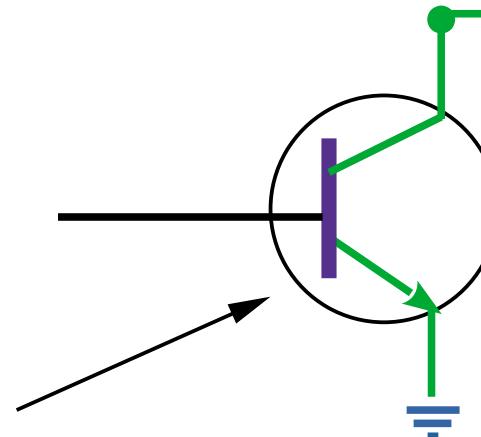
$V_2$  è basso  $\rightarrow V_{out} \approx 0 \text{ V}$



B

$V_1$  alto

Conduttore

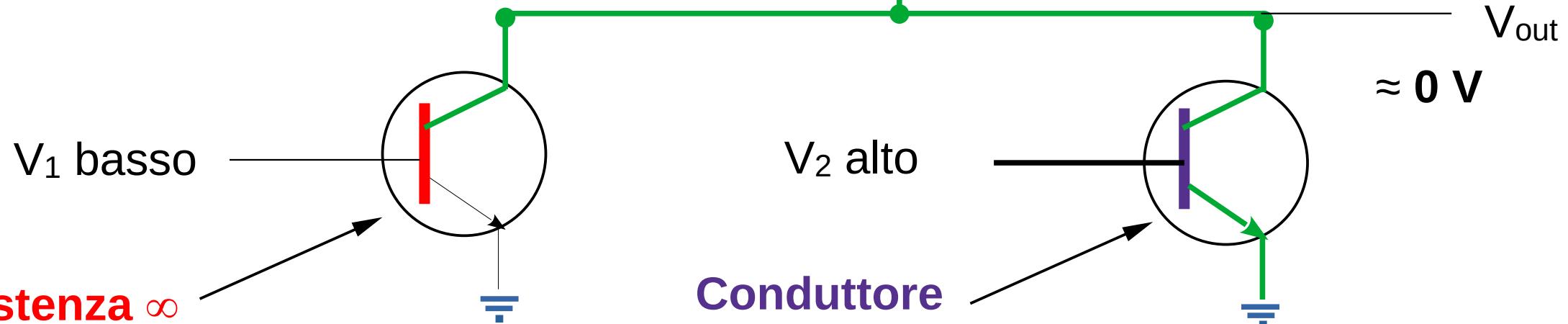
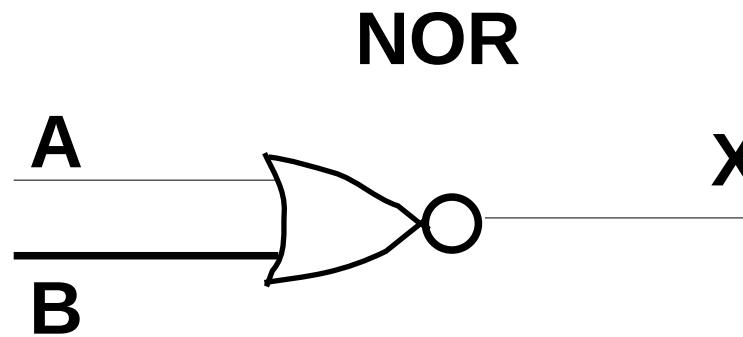


A	B	X
1	1	0
1	0	0
0	1	0
0	0	1

# Porta NOR

4 casi possibili:

$V_1$  è basso  $\rightarrow V_{out} \approx 0 \text{ V}$   
 $V_2$  è alto



$+V_{CC}$

$+5 \text{ V}$

A	B	X
1	1	0
1	0	0
0	1	0
0	0	1

$V_{out}$

$\approx 0 \text{ V}$

$V_2$  alto

Conduttore

# Porta NOR

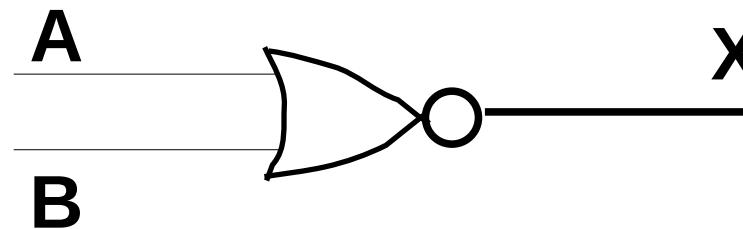
4 casi possibili:

$V_1$  è basso

$$\rightarrow V_{\text{out}} \approx +5 \text{ V}$$

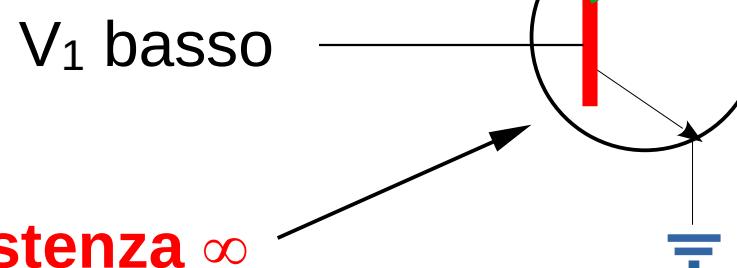
$V_2$  è basso

**NOR**



$V_1$  basso

**Resistenza  $\infty$**



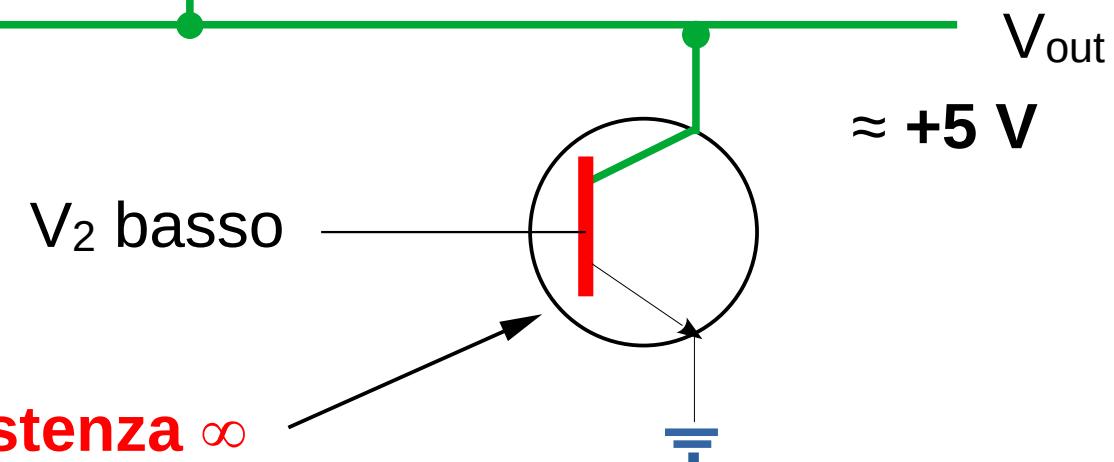
$+V_{\text{CC}}$

+5 V

A	B	X
1	1	0
1	0	0
0	1	0
0	0	1

$V_2$  basso

**Resistenza  $\infty$**



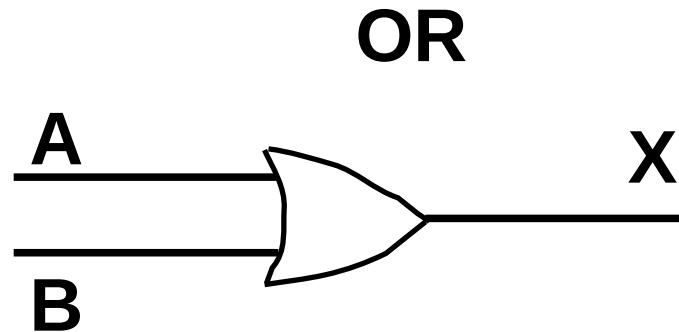
# Porta OR

4 casi possibili:

$V_1$  è alto

$V_2$  è alto

$$\rightarrow V_{\text{out}} \approx +6 \text{ V}$$

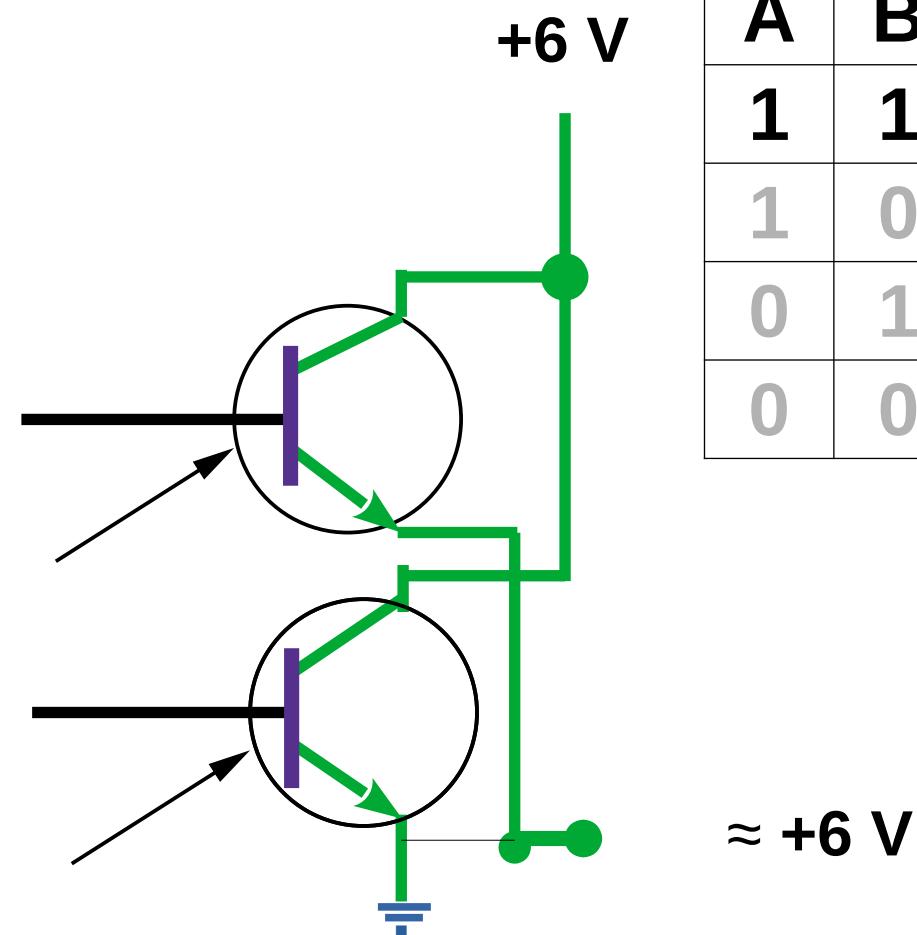


$V_1$  alto

**Conduttore**

$V_2$  alto

**Conduttore**



A	B	X
1	1	1
1	0	1
0	1	1
0	0	0

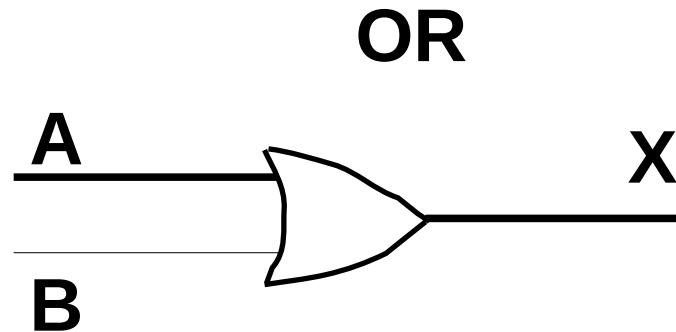
# Porta OR

4 casi possibili:

$V_1$  è alto

$V_2$  è basso

$$\rightarrow V_{\text{out}} \approx +6 \text{ V}$$

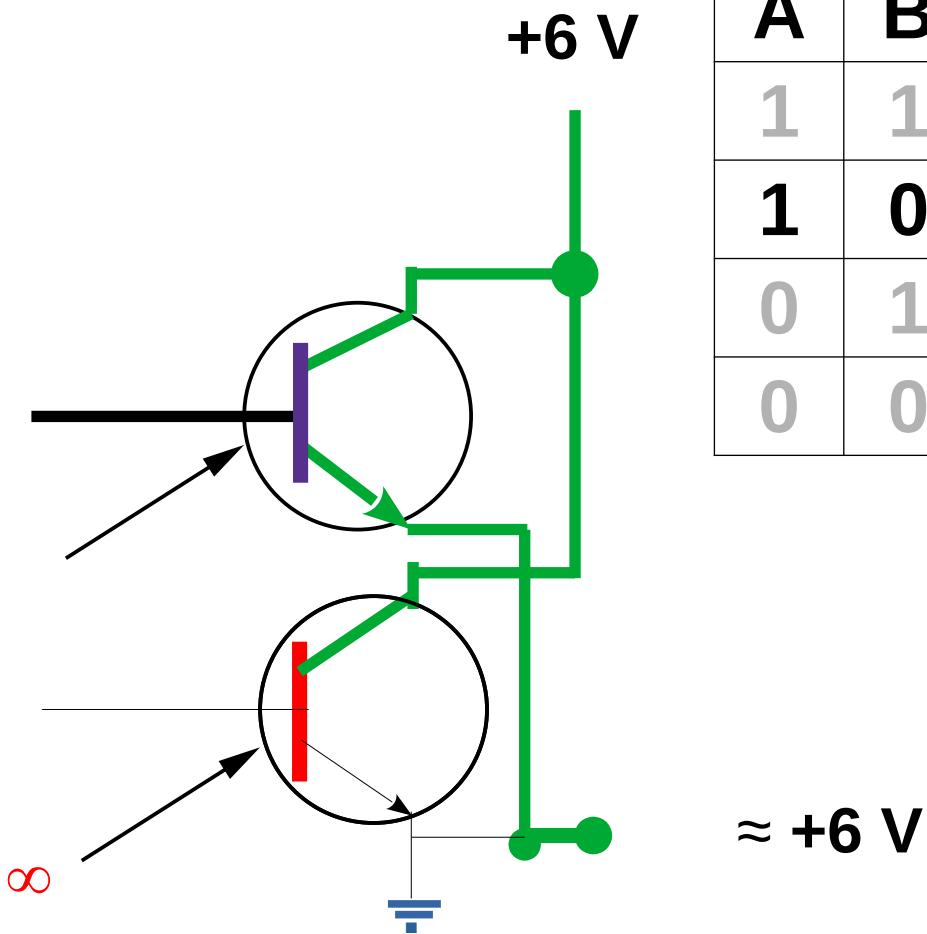


$V_1$  alto

**Conduttore**

$V_2$  basso

**Resistenza  $\infty$**



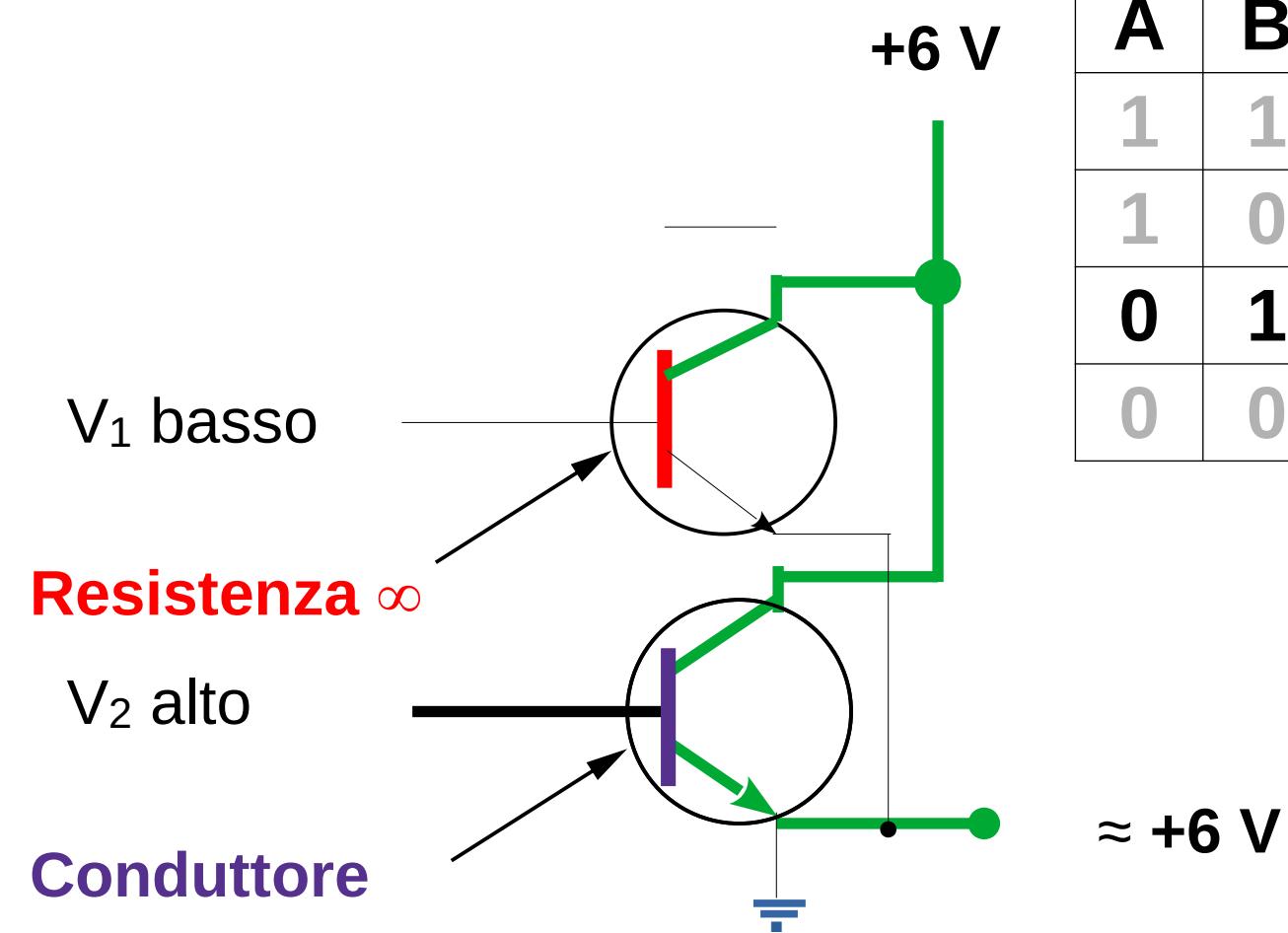
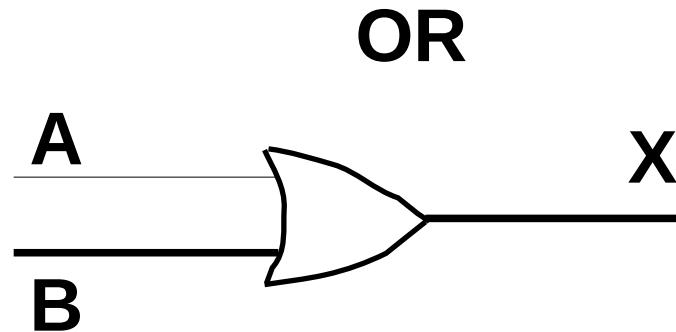
$$\approx +6 \text{ V}$$

A	B	X
1	1	1
1	0	1
0	1	1
0	0	0

# Porta OR

4 casi possibili:

$V_1$  è basso  $\rightarrow V_{out} \approx +6 \text{ V}$   
 $V_2$  è alto

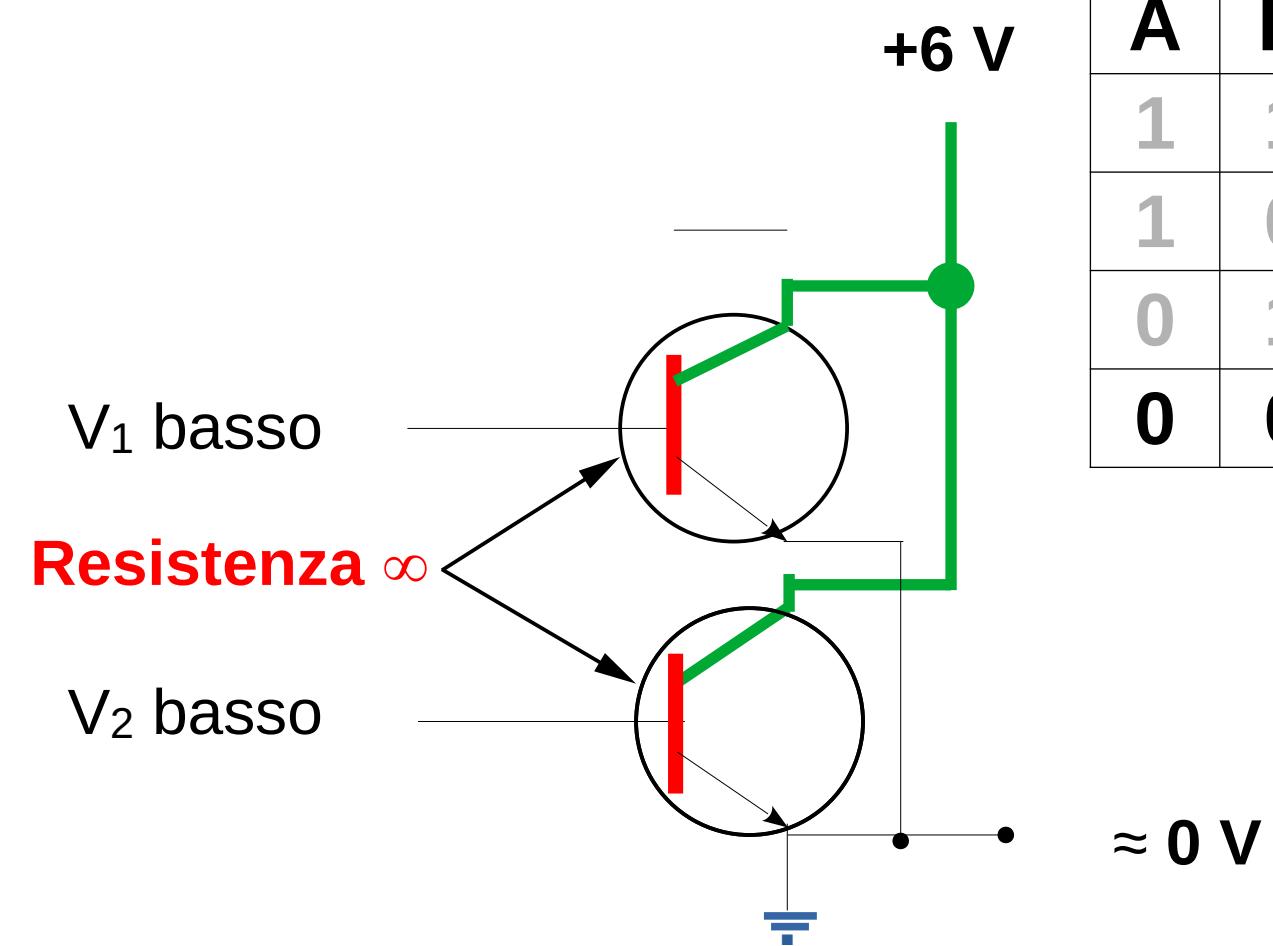
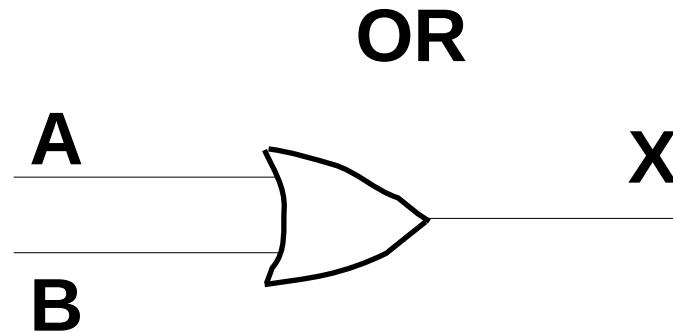


A	B	X
1	1	1
1	0	1
0	1	1
0	0	0

# Porta OR

4 casi possibili:

$V_1$  è basso  $\rightarrow V_{out} \approx 0 \text{ V}$   
 $V_2$  è basso



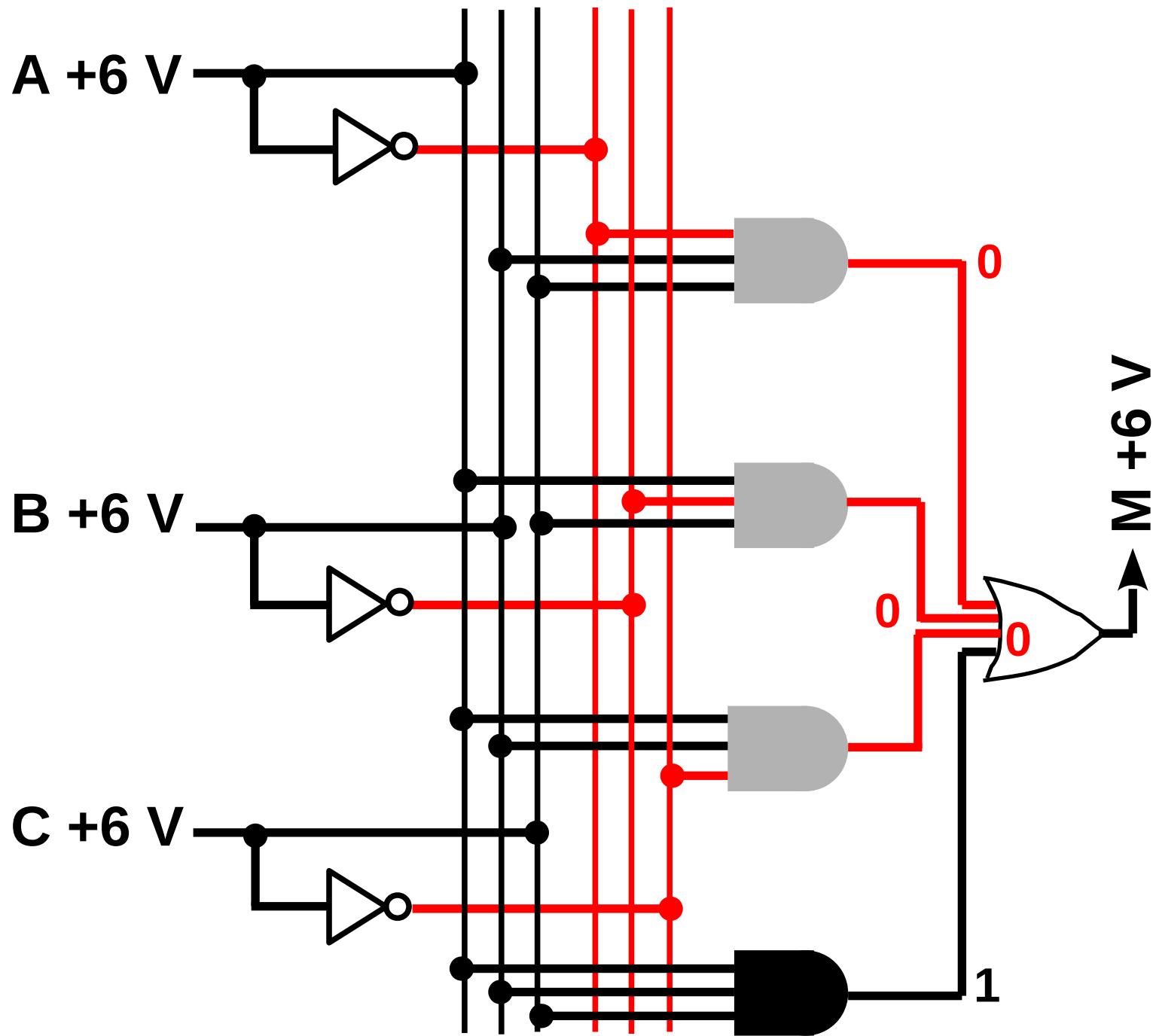
A	B	X
1	1	1
1	0	1
0	1	1
0	0	0

**NOT** = invertitore

**AND** = prodotto logico

**OR** = somma logica

A	B	C	M
1	1	1	1
1	1	0	1
1	0	1	1
1	0	0	0
0	1	1	1
0	1	0	0
0	0	1	0
0	0	0	0

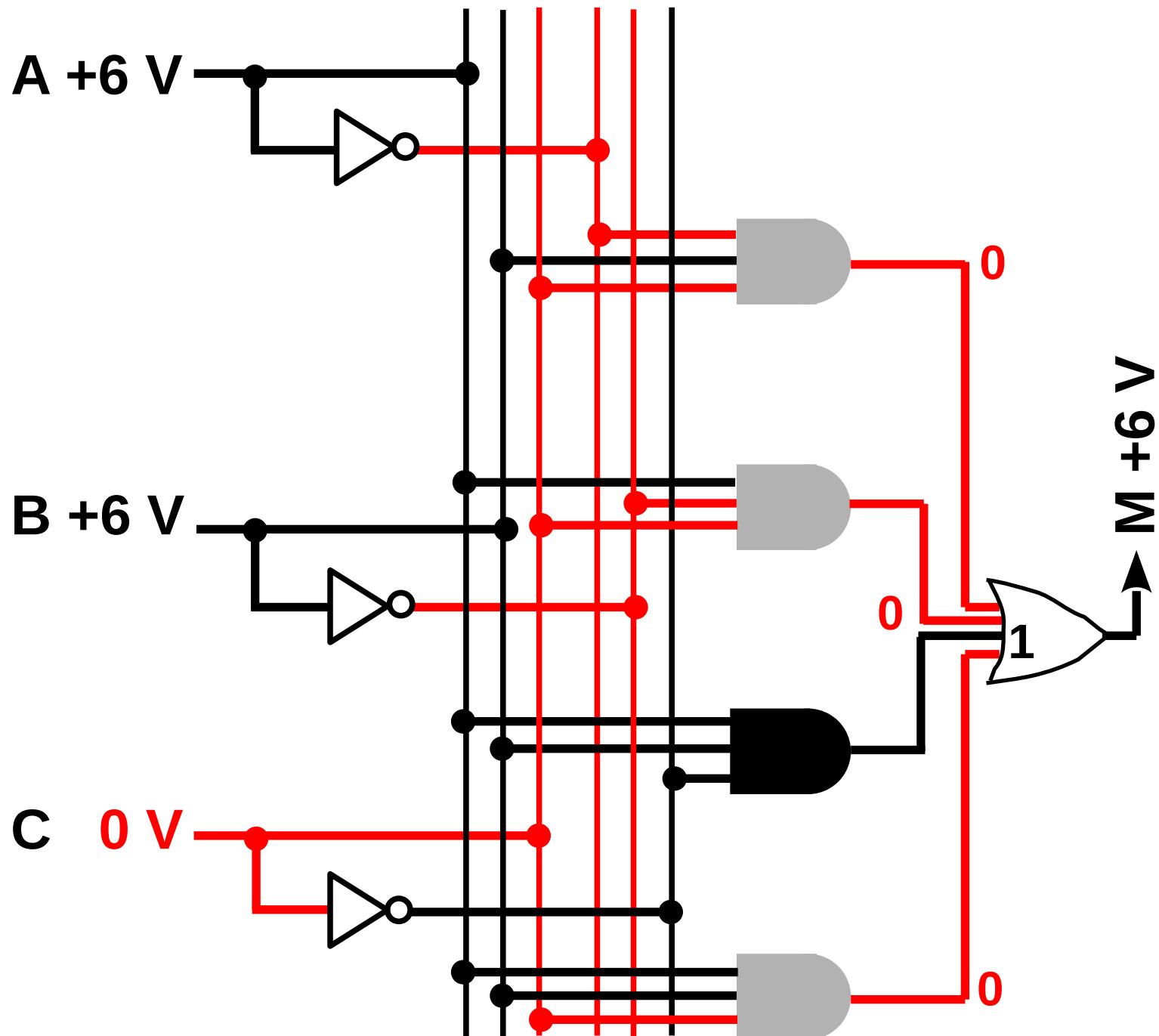


**NOT** = invertitore

**AND** = prodotto logico

**OR** = somma logica

A	B	C	M
1	1	1	1
1	1	0	1
1	0	1	1
1	0	0	0
0	1	1	1
0	1	0	0
0	0	1	0
0	0	0	0

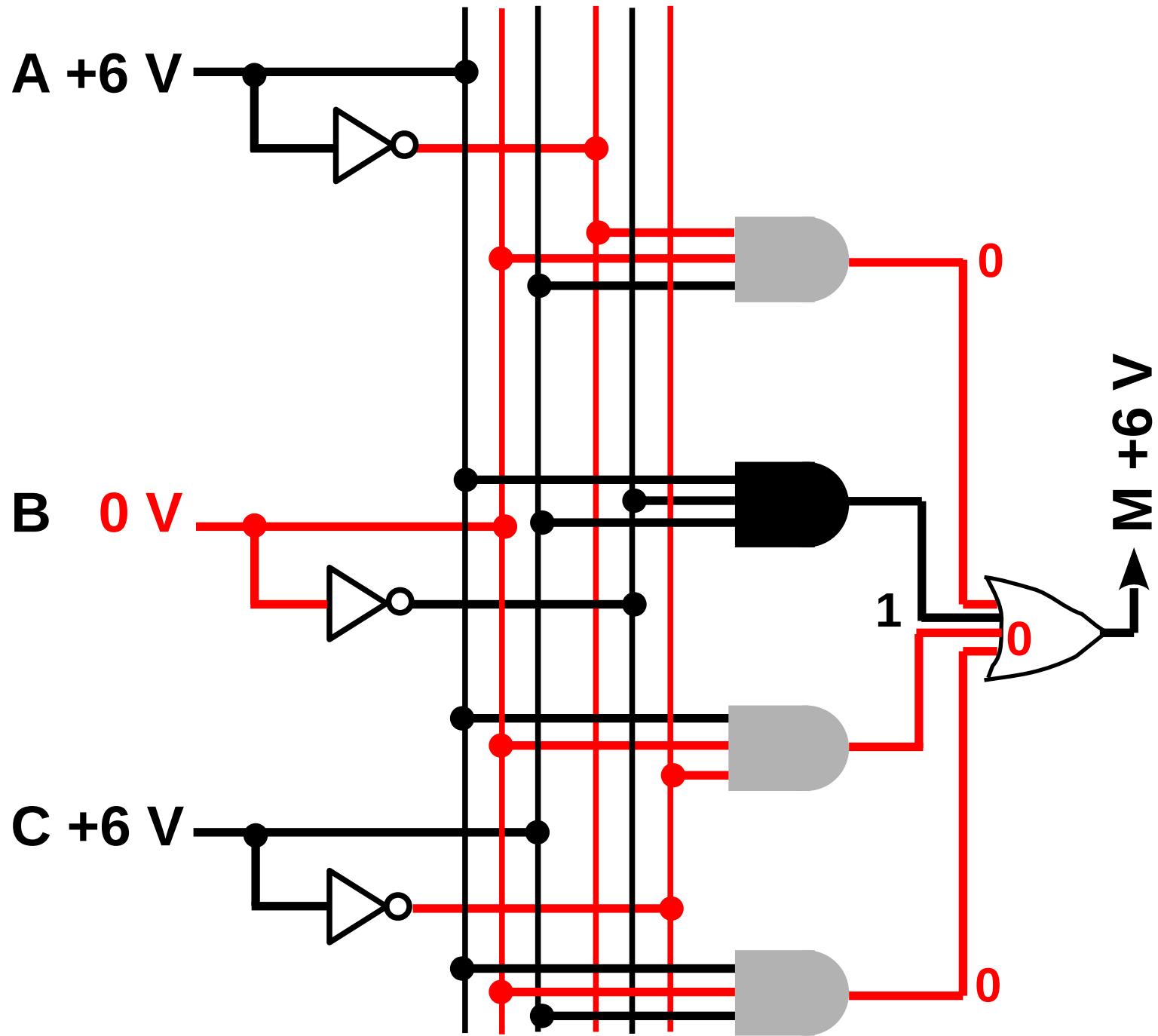


**NOT** = invertitore

**AND** = prodotto logico

**OR** = somma logica

A	B	C	M
1	1	1	1
1	1	0	1
1	0	1	1
1	0	0	0
0	1	1	1
0	1	0	0
0	0	1	0
0	0	0	0

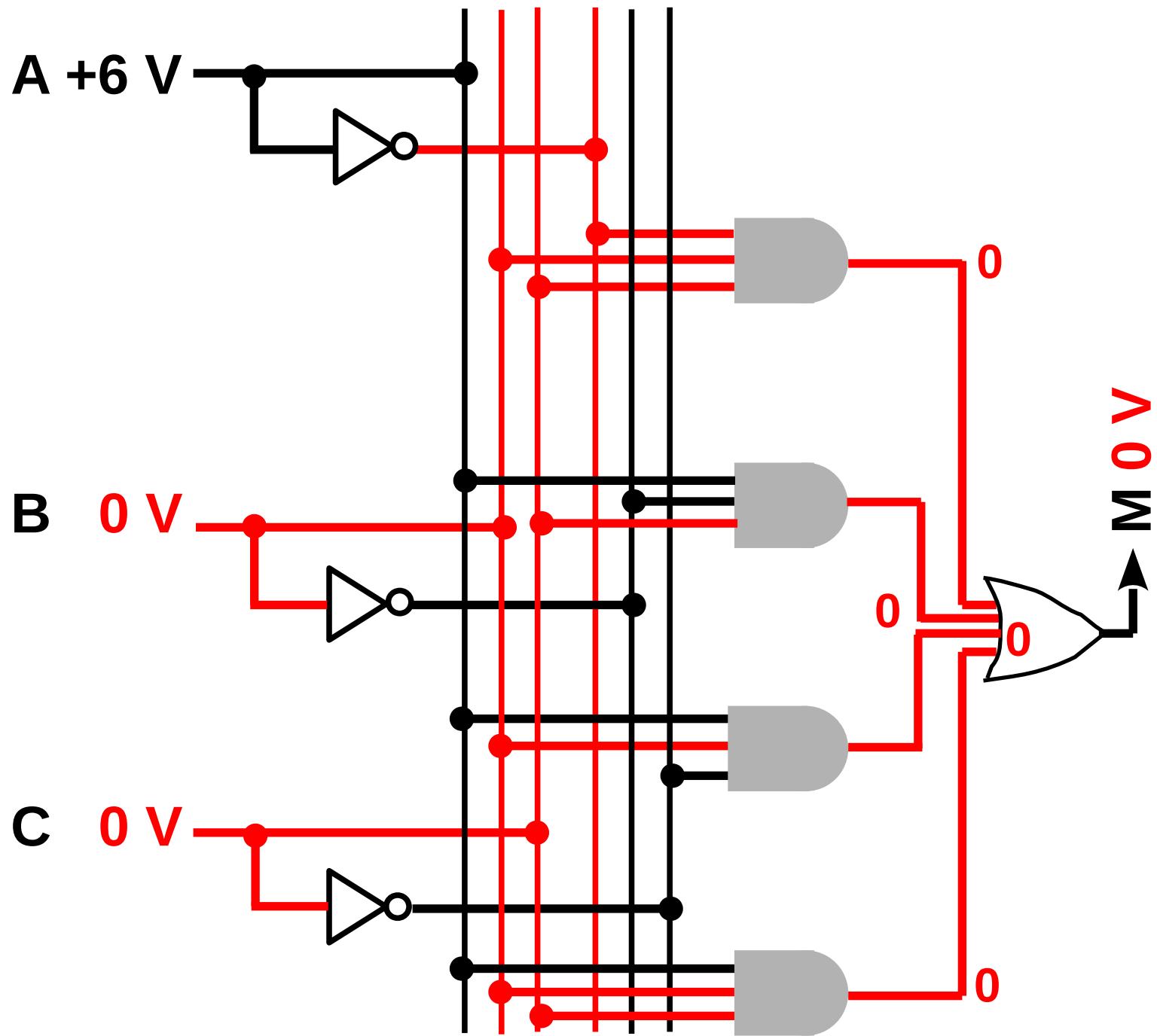


**NOT** = invertitore

**AND** = prodotto logico

**OR** = somma logica

A	B	C	M
1	1	1	1
1	1	0	1
1	0	1	1
1	0	0	0
0	1	1	1
0	1	0	0
0	0	1	0
0	0	0	0

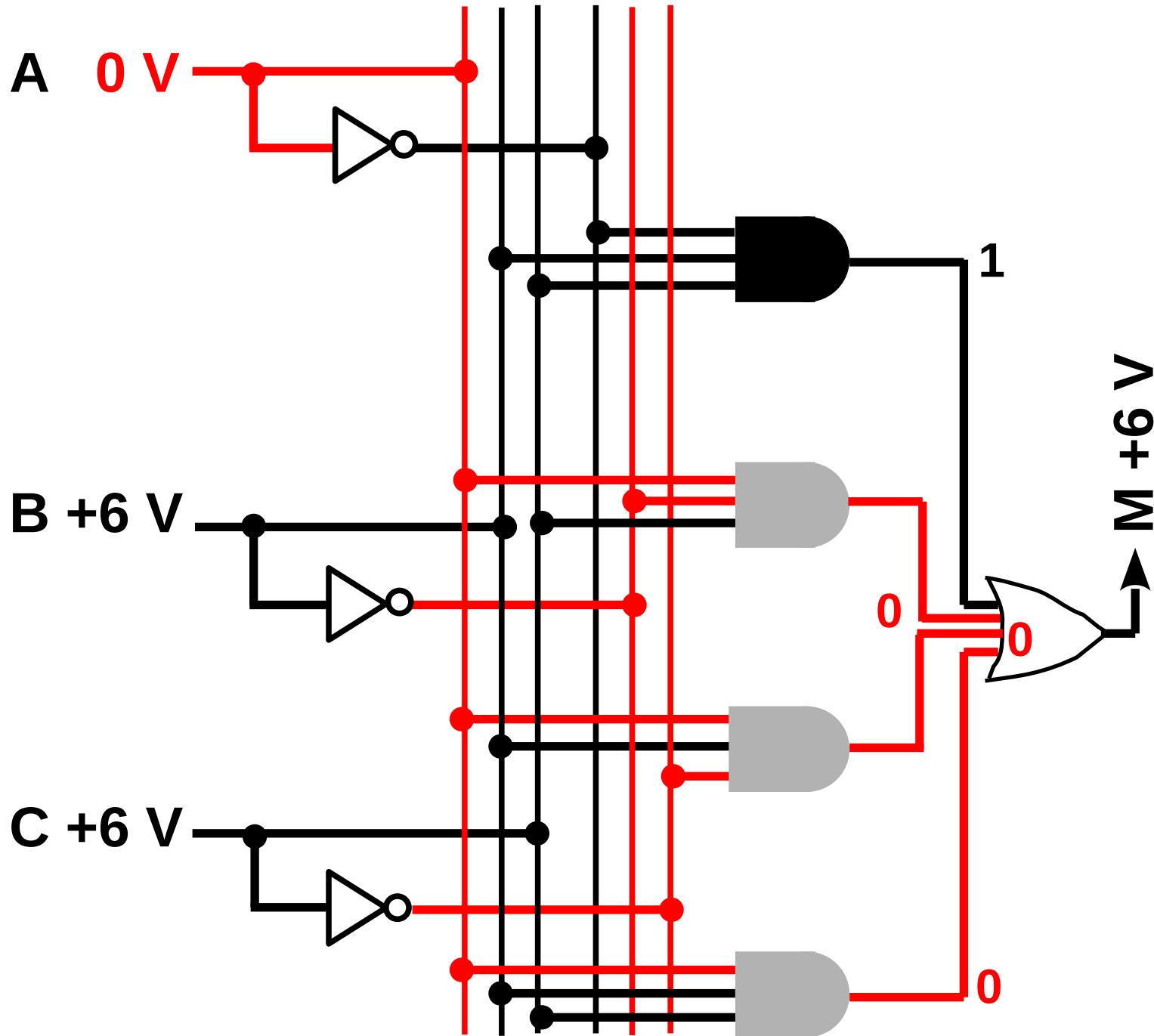


**NOT** = invertitore

**AND** = prodotto logico

**OR** = somma logica

A	B	C	M
1	1	1	1
1	1	0	1
1	0	1	1
1	0	0	0
0	1	1	1
0	1	0	0
0	0	1	0
0	0	0	0

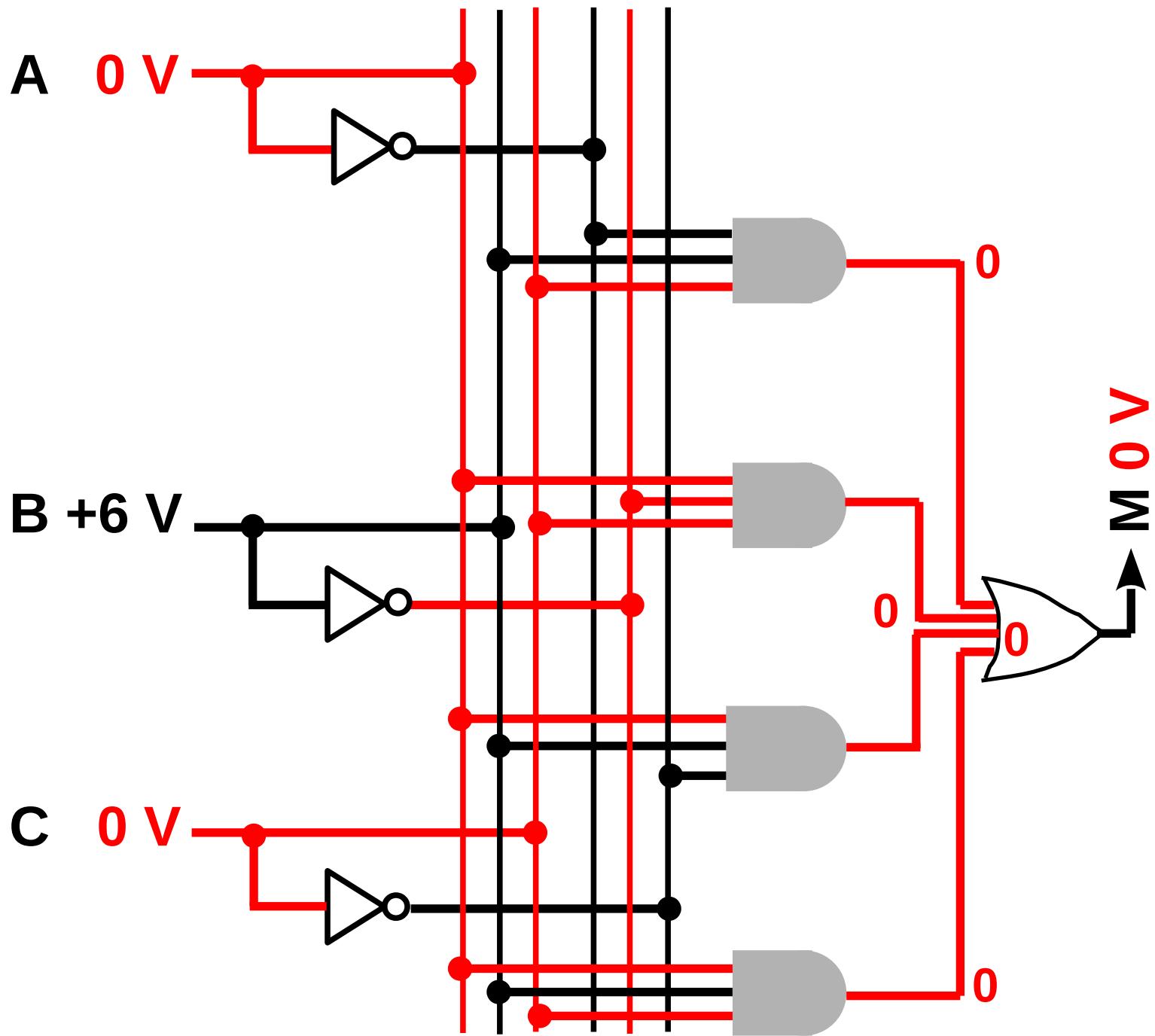


**NOT** = invertitore

**AND** = prodotto logico

**OR** = somma logica

A	B	C	M
1	1	1	1
1	1	0	1
1	0	1	1
1	0	0	0
0	1	1	1
0	1	0	0
0	0	1	0
0	0	0	0

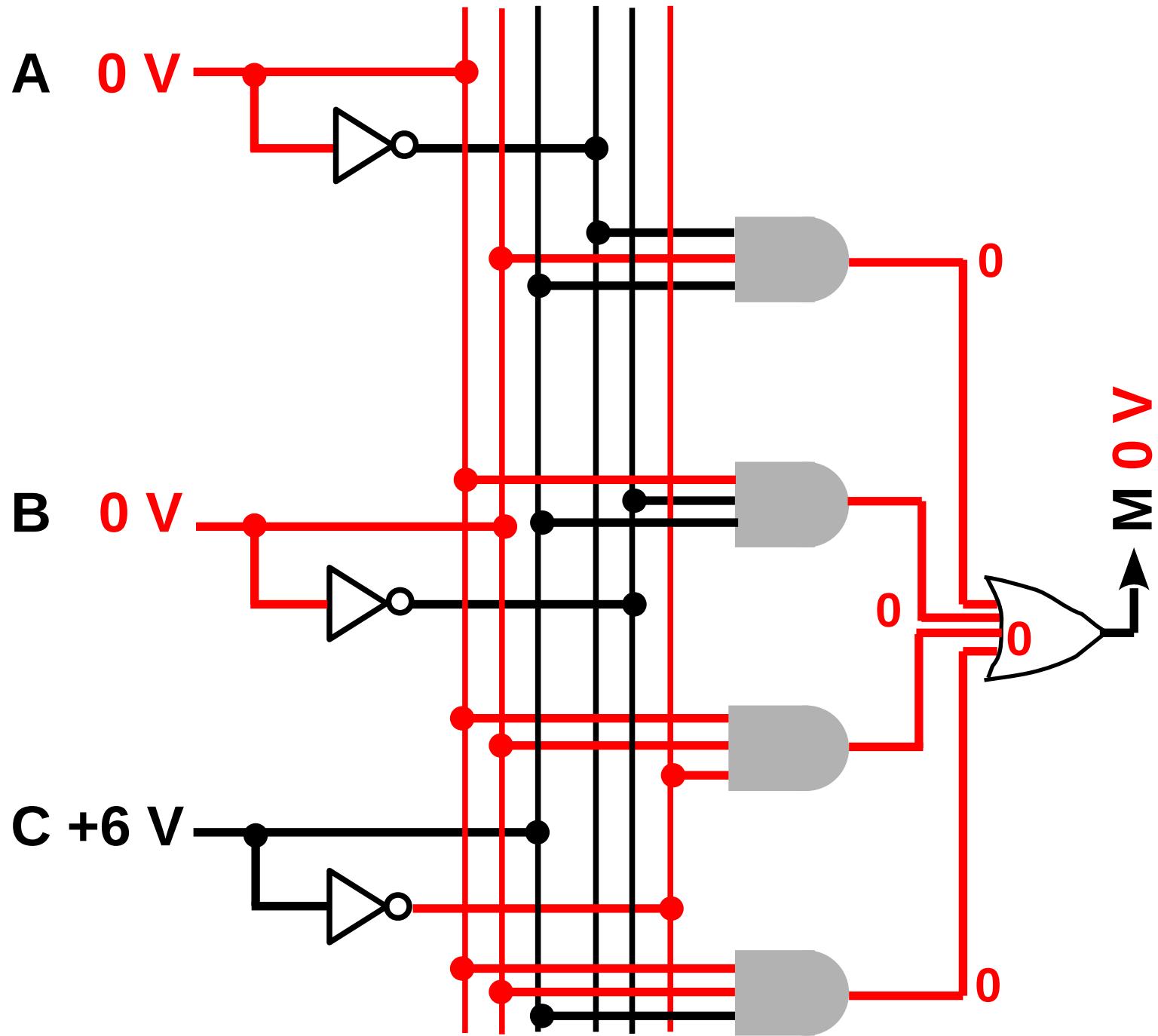


**NOT** = invertitore

**AND** = prodotto logico

**OR** = somma logica

A	B	C	M
1	1	1	1
1	1	0	1
1	0	1	1
1	0	0	0
0	1	1	1
0	1	0	0
0	0	1	0
0	0	0	0

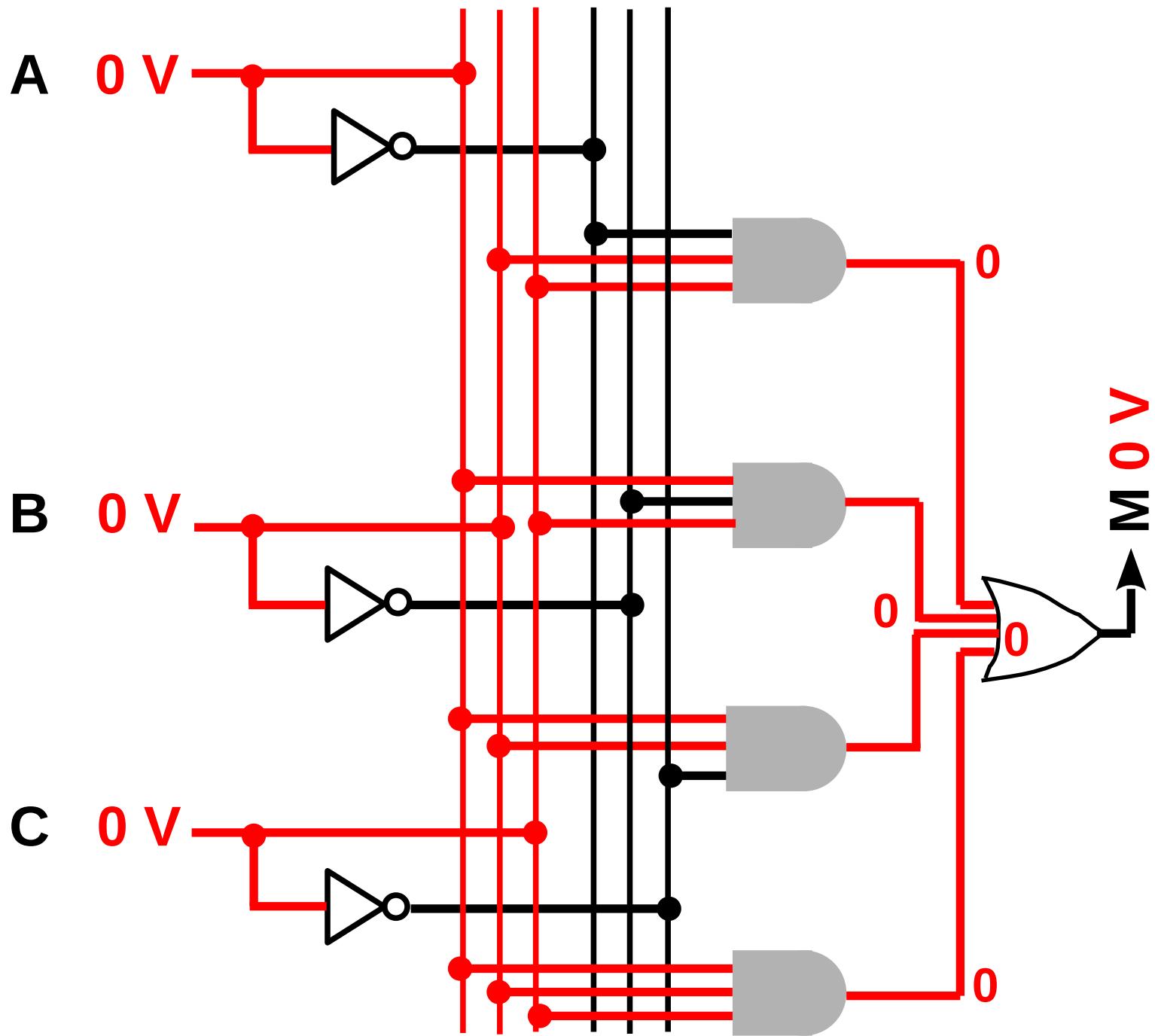


**NOT** = invertitore

**AND** = prodotto logico

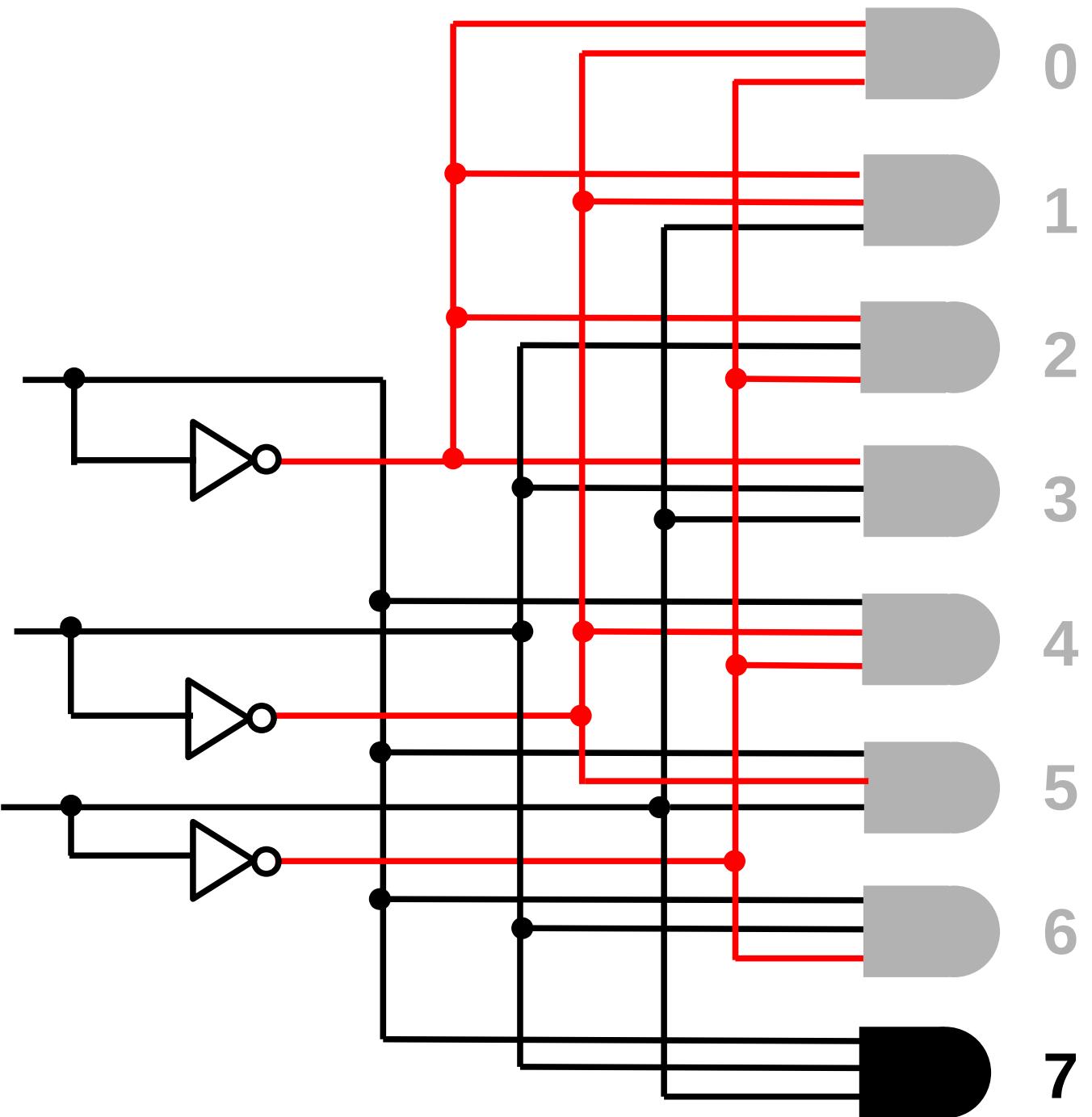
**OR** = somma logica

A	B	C	M
1	1	1	1
1	1	0	1
1	0	1	1
1	0	0	0
0	1	1	1
0	1	0	0
0	0	1	0
0	0	0	0



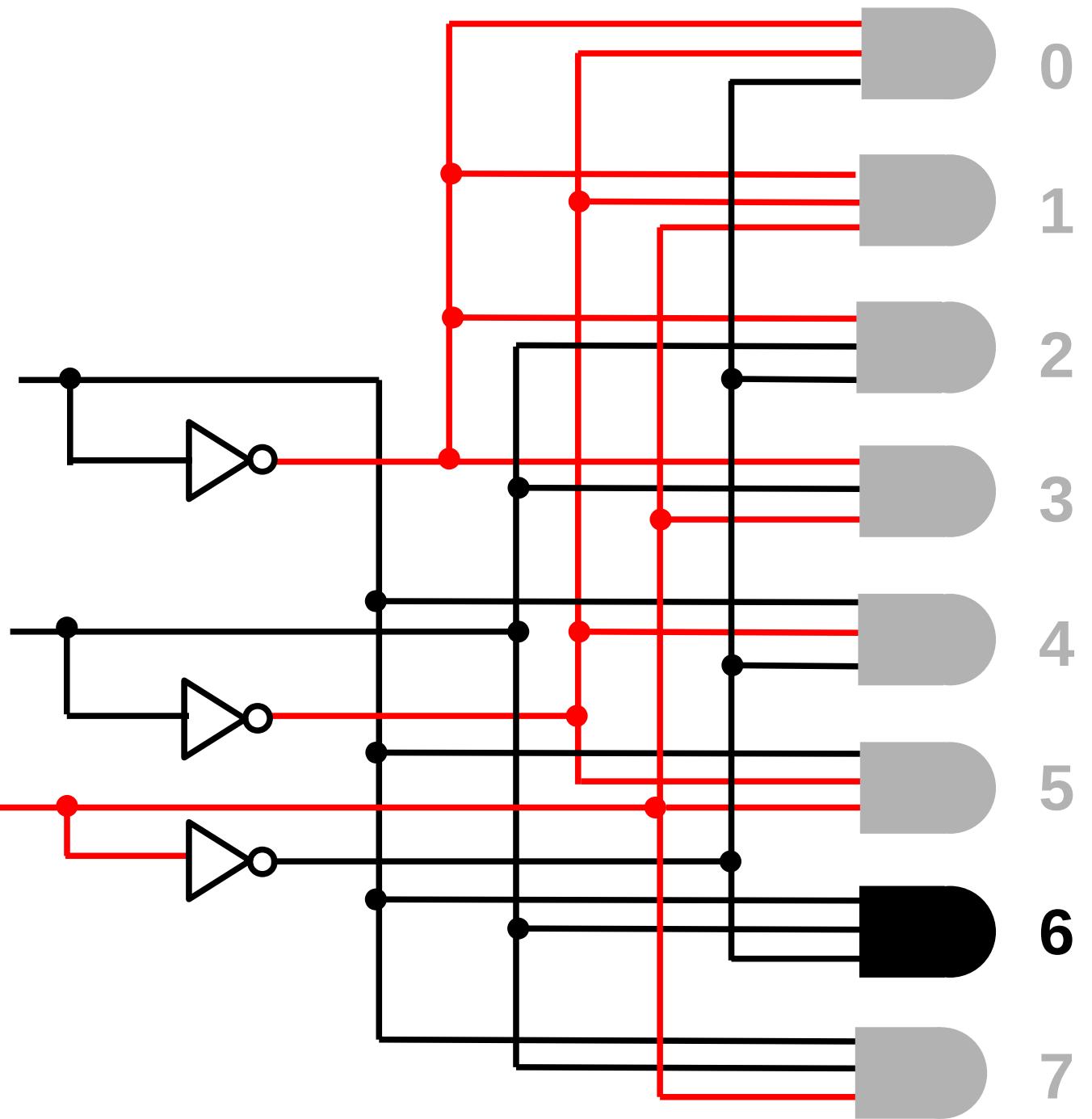
# Decodificatore

A	B	C	D	
1	1	1	7	
1	1	0	6	
1	0	1	5	
1	0	0	4	
0	1	1	3	
0	1	0	2	
0	0	1	1	
0	0	0	0	



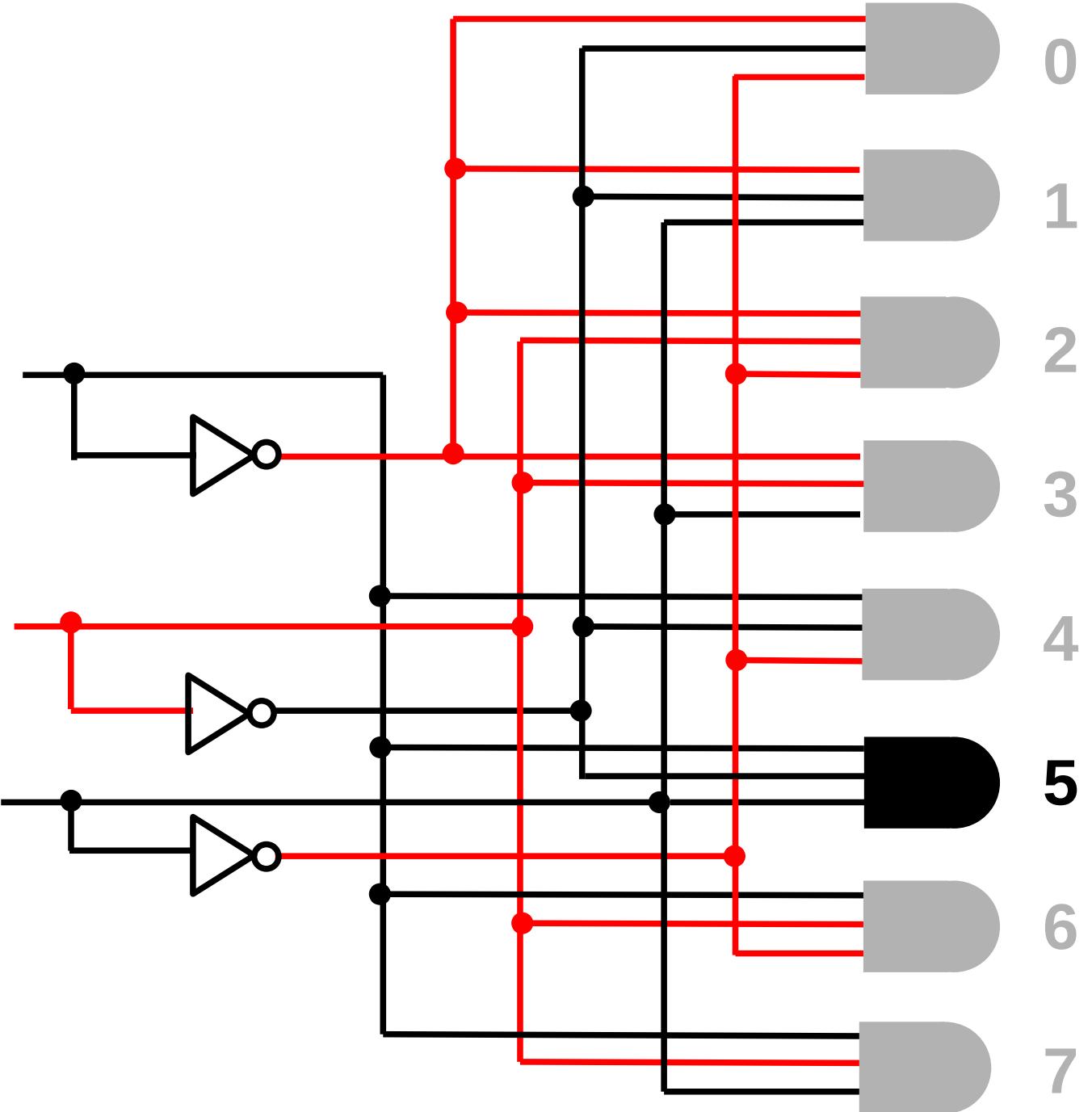
# Decodificatore

A	B	C	D
1	1	1	7
1	1	0	6
1	0	1	5
1	0	0	4
0	1	1	3
0	1	0	2
0	0	1	1
0	0	0	0



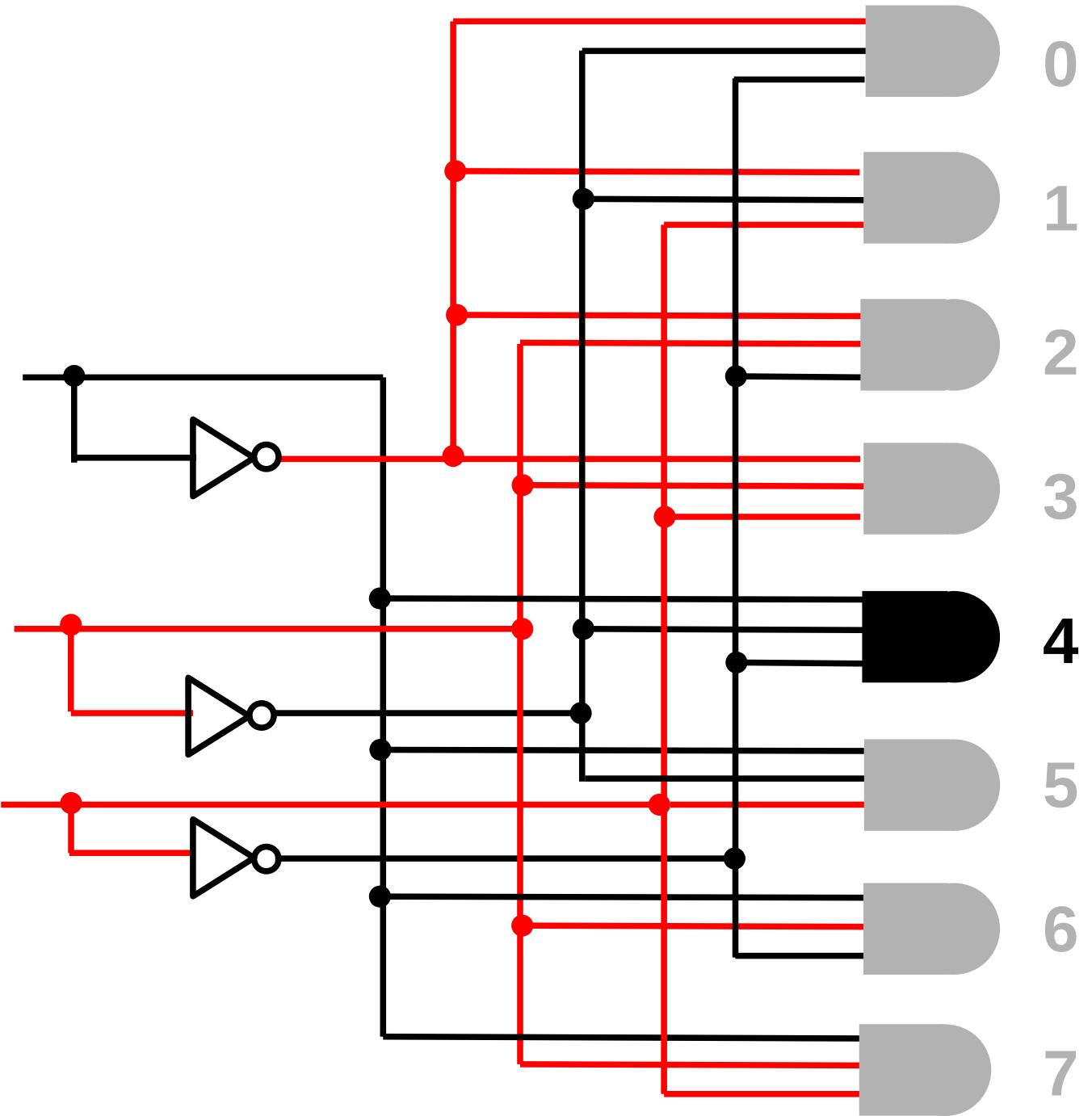
# Decodificatore

A	B	C	D	
1	1	1	7	
1	1	0	6	
1	0	1	5	
1	0	0	4	
0	1	1	3	
0	1	0	2	
0	0	1	1	
0	0	0	0	



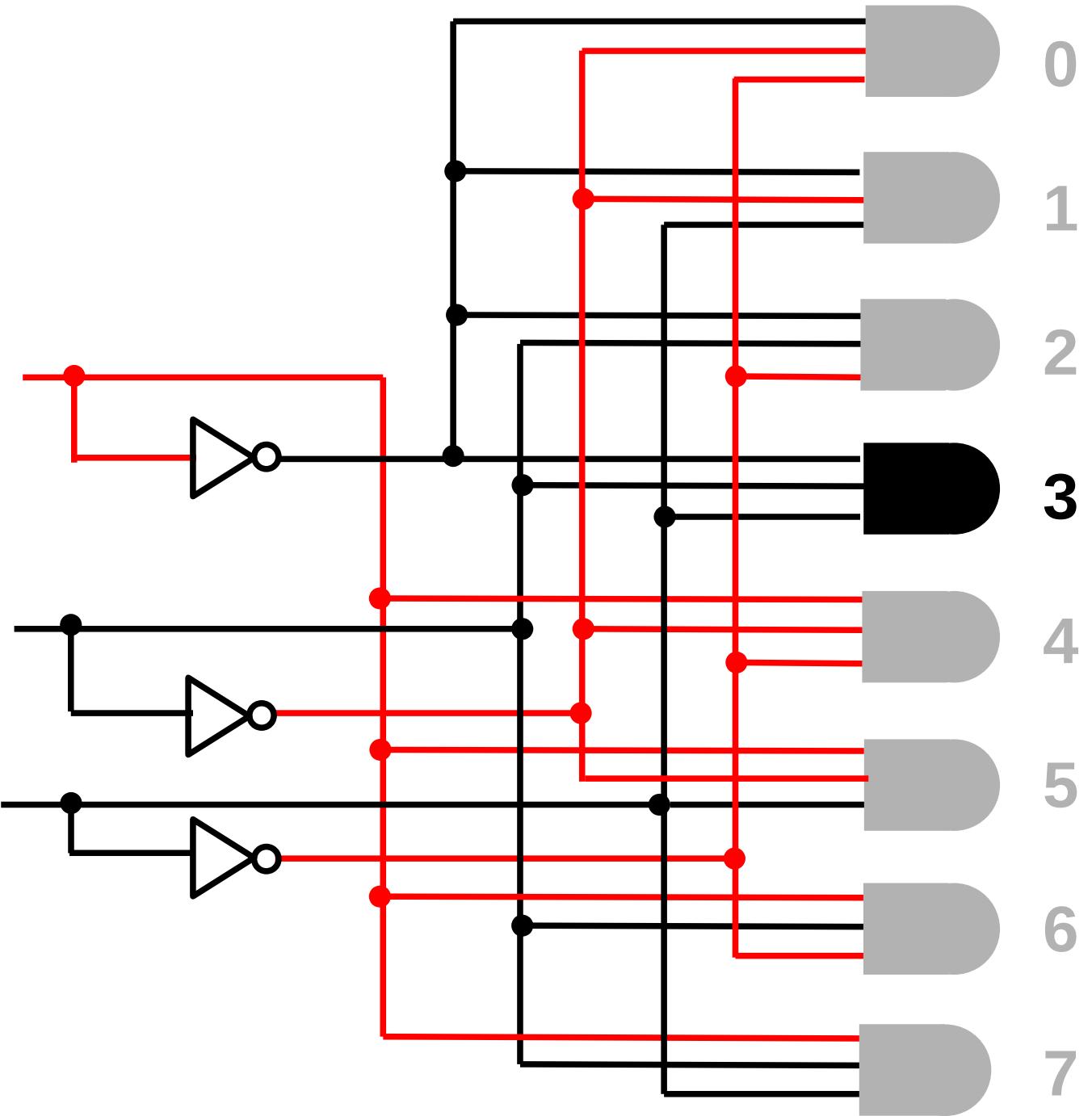
# Decodificatore

A	B	C	D	
1	1	1	7	
1	1	0	6	
1	0	1	5	
1	0	0	4	
0	1	1	3	
0	1	0	2	
0	0	1	1	
0	0	0	0	



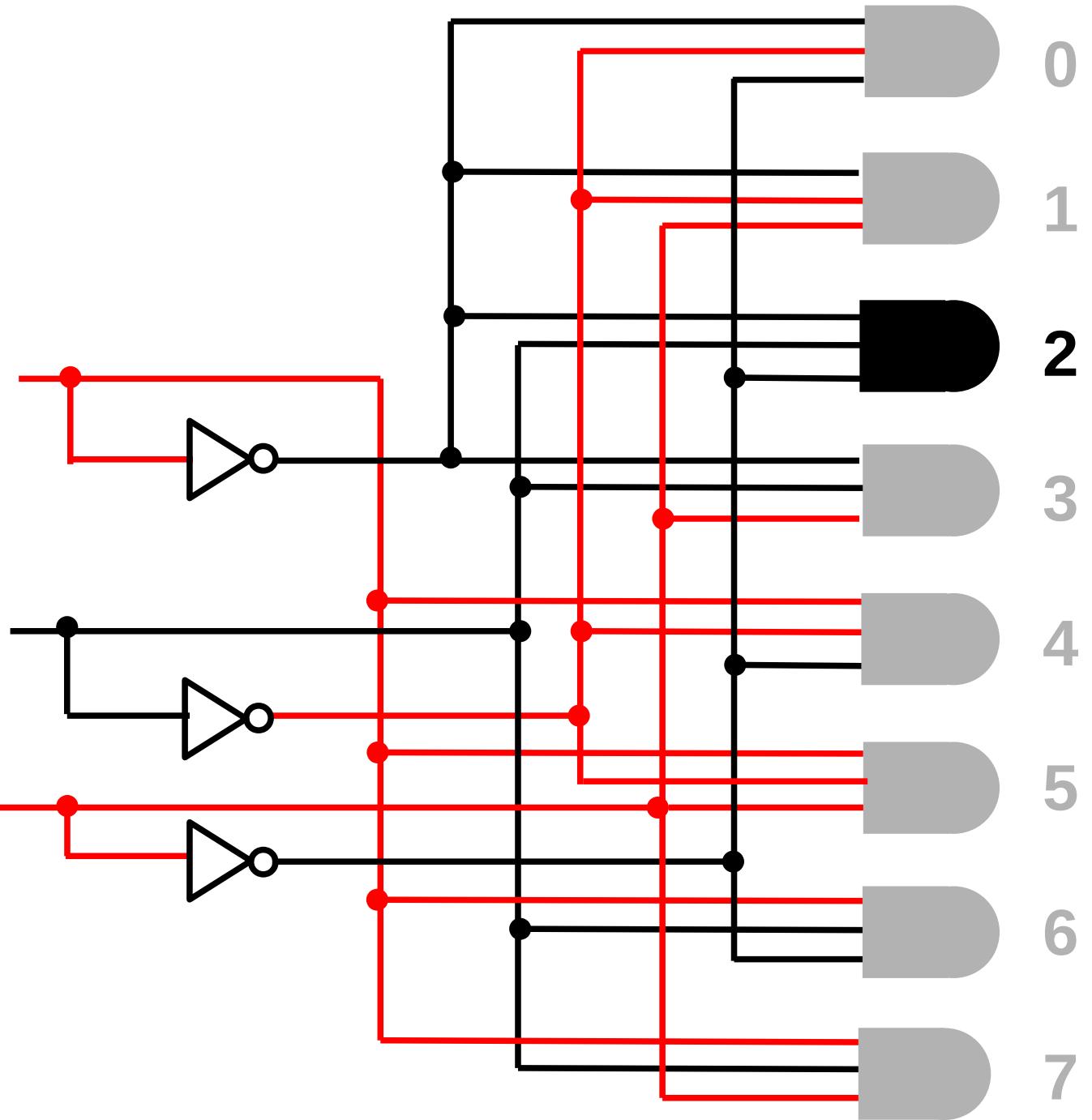
# Decodificatore

A	B	C	D
1	1	1	7
1	1	0	6
1	0	1	5
1	0	0	4
0	1	1	3
0	1	0	2
0	0	1	1
0	0	0	0



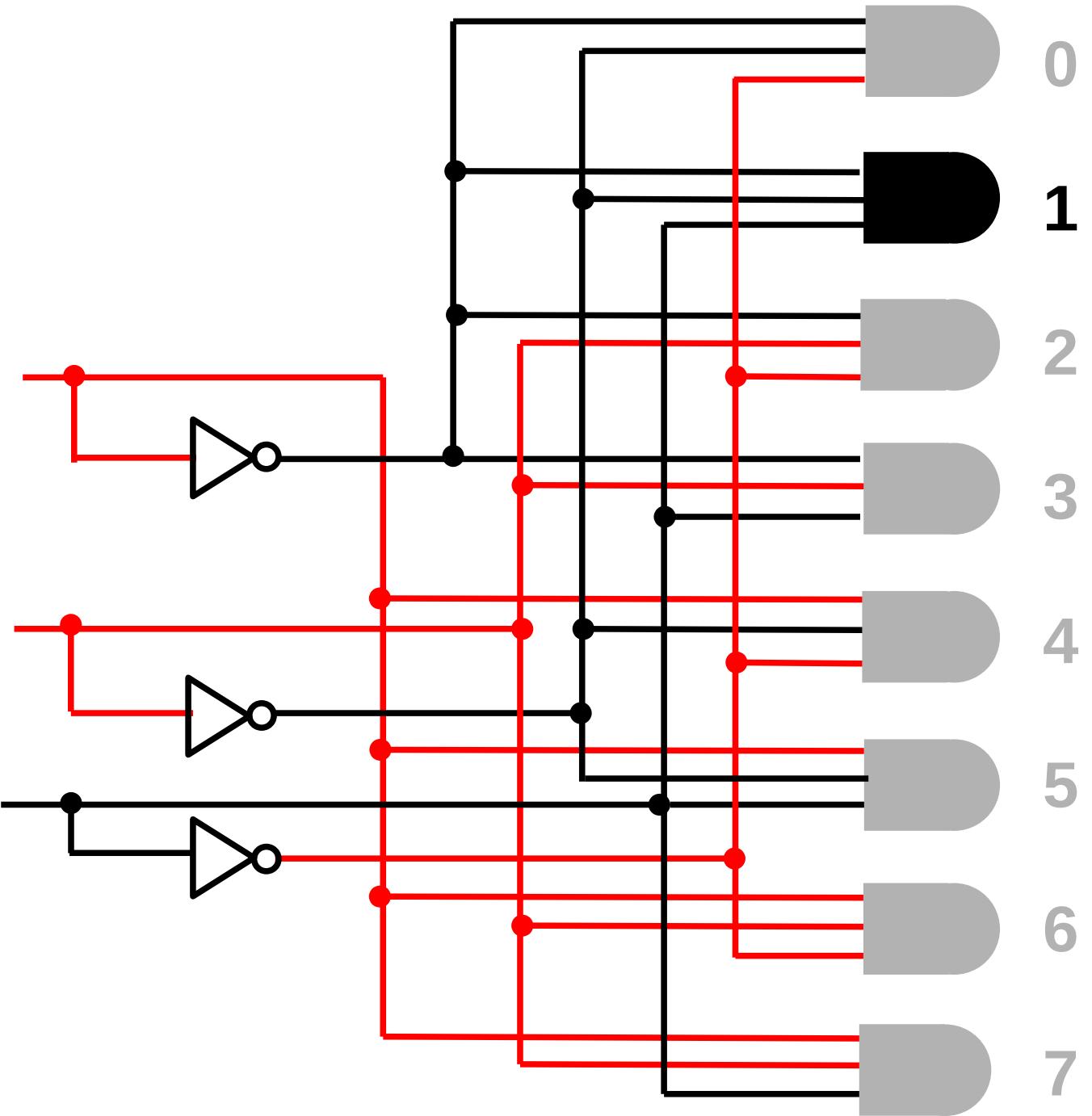
# Decodificatore

A	B	C	D
1	1	1	7
1	1	0	6
1	0	1	5
1	0	0	4
0	1	1	3
0	1	0	2
0	0	1	1
0	0	0	0



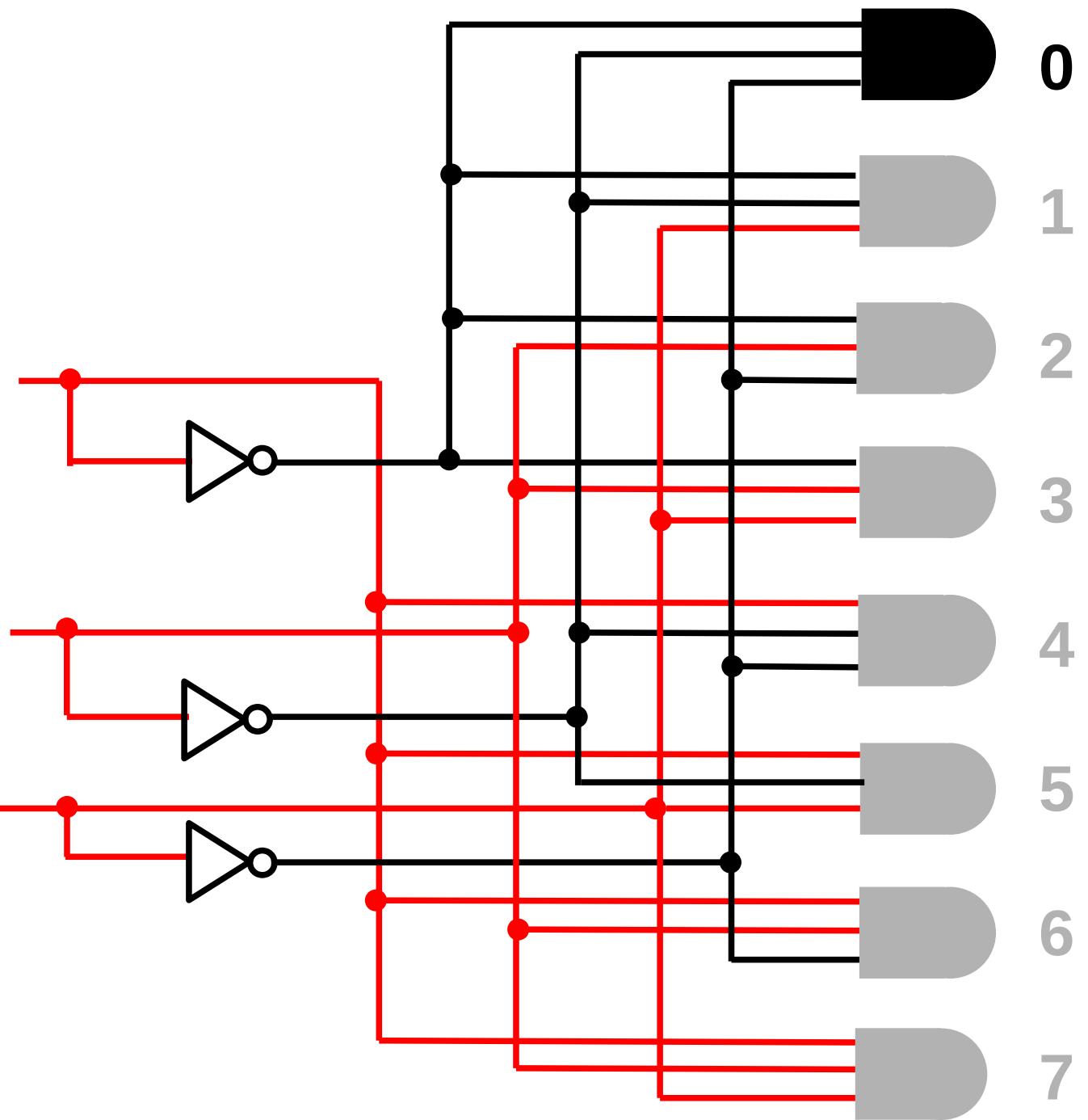
# Decodificatore

A	B	C	D	
1	1	1	7	
1	1	0	6	
1	0	1	5	
1	0	0	4	
0	1	1	3	
0	1	0	2	
0	0	1	1	
0	0	0	0	



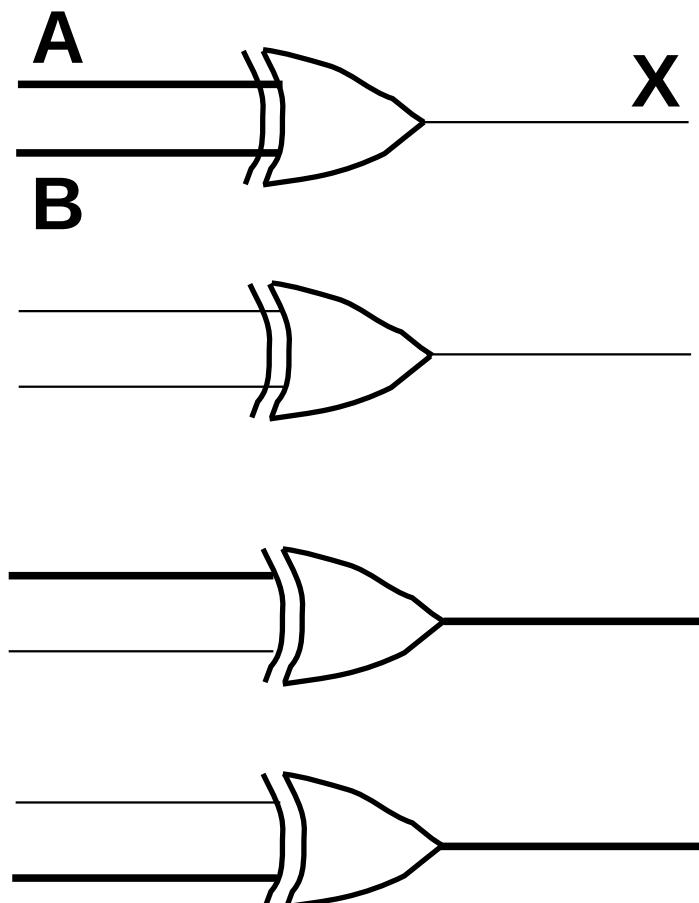
# Decodificatore

A	B	C	D	
1	1	1	1	7
1	1	0	1	6
1	0	1	1	5
1	0	0	1	4
0	1	1	1	3
0	1	0	1	2
0	0	1	1	1
0	0	0	1	0



# Porta OR ESCLUSIVO (XOR): sommatore

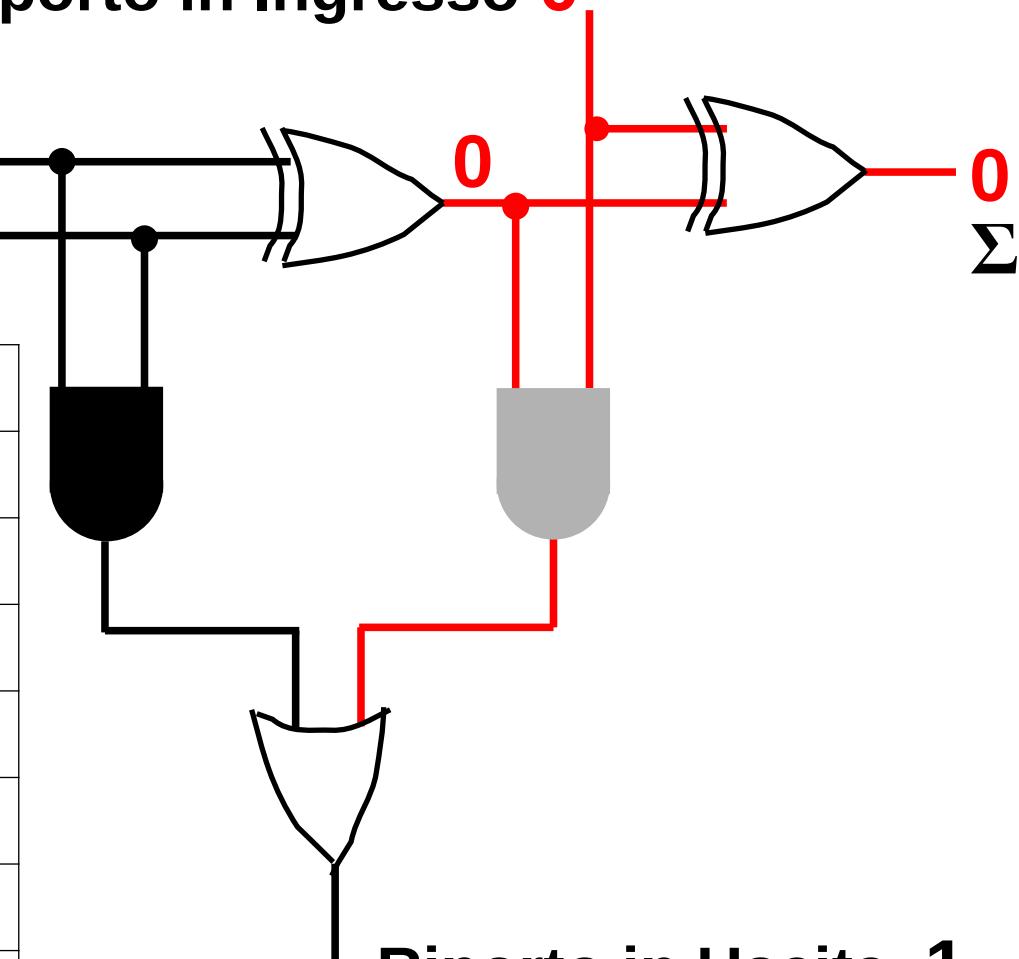
XOR



$$\begin{aligned}1+1 &= 10 \\&= 1 \cdot 2^1 + 0 \cdot 2^0 \\&= 2\end{aligned}$$

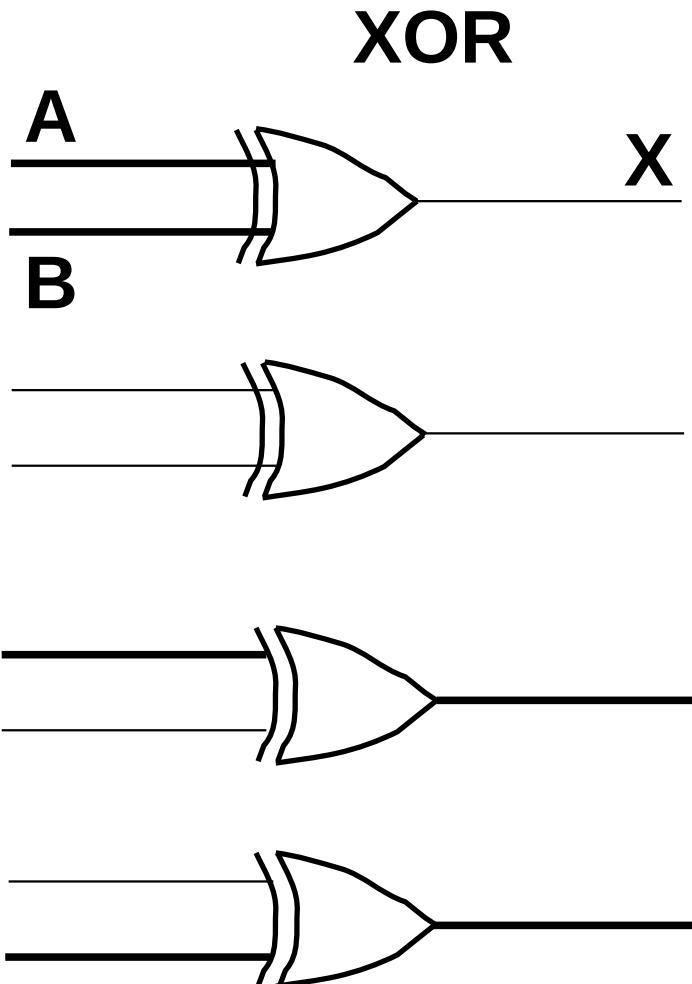
A	B	RI	$\Sigma$	RU
1	1	0	0	1
1	0	1	0	1
1	1	1	1	1
1	0	0	1	0
0	1	1	0	1
0	1	0	1	0
0	0	1	1	0
0	0	0	0	0

Riporto in Ingresso 0

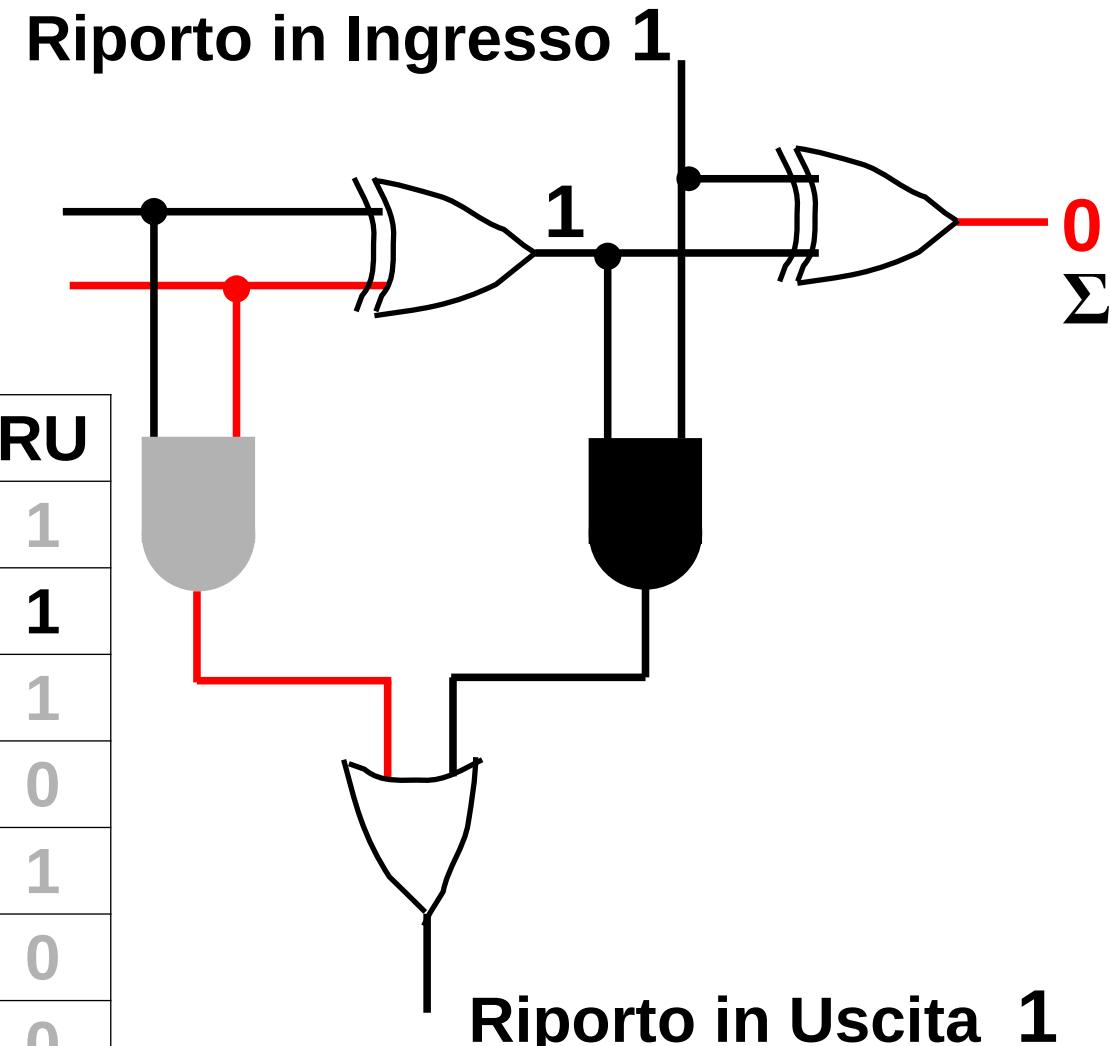


Riporto in Uscita 1

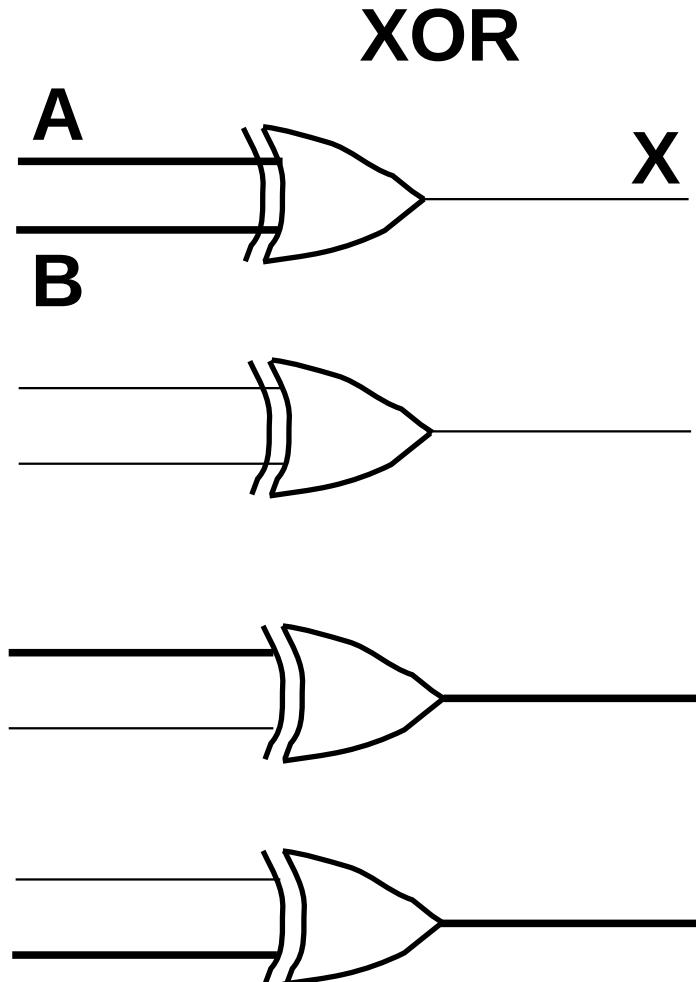
# Porta OR ESCLUSIVO (XOR): sommatore



A	B	RI	$\Sigma$	RU
1	1	0	0	1
1	0	1	0	1
1	1	1	1	1
1	0	0	1	0
0	1	1	0	1
0	1	0	1	0
0	0	1	1	0
0	0	0	0	0

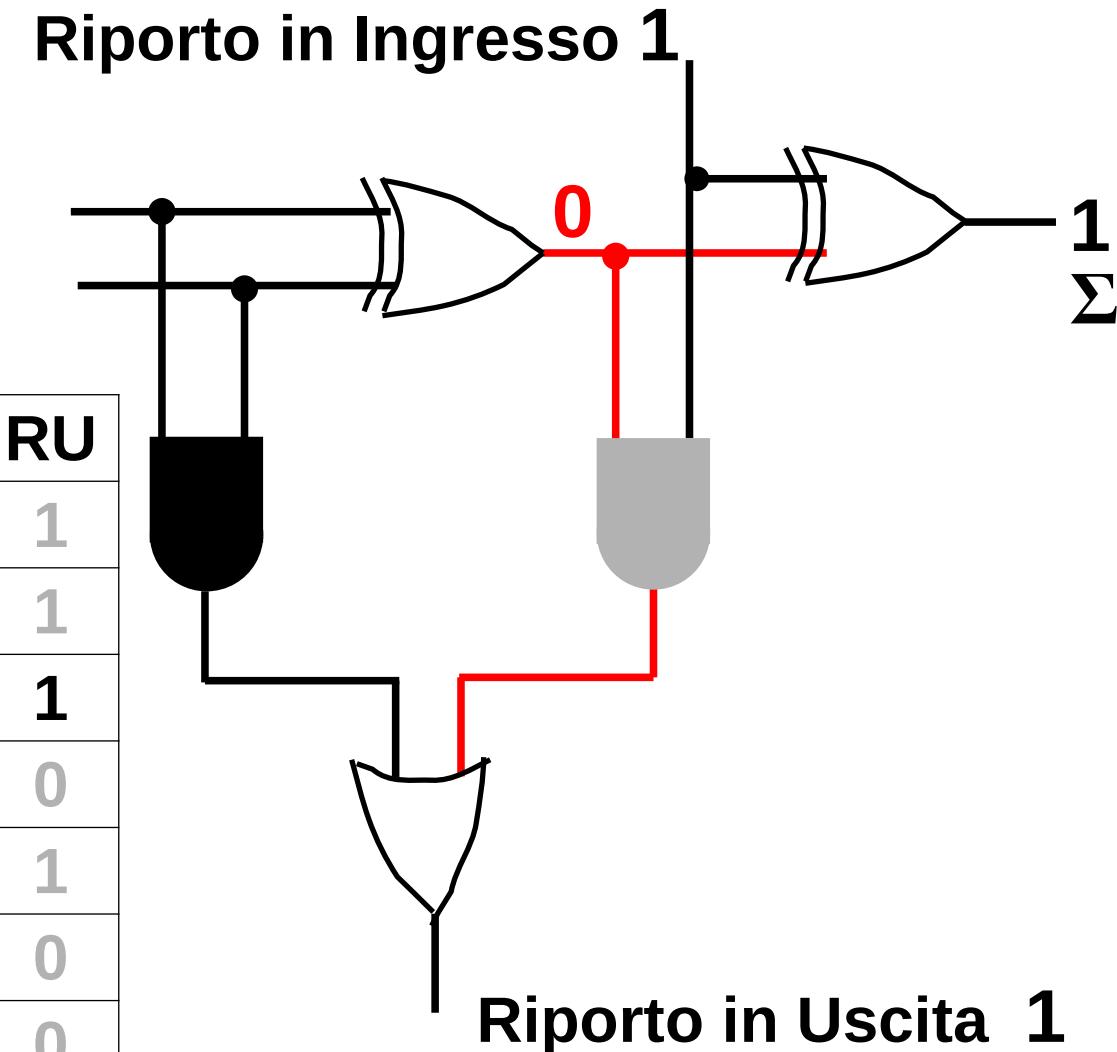


# Porta OR ESCLUSIVO (XOR): sommatore



$$\begin{aligned}11 + & \quad 1*2^1 + 1*2^0 = 3 \\11 = & \quad 1*2^1 + 1*2^0 = 3 \\110 = & 1*2^2 + 1*2^1 + 0*2^0 = 6\end{aligned}$$

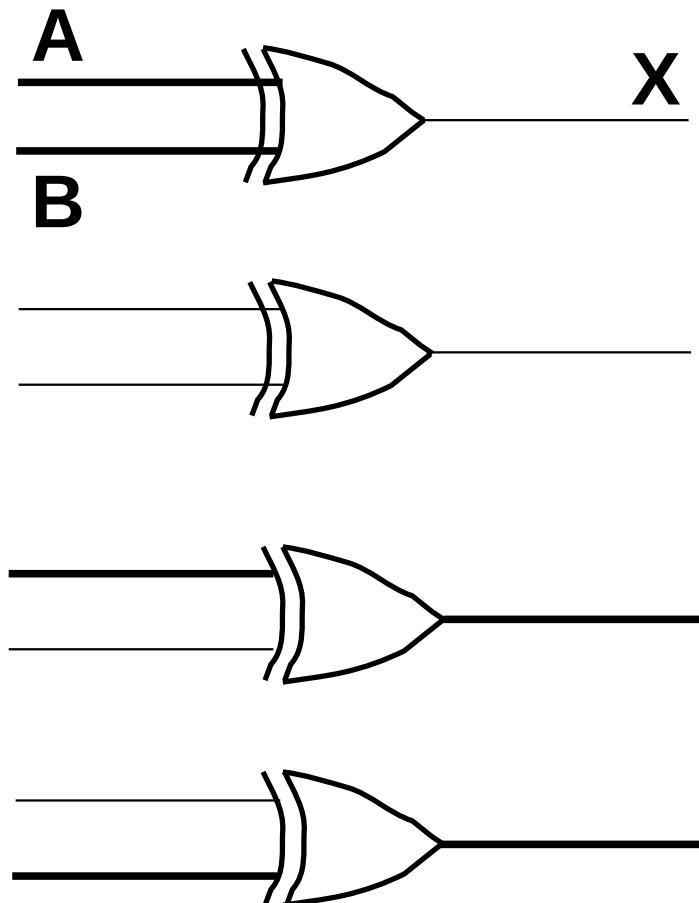
<b>A</b>	<b>B</b>	<b>RI</b>	$\Sigma$	<b>RU</b>
1	1	0	0	1
1	0	1	0	1
1	1	1	1	1
1	0	0	1	0
0	1	1	0	1
0	1	0	1	0
0	0	1	1	0
0	0	0	0	0



**Riporto in Uscita 1**

# Porta OR ESCLUSIVO (XOR): sommatore

XOR

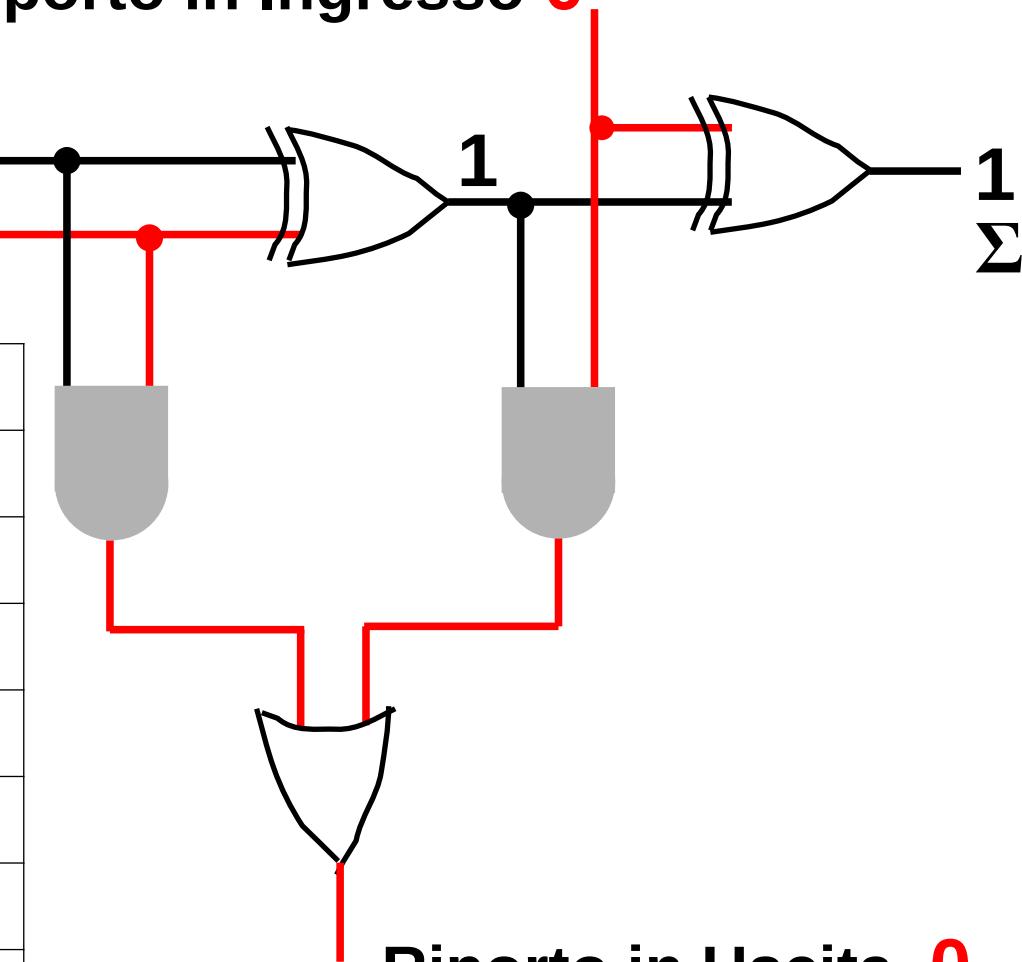


1 +  
0 =  
01 =

$$\begin{aligned} 1 &+ 1 * 2^0 = 1 \\ 0 &= 0 * 2^0 = 0 \\ 01 &= 0 * 2^1 + 1 * 2^0 = 1 \end{aligned}$$

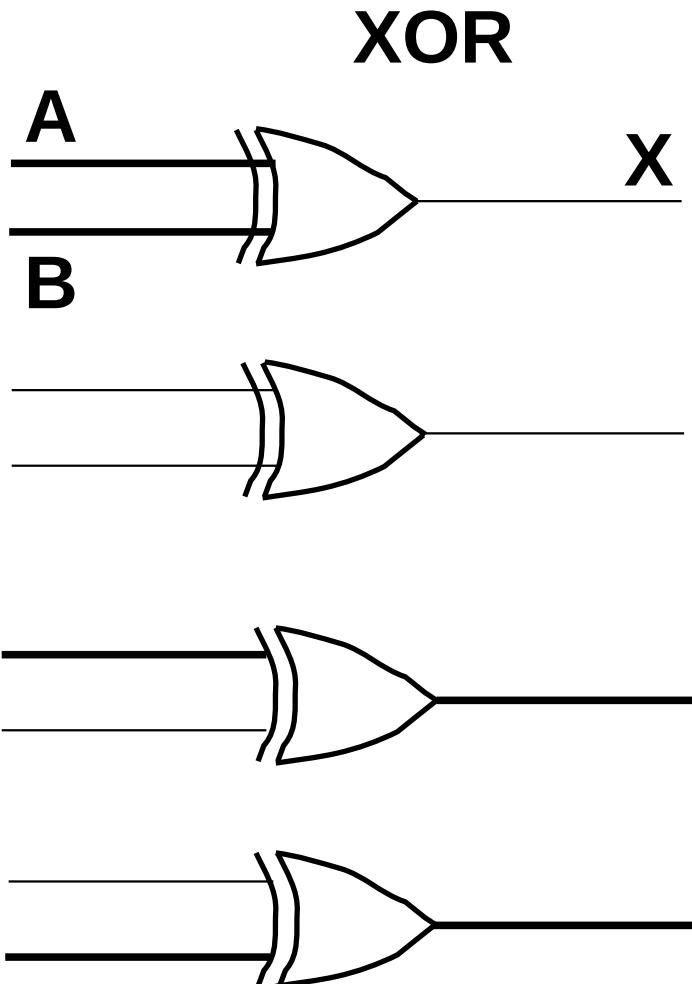
A	B	RI	$\Sigma$	RU
1	1	0	0	1
1	0	1	0	1
1	1	1	1	1
1	0	0	1	0
0	1	1	0	1
0	1	0	1	0
0	0	1	1	0
0	0	0	0	0

Riporto in Ingresso 0

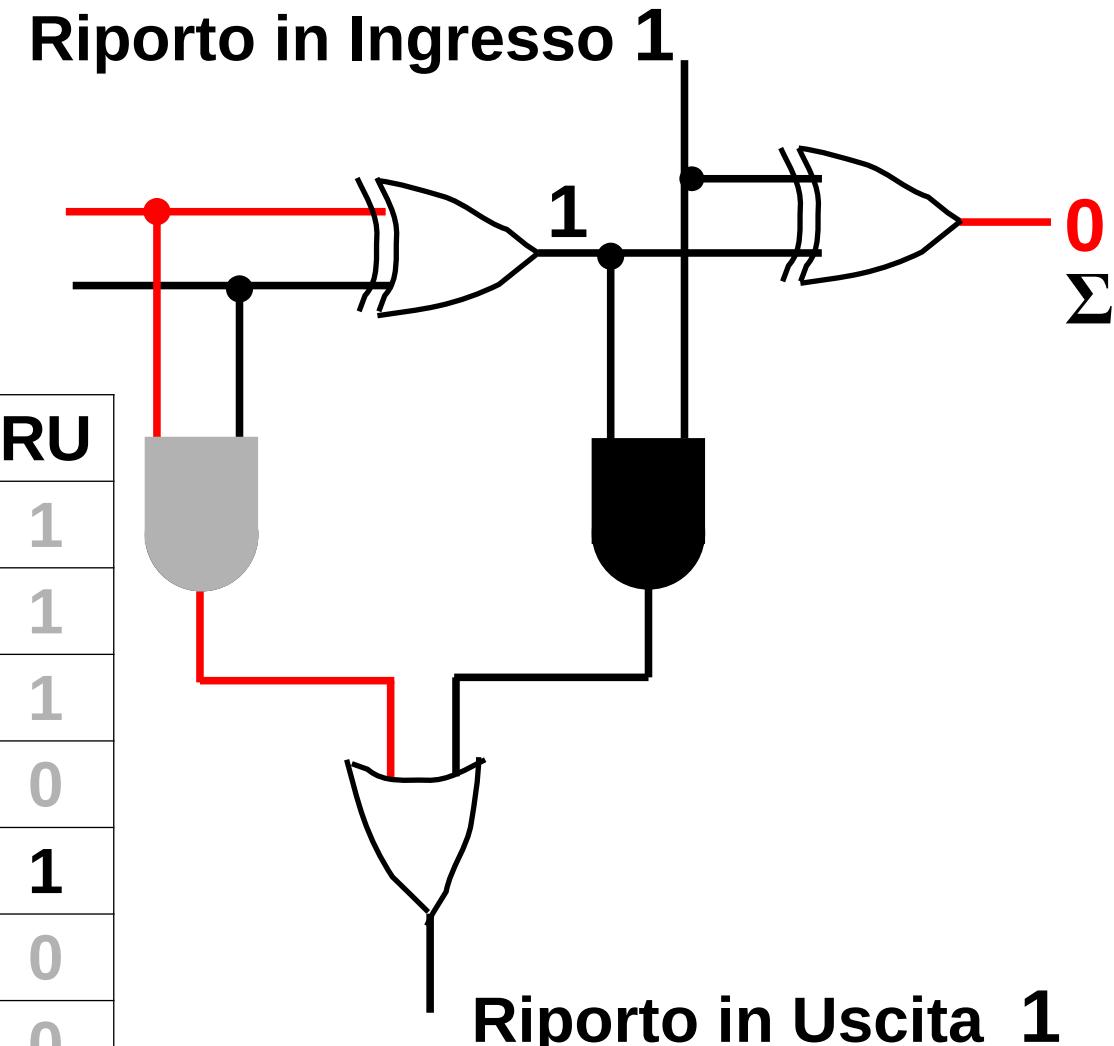


Riporto in Uscita 0

# Porta OR ESCLUSIVO (XOR): sommatore

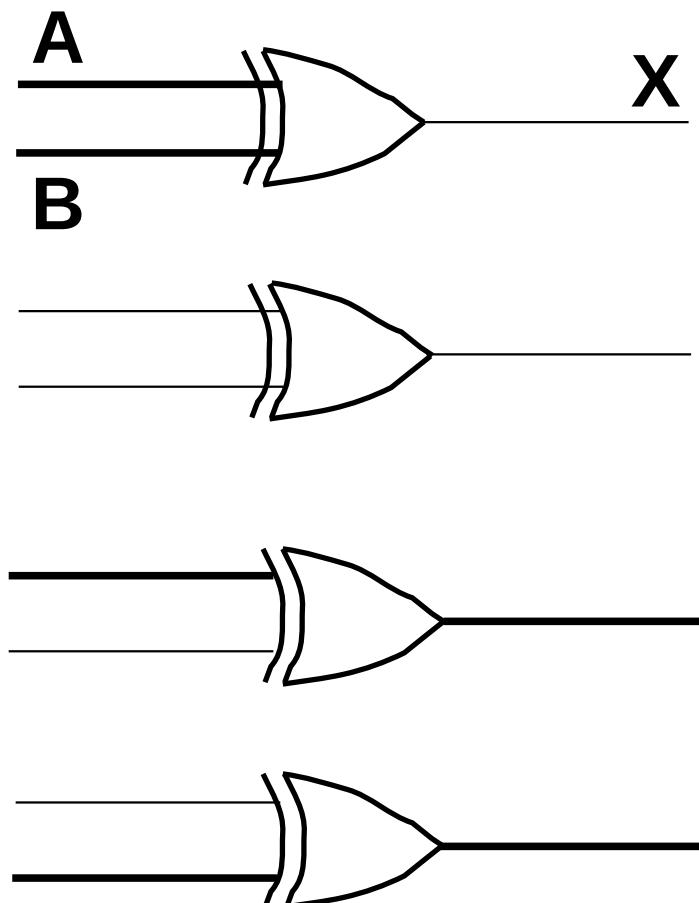


A	B	RI	$\Sigma$	RU
1	1	0	0	1
1	0	1	0	1
1	1	1	1	1
1	0	0	1	0
0	1	1	0	1
0	1	0	1	0
0	0	1	1	0
0	0	0	0	0



# Porta OR ESCLUSIVO (XOR): sommatore

XOR



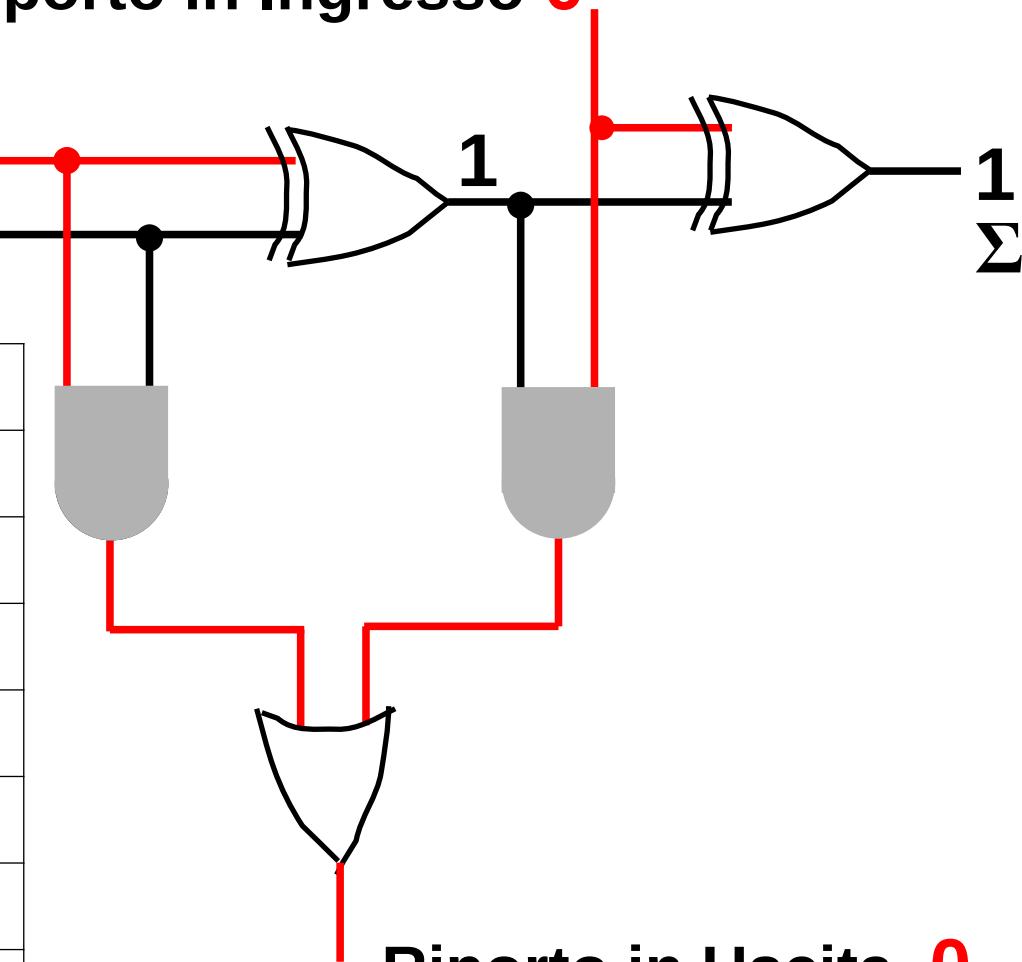
$$1 + \quad 1 * 2^0 = 1$$

$$0 = \quad 0 * 2^0 = 0$$

$$01 = \quad 0 * 2^1 + 1 * 2^0 = 1$$

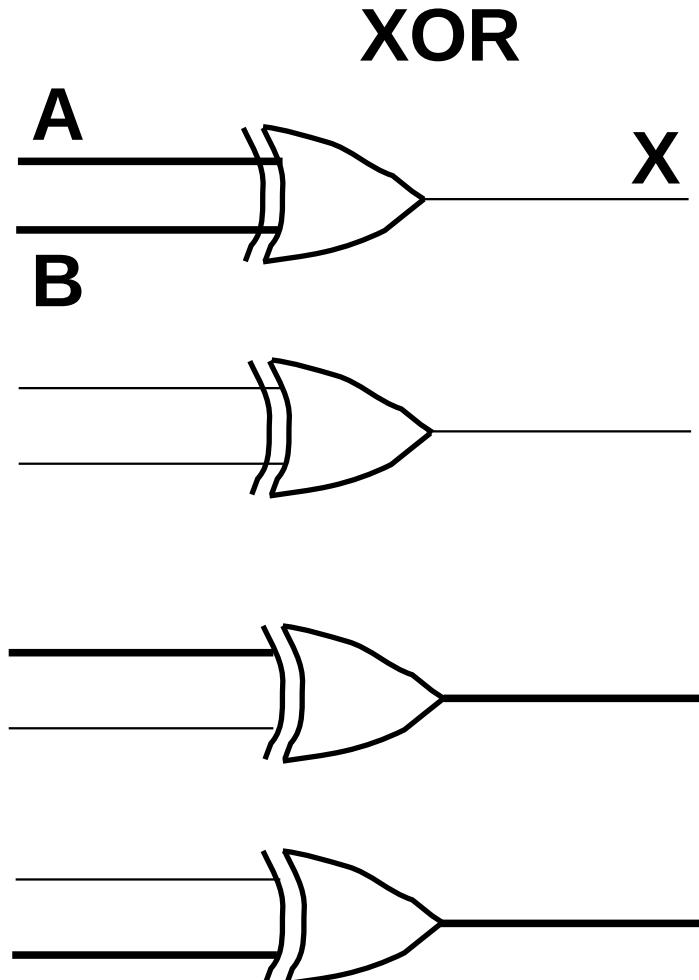
A	B	RI	$\Sigma$	RU
1	1	0	0	1
1	0	1	0	1
1	1	1	1	1
1	0	0	1	0
0	1	1	0	1
0	1	0	1	0
0	0	1	1	0
0	0	0	0	0

Riporto in Ingresso 0

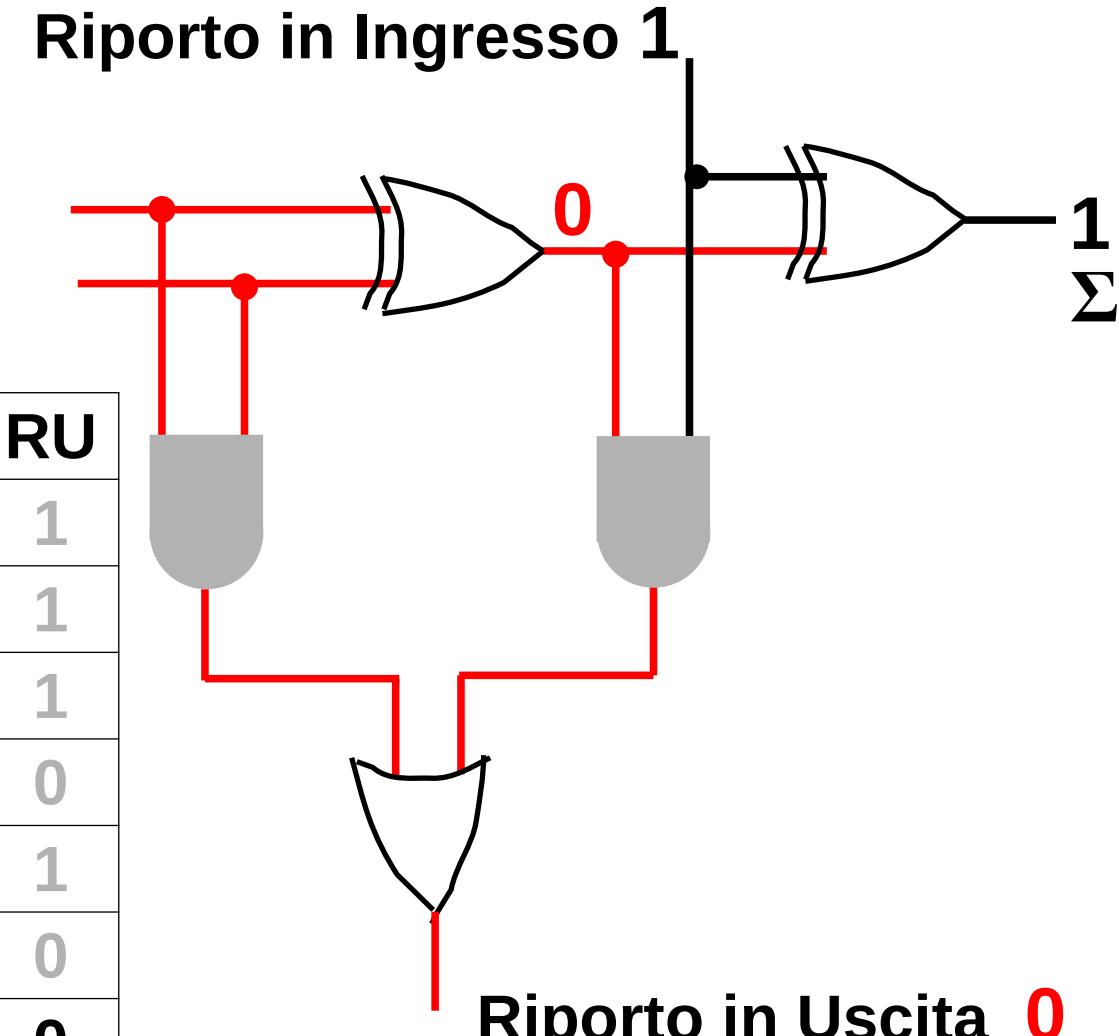


Riporto in Uscita 0

# Porta OR ESCLUSIVO (XOR): sommatore

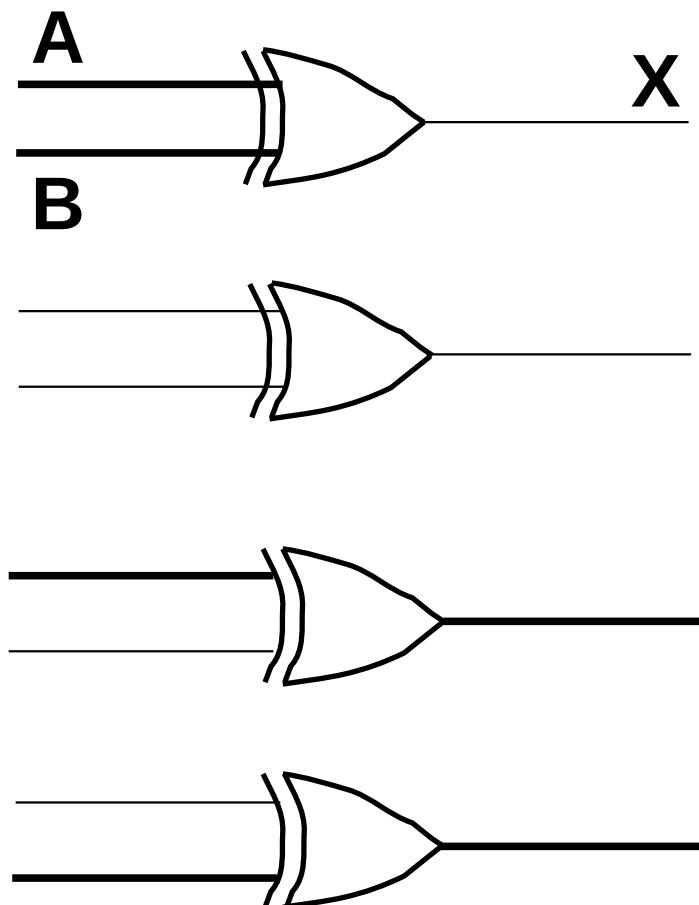


A	B	RI	$\Sigma$	RU
1	1	0	0	1
1	0	1	0	1
1	1	1	1	1
1	0	0	1	0
0	1	1	0	1
0	1	0	1	0
0	0	1	1	0
0	0	0	0	0



# Porta OR ESCLUSIVO (XOR): sommatore

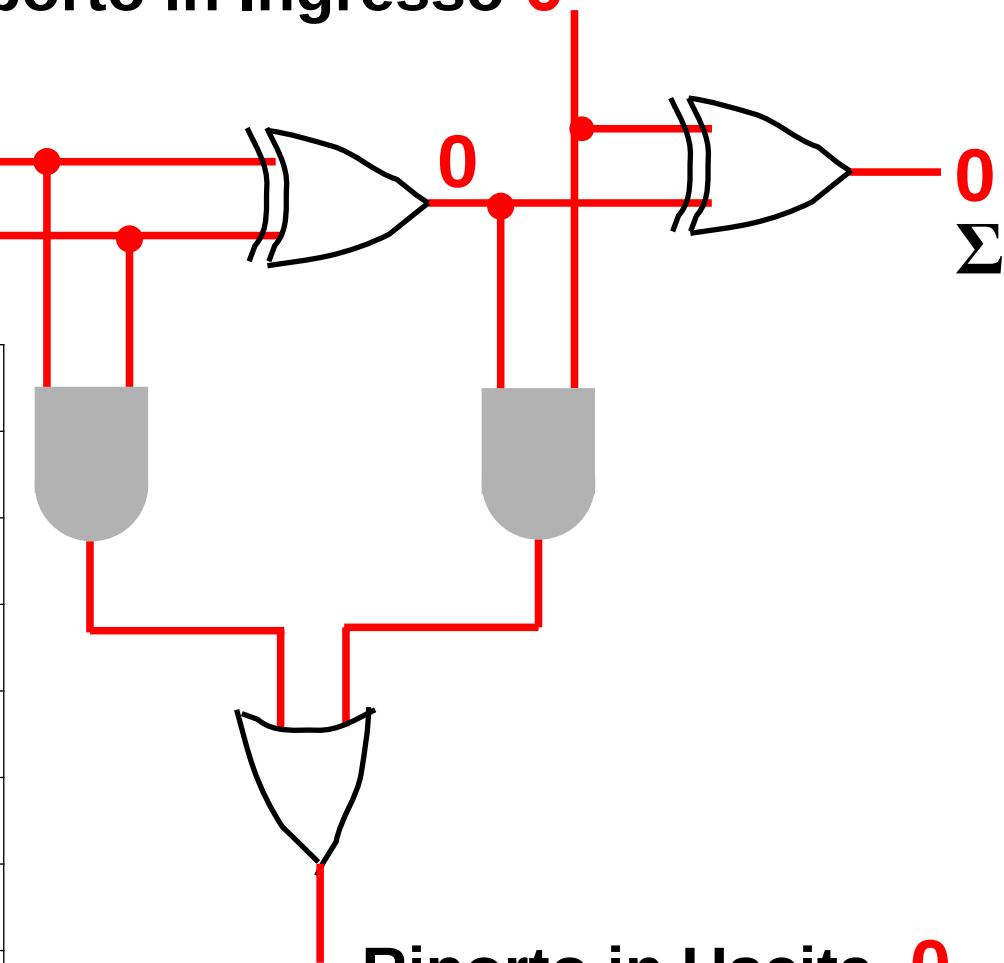
XOR



$$\begin{array}{l} 0+ \\ 0= \\ 0= \end{array} \quad 0^*2^0 = 0$$

A	B	RI	$\Sigma$	RU
1	1	0	0	1
1	0	1	0	1
1	1	1	1	1
1	0	0	1	0
0	1	1	0	1
0	1	0	1	0
0	0	1	1	0
0	0	0	0	0

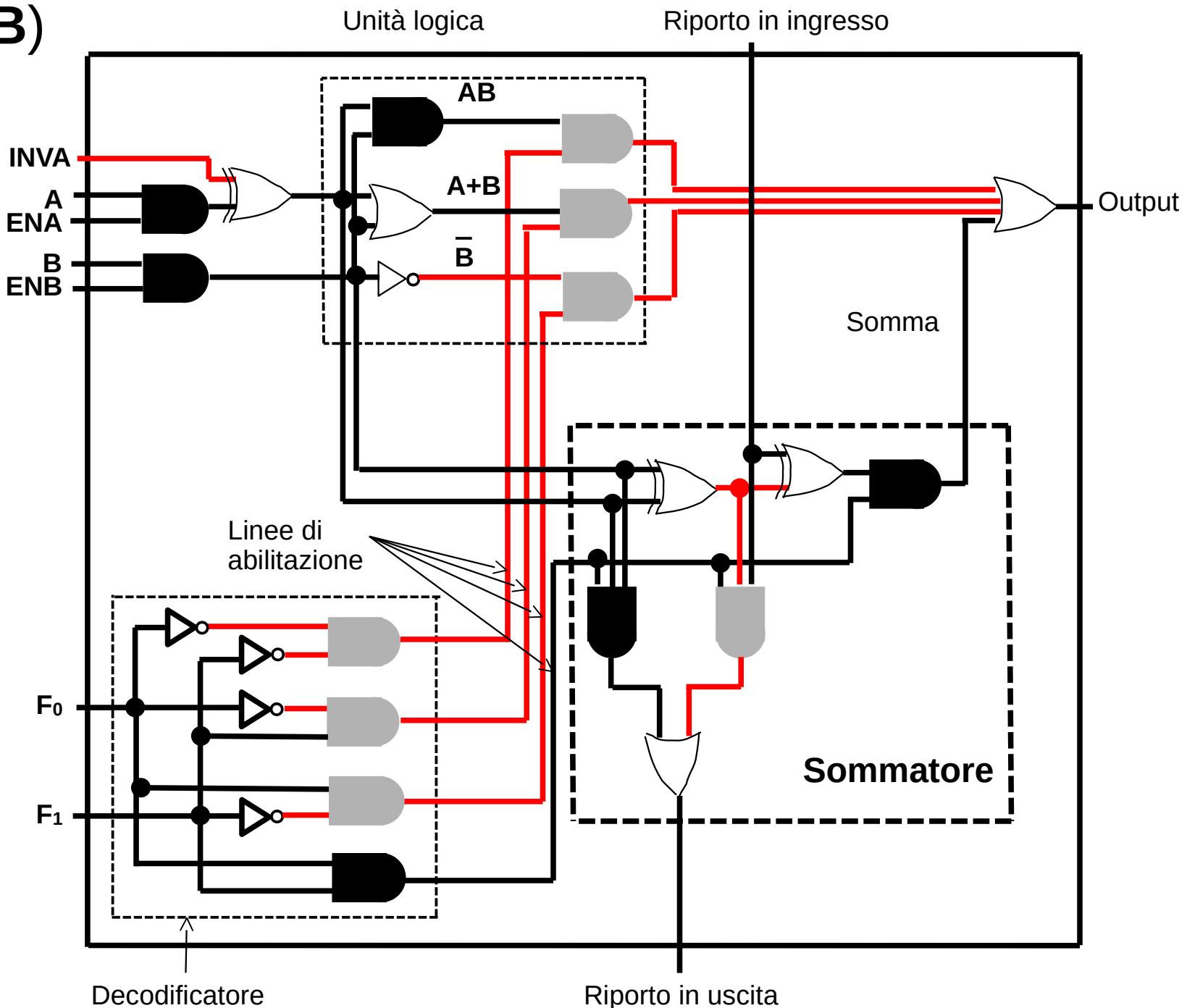
Riporto in Ingresso 0



# ALU a 1 bit: somma (A + B)

INVA	A	ENA	B	ENB	F <sub>0</sub>	F <sub>1</sub>	RI	RU
0	1	1	1	1	1	1	1	1

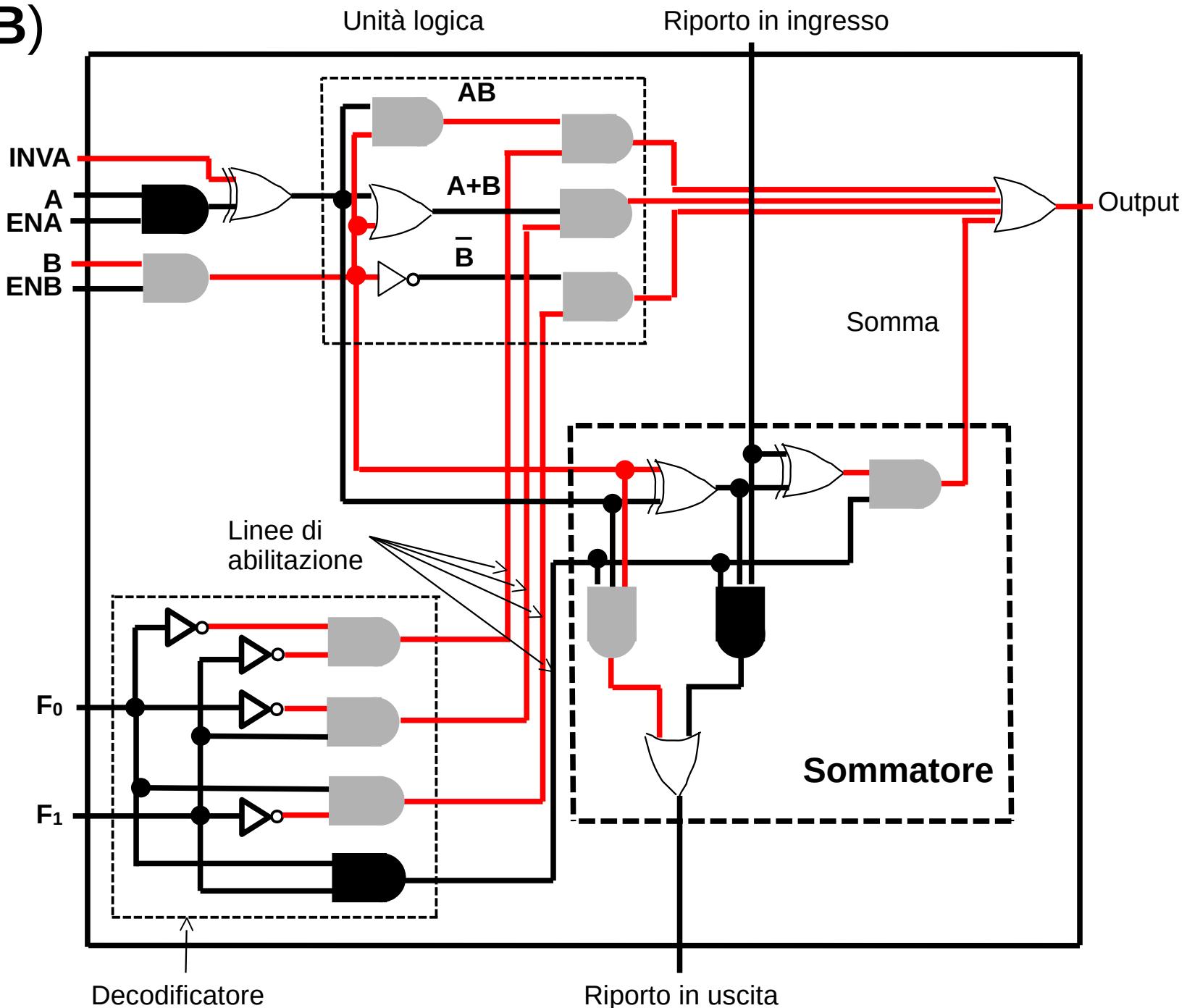
$$\begin{aligned}
 11 + & \quad 1*2^1 + 1*2^0 = 3 \\
 11 = & \quad 1*2^1 + 1*2^0 = 3 \\
 110 = & 1*2^2 + 1*2^1 + 0*2^0 = 6
 \end{aligned}$$



# ALU a 1 bit: somma (A + B)

INVA	A	ENA	B	ENB	F <sub>0</sub>	F <sub>1</sub>	RI	RU
0	1	1	0	1	1	1	1	1

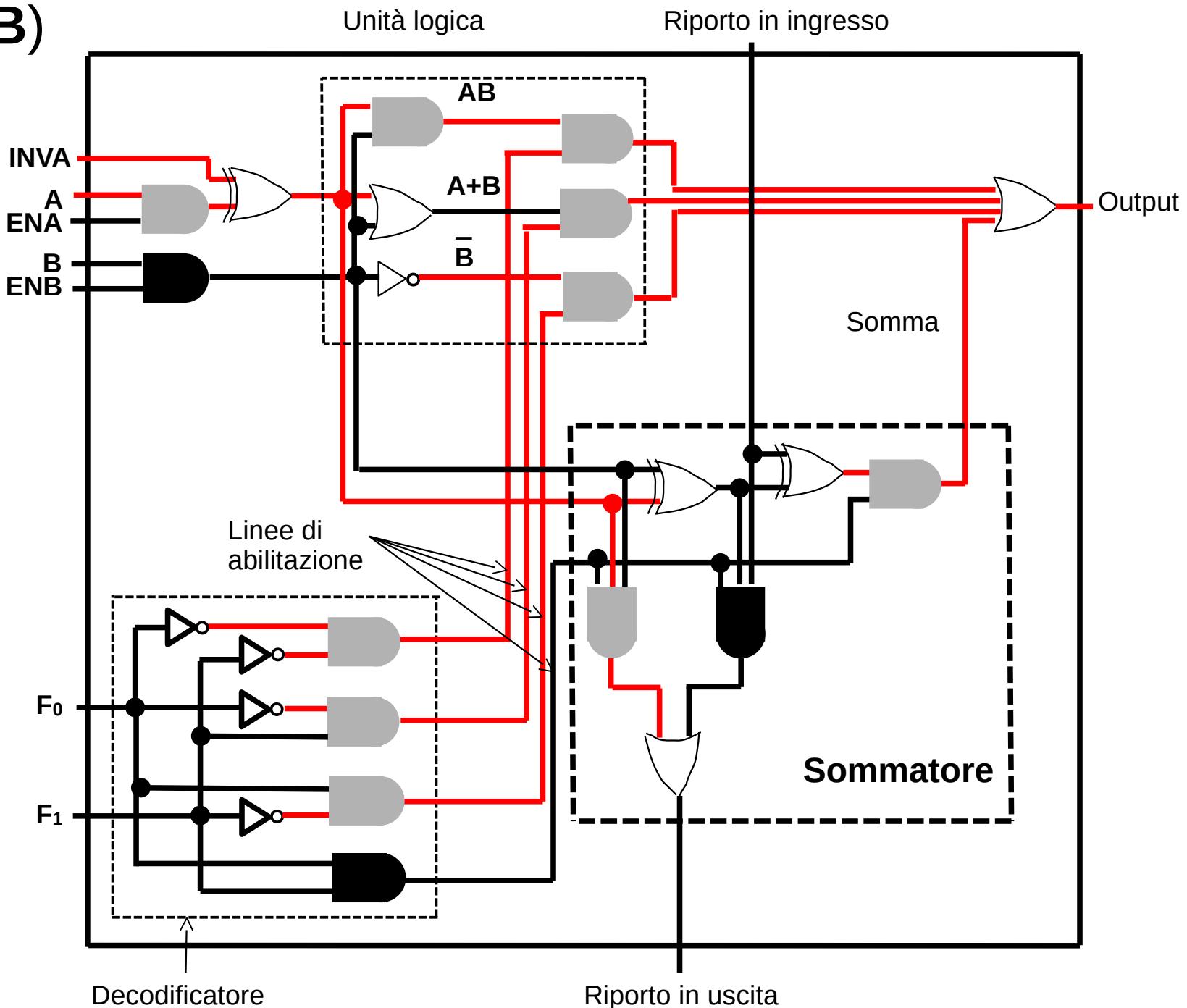
$$\begin{aligned}
 11 + & \quad 1*2^1 + 1*2^0 = 3 \\
 01 = & \quad 0*2^1 + 1*2^0 = 1 \\
 100 = & \quad 1*2^2 + 0*2^1 + 0*2^0 = 4
 \end{aligned}$$



# ALU a 1 bit: somma (A + B)

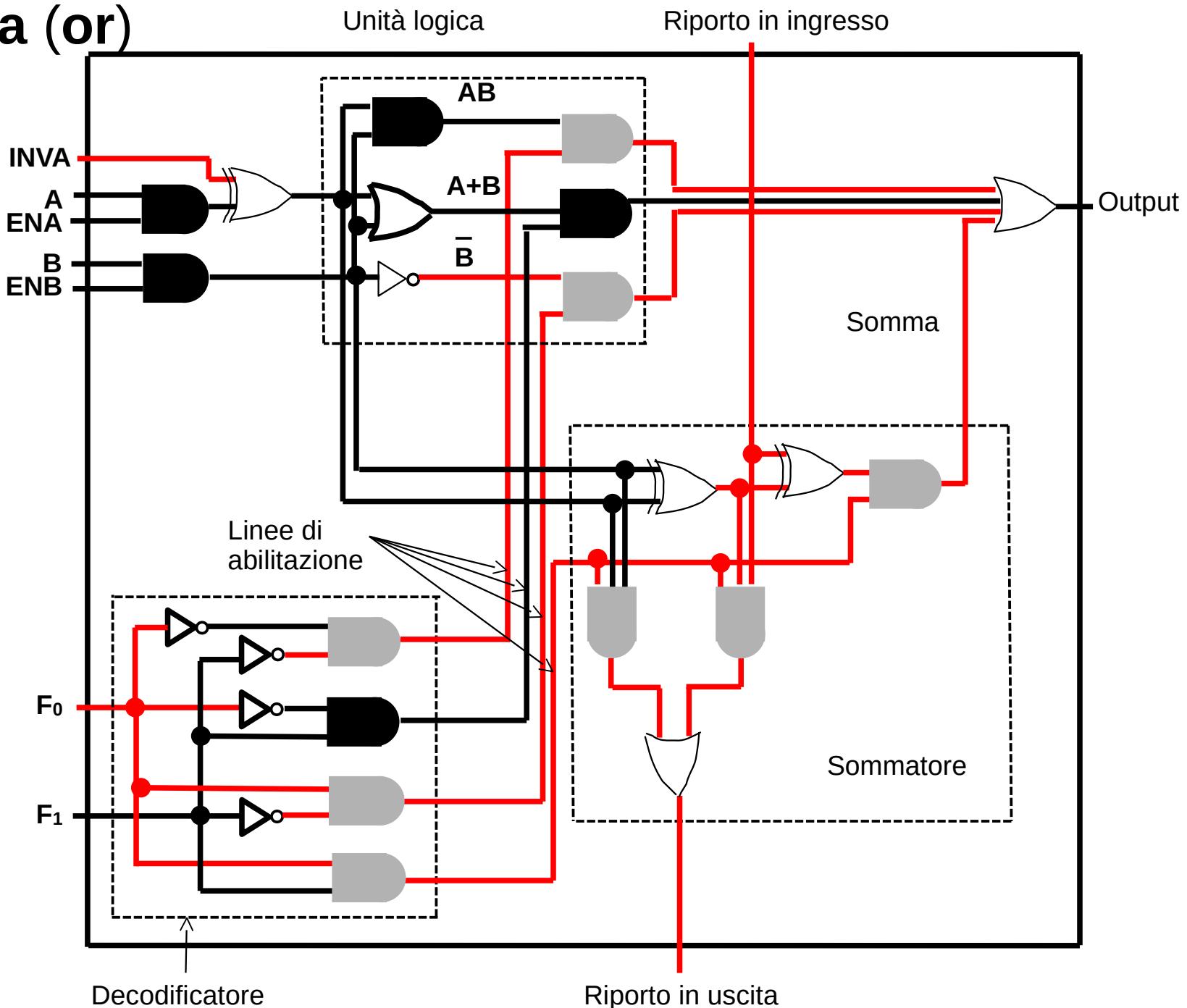
INVA	A	ENA	B	ENB	F <sub>0</sub>	F <sub>1</sub>	RI	RU
0	0	1	1	1	1	1	1	1

$$\begin{aligned}
 01 + & \quad 1*2^1 + 1*2^0 = 3 \\
 11 = & \quad 0*2^1 + 1*2^0 = 1 \\
 100 = & 1*2^2 + 0*2^1 + 0*2^0 = 4
 \end{aligned}$$



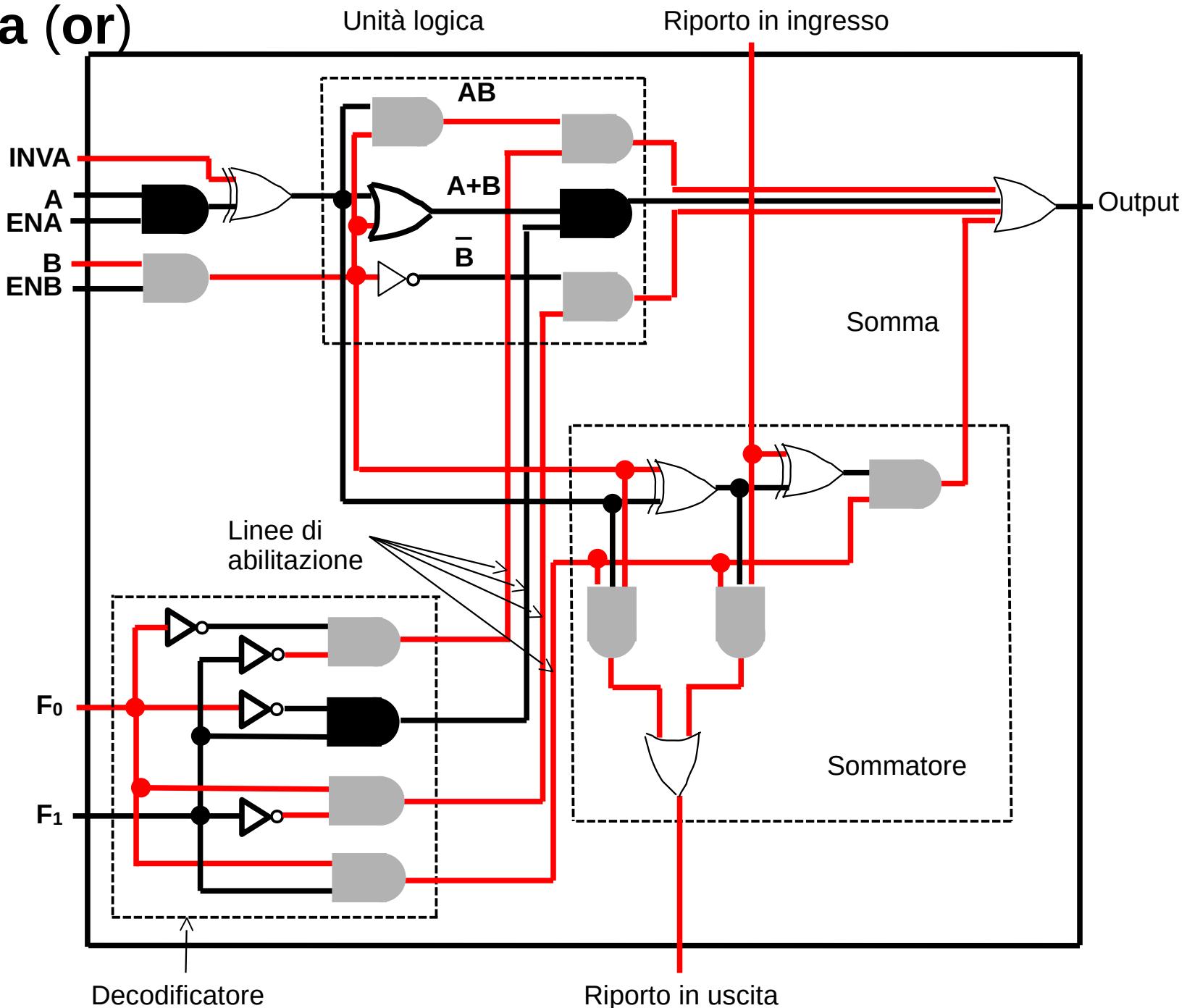
# ALU a 1 bit: somma logica (or)

INVA	A	ENA	B	ENB	$F_0$	$F_1$
0	1	1	1	1	0	1



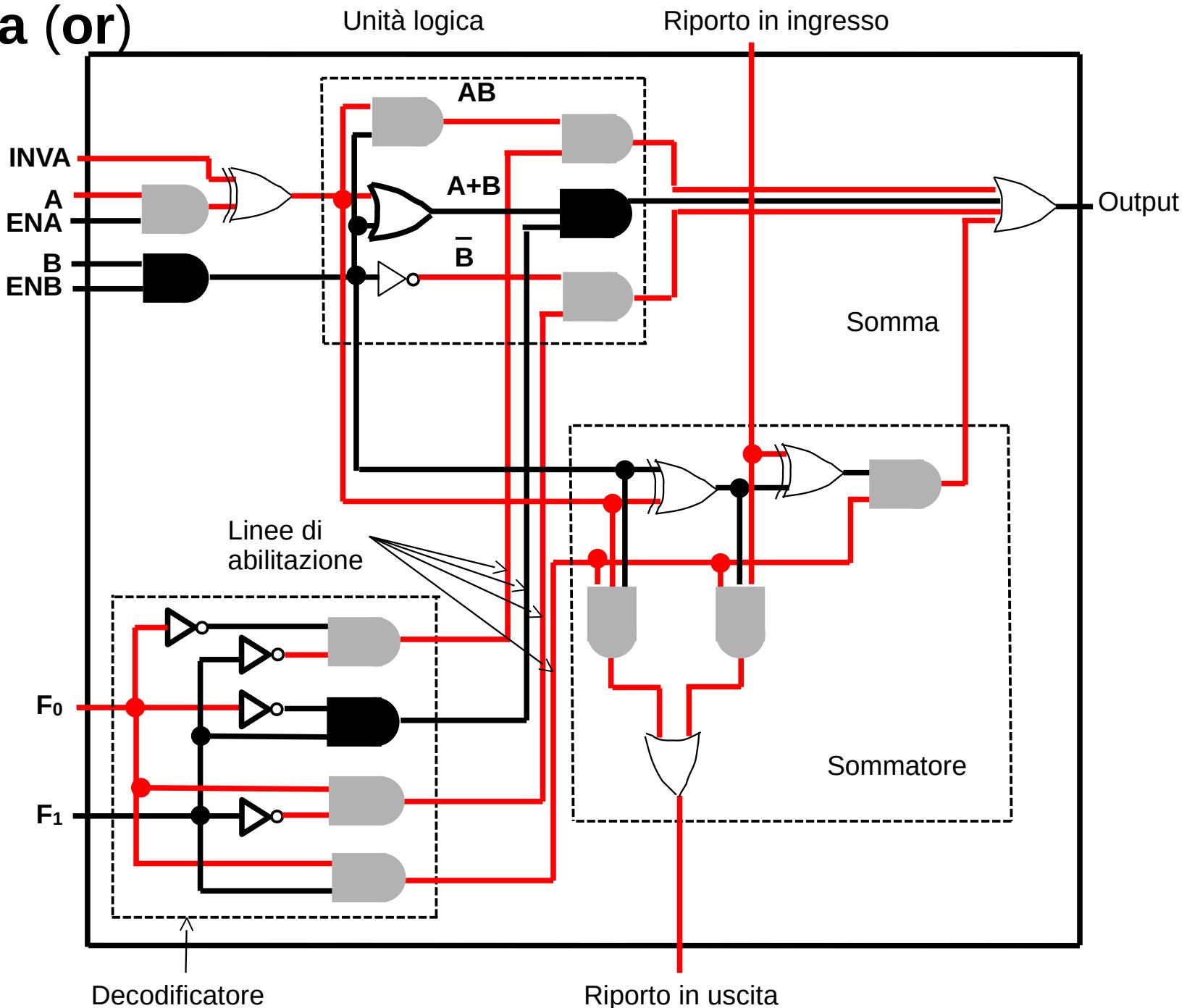
# ALU a 1 bit: somma logica (or)

INVA	A	ENA	B	ENB	$F_0$	$F_1$
0	1	1	0	1	0	1



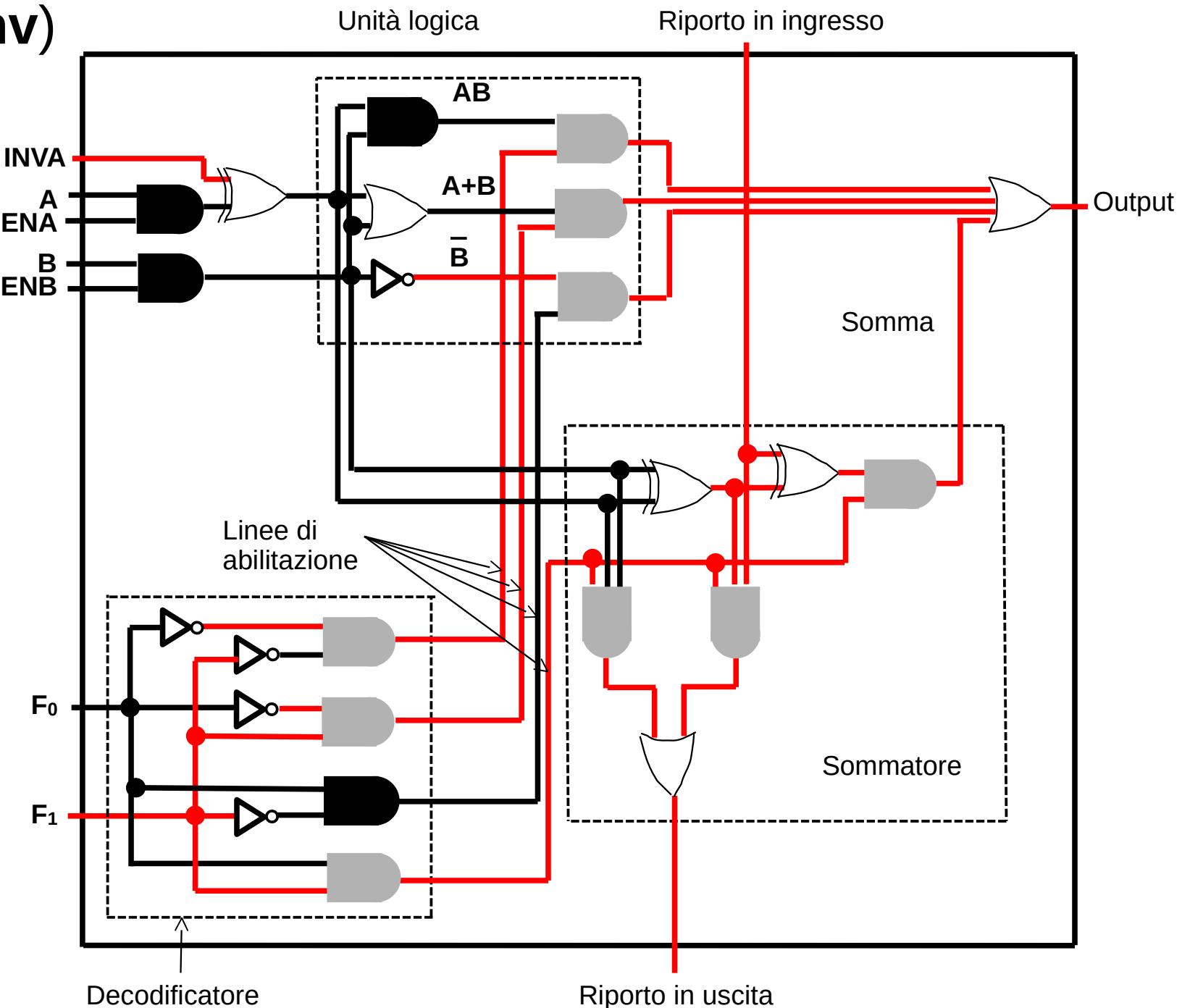
# ALU a 1 bit: somma logica (or)

INVA	A	ENA	B	ENB	$F_0$	$F_1$
0	0	1	1	1	0	1



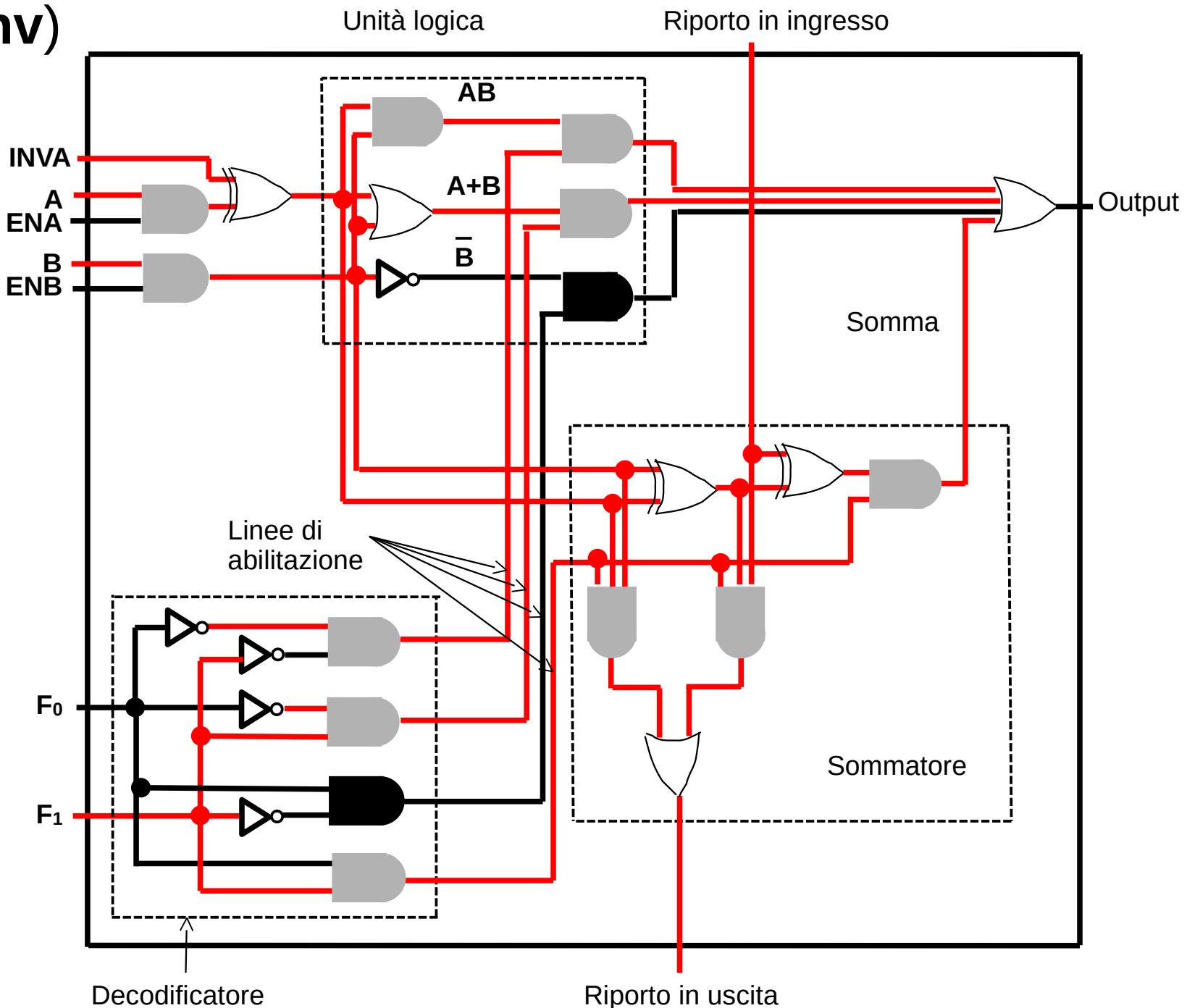
# ALU a 1 bit: invertitore (inv)

INVA	A	ENA	B	ENB	$F_0$	$F_1$
0	1	1	1	1	1	0



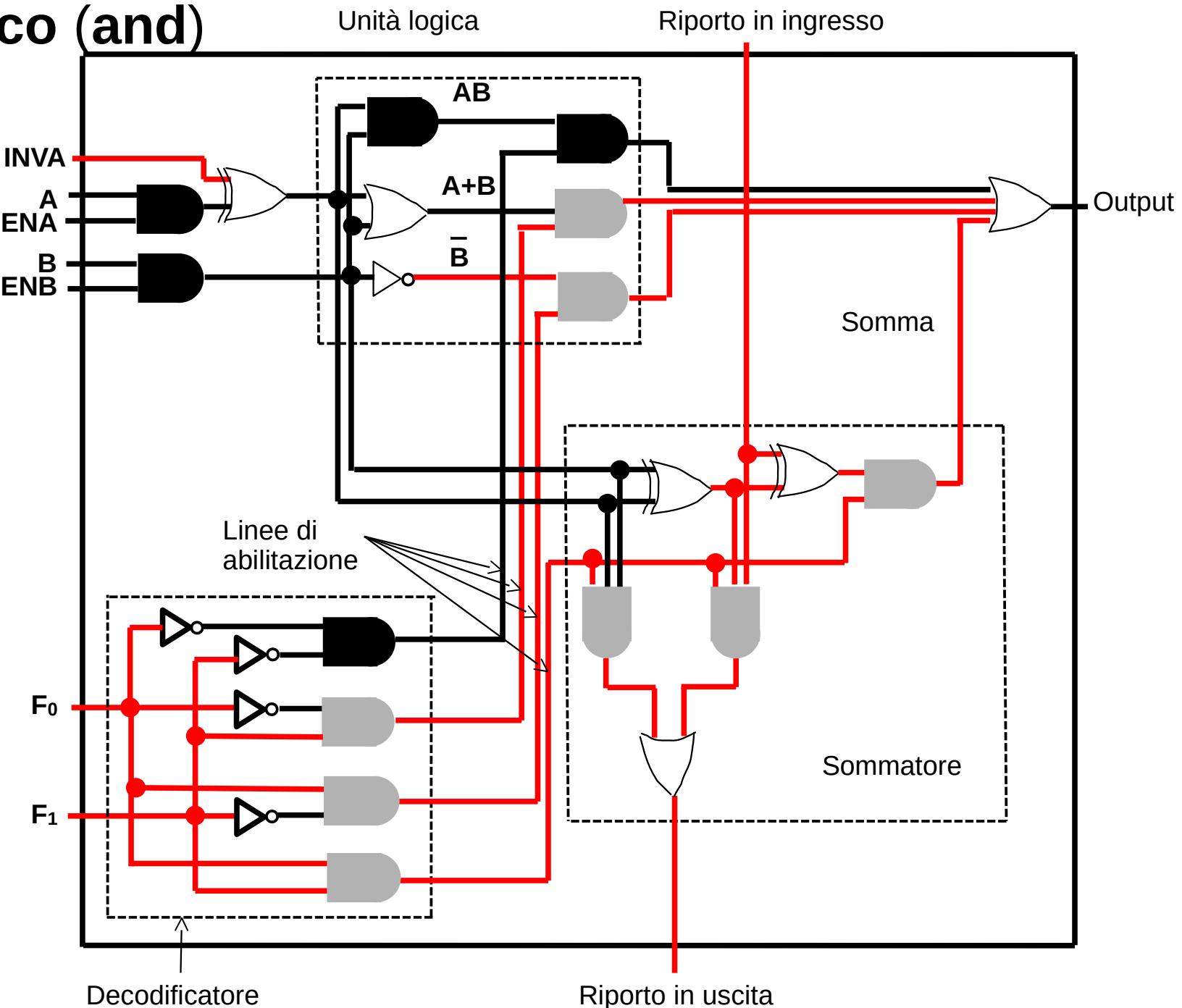
# ALU a 1 bit: invertitore (inv)

INVA	A	ENA	B	ENB	$F_0$	$F_1$
0	0	1	0	1	1	0



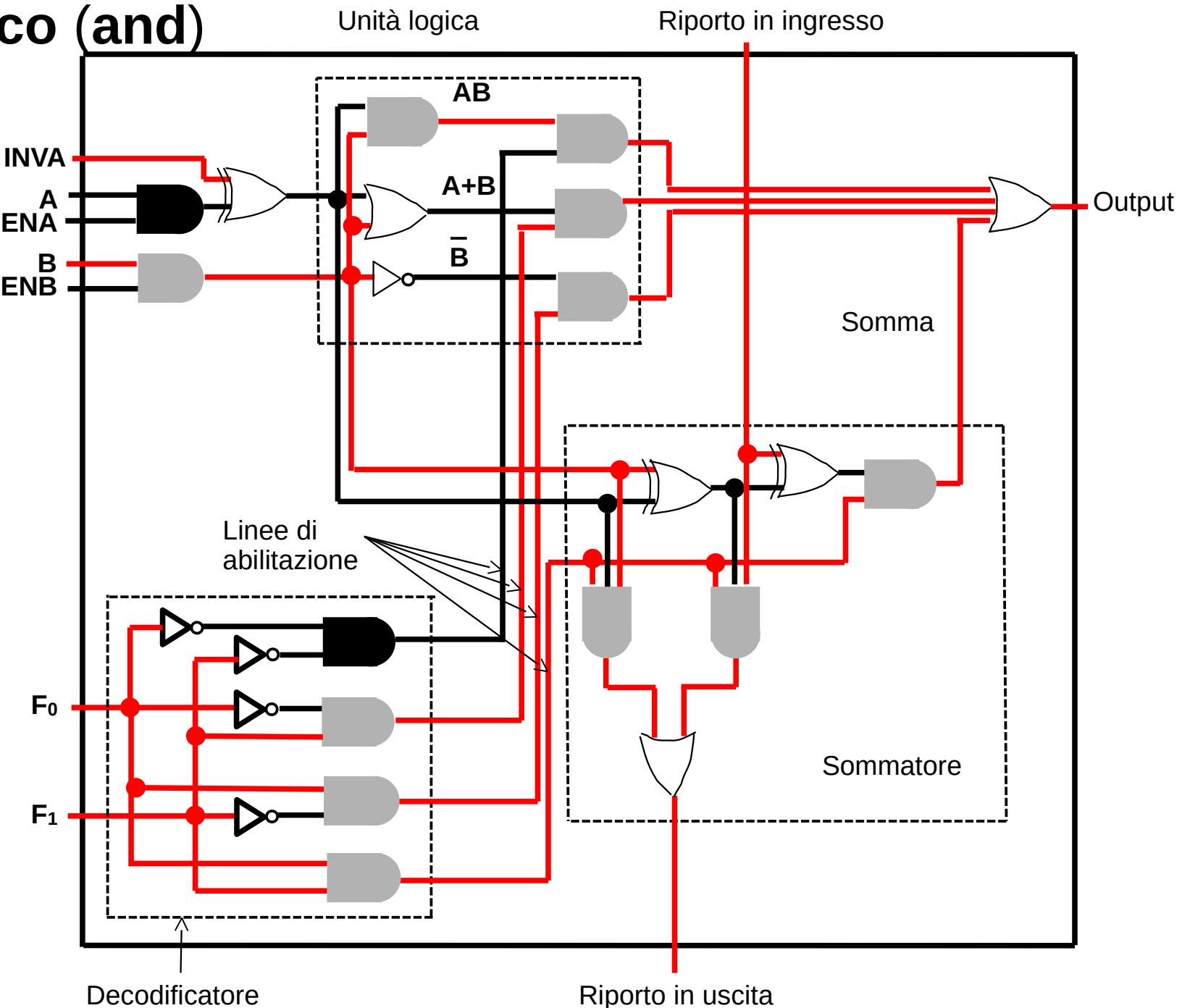
# ALU a 1 bit: prodotto logico (and)

INVA	A	ENA	B	ENB	$F_0$	$F_1$
0	1	1	1	1	0	0



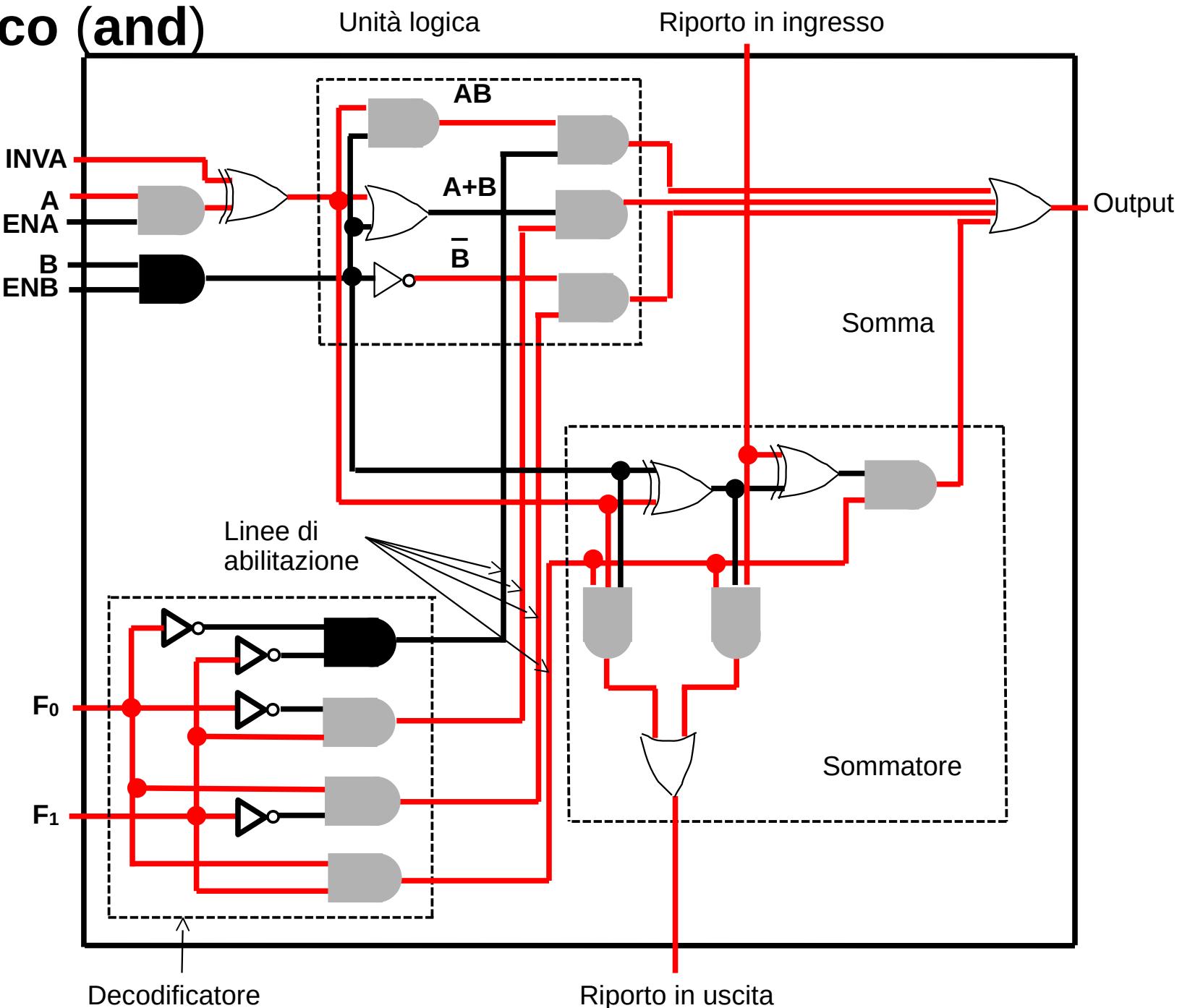
# ALU a 1 bit: prodotto logico (and)

INVA	A	ENA	B	ENB	$F_0$	$F_1$
0	1	1	0	1	0	0



# ALU a 1 bit: prodotto logico (and)

INVA	A	ENA	B	ENB	$F_0$	$F_1$
0	0	1	1	1	0	0



# Memoria

- Necessaria per conservare
  - Istruzioni da eseguire (programmi)
  - Dati

## Logica sequenziale

- I circuiti visti finora funzionano in modo prevedibile e deterministico: i dati di input producono quasi istantaneamente un output ma...
- non possono memorizzare un'informazione
- *Latch: a circuit which retains whatever output state results from a momentary input signal until reset by another signal*
- Per fornire una “memoria” ad un circuito, bisogna realizzare un dispositivo che convervi un’informazione in un determinato istante di tempo → logica sequenziale
- Sequenziale perché per immagazzinare e recuperare dati sotto forma di bits **una serie di passaggi devono verificarsi in un determinato ordine**

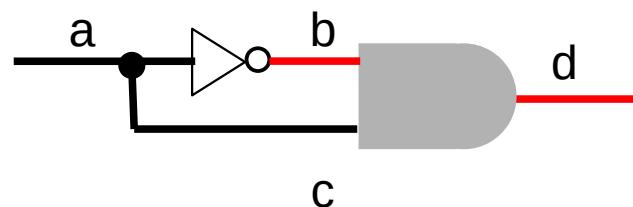
# Logica sequenziale

Es. di **sequenza di istruzioni per accedere ad una cella di memoria:**

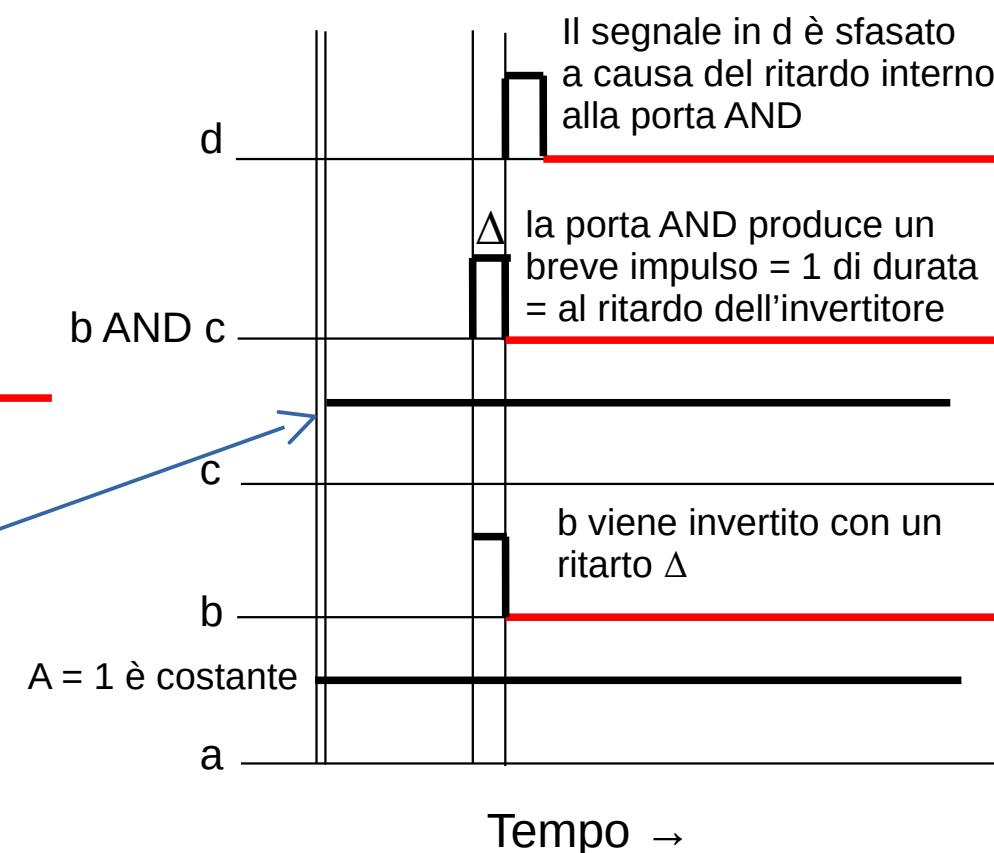
- Invio di un segnale di abilitazione al dispositivo di storage (memoria)
- Caricamento di un blocco di dati tutti insieme (*parallel load*) o uno per volta (*serial load*)
- Invio di un impulso di controllo alla memoria
- Altri impulsi possono essere necessari per recuperare i dati, estraendoli dalla memoria
- Necessario un **generatore di clock**
- ≈ al cuore di un animale
- Produce una serie di tensioni alte e basse (analoghe alle pressioni che la pompa cardiaca imprime al sangue arterioso) che attivano il circuito
- Il clock fornisce un riferimento temporale per tutte le operazioni sequenziali (un po' come un metronomo)

# Flip-flop

- Circuiti in cui il cambiamento di stato avviene durante la transizione del clock da 0 a 1 (fronte di salita) oppure da 1 a 0 (fronte di discesa)



$C = A = 1$ , costante, ha un piccolo ritardo dovuto alla velocità di propagazione del segnale ( $2 \cdot c/3$ )

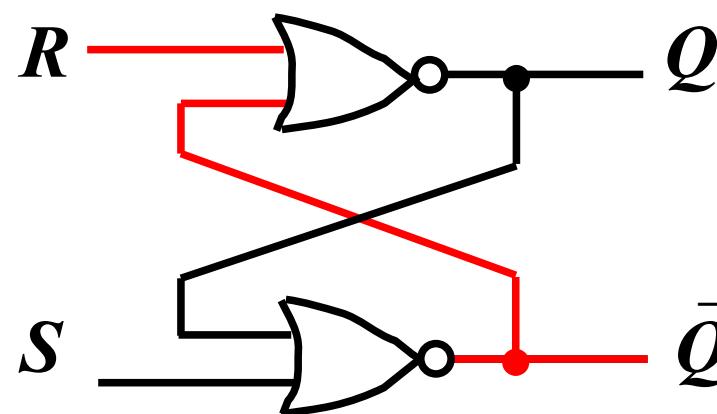


## Flip-Flops SR

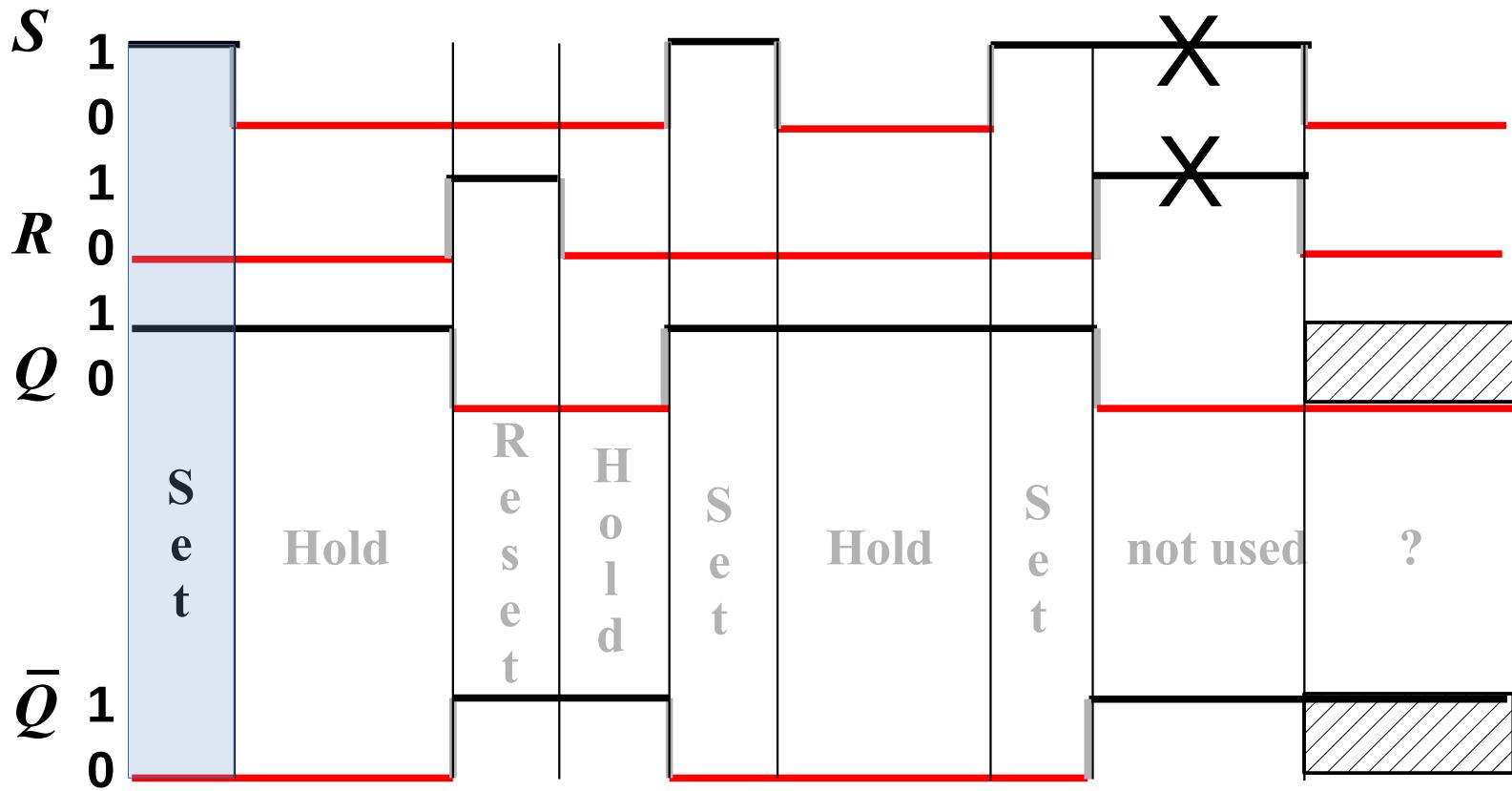
- Il circuito più elementare di data-storage è il *flip-flop set-reset* (SR), detto anche *latch* trasparente; due tipi:
  - **Cross-NOR SR flip-flop**
  - **Cross-NAND SR flip-flop**

# Cross-NOR SR flip-flop

A	B	NOR
1	1	0
1	0	0
0	1	0
0	0	1

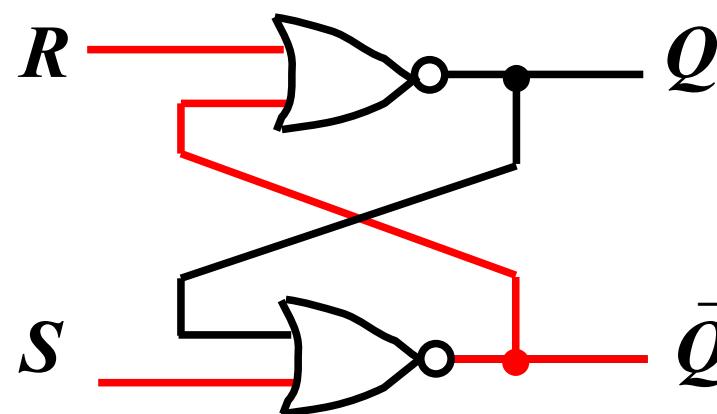


S	R	Q	$\bar{Q}$	condition
0	0	Q	$\bar{Q}$	Hold (no change)
0	1	0	1	Reset
1	0	1	0	Set
1	1	0	0	Not used

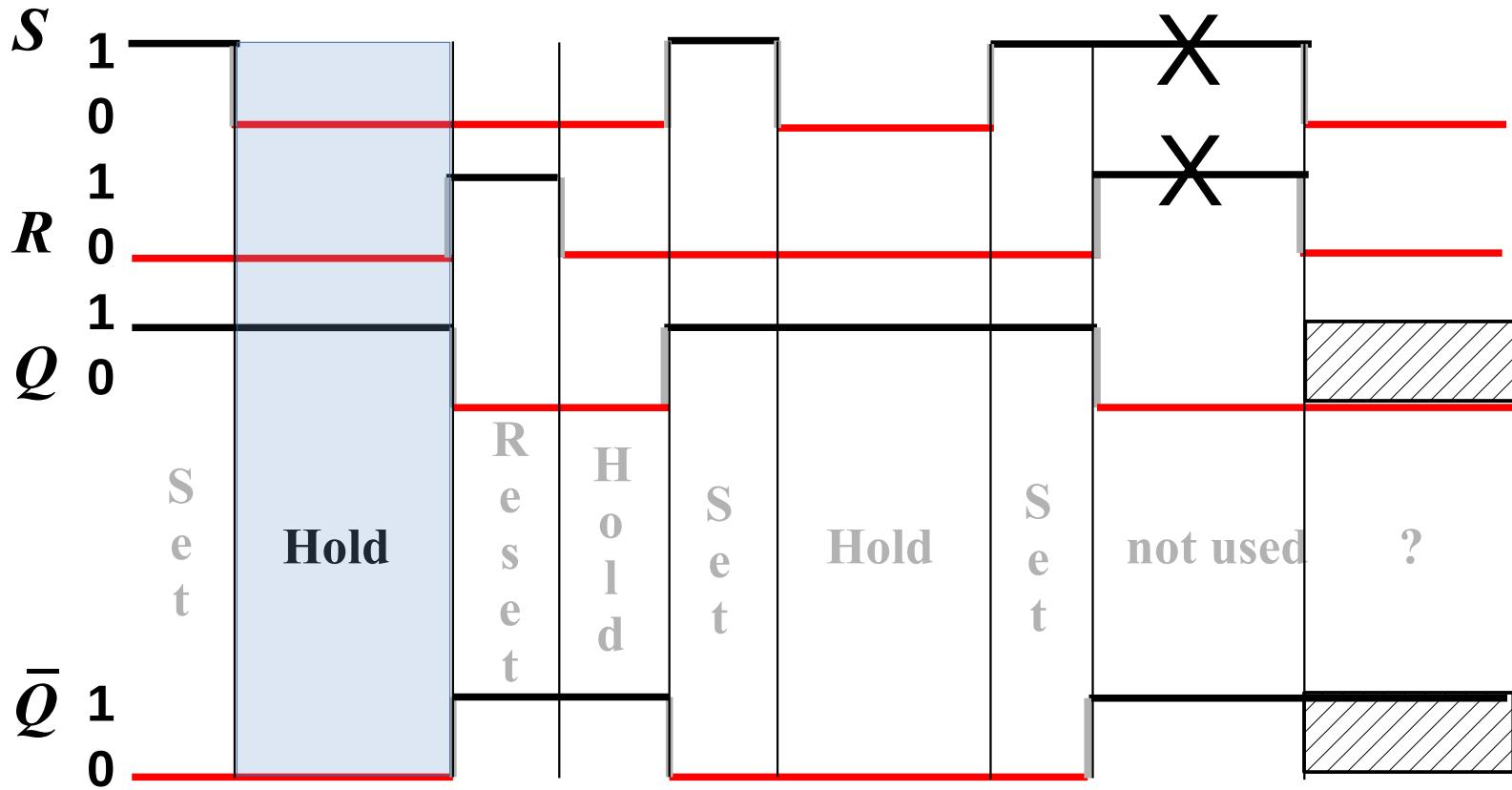


# Cross-NOR SR flip-flop

A	B	NOR
1	1	0
1	0	0
0	1	0
0	0	1

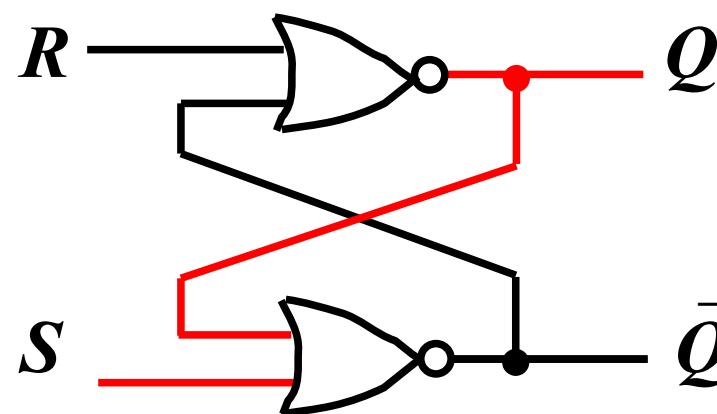


S	R	Q	$\bar{Q}$	condition
0	0	Q	$\bar{Q}$	Hold (no change)
0	1	0	1	Reset
1	0	1	0	Set
1	1	0	0	Not used

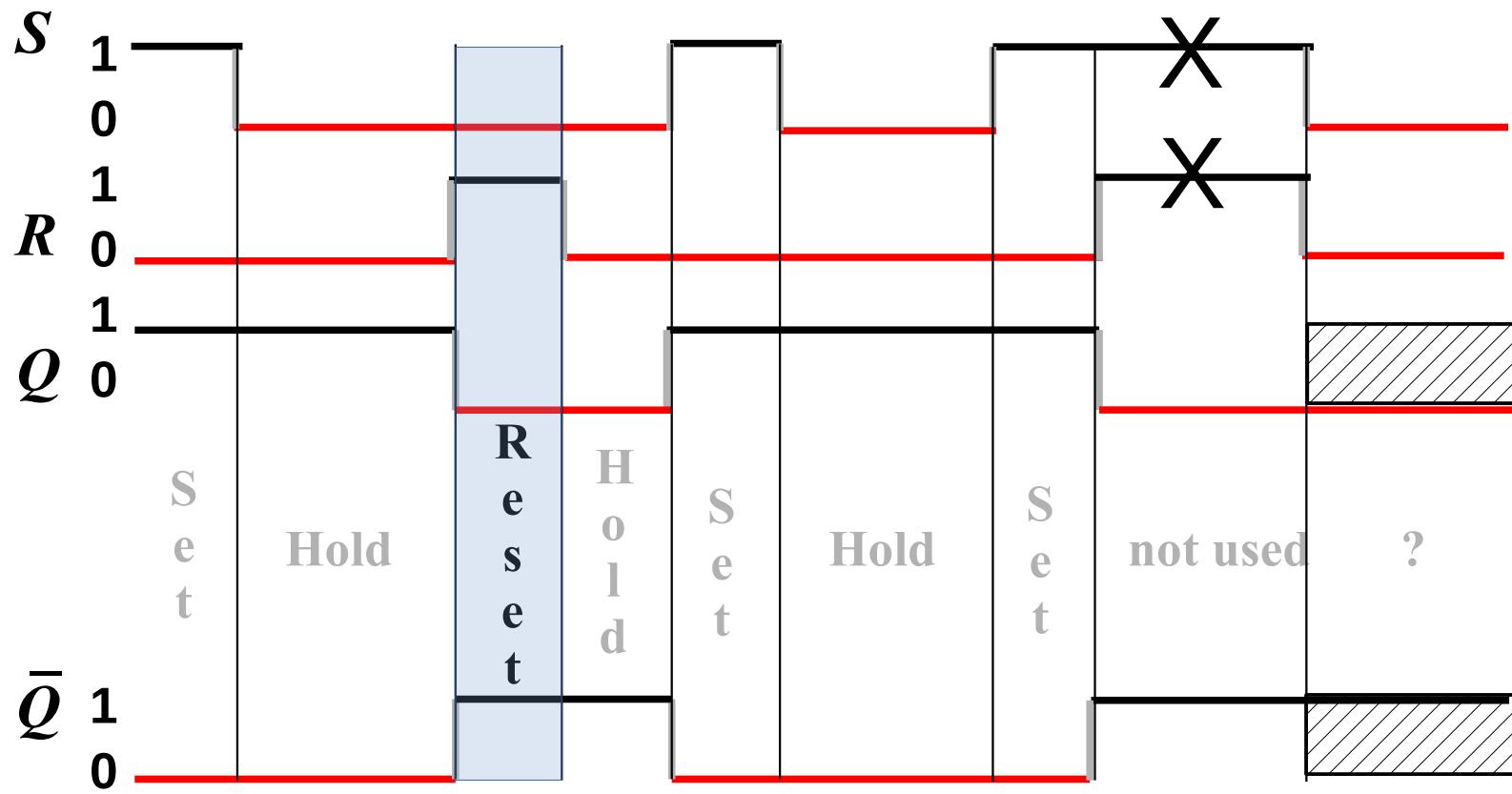


# Cross-NOR SR flip-flop

A	B	NOR
1	1	0
1	0	0
0	1	0
0	0	1

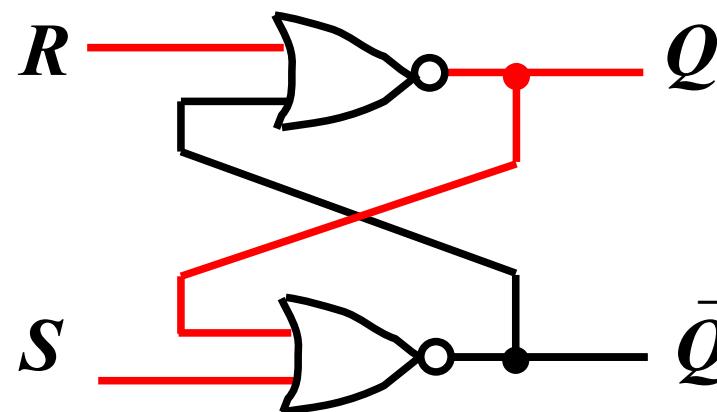


S	R	Q	$\bar{Q}$	condition
0	0	Q	$\bar{Q}$	Hold (no change)
0	1	0	1	Reset
1	0	1	0	Set
1	1	0	0	Not used



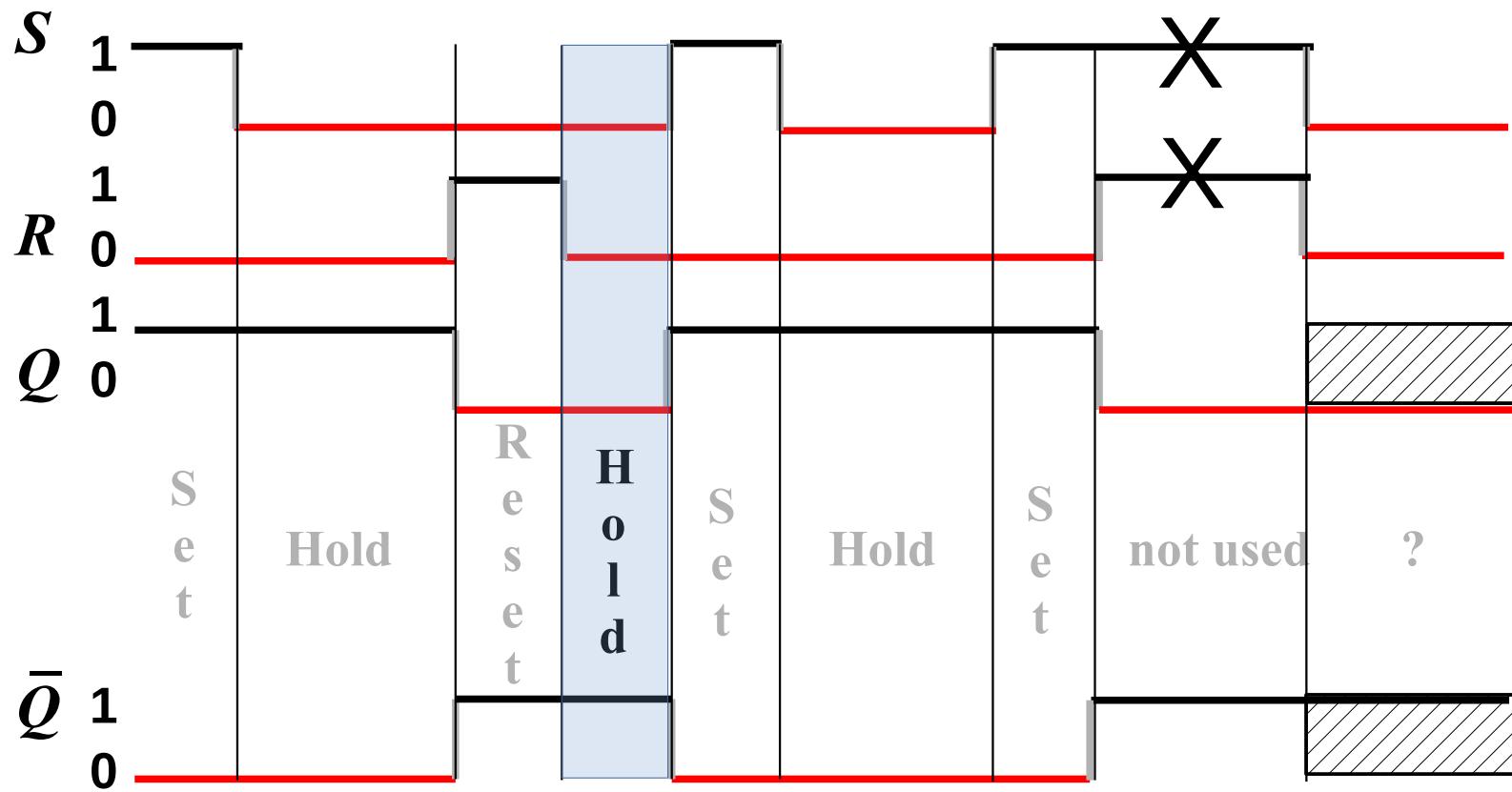
# Cross-NOR SR flip-flop

A	B	NOR
1	1	0
1	0	0
0	1	0
0	0	1



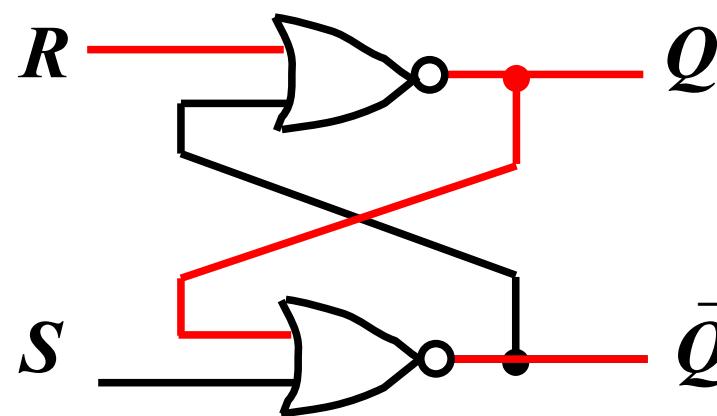
**Hold** conserva i valori precedenti degli output

S	R	Q	$\bar{Q}$	condition
0	0	Q	$\bar{Q}$	Hold (no change)
0	1	0	1	Reset
1	0	1	0	Set
1	1	0	0	Not used

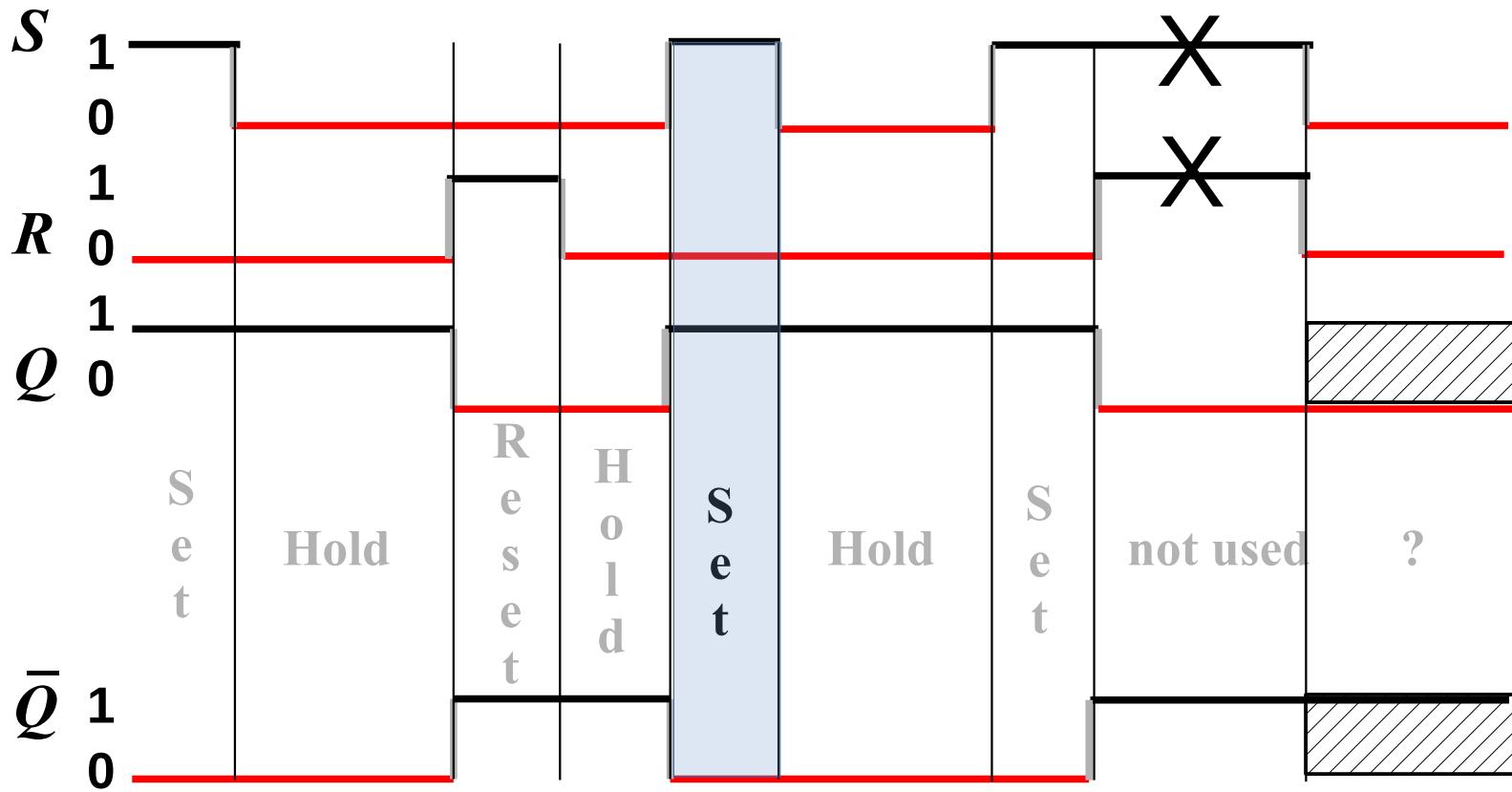


# Cross-NOR SR flip-flop

A	B	NOR
1	1	0
1	0	0
0	1	0
0	0	1

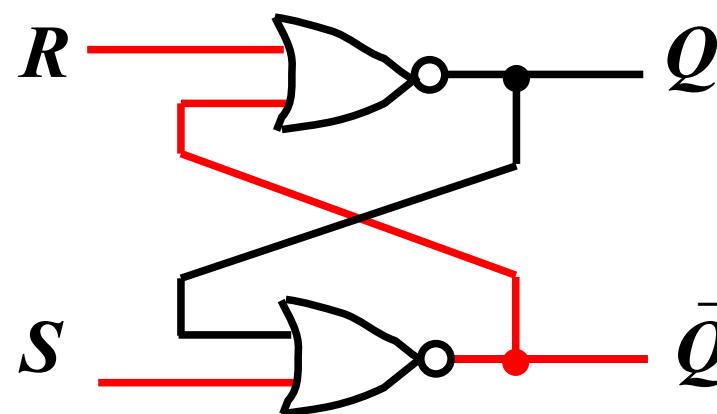


S	R	Q	$\bar{Q}$	condition
0	0	Q	$\bar{Q}$	Hold (no change)
0	1	0	1	Reset
1	0	1	0	Set
1	1	0	0	Not used

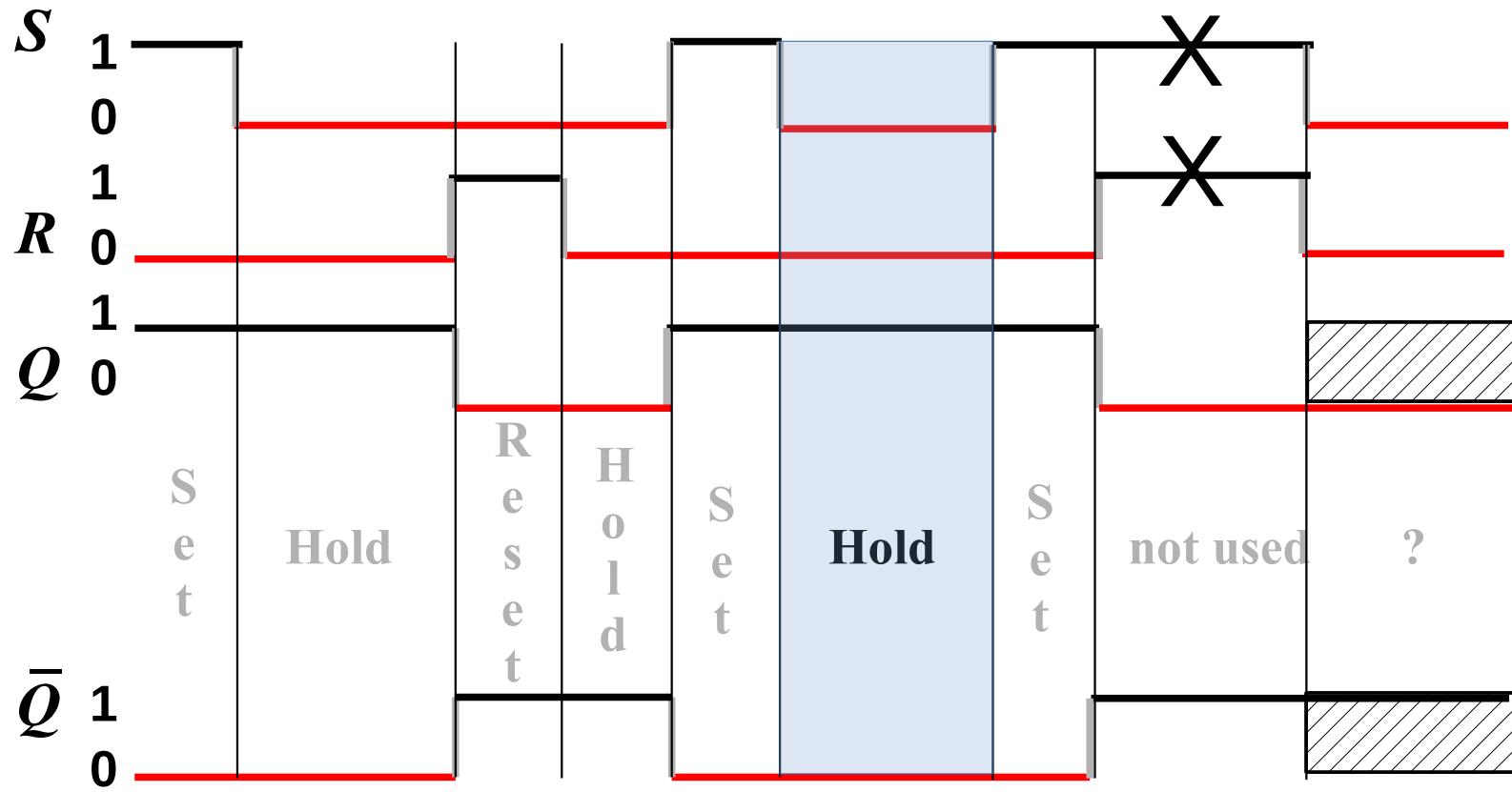


# Cross-NOR SR flip-flop

A	B	NOR
1	1	0
1	0	0
0	1	0
0	0	1

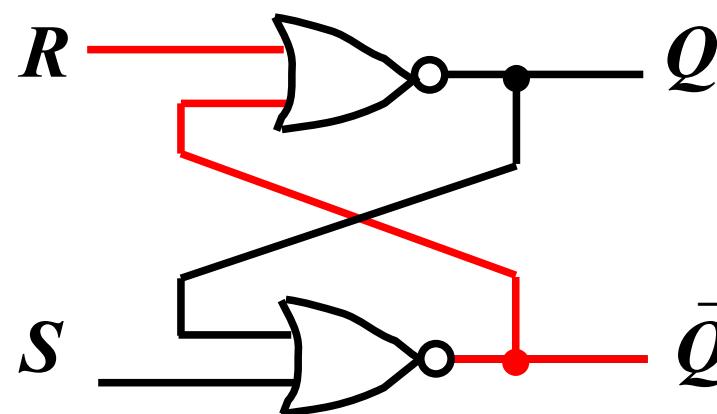


S	R	Q	$\bar{Q}$	condition
0	0	Q	$\bar{Q}$	Hold (no change)
0	1	0	1	Reset
1	0	1	0	Set
1	1	0	0	Not used

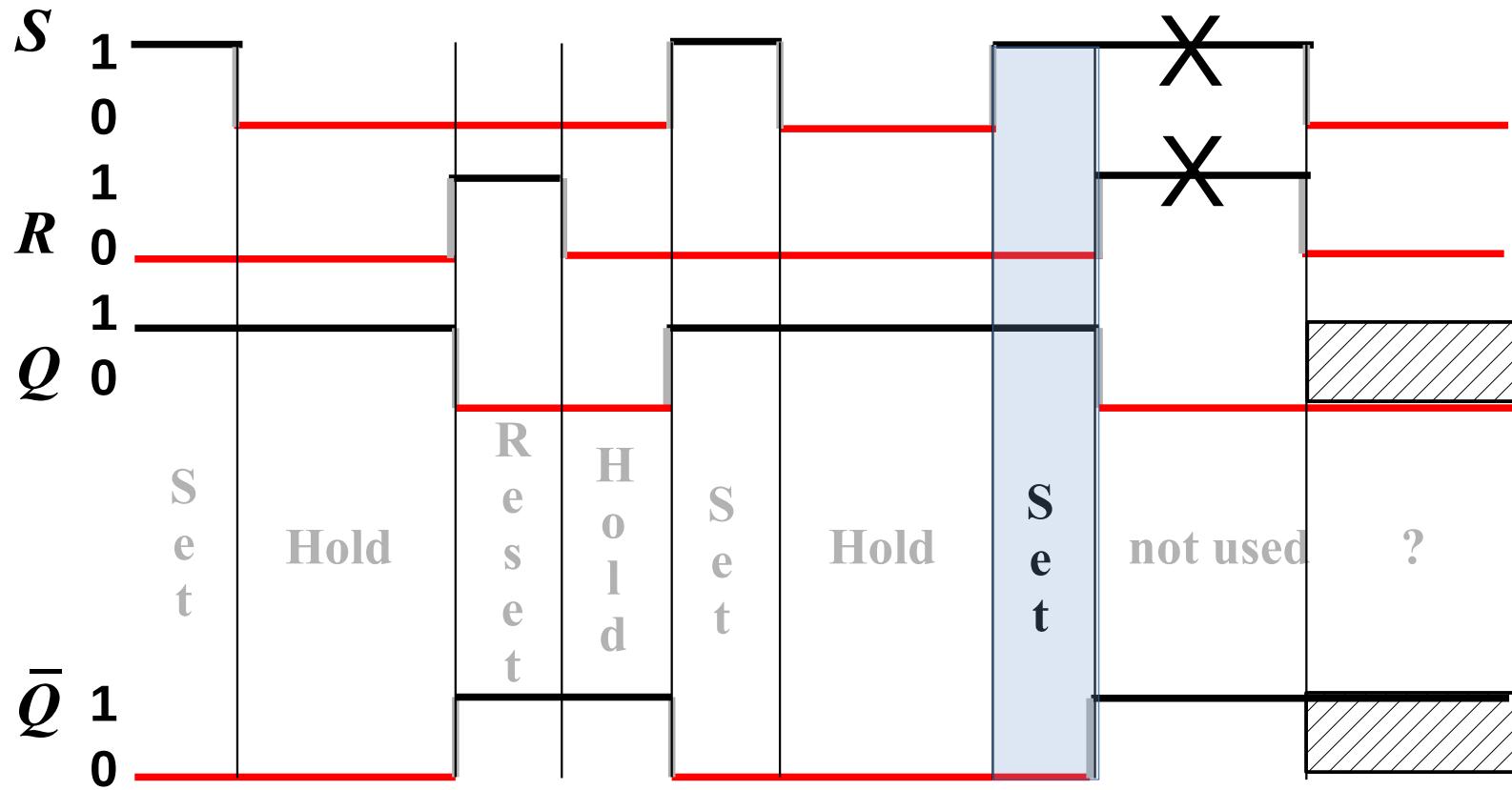


# Cross-NOR SR flip-flop

A	B	NOR
1	1	0
1	0	0
0	1	0
0	0	1

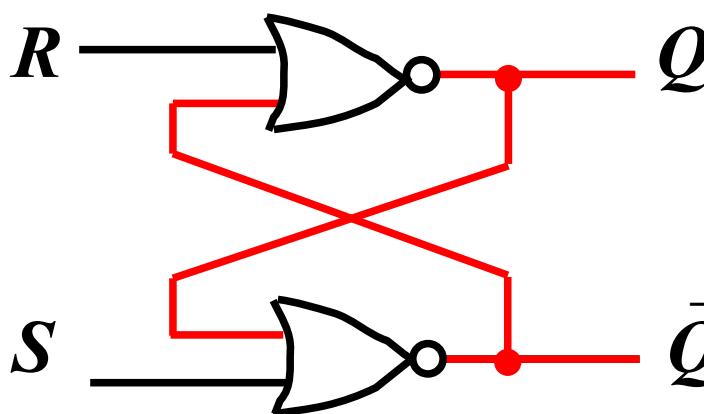


S	R	Q	$\bar{Q}$	condition
0	0	Q	$\bar{Q}$	Hold (no change)
0	1	0	1	Reset
1	0	1	0	Set
1	1	0	0	Not used



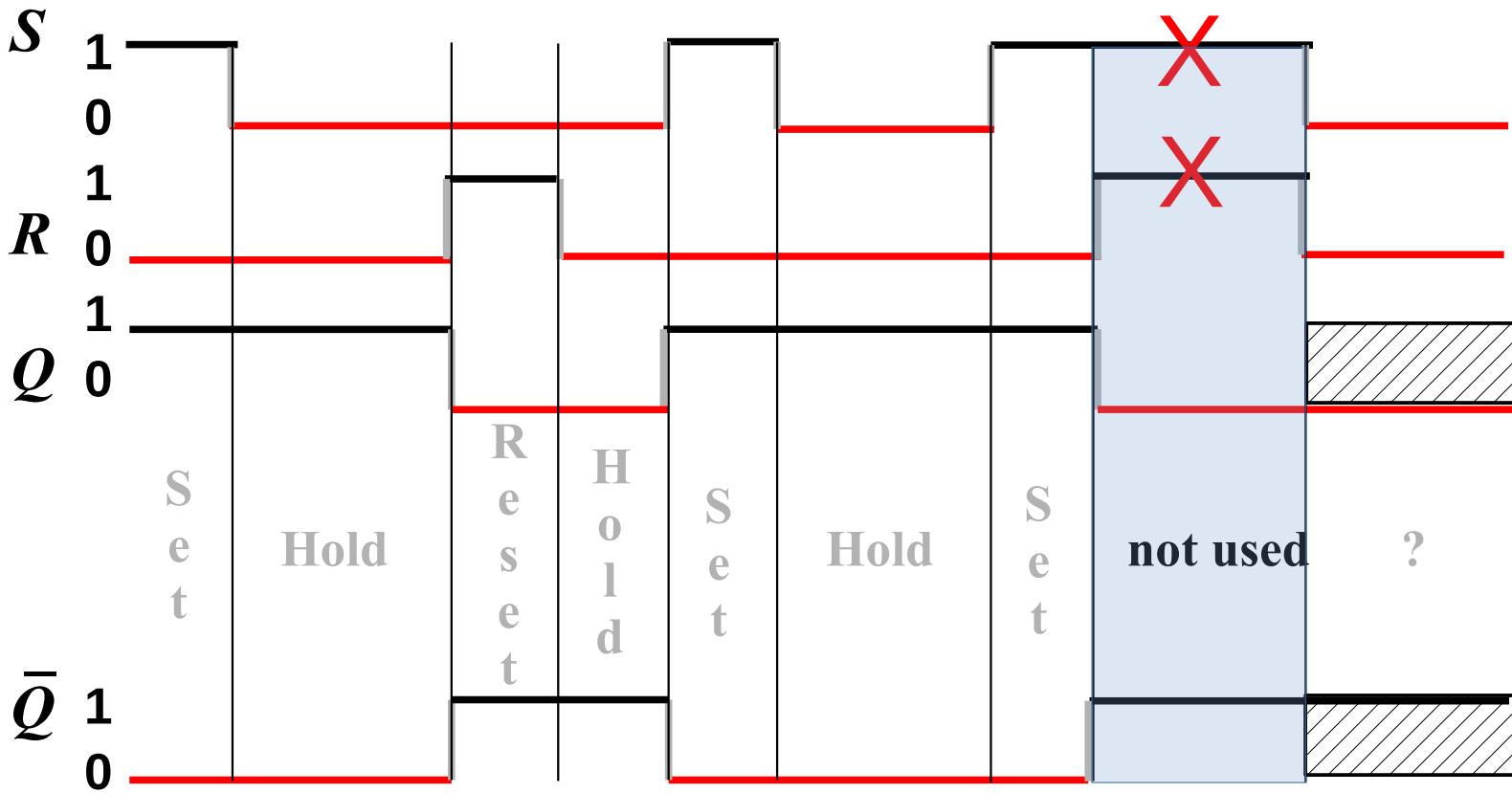
# Cross-NOR SR flip-flop

A	B	NOR
1	1	0
1	0	0
0	1	0
0	0	1



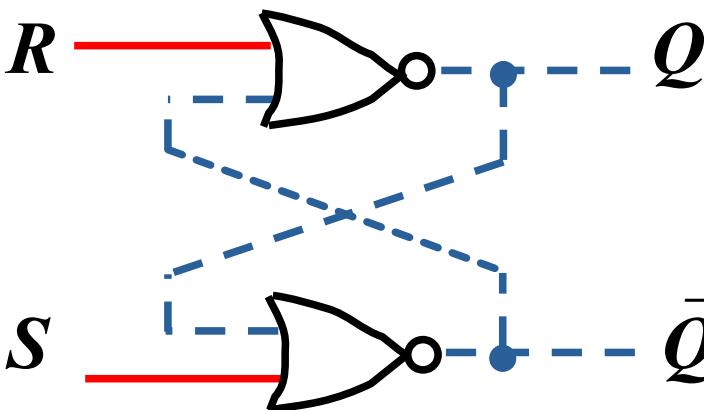
Why do you want to set and reset at the same time?

S	R	Q	$\bar{Q}$	condition
0	0	Q	$\bar{Q}$	Hold (no change)
0	1	0	1	Reset
1	0	1	0	Set
1	1	0	0	Not used



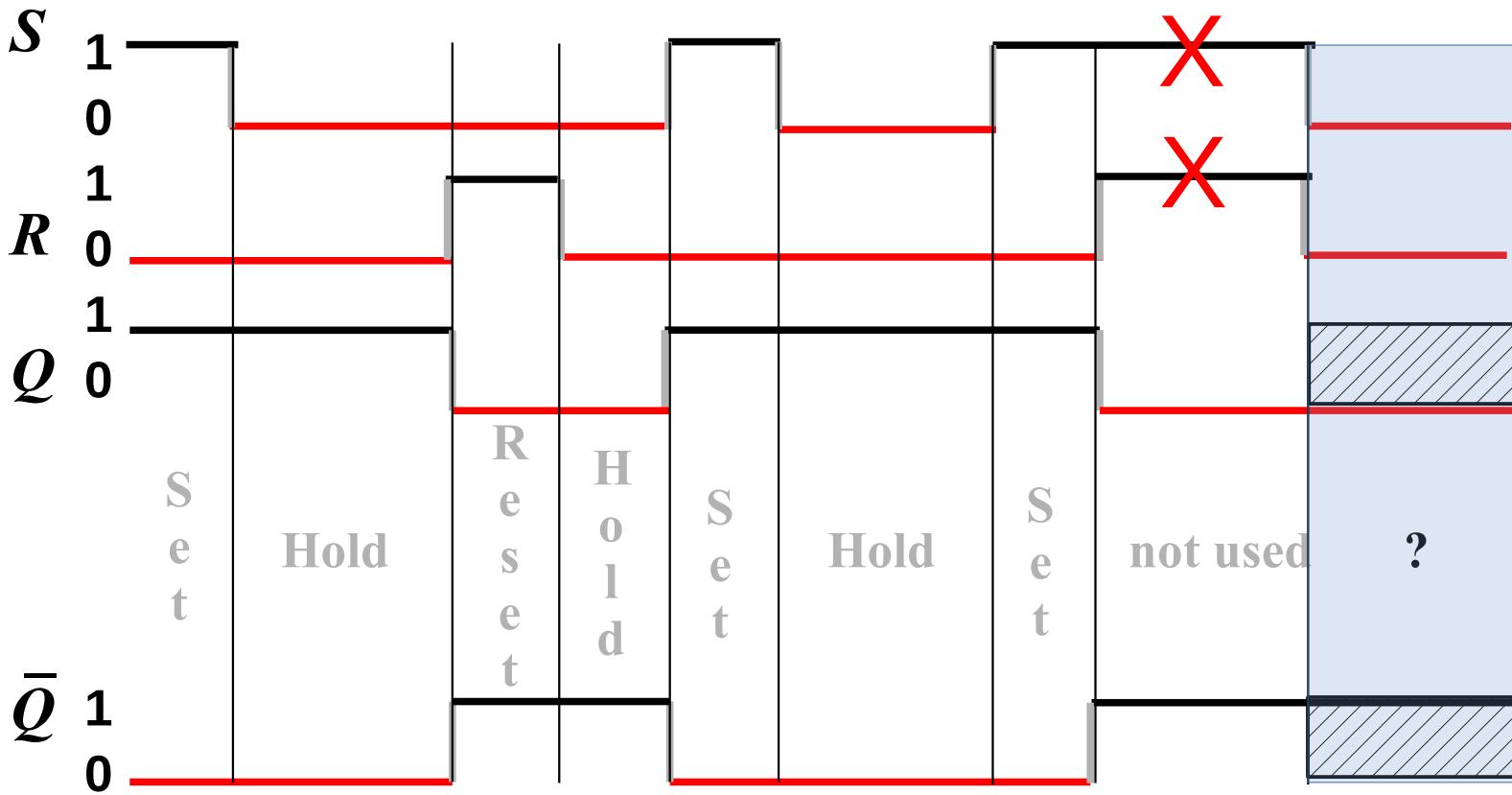
# Cross-NOR SR flip-flop

A	B	NOR
1	1	0
1	0	0
0	1	0
0	0	1



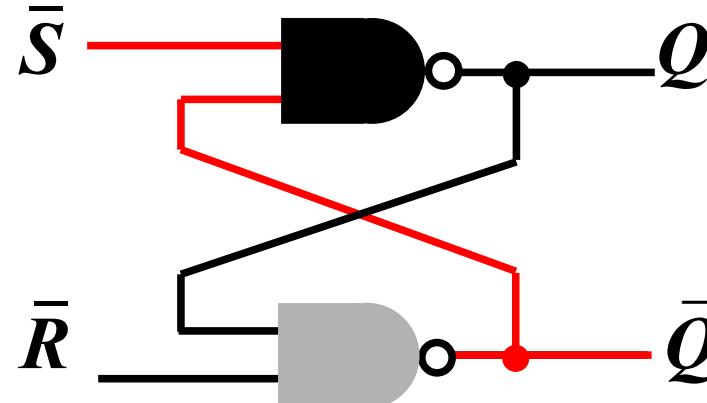
Se  $R$  ed  $S$  vanno entrambi a zero (**hold**), il valore degli output è imprevedibile!

S	R	Q	$\bar{Q}$	condition
0	0	Q	$\bar{Q}$	Hold (no change)
0	1	0	1	Reset
1	0	1	0	Set
1	1	0	0	Not used

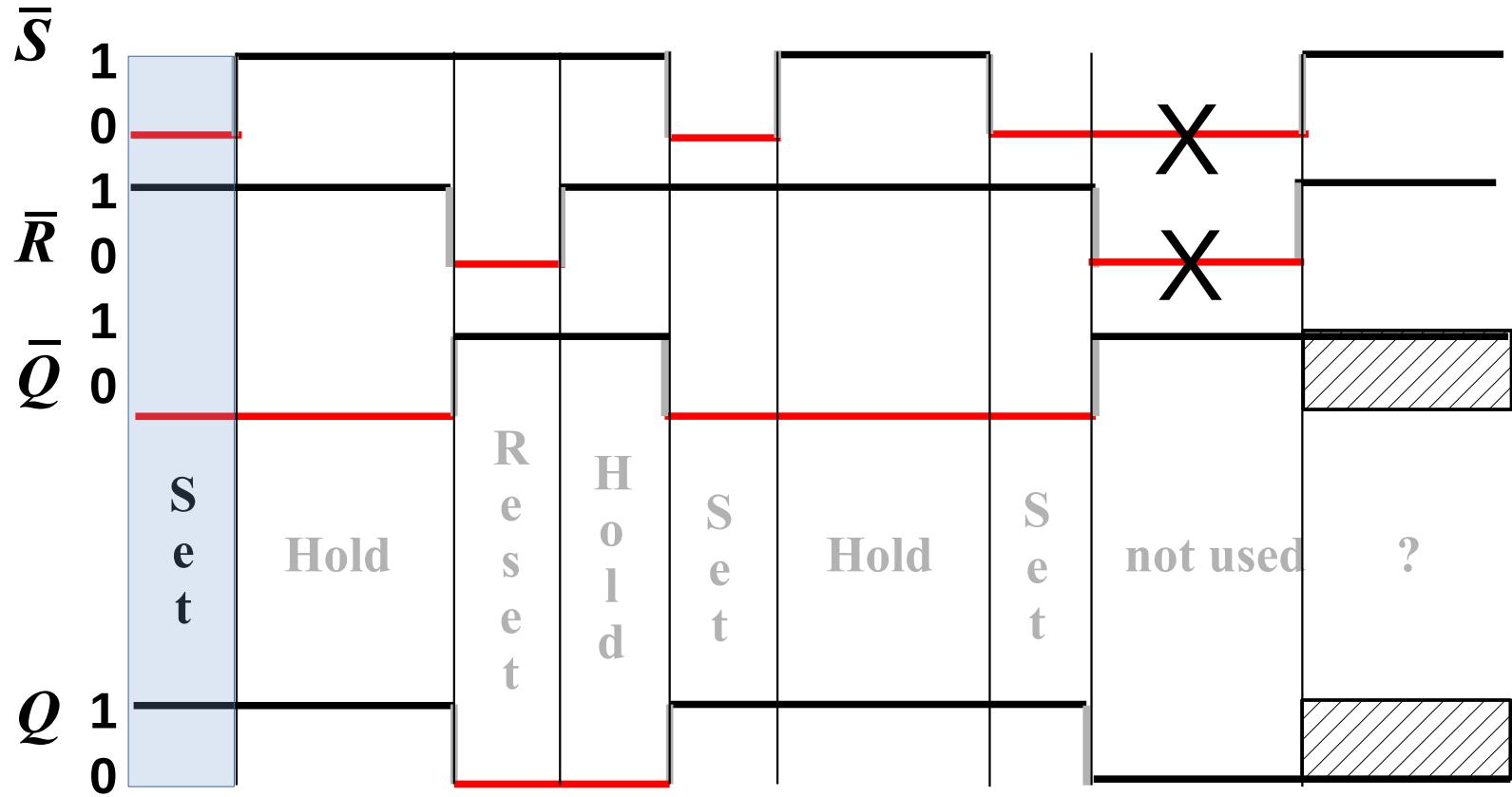


# Cross-NAND SR flip-flop $\bar{S}$

A	B	NAND
1	1	0
1	0	1
0	1	1
0	0	1

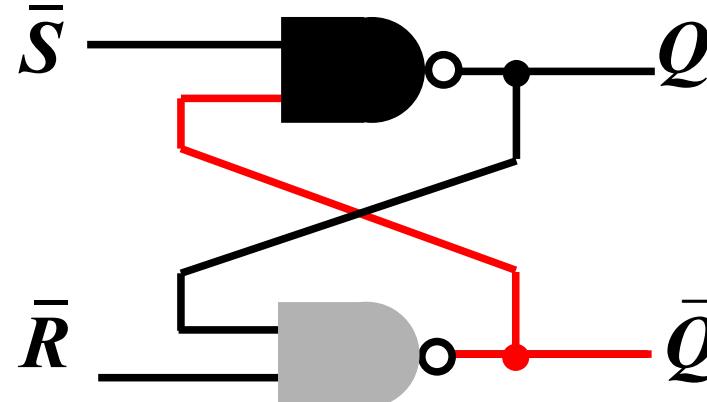


$\bar{S}$	$\bar{R}$	$Q$	$\bar{Q}$	condition
0	0	1	1	Not used
0	1	1	0	Set
1	0	0	1	Reset
1	1	$Q$	$\bar{Q}$	Hold (no change)

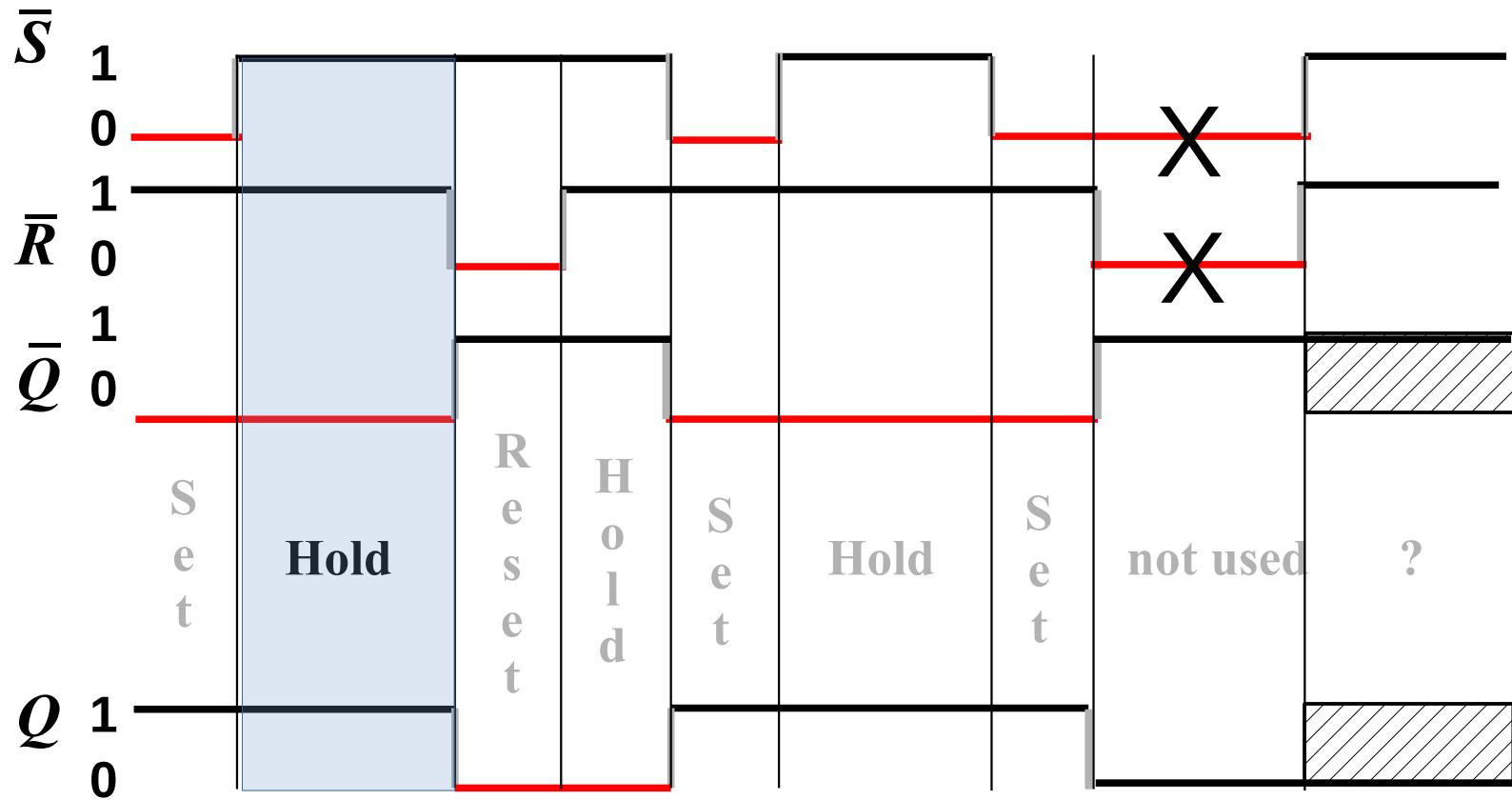


# Cross-NAND SR flip-flop $\bar{S}$

A	B	NAND
1	1	0
1	0	1
0	1	1
0	0	1

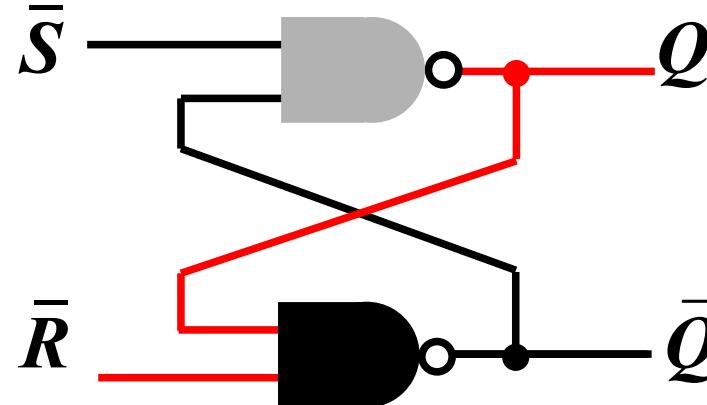


$\bar{S}$	$\bar{R}$	$Q$	$\bar{Q}$	condition
0	0	1	1	Not used
0	1	1	0	Set
1	0	0	1	Reset
1	1	$Q$	$\bar{Q}$	Hold (no change)

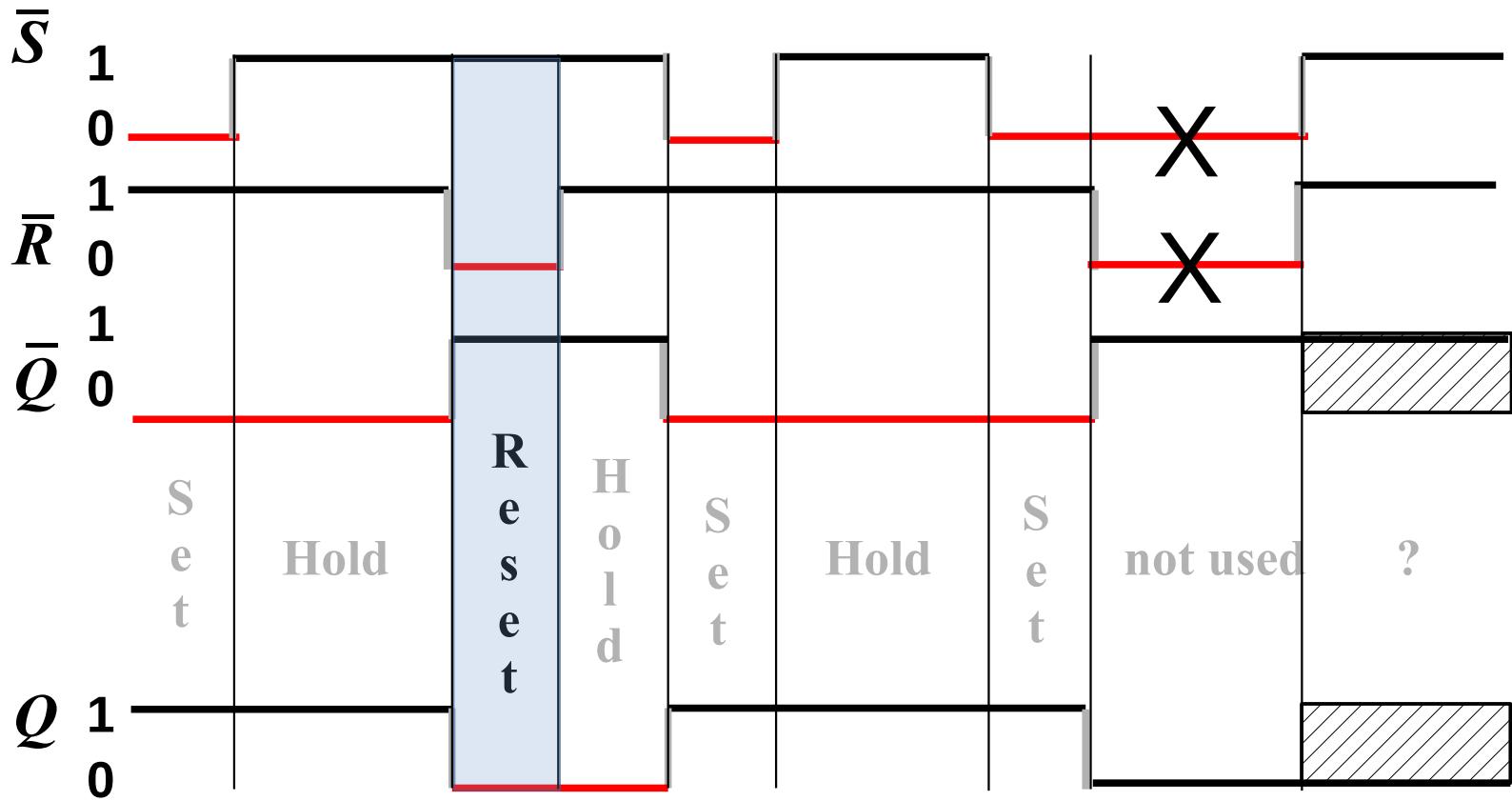


# Cross-NAND SR flip-flop $\bar{S}$

A	B	NAND
1	1	0
1	0	1
0	1	1
0	0	1

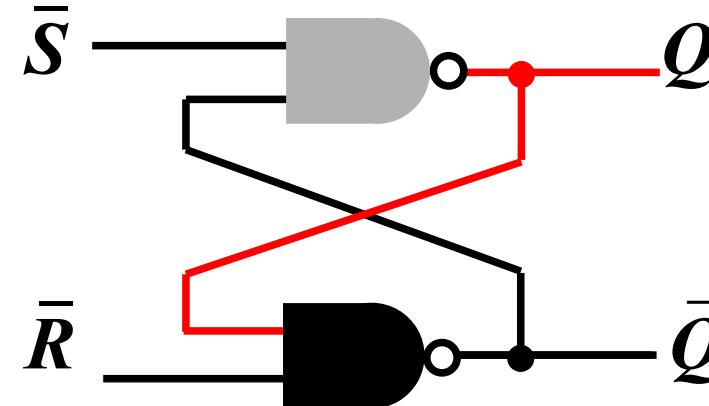


$\bar{S}$	$\bar{R}$	$Q$	$\bar{Q}$	condition
0	0	1	1	Not used
0	1	1	0	Set
1	0	0	1	Reset
1	1	$Q$	$\bar{Q}$	Hold (no change)

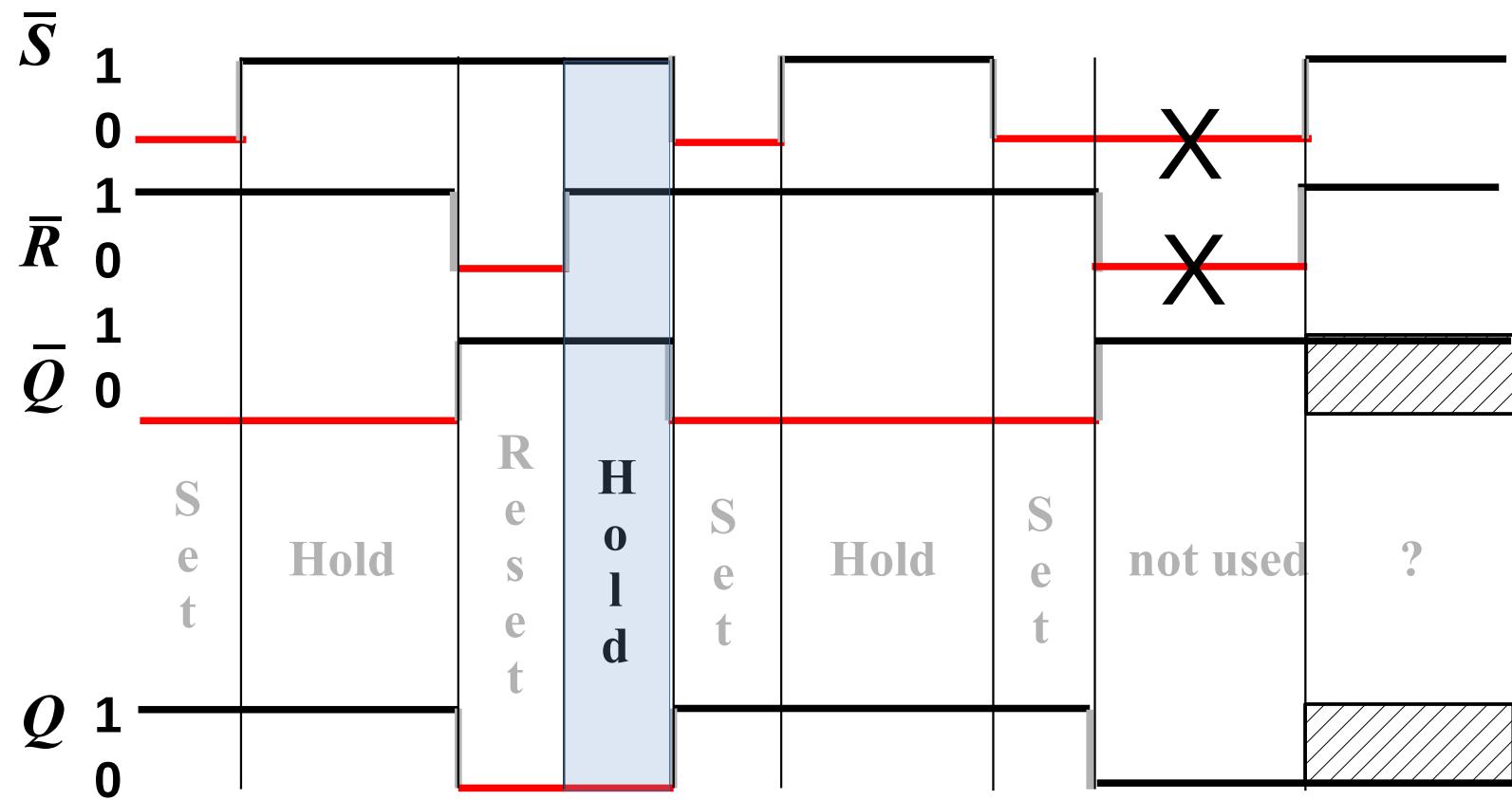


# Cross-NAND SR flip-flop $\bar{S}$

A	B	NAND
1	1	0
1	0	1
0	1	1
0	0	1

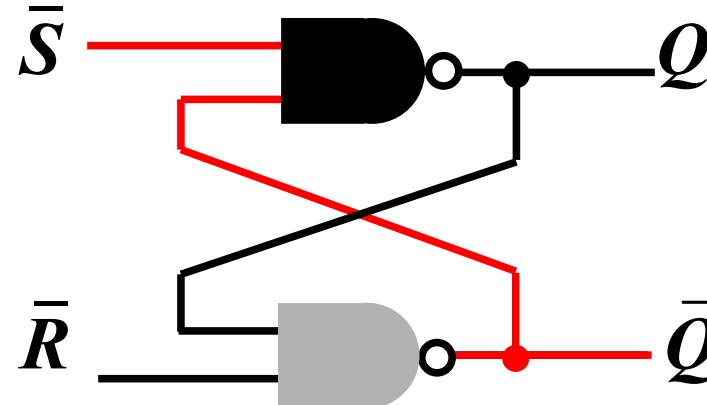


$\bar{S}$	$\bar{R}$	$Q$	$\bar{Q}$	condition
0	0	1	1	Not used
0	1	1	0	Set
1	0	0	1	Reset
1	1	$Q$	$\bar{Q}$	Hold (no change)

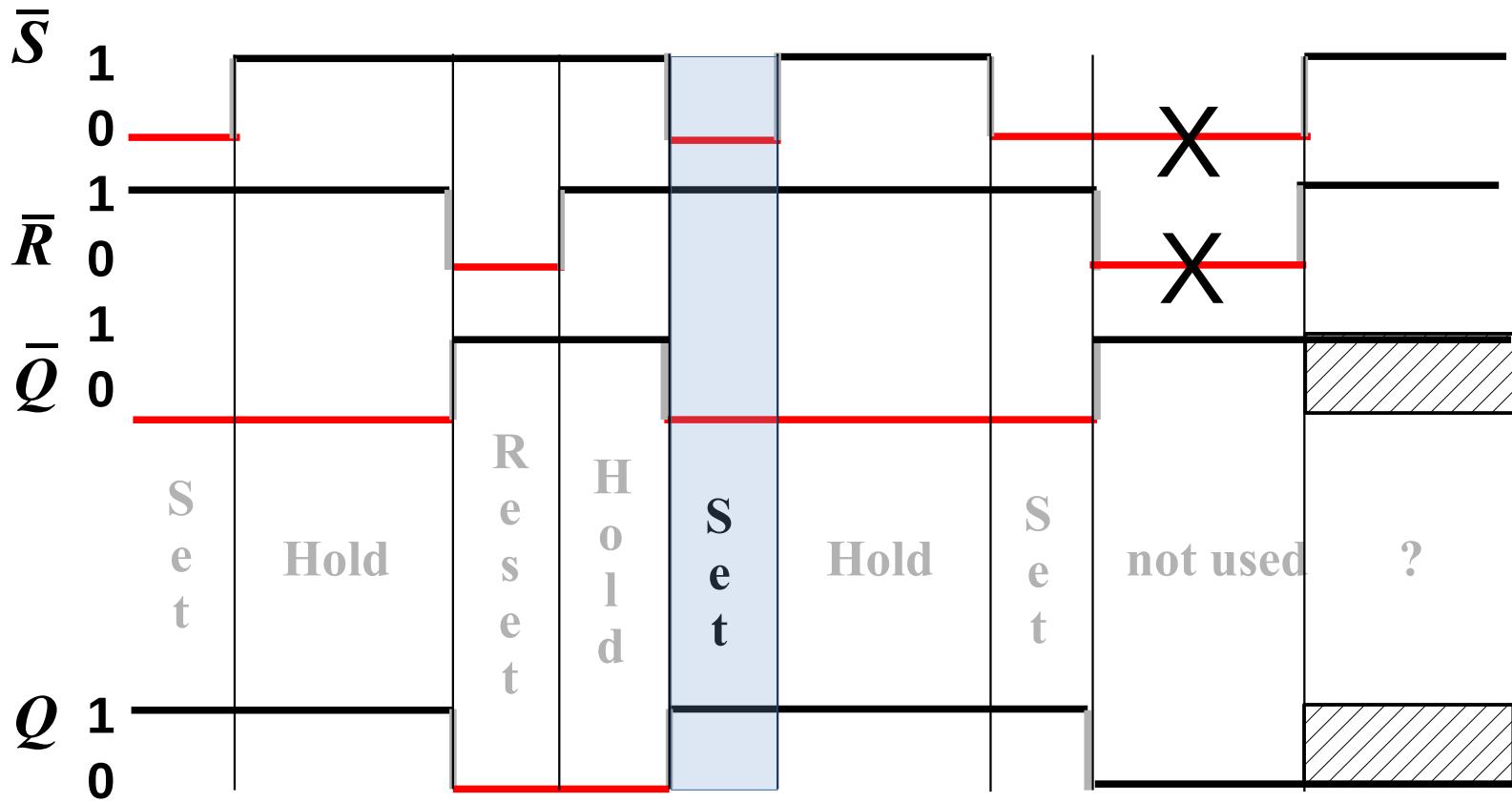


# Cross-NAND SR flip-flop $\bar{S}$

A	B	NAND
1	1	0
1	0	1
0	1	1
0	0	1

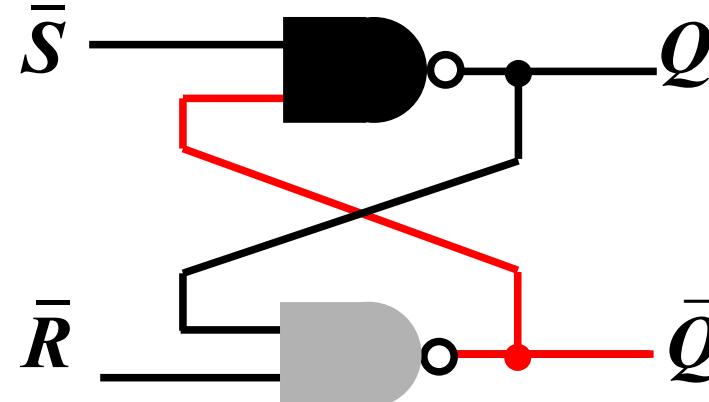


$\bar{S}$	$\bar{R}$	$Q$	$\bar{Q}$	condition
0	0	1	1	Not used
0	1	1	0	Set
1	0	0	1	Reset
1	1	$Q$	$\bar{Q}$	Hold (no change)

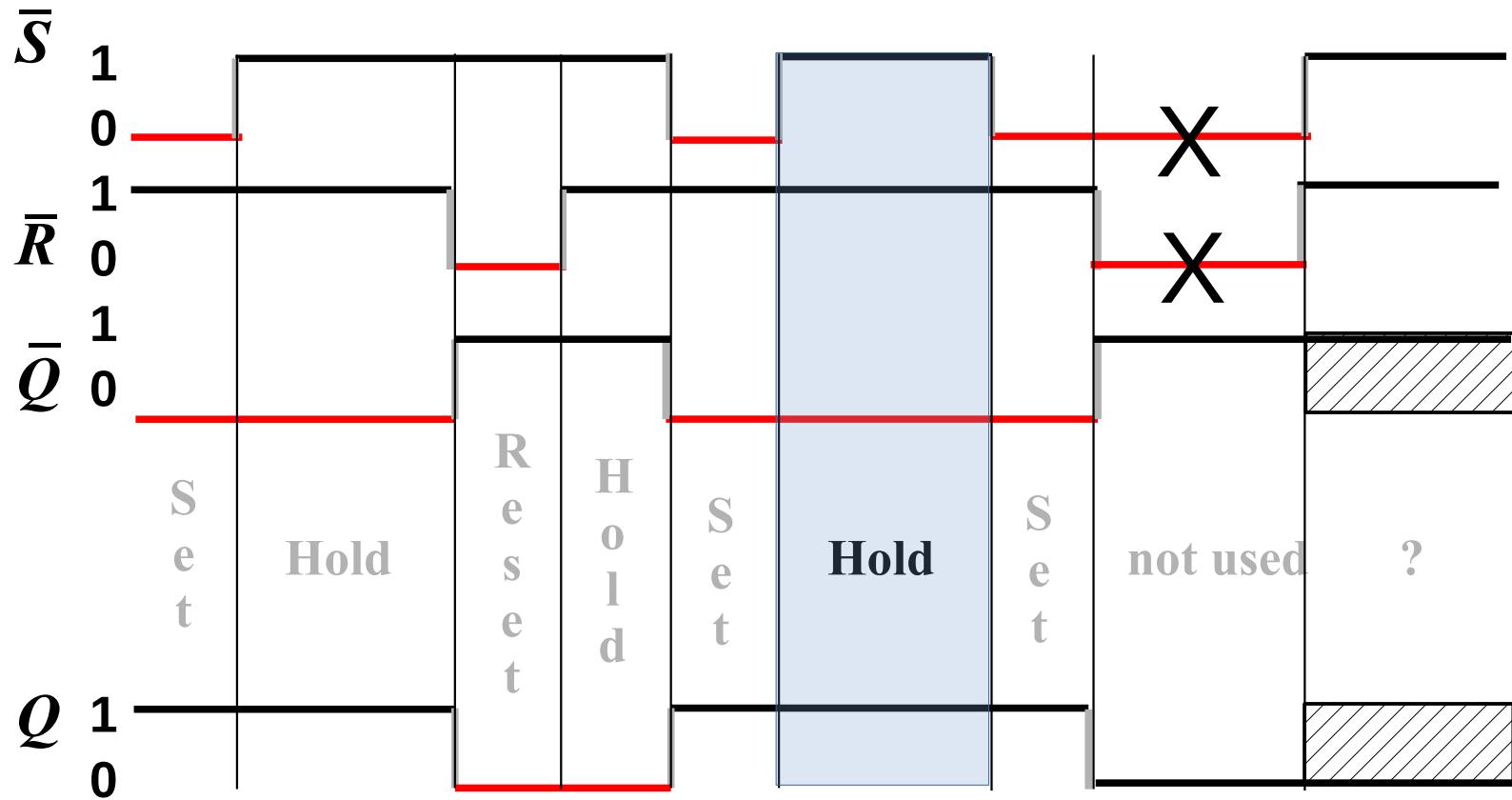


# Cross-NAND SR flip-flop $\bar{S}$

A	B	NAND
1	1	0
1	0	1
0	1	1
0	0	1

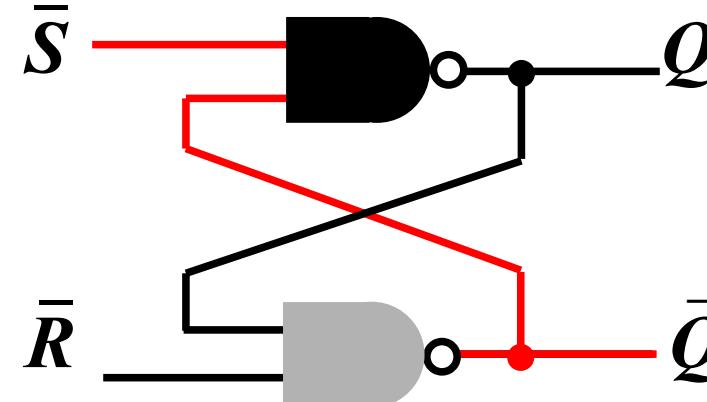


$\bar{S}$	$\bar{R}$	$Q$	$\bar{Q}$	condition
0	0	1	1	Not used
0	1	1	0	Set
1	0	0	1	Reset
1	1	$Q$	$\bar{Q}$	Hold (no change)

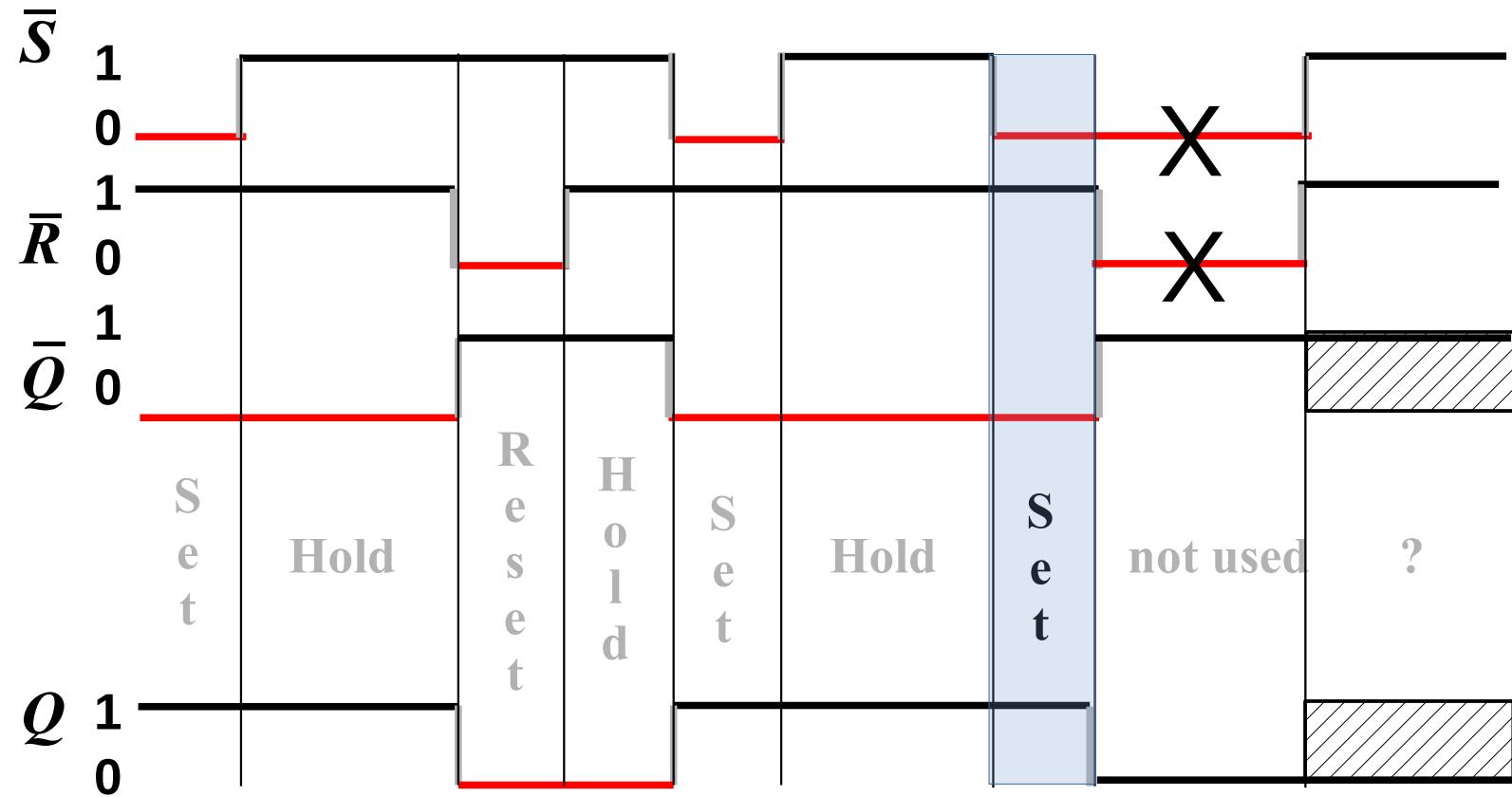


# Cross-NAND SR flip-flop $\bar{S}$

A	B	NAND
1	1	0
1	0	1
0	1	1
0	0	1



$\bar{S}$	$\bar{R}$	$Q$	$\bar{Q}$	condition
0	0	1	1	Not used
0	1	1	0	Set
1	0	0	1	Reset
1	1	$Q$	$\bar{Q}$	Hold (no change)

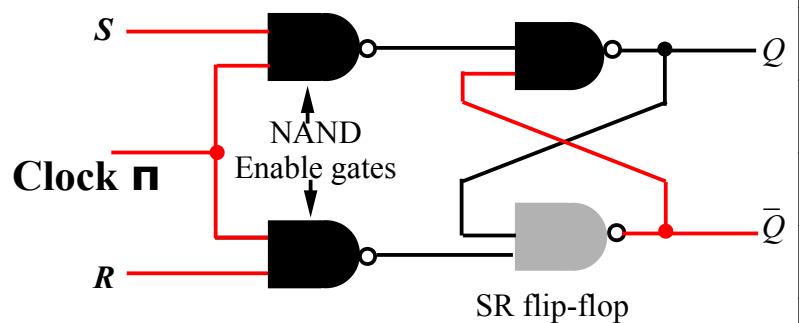


## **Latch SR attivati dal livello del segnale**

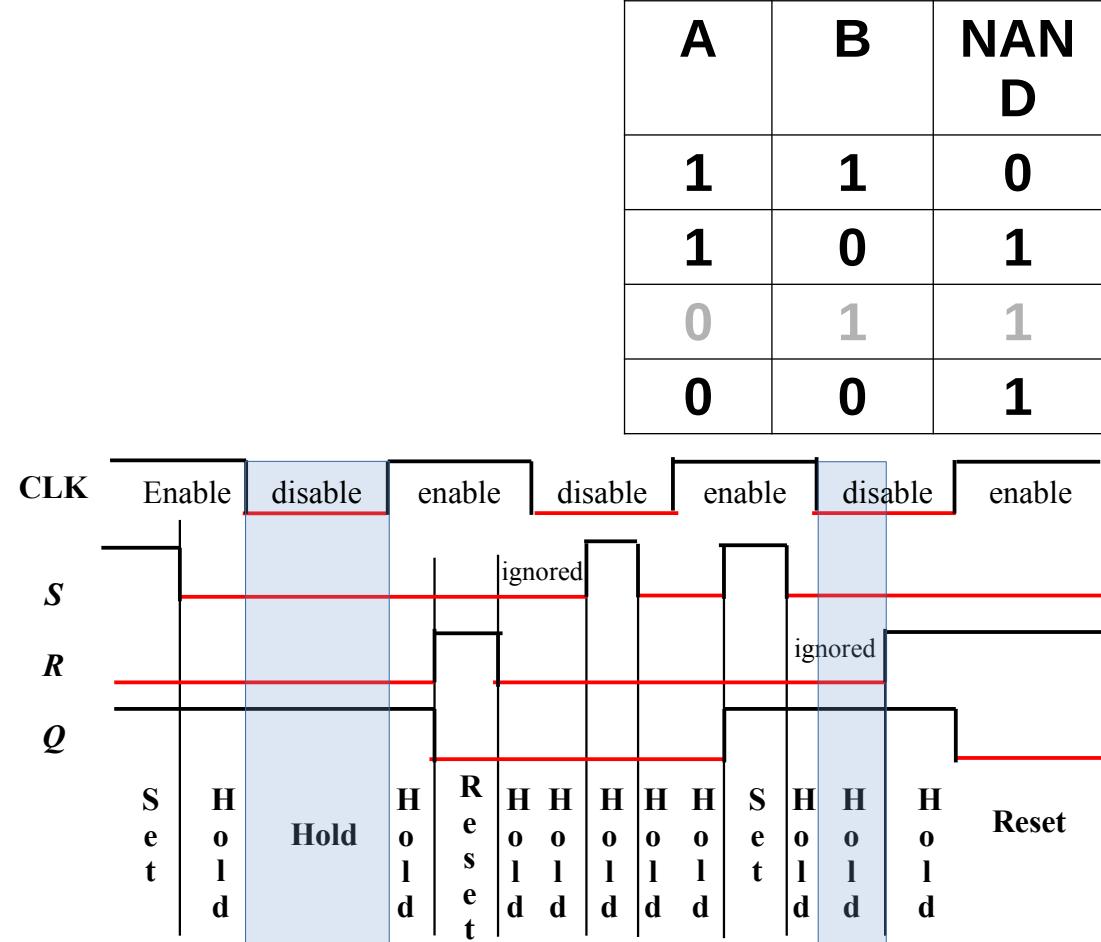
- Gli input S ed R vengono abilitati o disabilitati dal clock
- Gli input vengono campionati, attraverso la lettura dei loro bits, solo quando arriva l'impulso del clock (*flip-flops sincroni*)

# Latch SR attivati dal livello del segnale

- Hold funziona come memoria

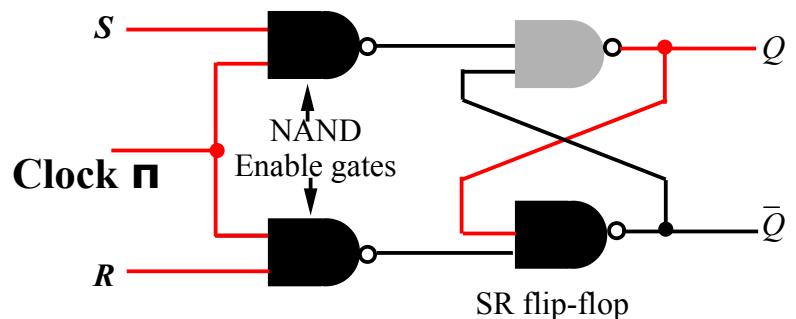


CLK	S	R	Q	$\bar{Q}$	Mode	
0	0	0	Q	$\bar{Q}$	hold	
0	0	1	Q	$\bar{Q}$	hold	SR-inputs disabled
0	1	0	Q	$\bar{Q}$		
0	1	1	Q	$\bar{Q}$		
1	0	0	0	0	hold	
1	0	0	1	0	hold	
1	0	1	0	1	RESET	
1	1	0	1	0	SET	SR-inputs enabled
1	1	1	Q	$\bar{Q}$	intdet.	

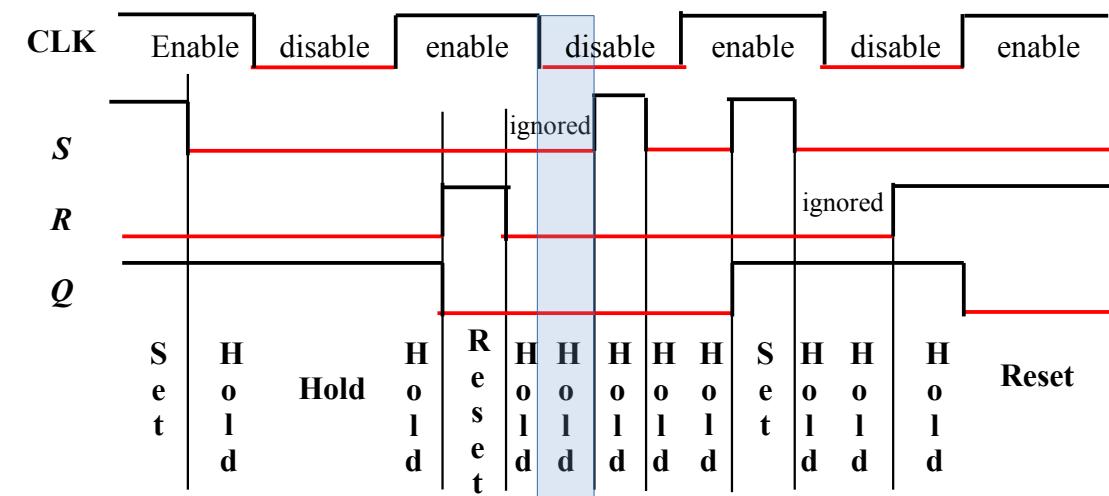


# Latch SR attivati dal livello del segnale

- Hold funziona come memoria



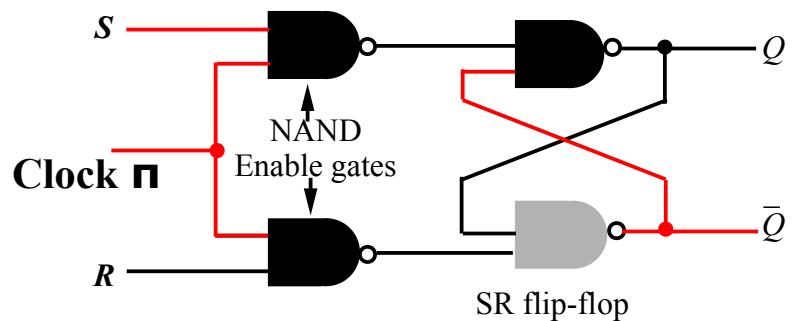
CLK	S	R	Q	$\bar{Q}$	Mode	
0	0	0	Q	$\bar{Q}$	hold	
0	0	1	Q	$\bar{Q}$	hold	SR-inputs disabled
0	1	0	Q	$\bar{Q}$	hold	
0	1	1	Q	$\bar{Q}$	hold	
1	0	0	0	0	hold	
1	0	0	1	0	hold	
1	0	1	0	1	RESET	
1	1	0	1	0	SET	SR-inputs enabled
1	1	1	Q	$\bar{Q}$	intdet.	



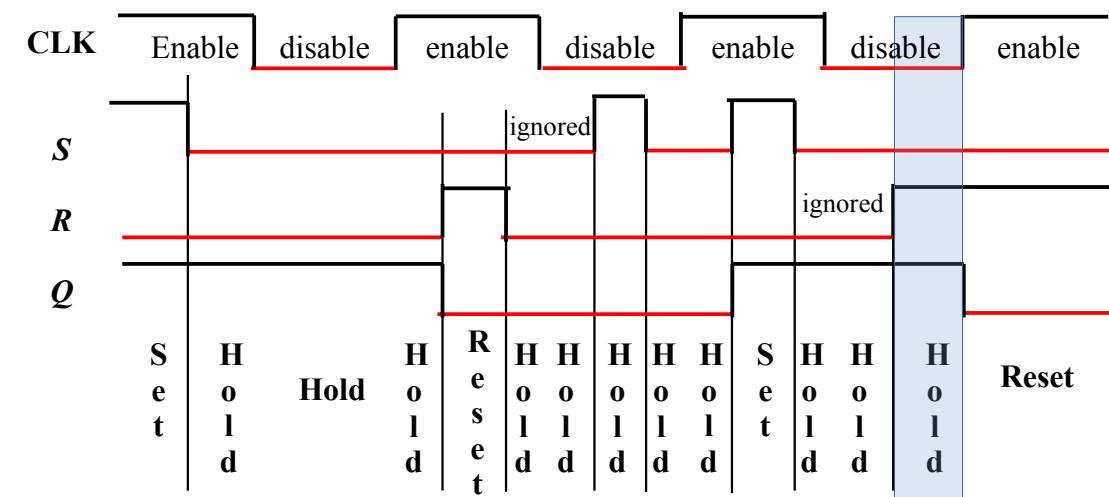
A	B	NAN D
1	1	0
1	0	1
0	1	1
0	0	1

# Latch SR attivati dal livello del segnale

- Hold funziona come memoria



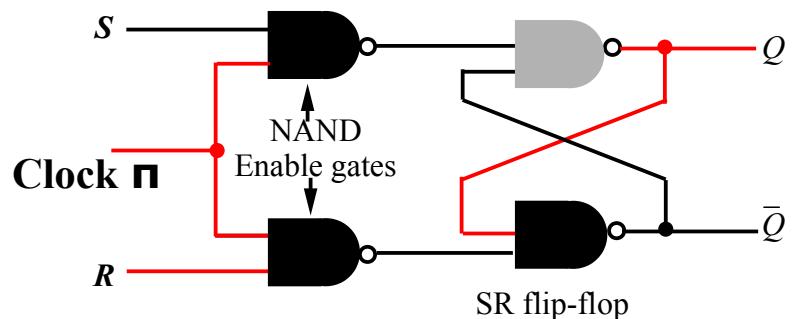
CLK	S	R	Q	$\bar{Q}$	Mode	
0	0	0	Q	$\bar{Q}$	hold	
0	0	1	Q	$\bar{Q}$	hold	SR-inputs disabled
0	1	0	Q	$\bar{Q}$	hold	
0	1	1	Q	$\bar{Q}$	hold	
1	0	0	0	0	hold	
1	0	0	1	0	hold	
1	0	1	0	1	RESET	
1	1	0	1	0	SET	SR-inputs enabled
1	1	1	Q	$\bar{Q}$	intdet.	



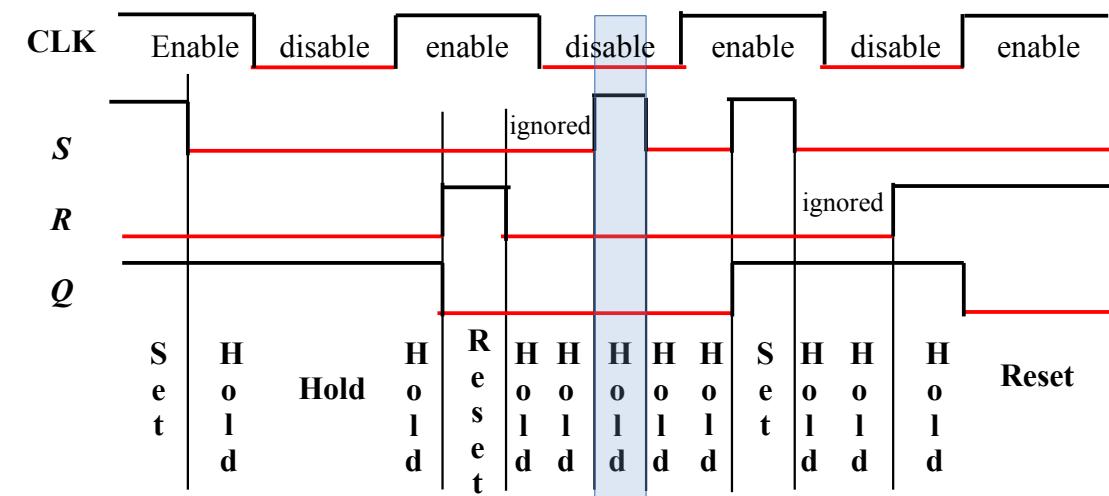
A	B	NAN D
1	1	0
1	0	1
0	1	1
0	0	1

# Latch SR attivati dal livello del segnale

- Hold funziona come memoria



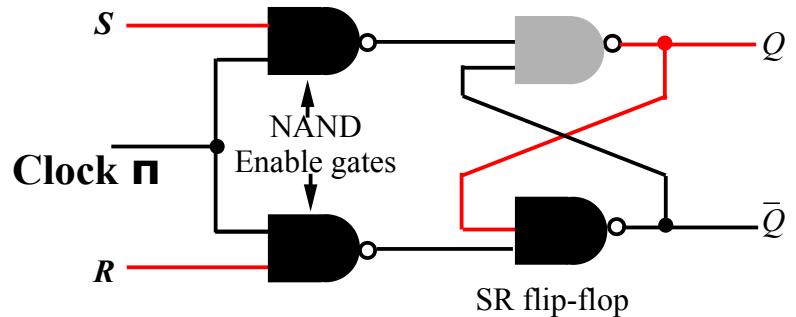
CLK	S	R	Q	$\bar{Q}$	Mode	
0	0	0	Q	$\bar{Q}$	hold	
0	0	1	Q	$\bar{Q}$	hold	
<b>0</b>	<b>1</b>	<b>0</b>	<b>Q</b>	<b><math>\bar{Q}</math></b>	<b>hold</b>	<b>SR- inputs disabled</b>
0	1	1	Q	$\bar{Q}$	hold	
1	0	0	0	0	hold	
1	0	0	1	0	hold	
1	0	1	0	1	RESET	
1	1	0	1	0	SET	<b>SR- inputs enabled</b>
1	1	1	Q	Q	intdet.	



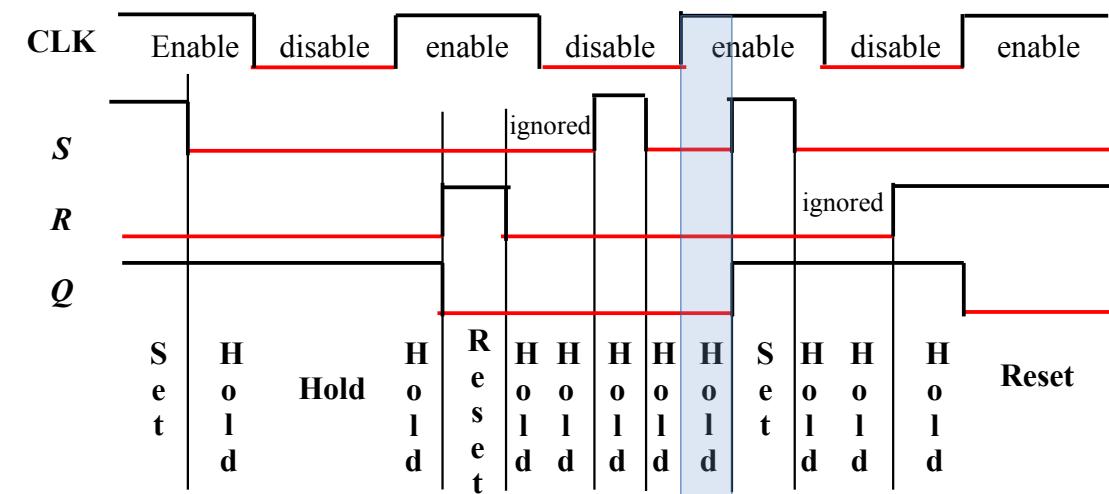
A	B	NAN D
1	1	0
1	0	1
0	1	1
0	0	1

# Latch SR attivati dal livello del segnale

- Hold funziona come memoria



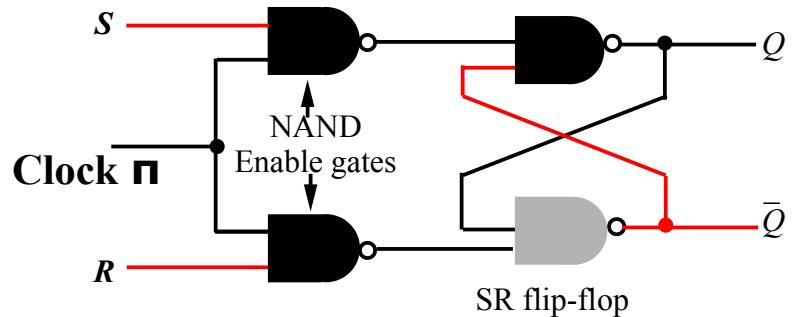
CLK	S	R	Q	$\bar{Q}$	Mode	
0	0	0	Q	$\bar{Q}$	hold	
0	0	1	Q	$\bar{Q}$	hold	SR-inputs disabled
0	1	0	Q	$\bar{Q}$	hold	
0	1	1	Q	$\bar{Q}$	hold	
1	0	0	0	1	hold	
1	0	0	1	0	hold	
1	0	1	0	1	RESET	
1	1	0	1	0	SET	SR-inputs enabled
1	1	1	Q	Q	intdet.	



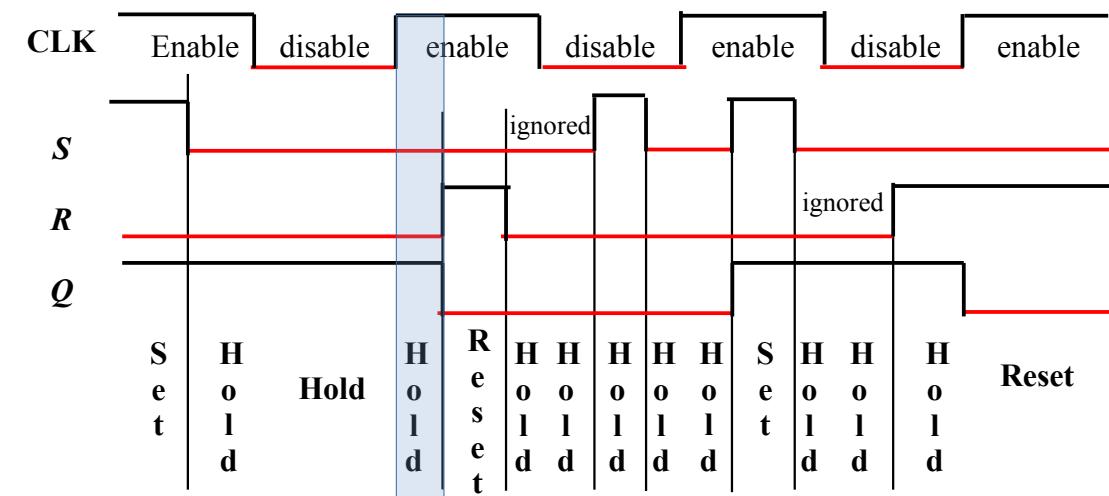
A	B	NAN D
1	1	0
1	0	1
0	1	1
0	0	1

# Latch SR attivati dal livello del segnale

- Hold funziona come memoria



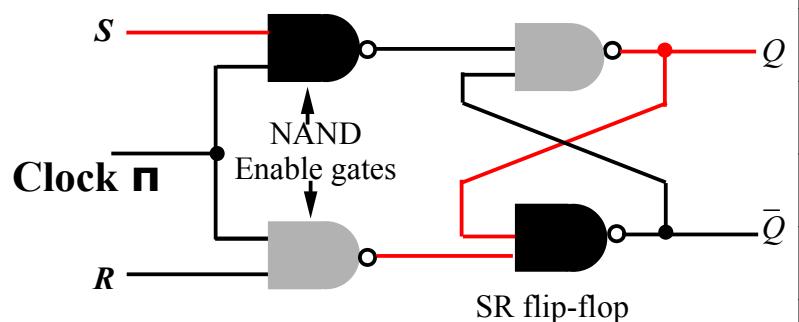
CLK	S	R	Q	$\bar{Q}$	Mode	
0	0	0	Q	$\bar{Q}$	hold	
0	0	1	Q	$\bar{Q}$	hold	SR-inputs disabled
0	1	0	Q	$\bar{Q}$	hold	
0	1	1	Q	$\bar{Q}$	hold	
1	0	0	0	1	hold	
1	0	0	1	0	hold	
1	0	1	0	1	RESET	
1	1	0	1	0	SET	SR-inputs enabled
1	1	1	Q	Q	intdet.	



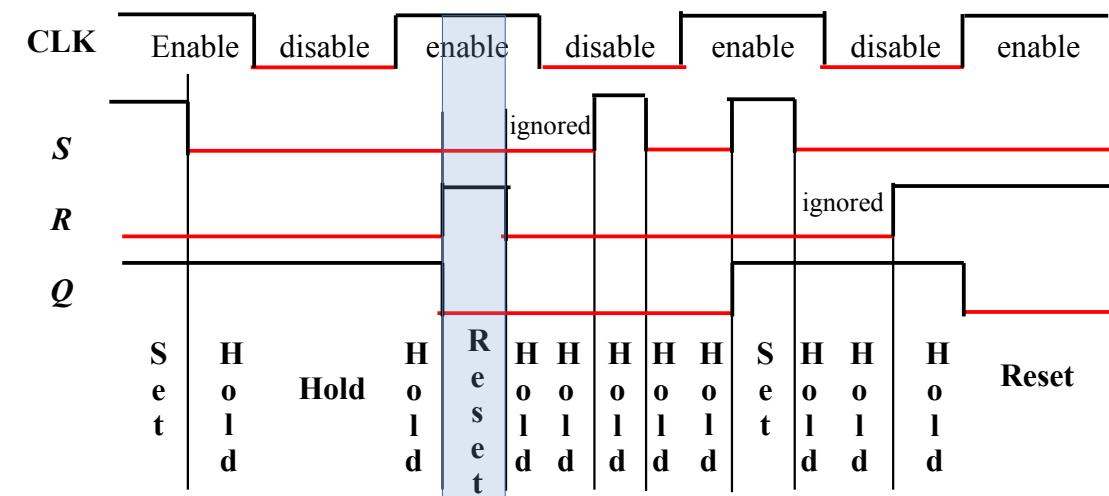
A	B	NAN D
1	1	0
1	0	1
0	1	1
0	0	1

# Latch SR attivati dal livello del segnale

- RESET azzerava un valore



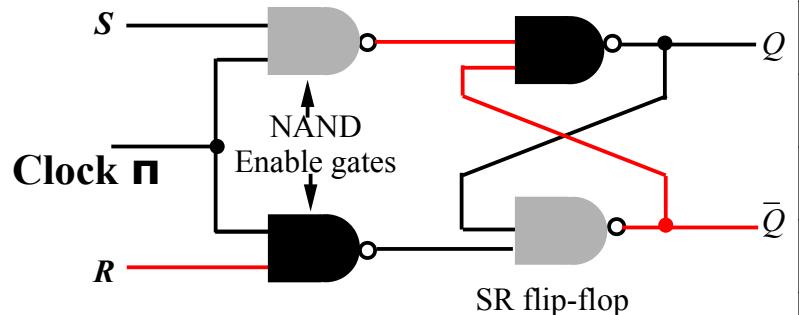
CLK	S	R	Q	$\bar{Q}$	Mode	
0	0	0	Q	$\bar{Q}$	hold	
0	0	1	Q	$\bar{Q}$	hold	SR-inputs disabled
0	1	0	Q	$\bar{Q}$	hold	
0	1	1	Q	$\bar{Q}$	hold	
1	0	0	0	1	hold	
1	0	0	1	0	hold	
<b>1</b>	<b>0</b>	<b>1</b>	<b>0</b>	<b>1</b>	<b>RESET</b>	SR-inputs enabled
1	1	0	1	0	SET	
1	1	1	Q	Q	intdet.	



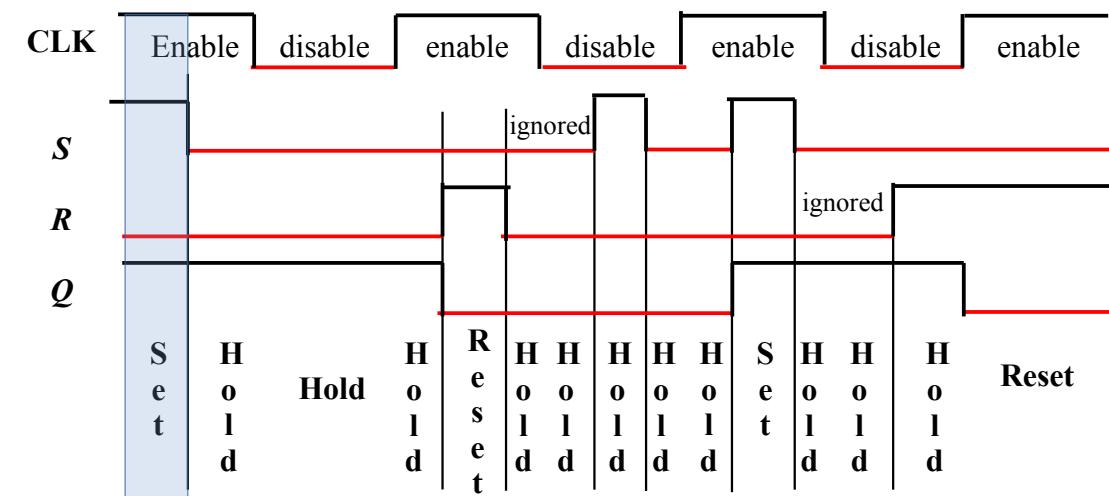
A	B	NAN D
<b>1</b>	<b>1</b>	<b>0</b>
<b>1</b>	<b>0</b>	<b>1</b>
<b>0</b>	<b>1</b>	<b>1</b>
<b>0</b>	<b>0</b>	<b>1</b>

# Latch SR attivati dal livello del segnale

- SET imposta un valore



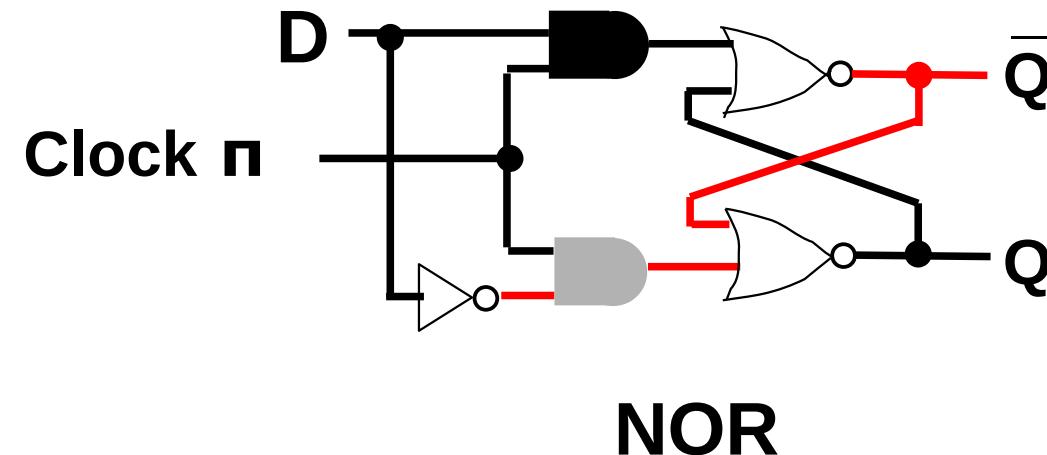
CLK	S	R	Q	$\bar{Q}$	Mode	
0	0	0	Q	$\bar{Q}$	hold	SR-inputs disabled
0	0	1	Q	$\bar{Q}$		
0	1	0	Q	$\bar{Q}$		
0	1	1	Q	$\bar{Q}$	hold	SR- inputs enabled
1	0	0	0	1		
1	0	0	1	0		
1	0	1	0	1	RESET	SET
1	1	0	1	$\bar{0}$		
1	1	1	Q	Q	intdet.	



A	B	NAN D
1	1	0
1	0	1
0	1	1
0	0	1

# Latch D temporizzato

- Un solo input (D)
- L'input della porta AND inferiore è il complemento di quello della porta AND superiore
- Non è possibile che entrambi valgano 1

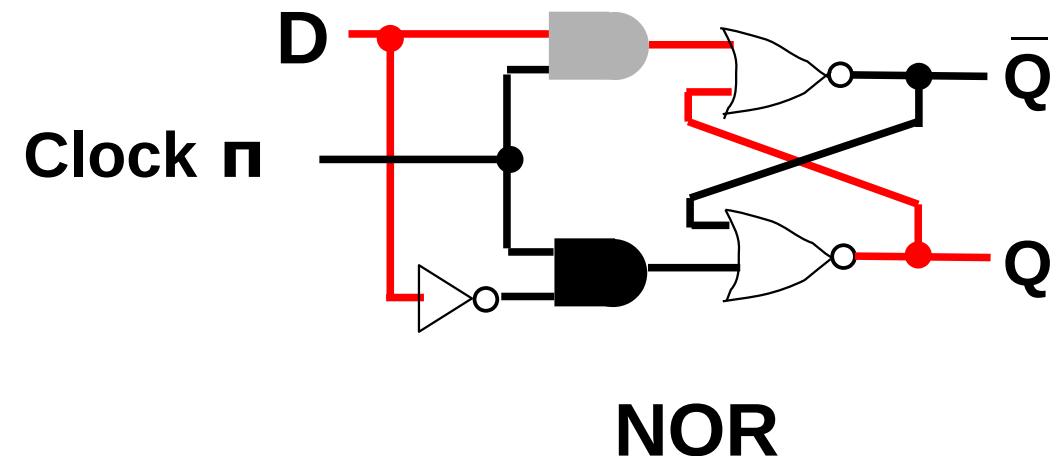


Clock	D	Q	$\bar{Q}$
1	1	1	0
1	0	0	1

A	B	X
1	1	0
1	0	0
0	1	0
0	0	1

# Latch D temporizzato

- Quando il clock vale 1 il valore corrente di D viene campionato e memorizzato nel latch, ossia Q assume lo stesso valore di D → memoria a 1 bit



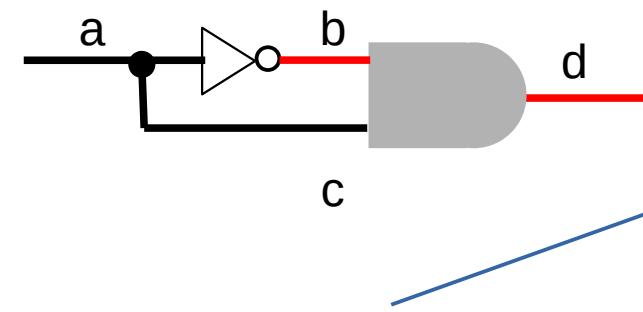
Clock	D	Q	$\bar{Q}$
1	1	1	0
1	0	0	1

A	B	X
1	1	0
1	0	0
0	1	0
0	0	1

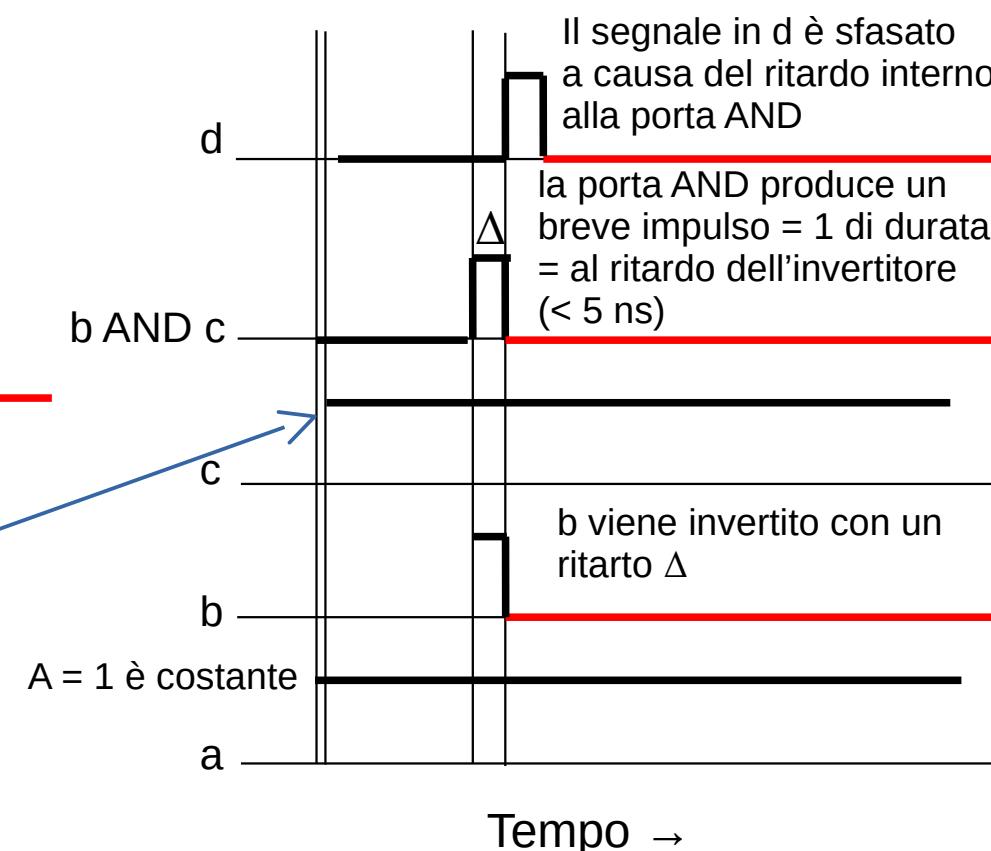
# Flip-flop

- Circuiti in cui il cambiamento di stato non avviene quando il clock vale 1 ma durante la transizione del clock da 0 a 1 (fronte di salita) oppure da 1 a 0 (fronte di discesa)

- In molti circuiti, infatti, è necessario campionare (e memorizzare) il valore di una certa linea in un istante preciso

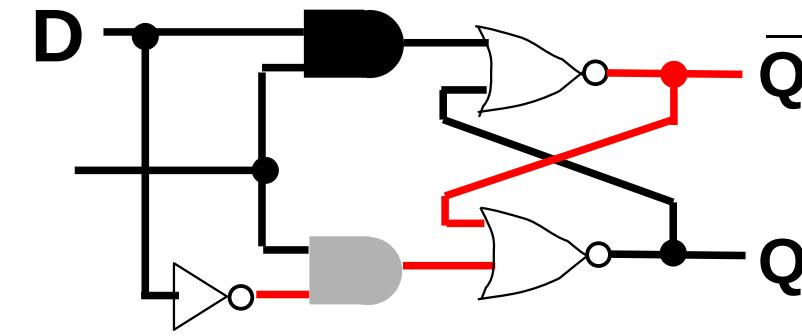
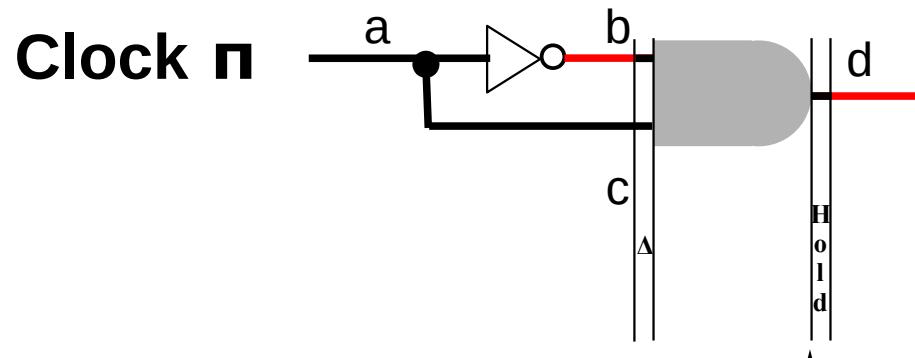


$C = A = 1$ , costante, ha un piccolo ritardo dovuto alla velocità di propagazione del segnale ( $2 \cdot c / 3$ )



# Flip-flop D

- Per  $< 5 \text{ ns}$   $b = 1$  a causa del ritardo  $\Delta$  dell'invertitore ( $d$  passa da 0 a 1)



NOR

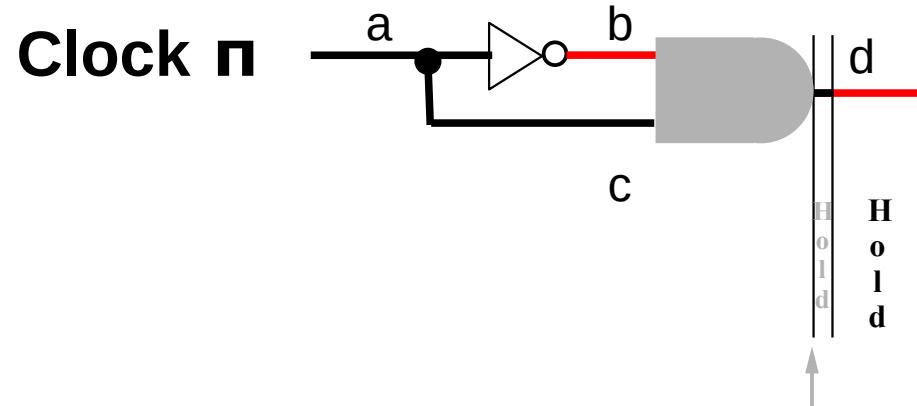
Clock	d	D	Q	$\bar{Q}$
1	1	1	1	0
1	0	1	1	0
0	0	1	1	0
1	1	1	1	0
1	1	0	0	1

Cambio di stato

A	B	X
1	1	0
1	0	0
0	1	0
0	0	1

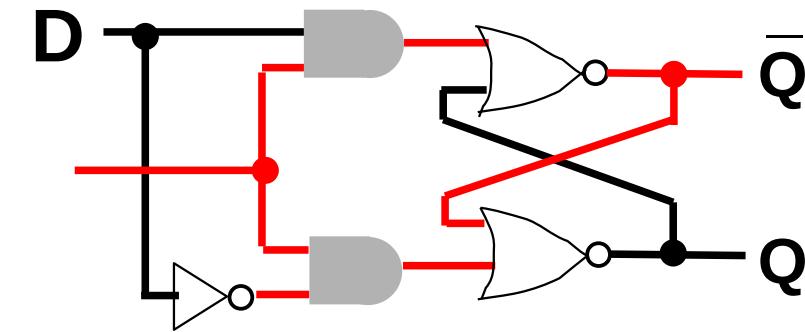
# Flip-flop D

- L'invertitore inverte il segnale **a** e **b** diventa = 0  
(d passa da 1 a 0)



Clock	d	D	Q	$\bar{Q}$
1	1	1	1	0
1	0	1	1	0
0	0	1	1	0
1	1	1	1	0
1	1	0	0	1

Cambio di stato

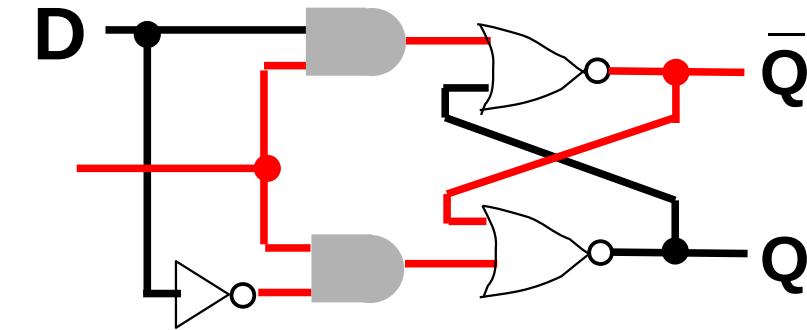
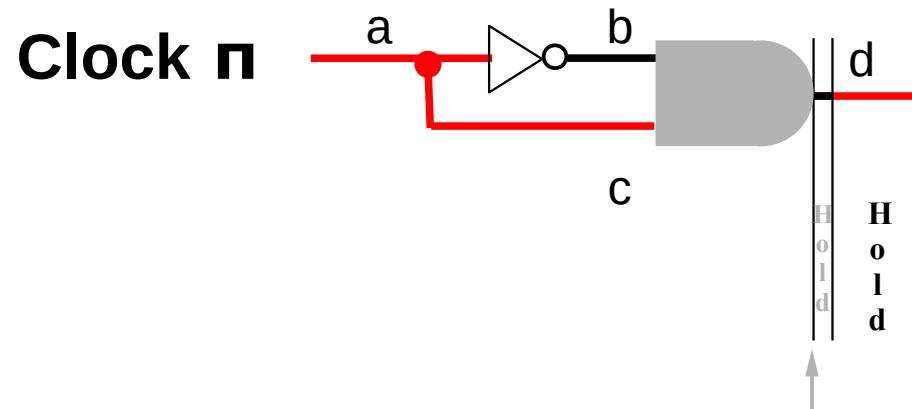


NOR

A	B	X
1	1	0
1	0	0
0	1	0
0	0	1

# Flip-flop D

- Per  $< 5$  ns  $b = 0$  a causa del ritardo dell'invertitore ma ciò è irrilevante



**NOR**

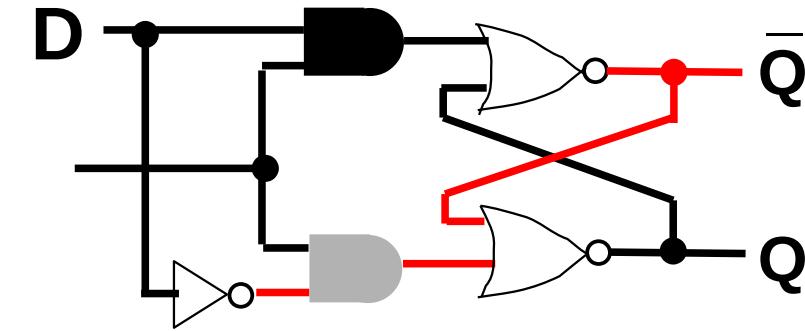
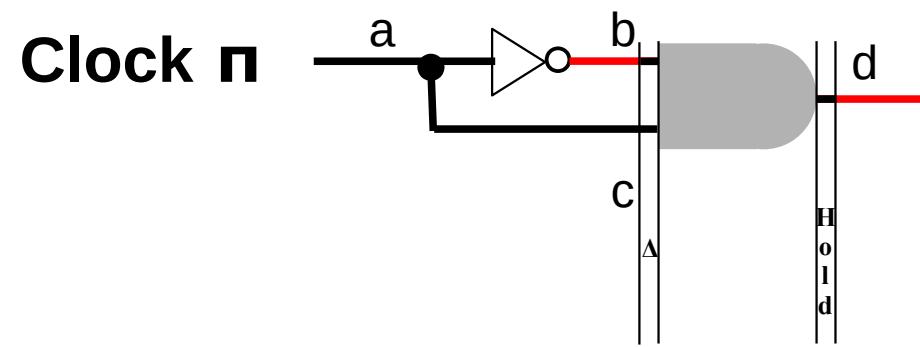
<b>Clock</b>	<b>d</b>	<b>D</b>	<b>Q</b>	<b><math>\bar{Q}</math></b>
1	1	1	1	0
1	0	1	1	0
0	0	1	1	0
1	1	1	1	0
1	1	0	0	1

Cambio di stato

<b>A</b>	<b>B</b>	<b>X</b>
1	1	0
1	0	0
0	1	0
0	0	1

# Flip-flop D

- Quando il clock passa da 0 a 1 il ciclo ricomincia



**NOR**

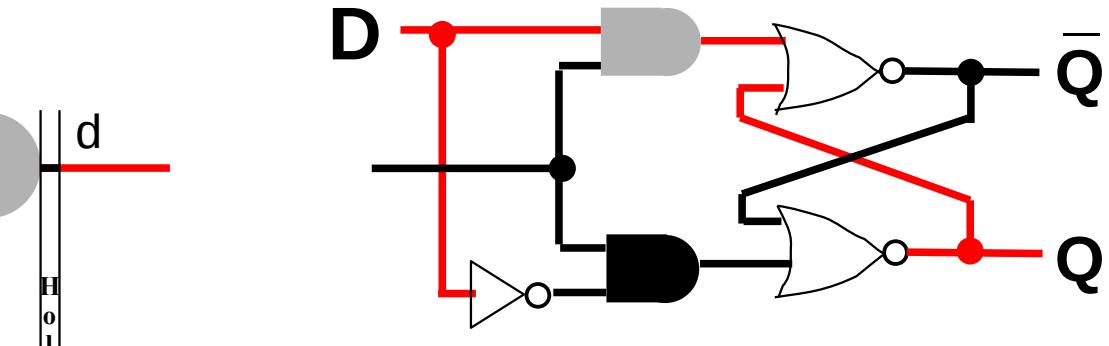
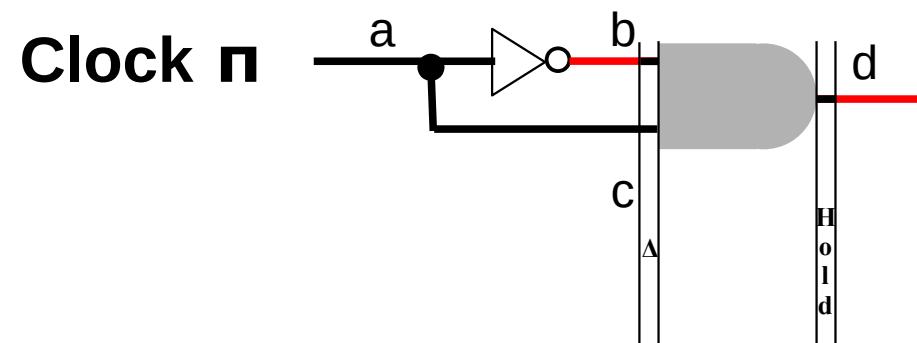
<b>Clock</b>	<b>d</b>	<b>D</b>	<b>Q</b>	<b><math>\bar{Q}</math></b>
1	1	1	1	0
1	0	1	1	0
0	0	1	1	0
1	1	1	1	0
1	1	0	0	1

Cambio di stato

<b>A</b>	<b>B</b>	<b>X</b>
<b>1</b>	<b>1</b>	<b>0</b>
1	0	0
0	1	0
0	0	1

# Flip-flop D

- Quando il clock passa da 0 a 1 il ciclo ricomincia



NOR

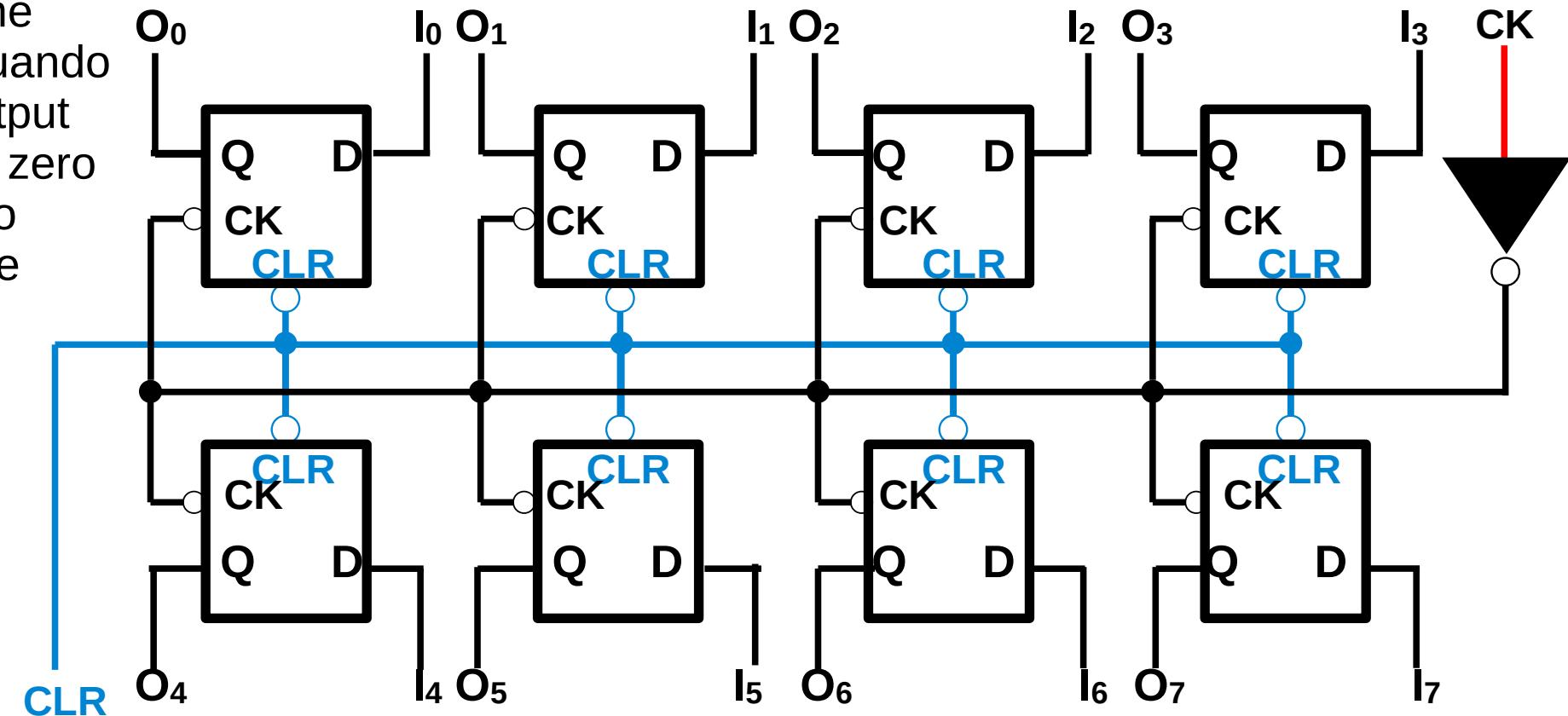
Clock	d	D	Q	$\bar{Q}$
1	1	1	1	0
1	0	1	1	0
0	0	1	1	0
1	1	1	1	0
1	1	0	0	1

Cambio di stato

A	B	X
1	1	0
1	0	0
0	1	0
0	0	1

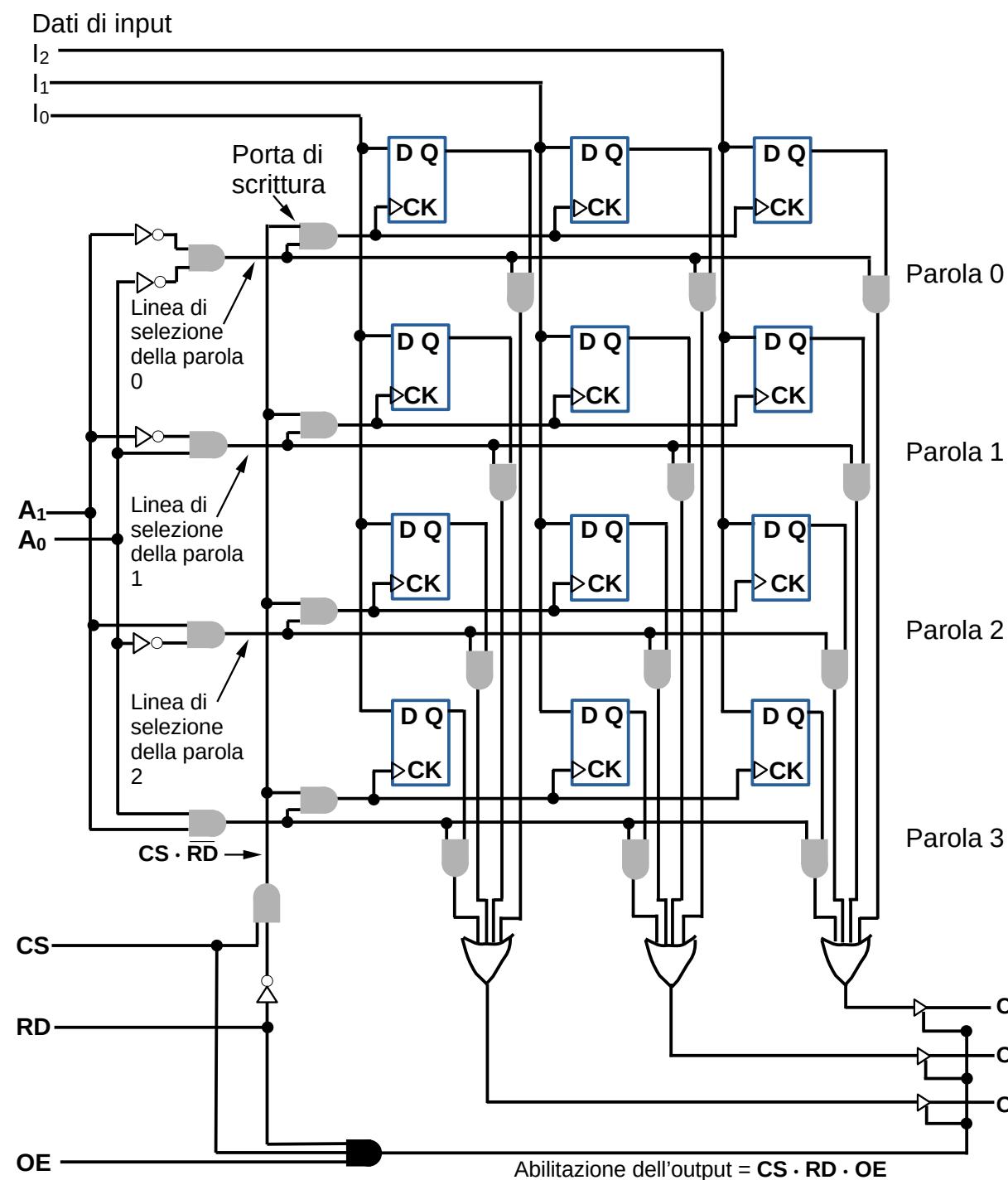
# Registri da 8 bit

- I flip-flops possono essere combinati per creare dei registri che memorizzano dati composti da più bit
- CLR** = clear (linea di cancellazione); **CK** = clock
- Segnale in ingresso di 8 bit quando si verifica una transizione del clock  
(tutte le linee di clock sono collegate allo stesso segnale in ingresso di **CK**)
- Tutti i canali di cancellazione sono collegati fra loro → quando **CLR** va a zero tutti gli 8 output assumono lo stesso valore zero
- Più registri da 8 bit possono essere combinati per creare registri da 16, 32 o 64 bit



# Memoria 4 x 3

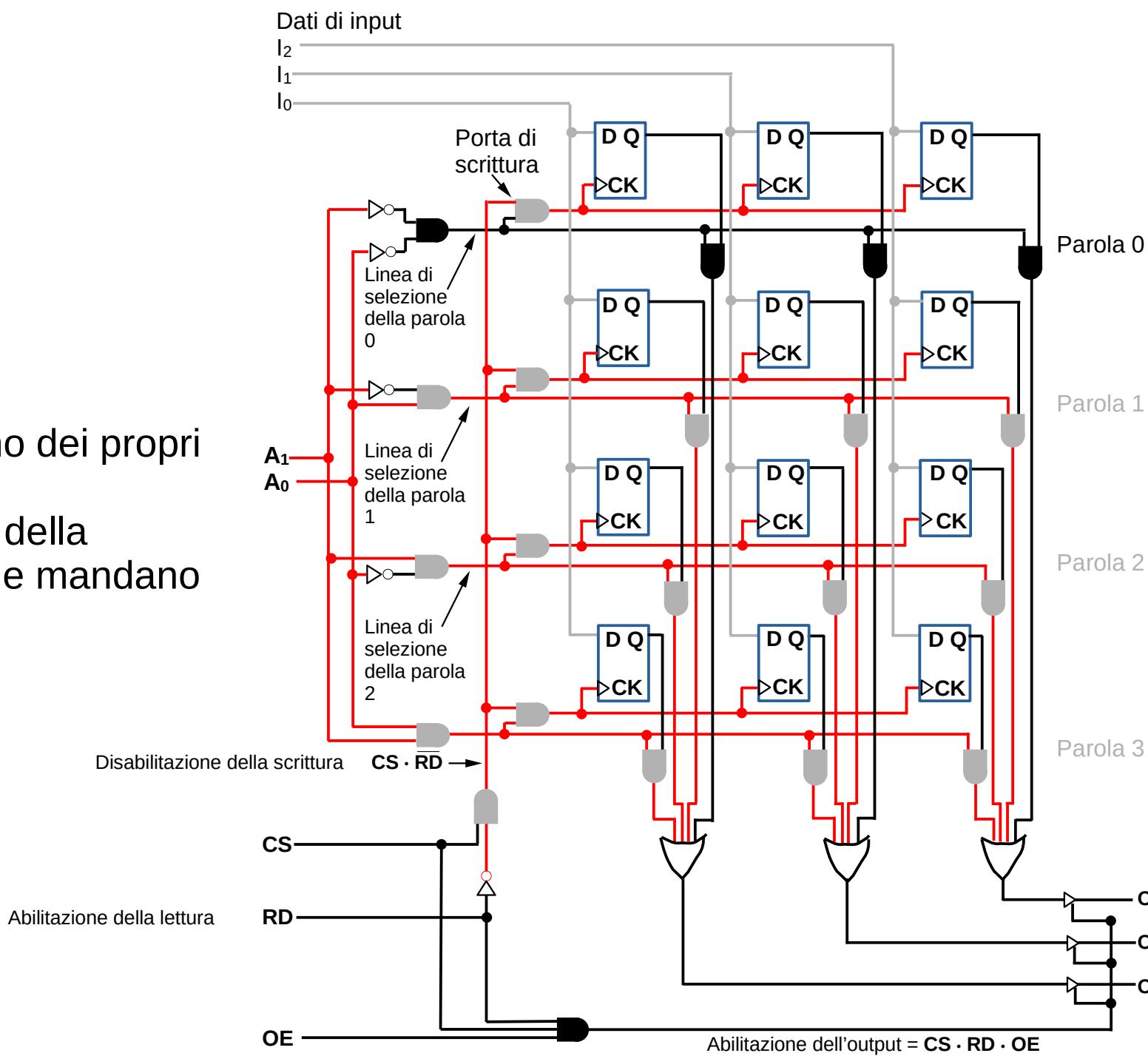
- Memoria con 4 parole a 3 bit
- Ogni operazione legge o scrive un'intera parola
- 3 linee di input per i dati:  $I_0$ ,  $I_1$ , e  $I_2$
- 2 linee di input per l'indirizzo:  $A_0$  e  $A_1$
- 3 linee di input per i controlli:
  - **CS** per la selezione del chip
  - **RD** per distinguere fra read e write
  - **OE** per l'abilitazione dell'output (**Output Enable**)
- 3 linee di output per i dati:  $O_0$ ,  $O_1$ , e  $O_2$
- Decodificatore formato dalle 4 porte AND a sinistra per selezionare la parola
  - 4 invertitori collocati per far sì che ciascuna porta AND sia abilitata da una diversa combinazione degli indirizzi  $A_0$  e  $A_1$
  - Ogni porta è collegata ad una linea di selezione della parola corrispondente



# Memoria 4 x 3

Esempio: lettura parola 0 = 111

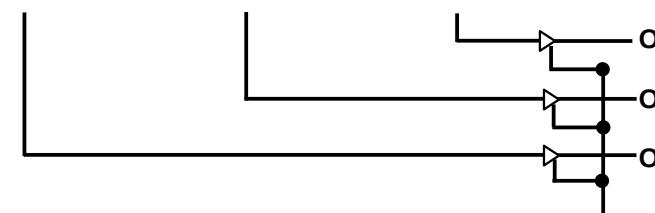
- **CS** = 1 per selezionare il chip
- **RD** = 1 → read
- **OE** = 1 → abilitazione dell'output
- Impostazione dell'indirizzo
- No input: **I<sub>0</sub>**, **I<sub>1</sub>**, e **I<sub>2</sub>** non utilizzate
- La linea **CS • RD** assume valore basso disabilitando tutte le porte di scrittura
- La parola selezionata spedisce ciascuno dei propri 3 bit ad una porta OR
- L'output delle porte OR è identico ai bit della parola selezionata, poiché le altre parole mandano in input alle porte OR valori uguali a 0



# Memoria 4 x 3

**Buffer non invertente:** interruttore elettronico che, in una frazione di ns, collega le porte OR alle linee di output durante le operazioni di lettura e le disconnette durante le scritture

- Dato di input
- Dato di output
- Linea di controllo: **CS • RD • OE**
  - Input di controllo alto: collegamento
  - Input di controllo basso: circuito aperto (le porte OR vengono scollegate dalle linee di output)

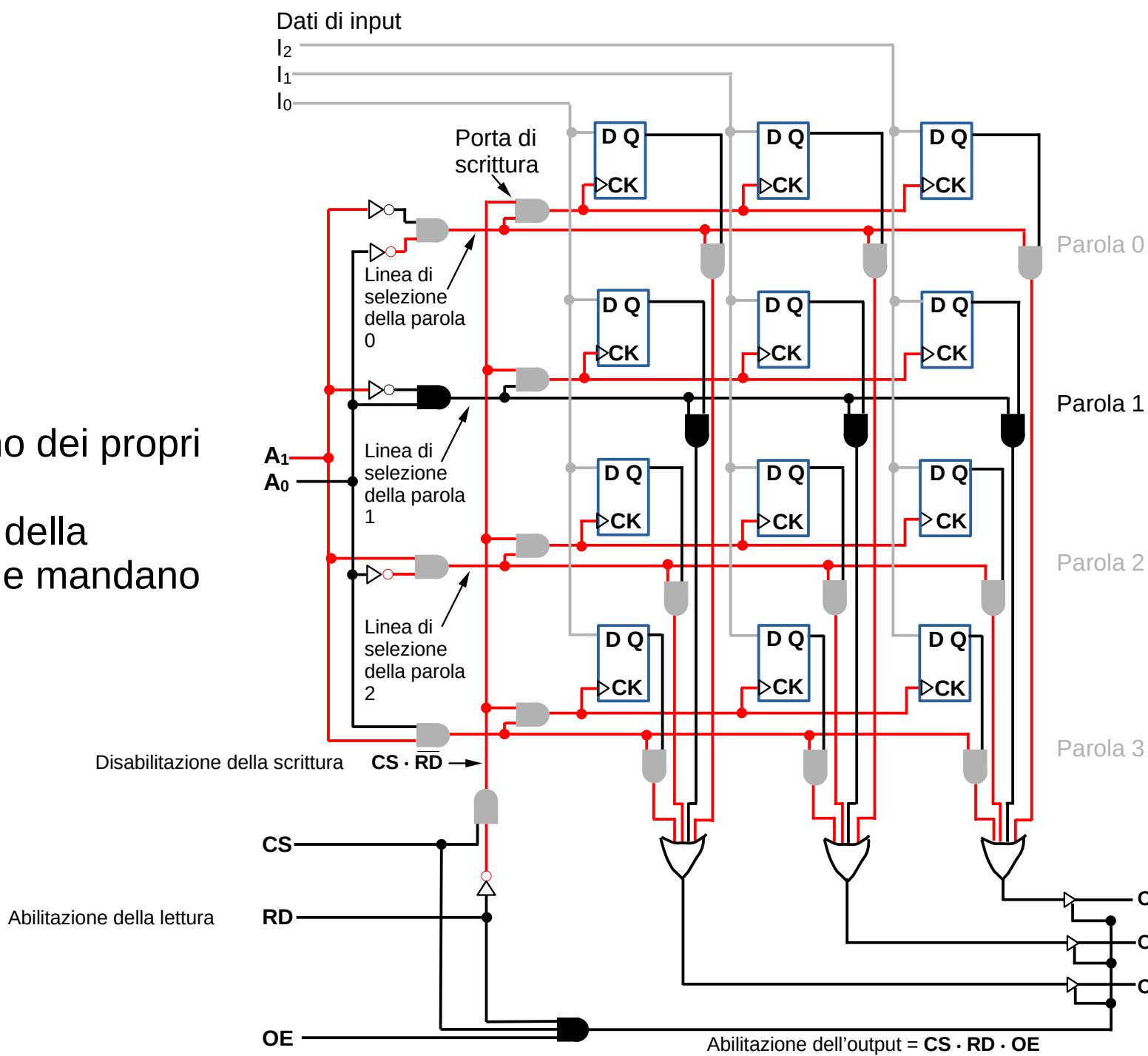


Abilitazione dell'output = **CS • RD • OE**

# Memoria 4 x 3

Esempio: lettura parola 1 = 111

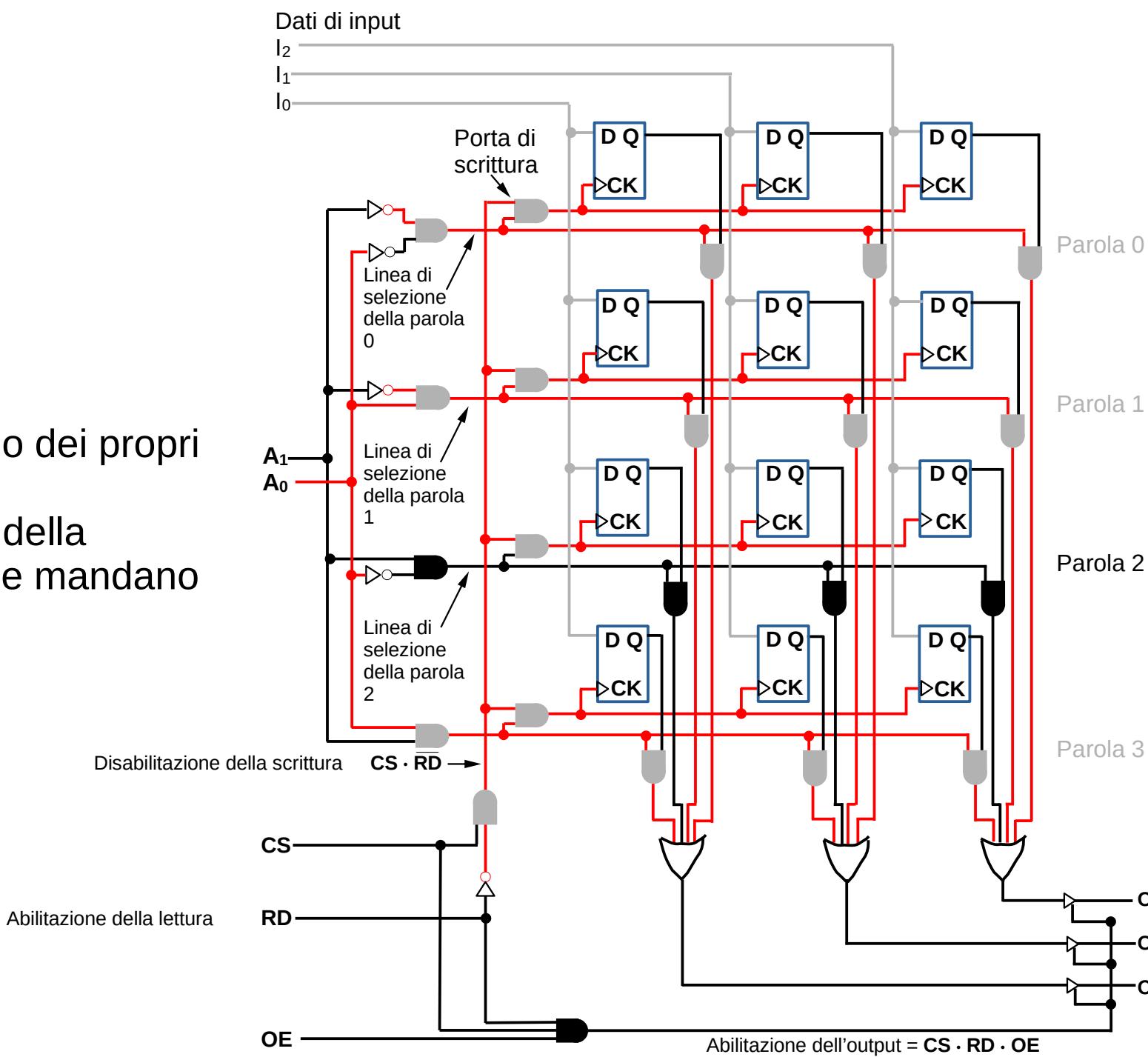
- **CS** = 1 per selezionare il chip
- **RD** = 1 → read
- **OE** = 1 → abilitazione dell'output
- Impostazione dell'indirizzo
- No input: **I<sub>0</sub>**, **I<sub>1</sub>**, e **I<sub>2</sub>** non utilizzate
- La linea **CS • RD** assume valore basso disabilitando tutte le porte di scrittura
- La parola selezionata spedisce ciascuno dei propri 3 bit ad una porta OR
- L'output delle porte OR è identico ai bit della parola selezionata, poiché le altre parole mandano in input alle porte OR valori uguali a 0



# Memoria 4 x 3

Esempio: lettura parola 2 = 111

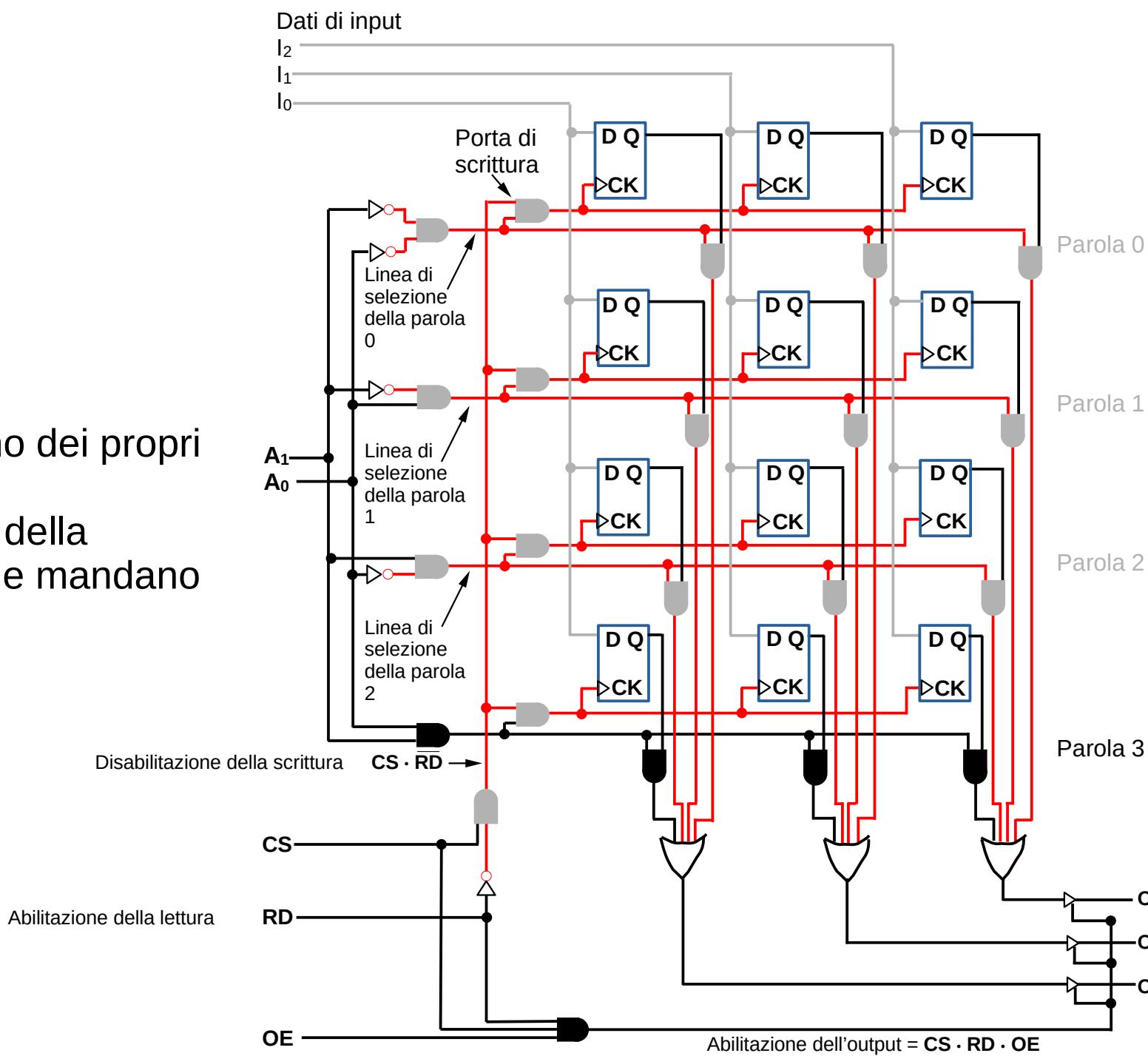
- **CS** = 1 per selezionare il chip
- **RD** = 1 → read
- **OE** = 1 → abilitazione dell'output
- Impostazione dell'indirizzo
- No input: **I<sub>0</sub>**, **I<sub>1</sub>**, e **I<sub>2</sub>** non utilizzate
- La linea **CS • RD** assume valore basso disabilitando tutte le porte di scrittura
- La parola selezionata spedisce ciascuno dei propri 3 bit ad una porta OR
- L'output delle porte OR è identico ai bit della parola selezionata, poiché le altre parole mandano in input alle porte OR valori uguali a 0



# Memoria 4 x 3

Esempio: lettura parola 3 = 111

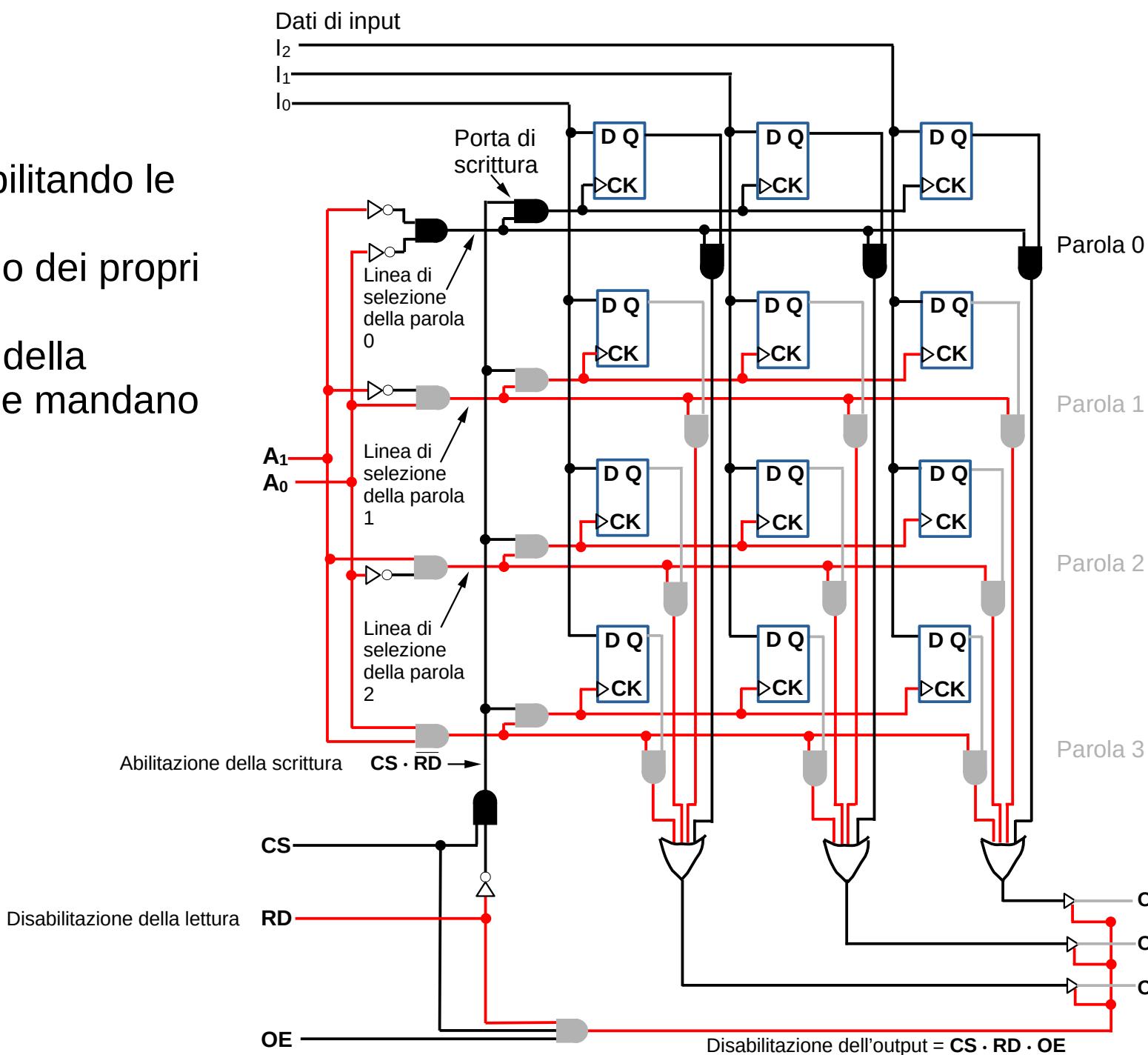
- **CS** = 1 per selezionare il chip
- **RD** = 1 → read
- **OE** = 1 → abilitazione dell'output
- Impostazione dell'indirizzo
- No input: **I<sub>0</sub>**, **I<sub>1</sub>**, e **I<sub>2</sub>** non utilizzate
- La linea **CS • RD** assume valore basso disabilitando tutte le porte di scrittura
- La parola selezionata spedisce ciascuno dei propri 3 bit ad una porta OR
- L'output delle porte OR è identico ai bit della parola selezionata, poiché le altre parole mandano in input alle porte OR valori uguali a 0



# Memoria 4 x 3

Esempio: scrittura 111 nella parola 0

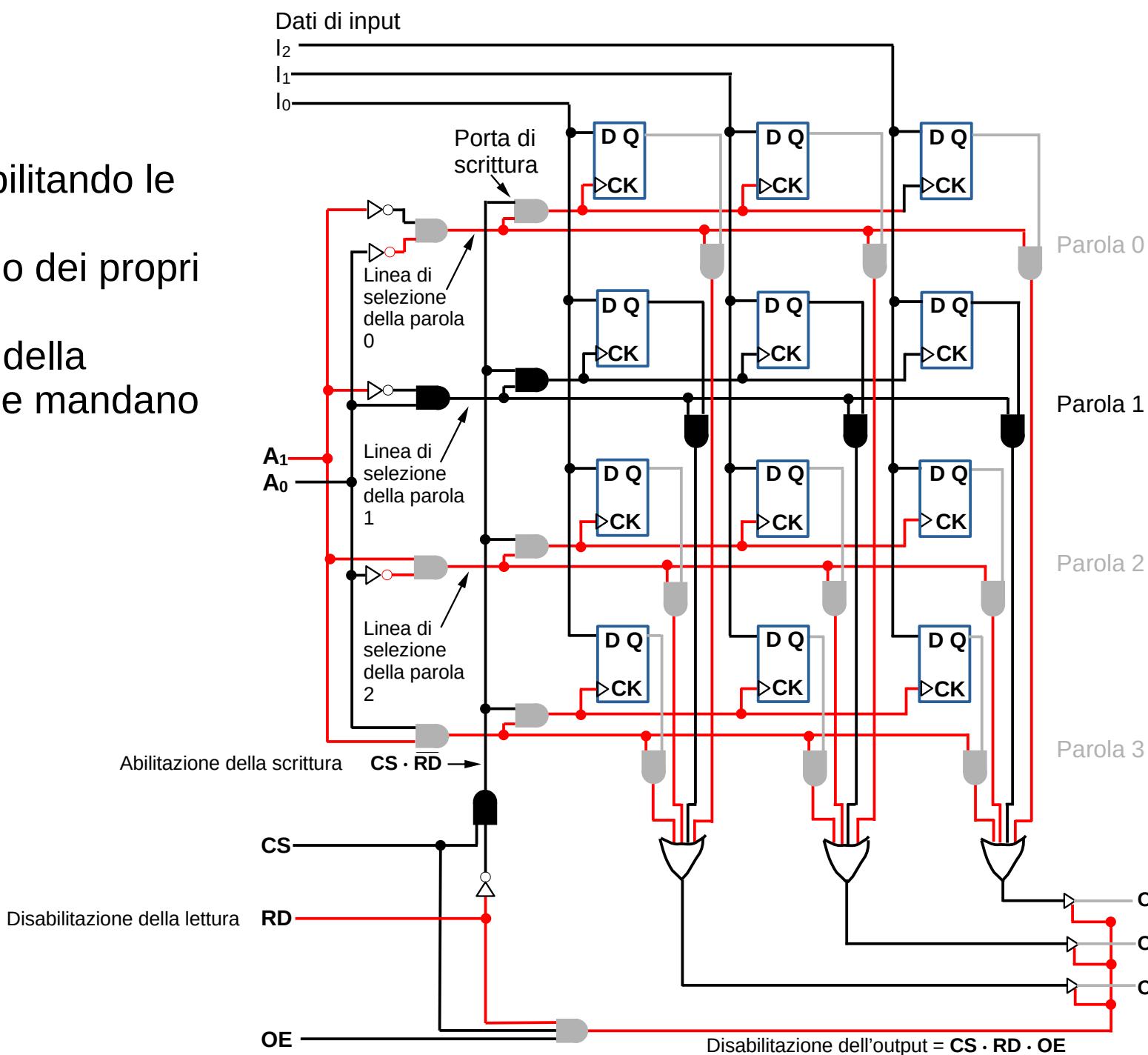
- $RD = 0 \rightarrow$  don't read
- La linea  $CS \cdot RD$  assume valore alto abilitando le porte di scrittura
- La parola selezionata spedisce ciascuno dei propri 3 bit ad una porta OR
- L'output delle porte OR è identico ai bit della parola selezionata, poiché le altre parole mandano in input alle porte OR valori uguali a 0



# Memoria 4 x 3

Esempio: scrittura 111 nella parola 1

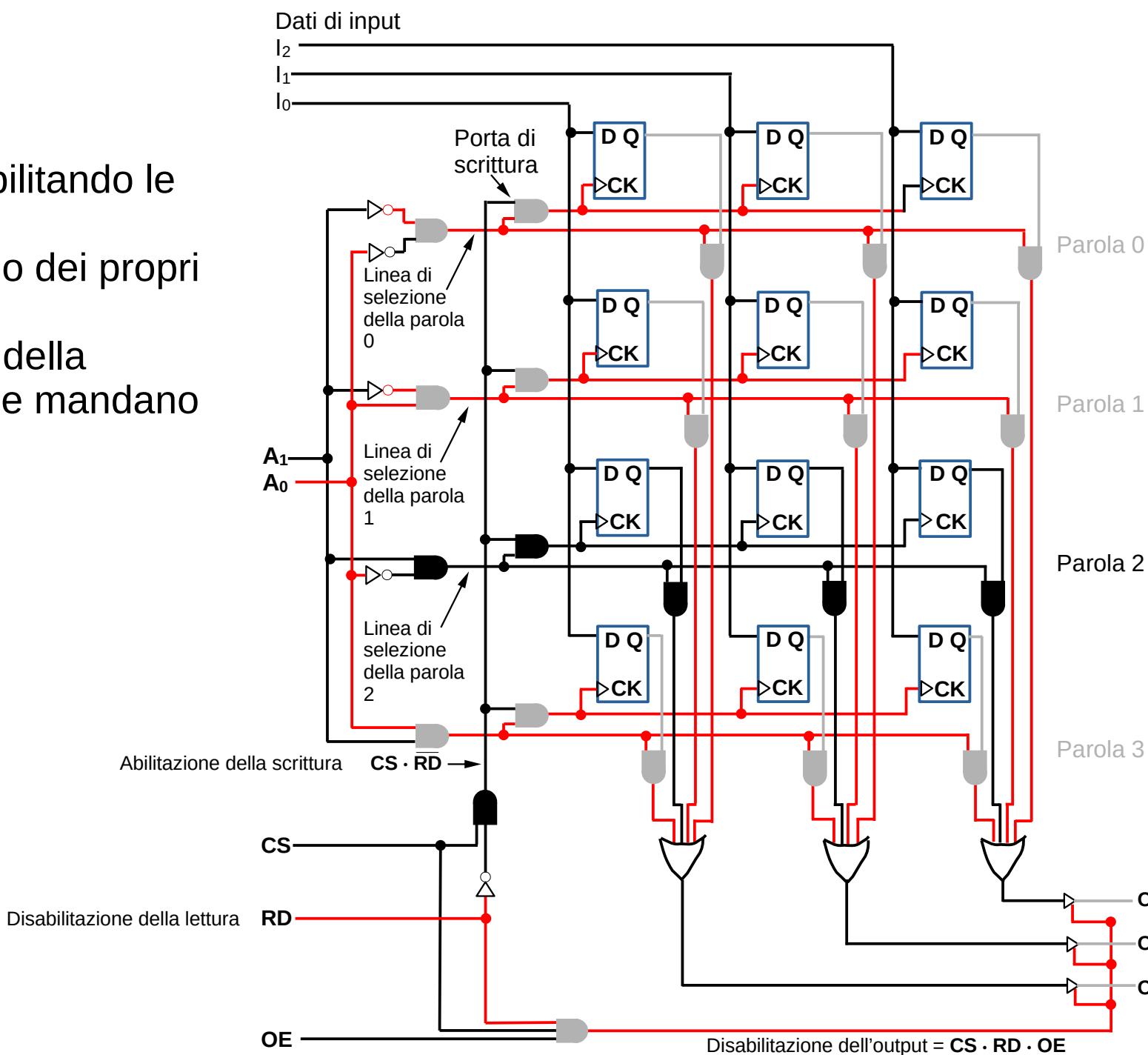
- $RD = 0 \rightarrow$  don't read
- La linea  $CS \cdot RD$  assume valore alto abilitando le porte di scrittura
- La parola selezionata spedisce ciascuno dei propri 3 bit ad una porta OR
- L'output delle porte OR è identico ai bit della parola selezionata, poiché le altre parole mandano in input alle porte OR valori uguali a 0



# Memoria 4 x 3

Esempio: scrittura 111 nella parola 2

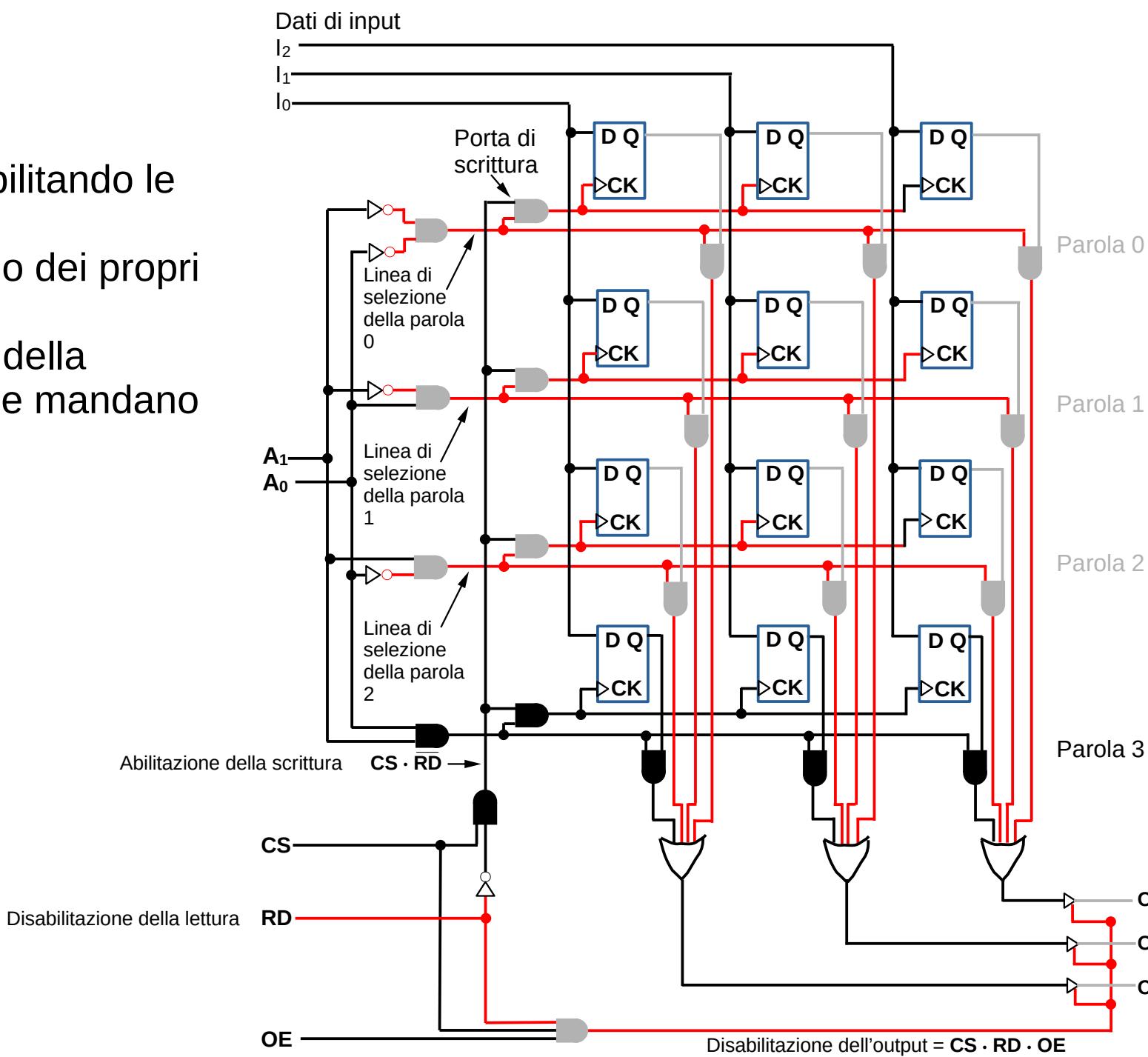
- $RD = 0 \rightarrow$  don't read
- La linea  $CS \cdot RD$  assume valore alto abilitando le porte di scrittura
- La parola selezionata spedisce ciascuno dei propri 3 bit ad una porta OR
- L'output delle porte OR è identico ai bit della parola selezionata, poiché le altre parole mandano in input alle porte OR valori uguali a 0



# Memoria 4 x 3

Esempio: scrittura 111 nella parola 3

- $RD = 0 \rightarrow$  don't read
- La linea  $CS \cdot RD$  assume valore alto abilitando le porte di scrittura
- La parola selezionata spedisce ciascuno dei propri 3 bit ad una porta OR
- L'output delle porte OR è identico ai bit della parola selezionata, poiché le altre parole mandano in input alle porte OR valori uguali a 0



**Memoria: 1 bit alla volta  
64 celle, matrice 8x8**

**Ogni cella: 1 oppure 0**

3 linee per l'indirizzo sull'asse Y

Ogni linea: 1 bit

Es.: 101

$$1*2^0+0*2^1+1*2^2=1+0+4=\textcolor{red}{5}$$

3 linee per l'indirizzo sull'asse X

Ogni linea: 1 bit

Es.: 001

$$0*2^0+0*2^1+1*2^2=0+0+4=\textcolor{blue}{4}$$

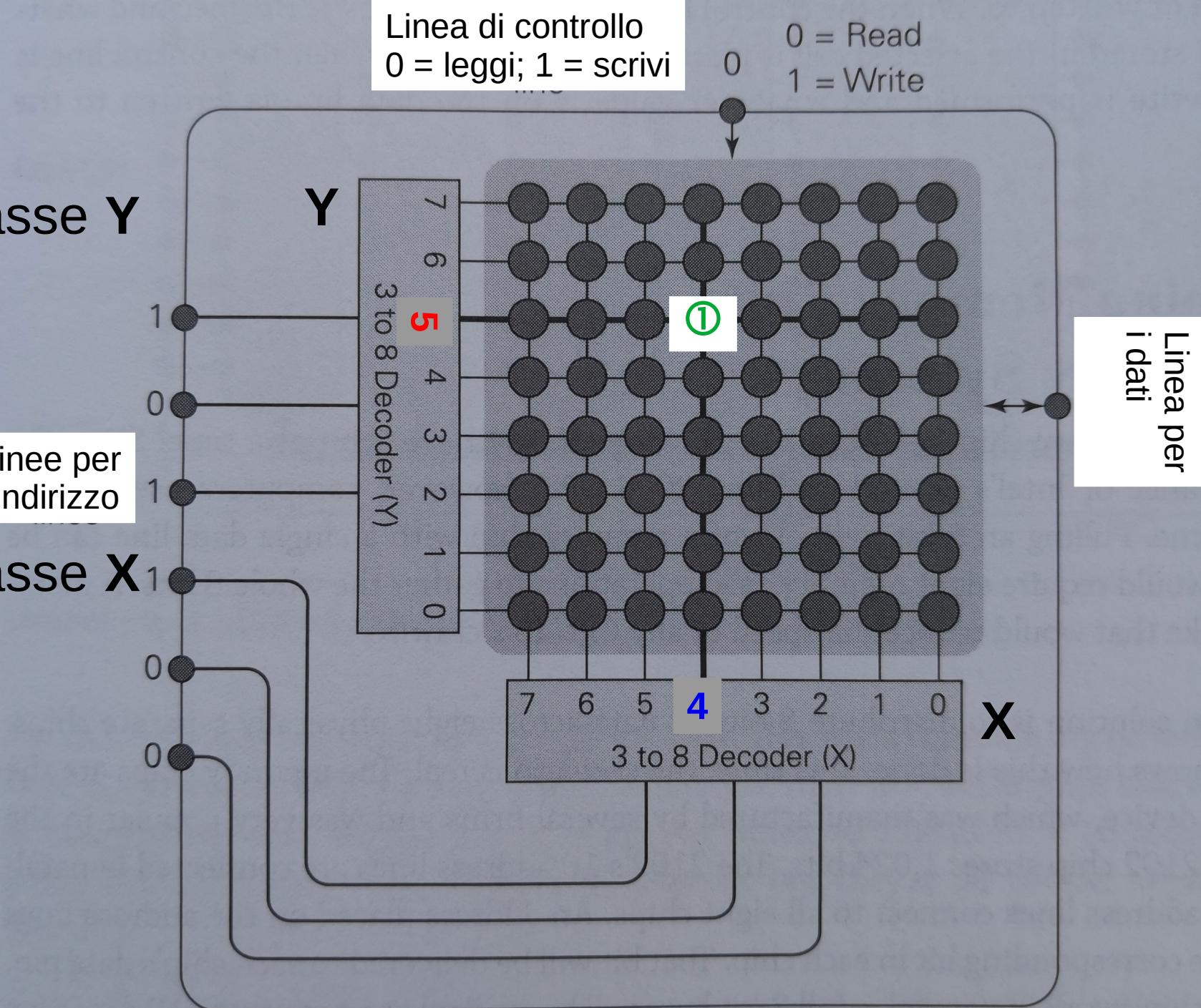
1 linea per i dati:

Input = scrivi

Output = leggi

Linea di controllo  
0 = leggi; 1 = scrivi

0 = Read  
1 = Write



# Memoria: 1 bit alla volta

6 linee di indirizzo (6 bit) perché:

Con 3 bit si possono codificare tutti gli interi compresi fra  
 $000 = 0 \cdot 2^0 + 0 \cdot 2^1 + 0 \cdot 2^2 = 0 + 0 + 0 = 0$

e

$$111 = 1 \cdot 2^0 + 1 \cdot 2^1 + 1 \cdot 2^2 = 1 + 2 + 4 = 7$$

Ossia

0 1 2 3 4 5 6 7

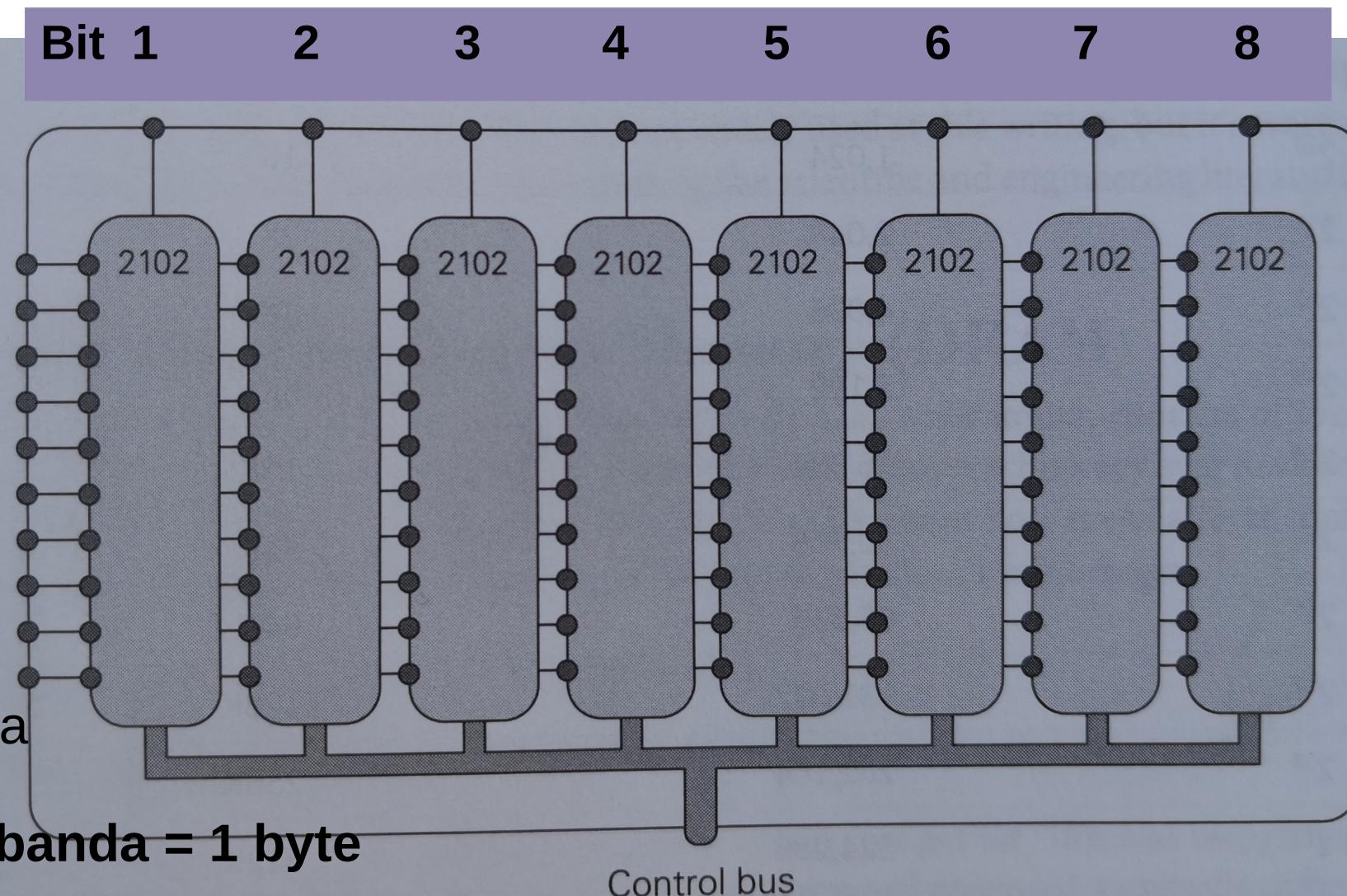
E, quindi, tutti i 64 ingressi della matrice di memoria 8x8

# Aprile 1972: arriva il processore Intel 8008

- Velocità di clock: 0.5 MHz
- 3500 transistors
- Linea dati (bus) da 8 bit
- Ogni unità di memoria 2102 contiene 1024 celle (**profondità = 1K**, per es. 256x4)
- Bus indirizzi da 10 bit
- Le 10 linee degli indirizzi sono collegate in parallelo alle 8 unità di memoria: lo stesso indirizzo viene letto (o scritto) contemporaneamente in tutte le 8 unità con una sola operazione di lettura (o scrittura) → **larghezza di banda = 1 byte**

Dimensione memoria = profondità \* larghezza

$$1024 * 8 = 8192 \text{ bit}$$



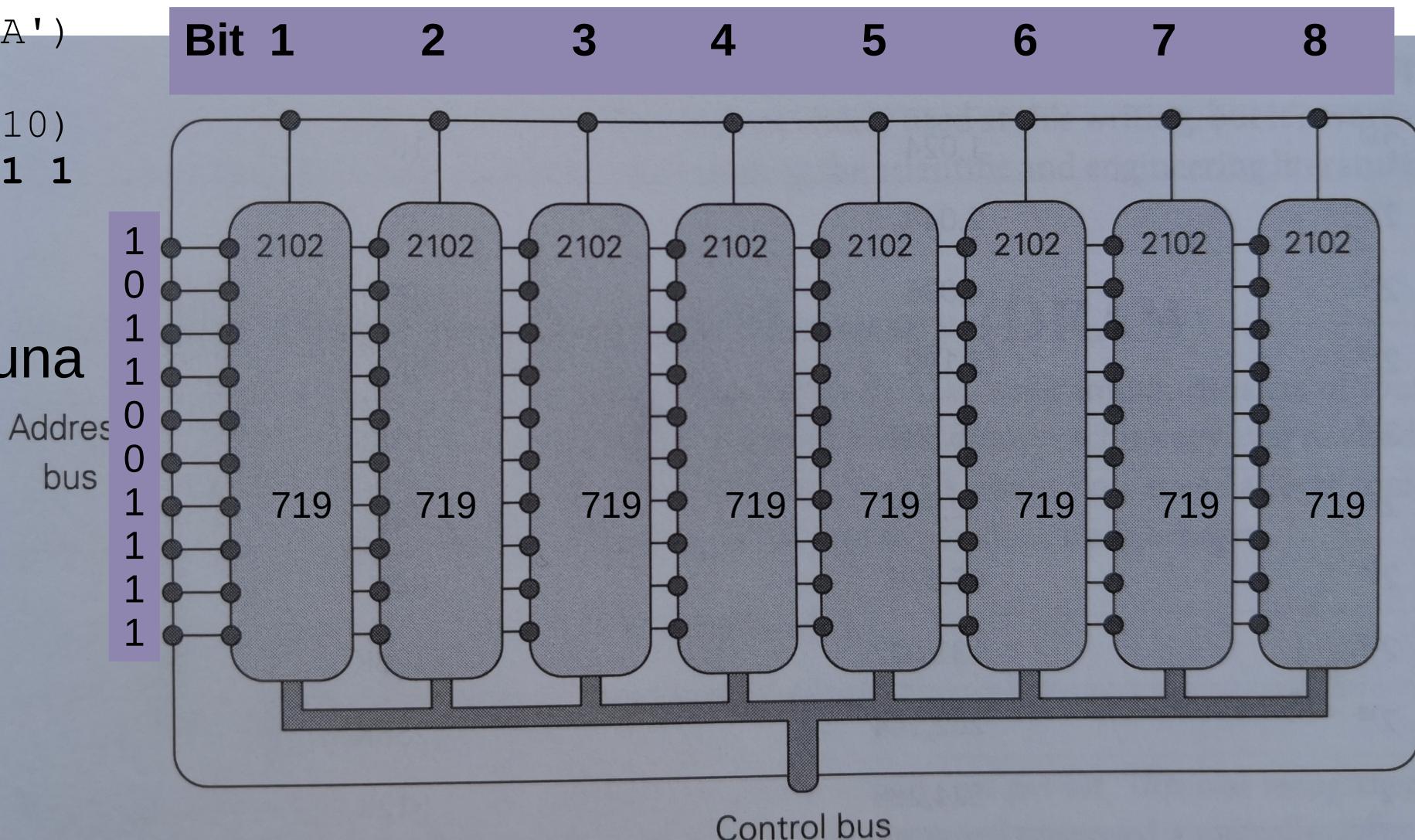
# Aprile 1972: arriva il processore Intel 8008

E., vogliamo accedere all'indirizzo di memoria <719>

In R:

```
> install.packages('GA')  
> library(GA)  
> decimal2binary(719, 10)  
[1] 1 0 1 1 0 0 1 1 1 1
```

- Lo stesso indirizzo viene letto in ciascuna delle 8 unità di memoria
- 8 bit = 1 byte



# Memoria: 8 bit alla volta

Con 10 bit si possono codificare 1024 numeri decimali, compresi fra

0 (0000000000)

e

1023 (1111111111)

In R:

```
> esponente = seq(0,9,1)
> esponente
[1] 0 1 2 3 4 5 6 7 8 9
> out[] = lapply(esponente, function(esponente) 2^esponente)
> unlist(out)
[1] 1 2 4 8 16 32 64 128 256 512
> sum(unlist(out))
[1] 1023
```

# Aprile 1972: arriva il processore Intel 8008

Esempio, vogliamo accedere all'indirizzo di memoria <719>  
1111001101 = ?

$$1*2^0+1*2^1+1*2^2+1*2^3+0*2^4+0*2^5+1*2^6+1*2^7+0*2^8+1*2^9 =$$

```
> unlist(out)
[1]    1    2    4    8   16   32   64  128  256  512
```

$$\begin{aligned} &= 1*1 + 1*2 + 1*4 + 1*8 + 0*16 + 0*32 + 1*64 + 1*128 + 0*256 + 1*512 \\ &= 719 \end{aligned}$$

# Tipi di memoria

- **ROM** (Read-Only Memory): sola lettura, non volatile, contiene le istruzioni per l'avvio del sistema, ROM della **BIOS** (Basic Input Output System) che fa parte della scheda madre (*firmware*)
- **DRAM** (Dynamic Random Access Memory)
- **SRAM** (Static Random Access Memory)

# Tipi di memoria

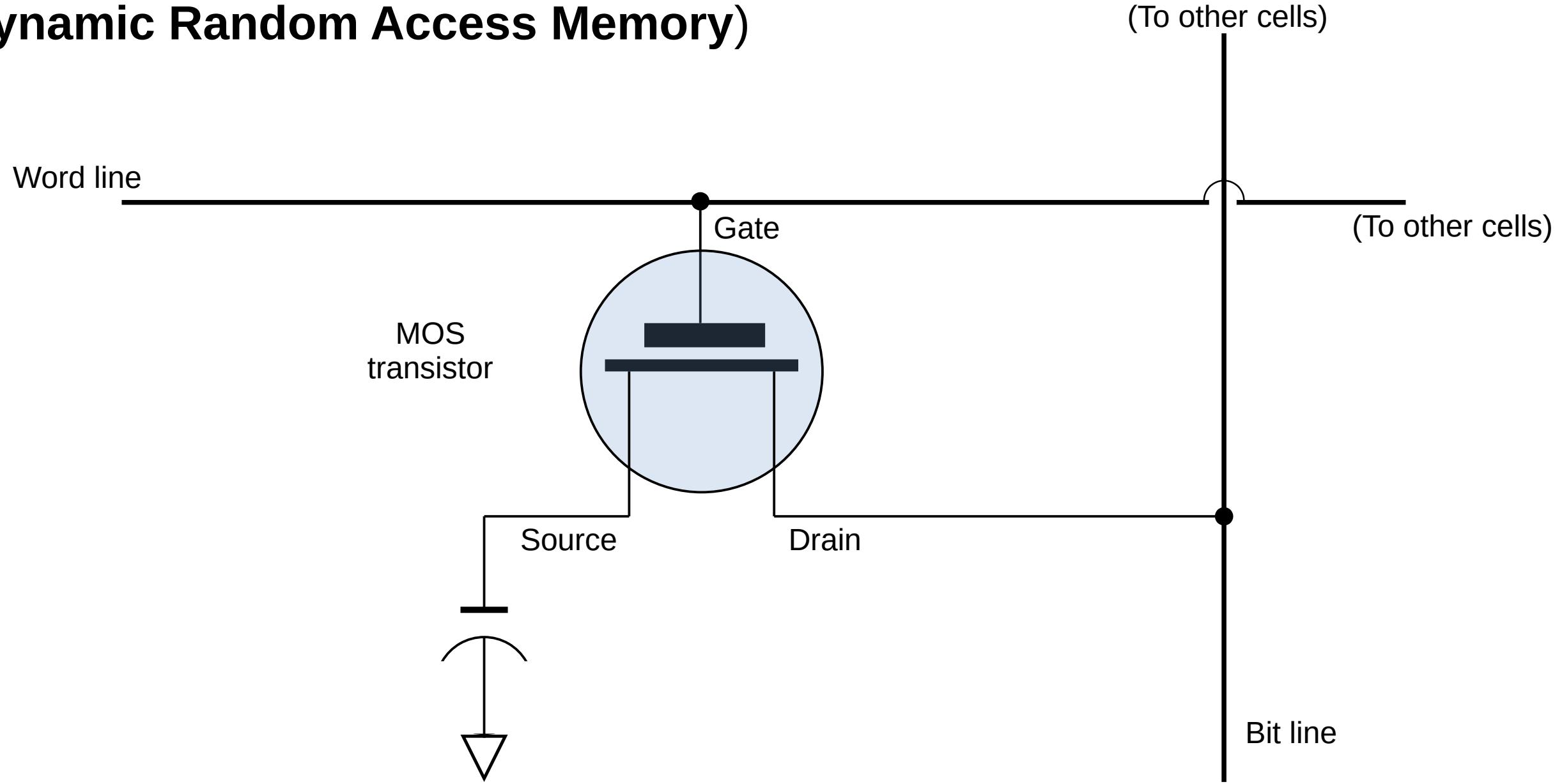
- **ROM** (Read-Only Memory): sola lettura, non volatile, contiene le istruzioni per l'avvio del sistema, ROM della **BIOS** (Basic Input Output System) che fa parte della scheda madre (*firmware*)

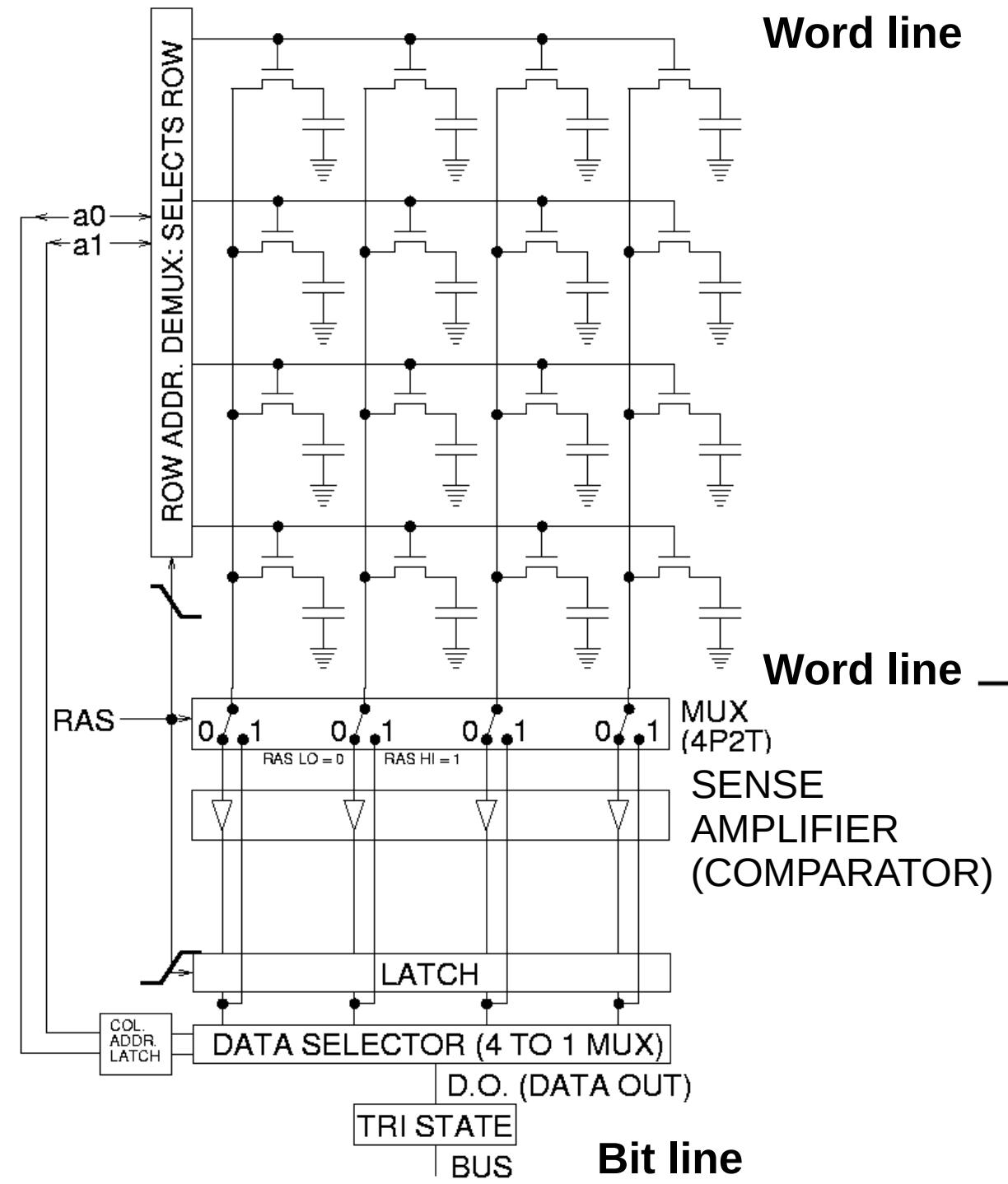
```
lab15@lab15:~$ sudo lshw -C memory
[sudo] password di lab15:
  *-firmware
    descrizione: BIOS
    fornitore: Insyde
    id fisico: 0
    versione: F.59
    date: 02/04/2023
    dimensione: 128KiB
    capacità: 10MiB
    capacità: pci upgrade shadowing cdboot bootselect edd int13floppynec
    pi usb biosbootspecification uefi
```

# Tipi di memoria

- **ROM** (Read-Only Memory): sola lettura, non volatile, contiene le istruzioni per l'avvio del sistema, ROM della **BIOS** (Basic Input Output System) che fa parte della scheda madre (*firmware*)
- **DRAM (Dynamic Random Access Memory)**
- **SRAM (Static Random Access Memory)**

# Una cella di memoria DRAM (Dynamic Random Access Memory)

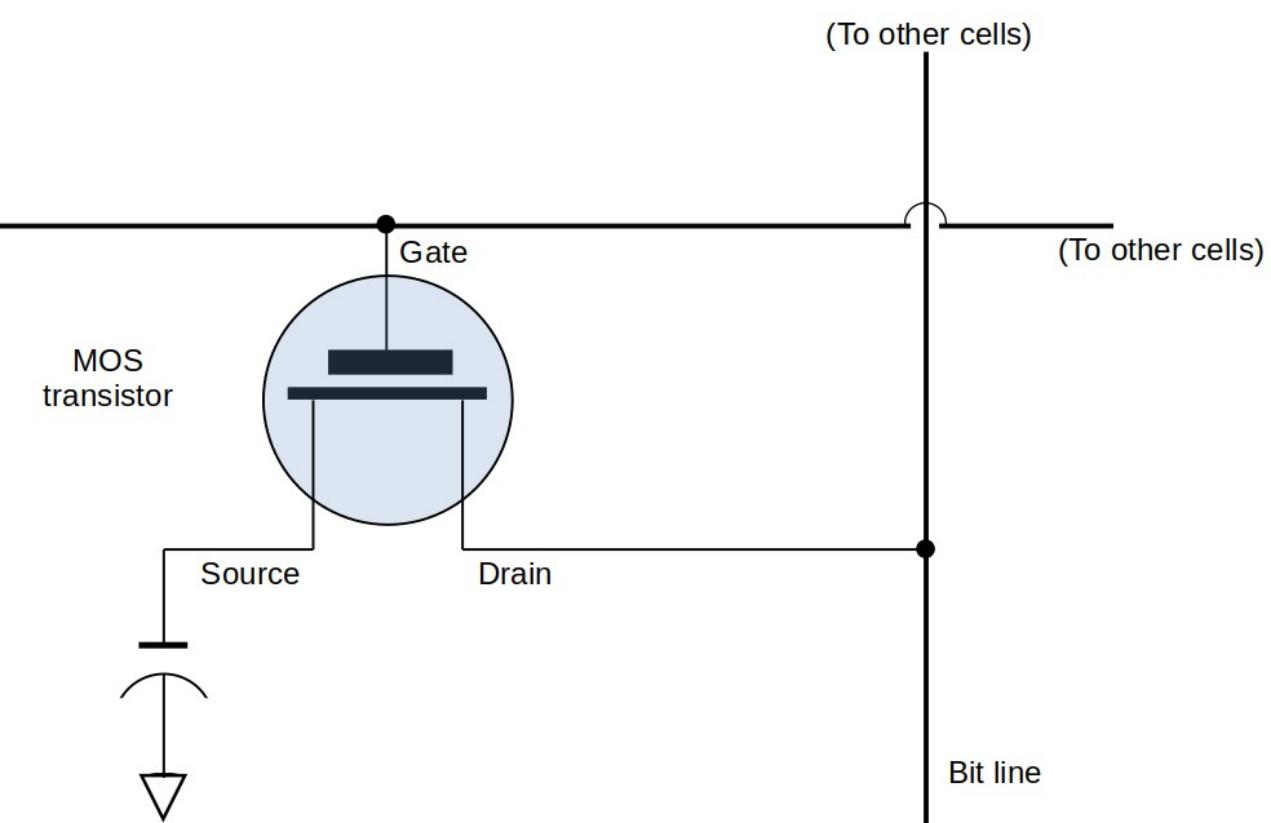


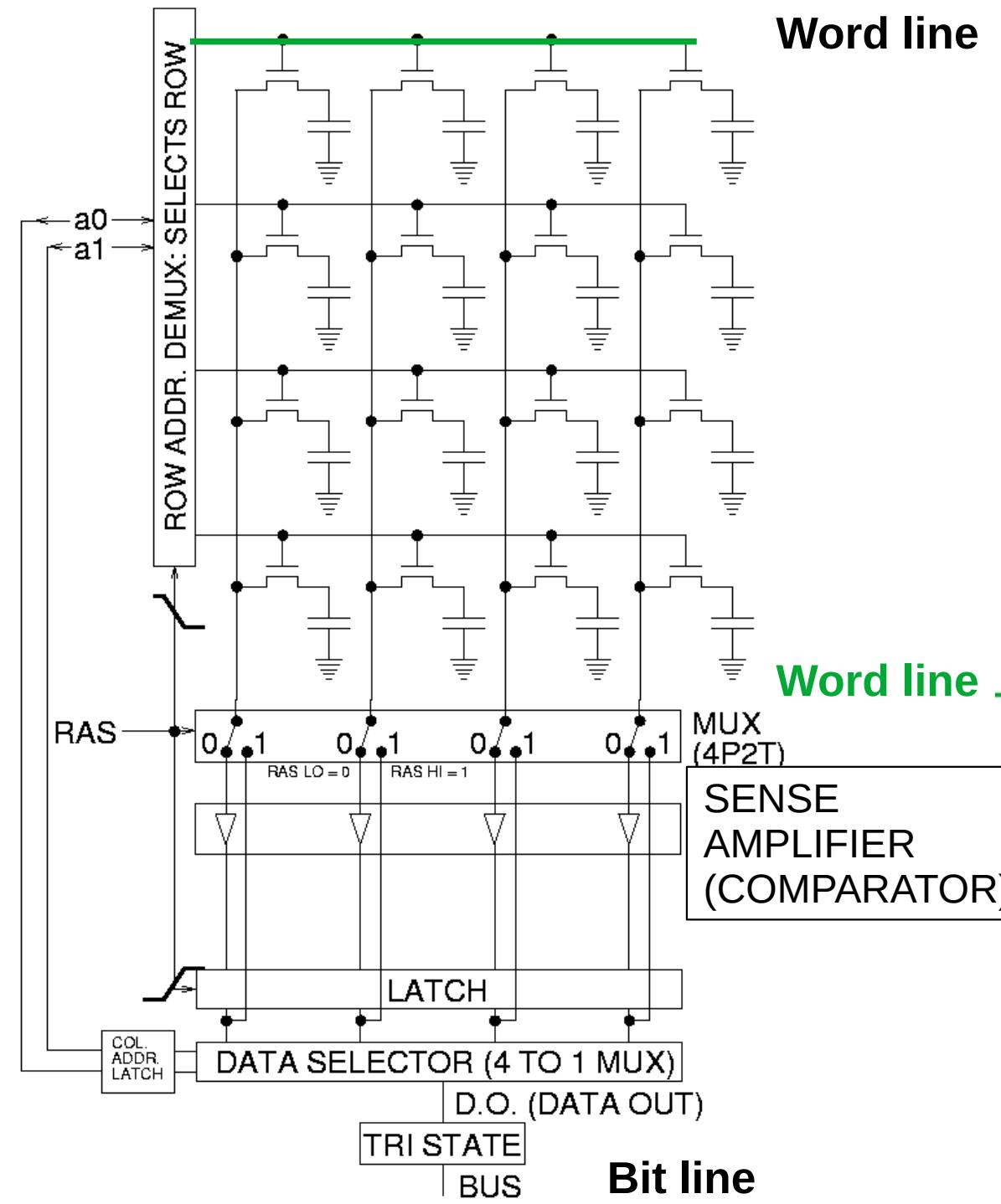


**Word line**

**Bit line**

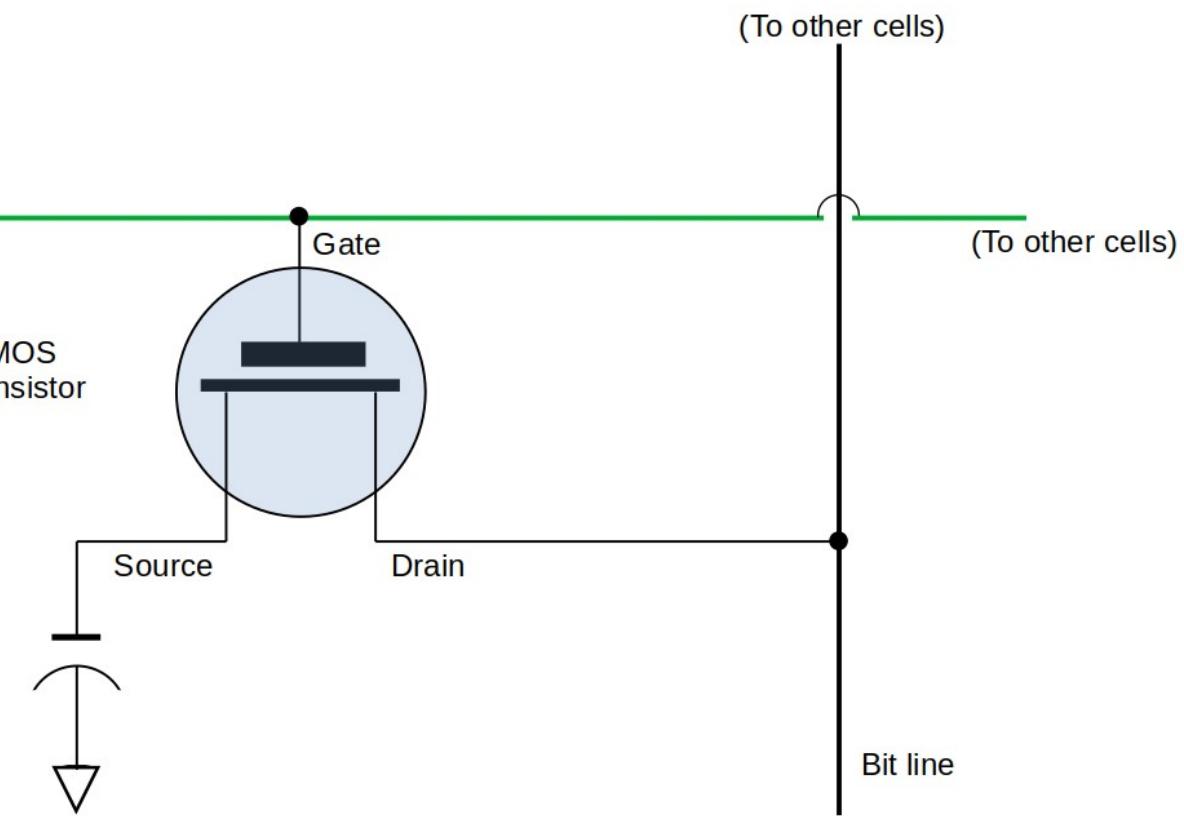
- Solo un condensatore e un transistor per bit → alte densità ( $10^9$  su piccole superf.) → basso costo
- La carica del condensatore dura poco → i dati devono essere periodicamente riscritti (ogni 15 ms) → consumo E → calore e richiede più cicli della CPU
- **Lenta**

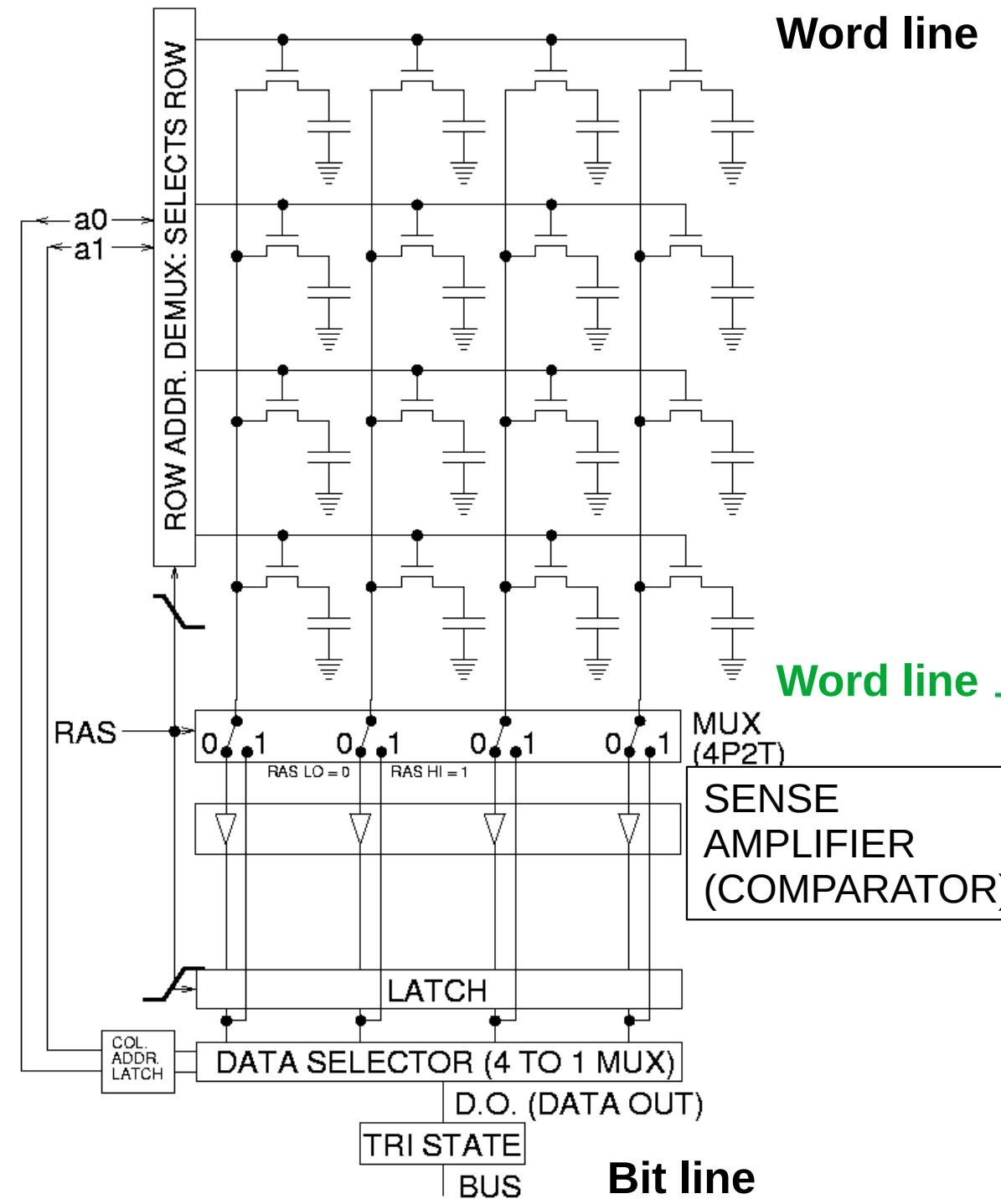




## Lettura di un bit

- L'amplificatore viene disconnesso
- Le bit lines vengono precaricate a 0.5 V, valore intermedio fra i livelli logici basso (0 V) e alto (1 V)
- Il circuito di precarica viene disconnesso e la bit line collegata all'amplificatore
- Viene selezionata una **word line**



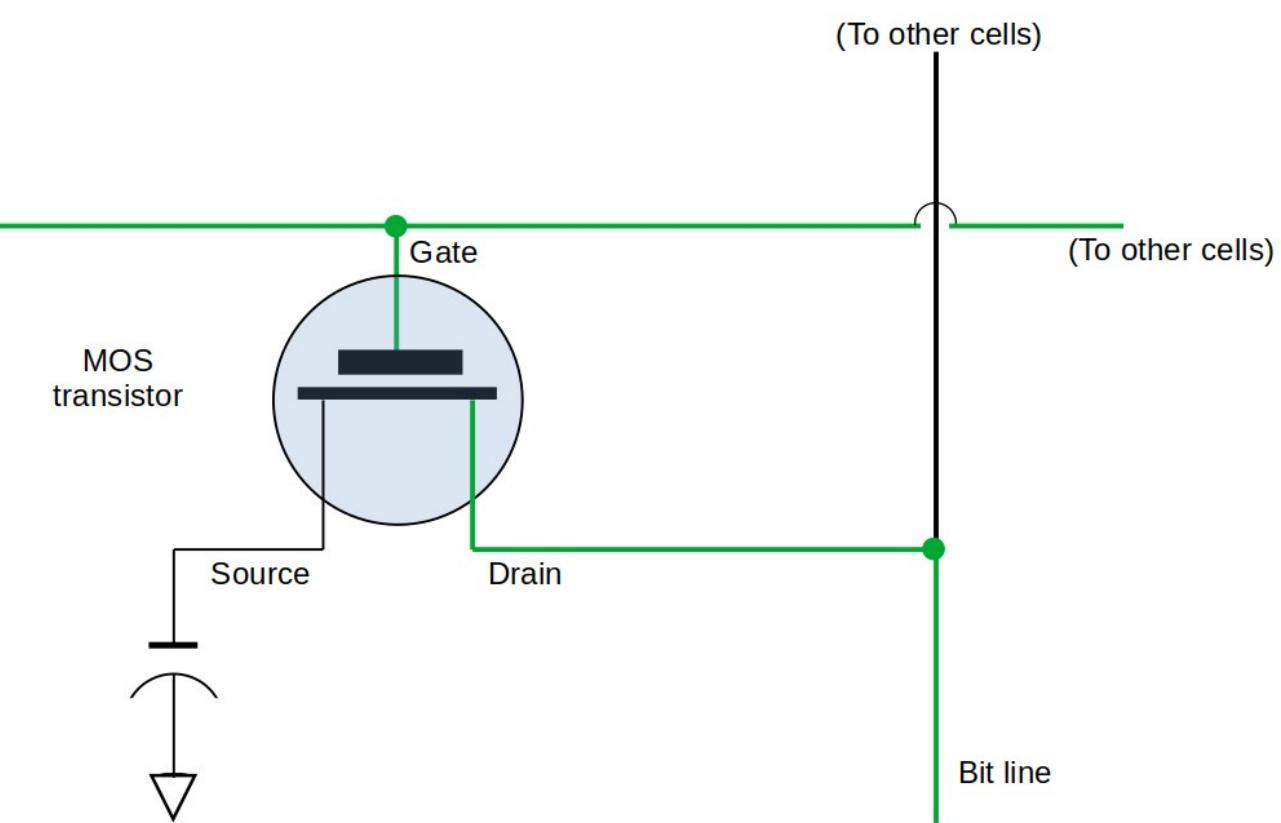


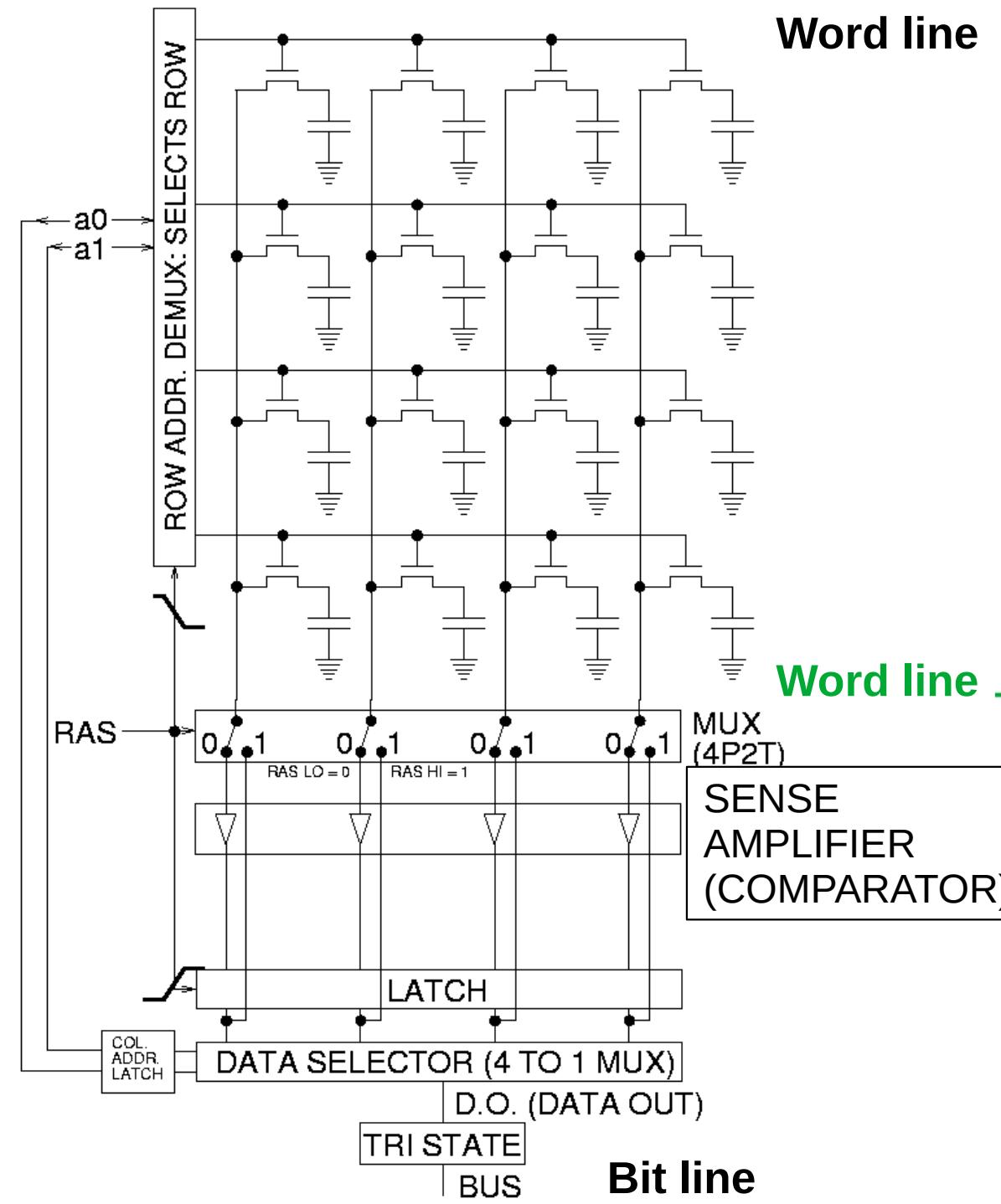
Word line

Bit line

## Lettura di un bit

- La word line attiva i transistor di un'intera riga
- La tensione (in V) della bit line dipende dallo stato di carica del condensatore
- Condensatore carico  $\rightarrow > V$  nella bit line
- Condensatore scarico  $\rightarrow < V$  nella bit line ( $\Delta \approx 10^6 e^-$ )





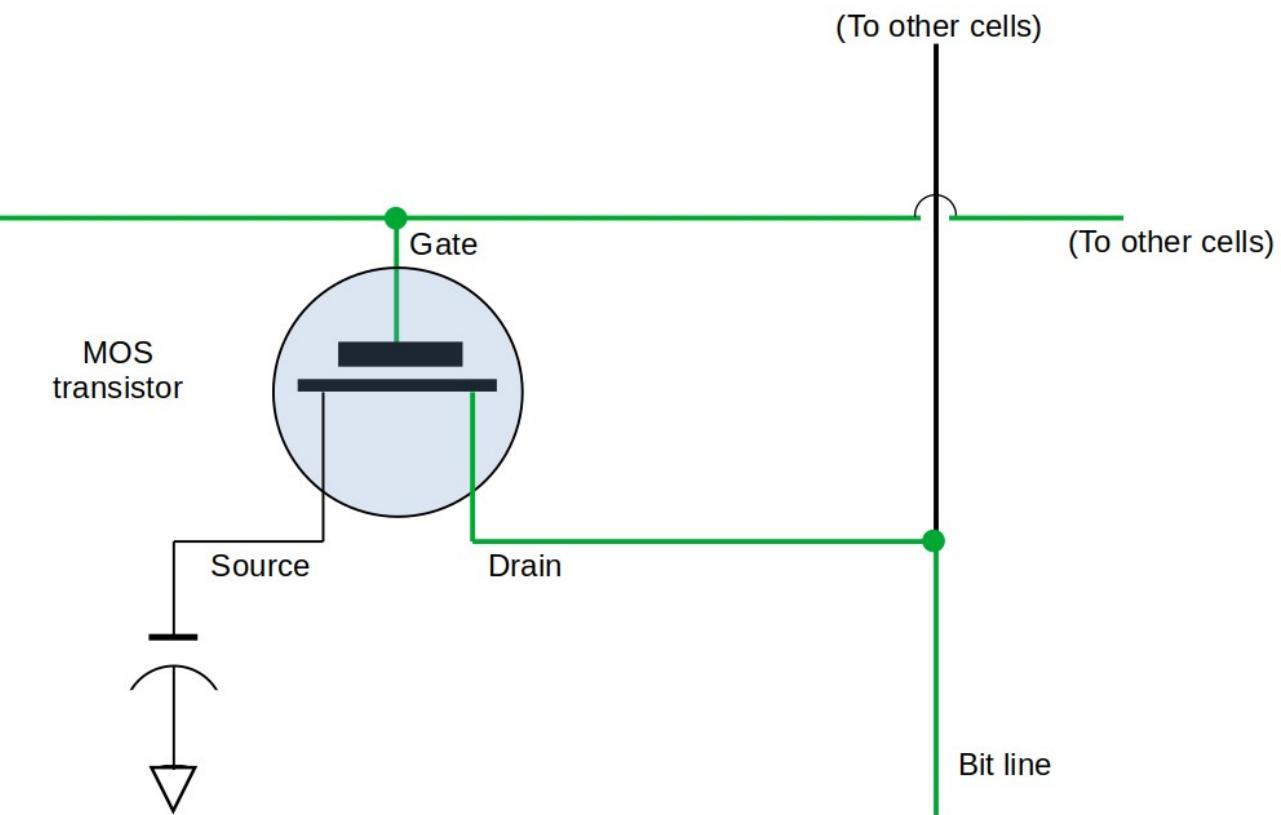
Word line

Bit line

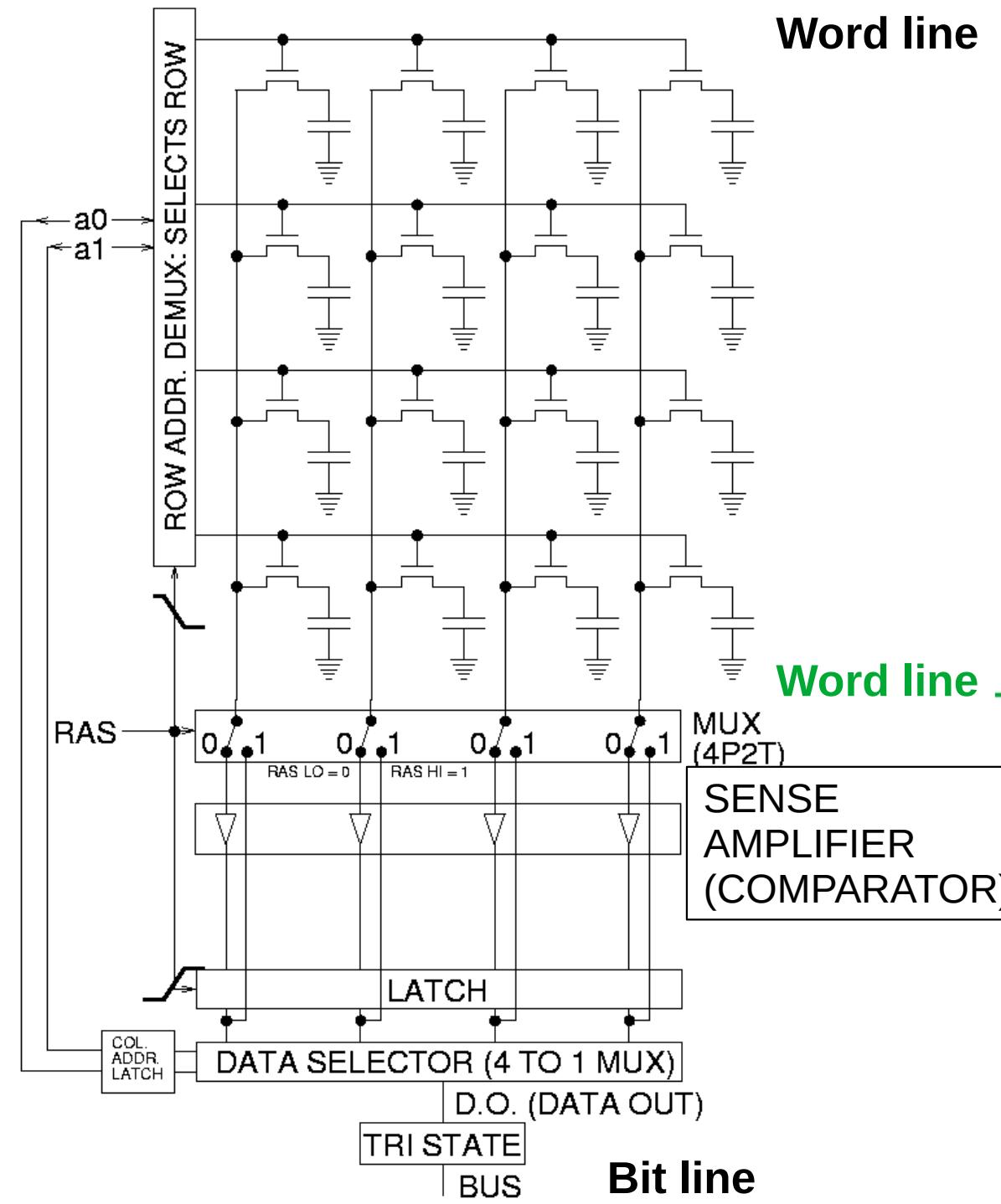
Word line

## Lettura di un bit

- La lettura distrugge la carica dei condensatori di tutta la riga
- La carica dev'essere ripristinata (refreshed)
- Il ripristino viene effettuato comunque ogni 5÷50 millisecondi da parte del *memory controller* (non della CPU)

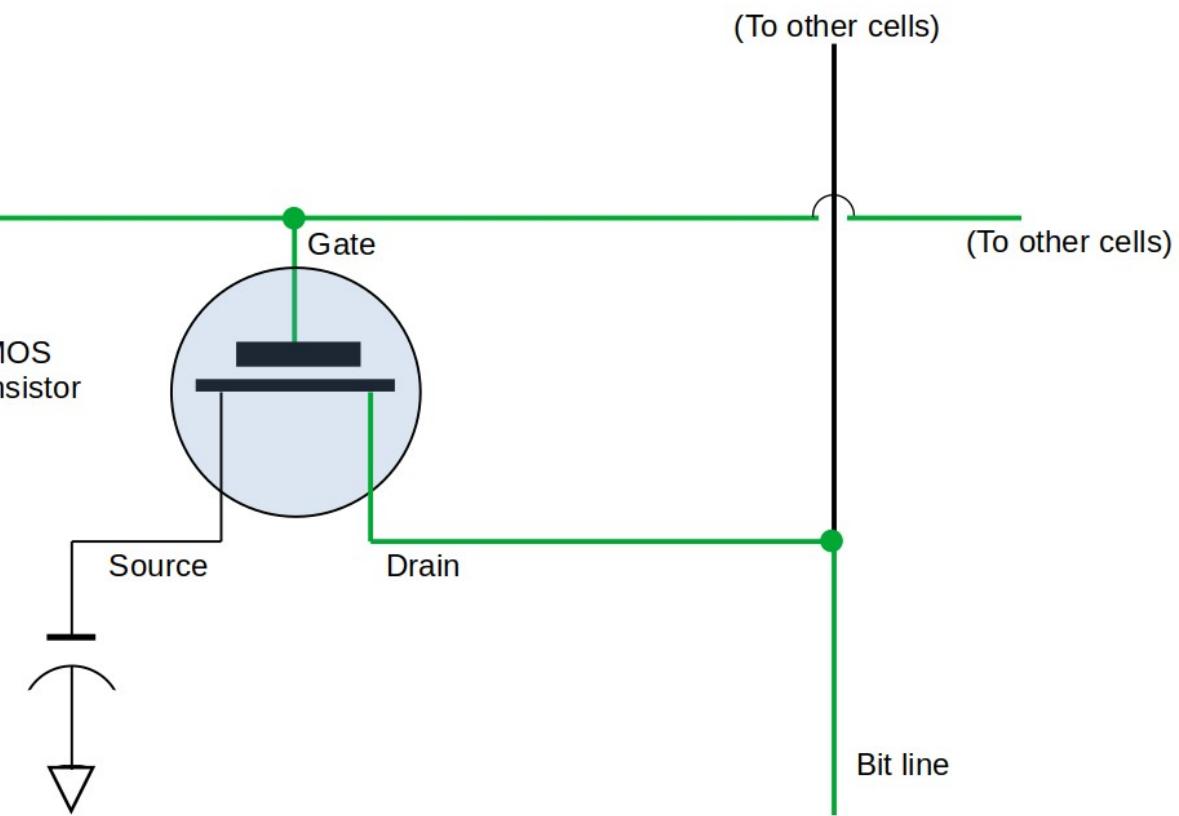


Bit line

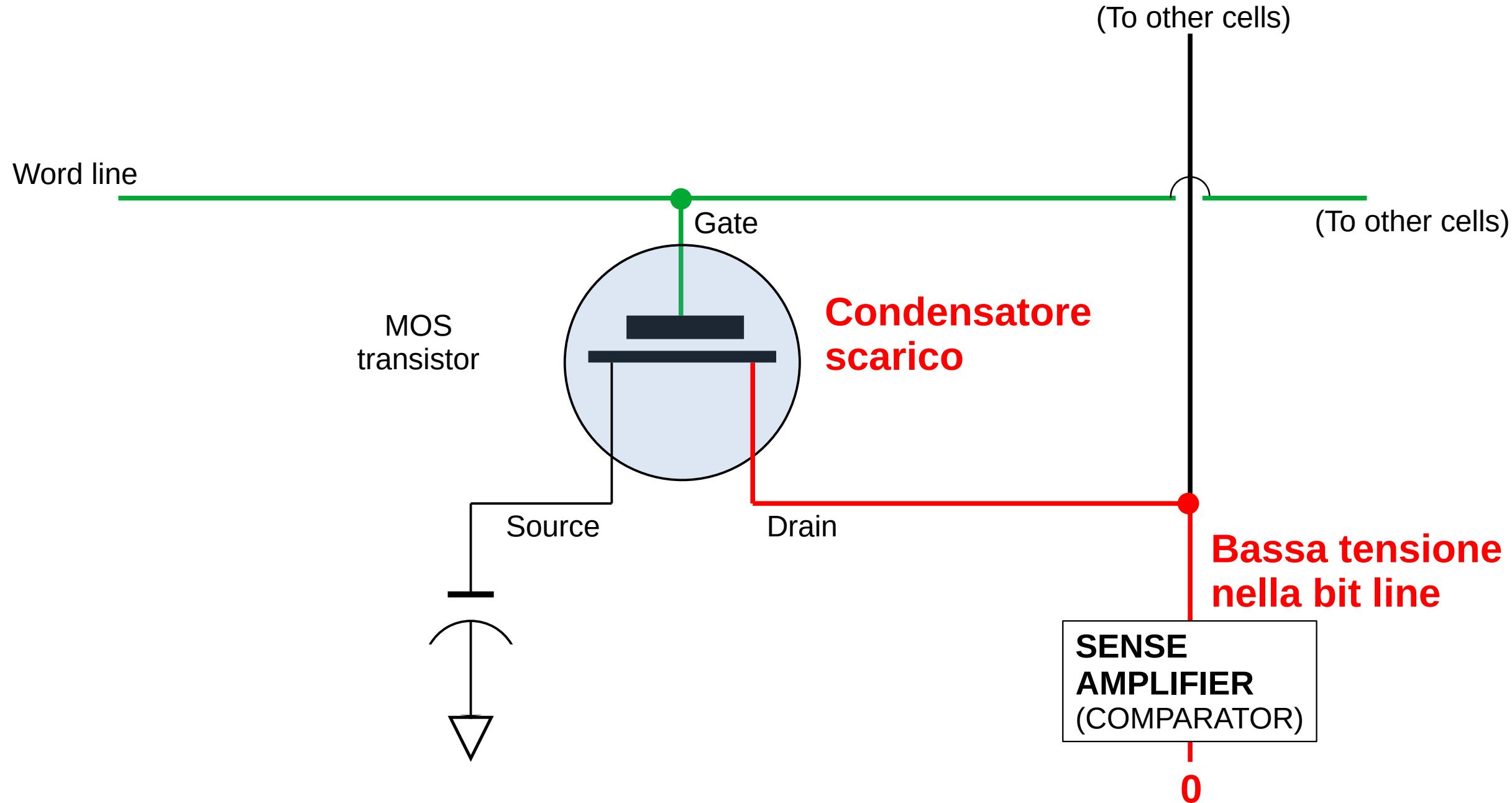


## Lettura di un bit

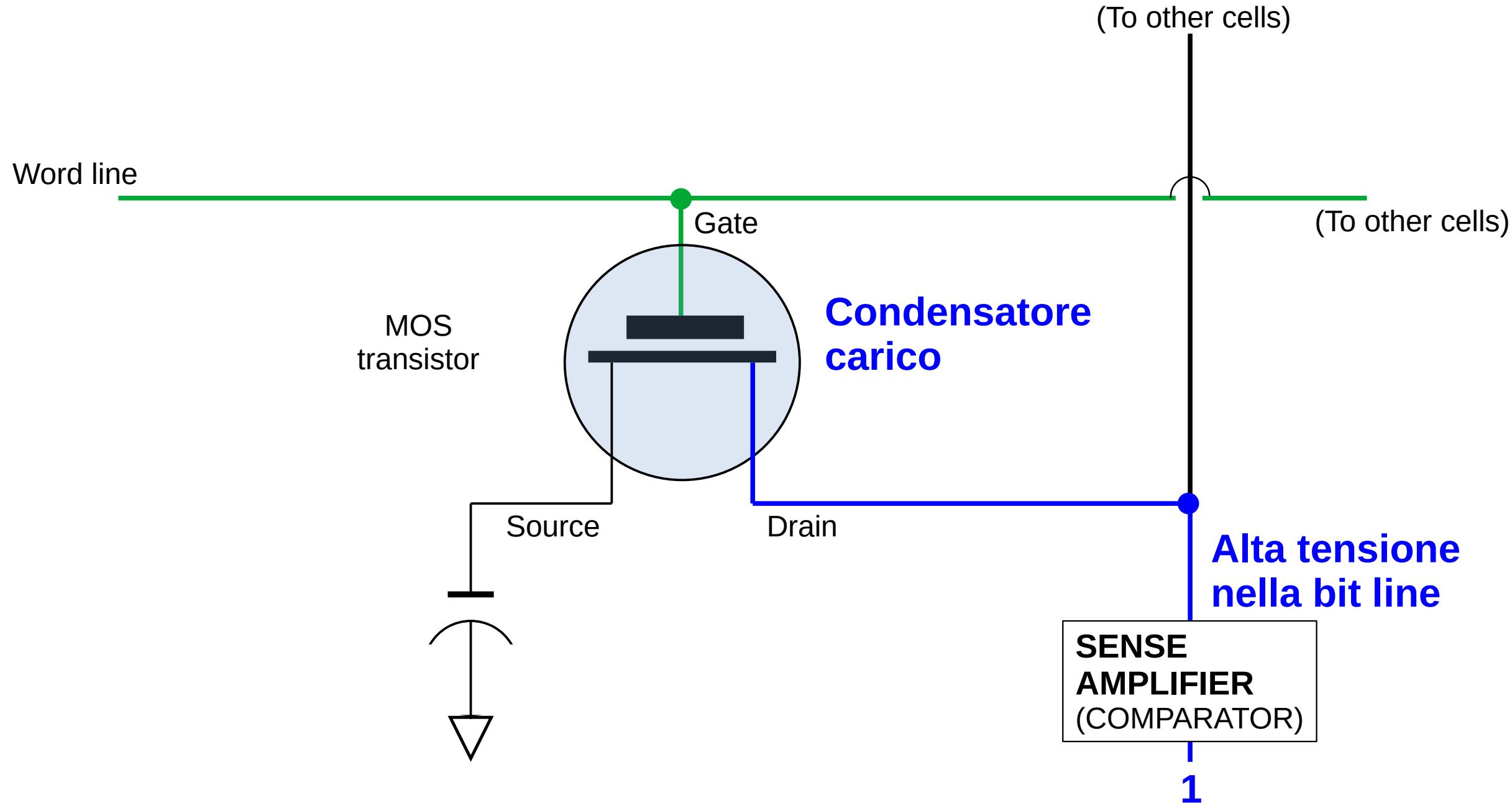
- L'operazione di refreshing viene svolta mediante la lettura dello stato delle celle della riga (carica dei condensatori) e la successiva riscrittura dello stesso



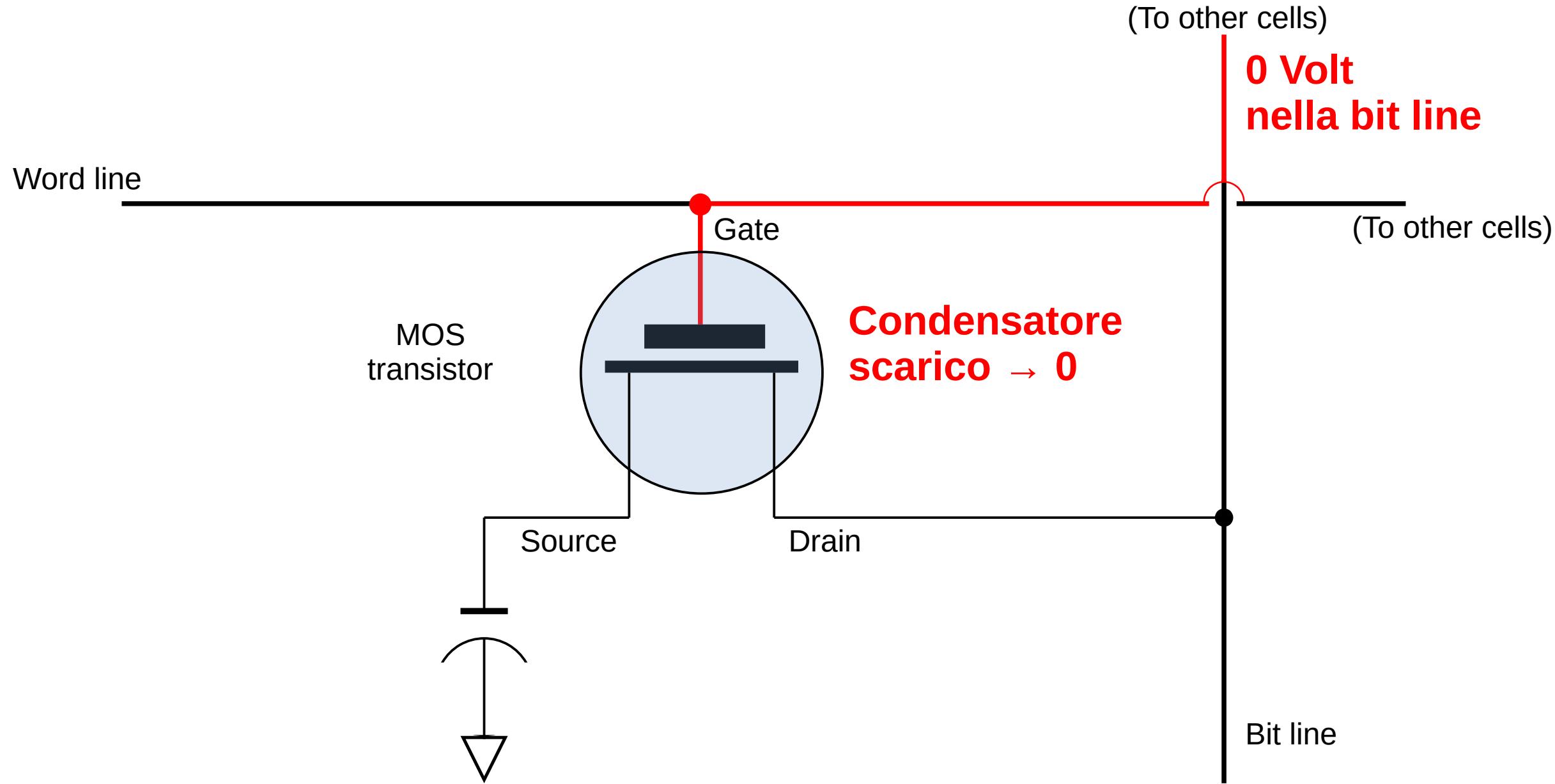
# Lettura di un bit da una cella DRAM: 0



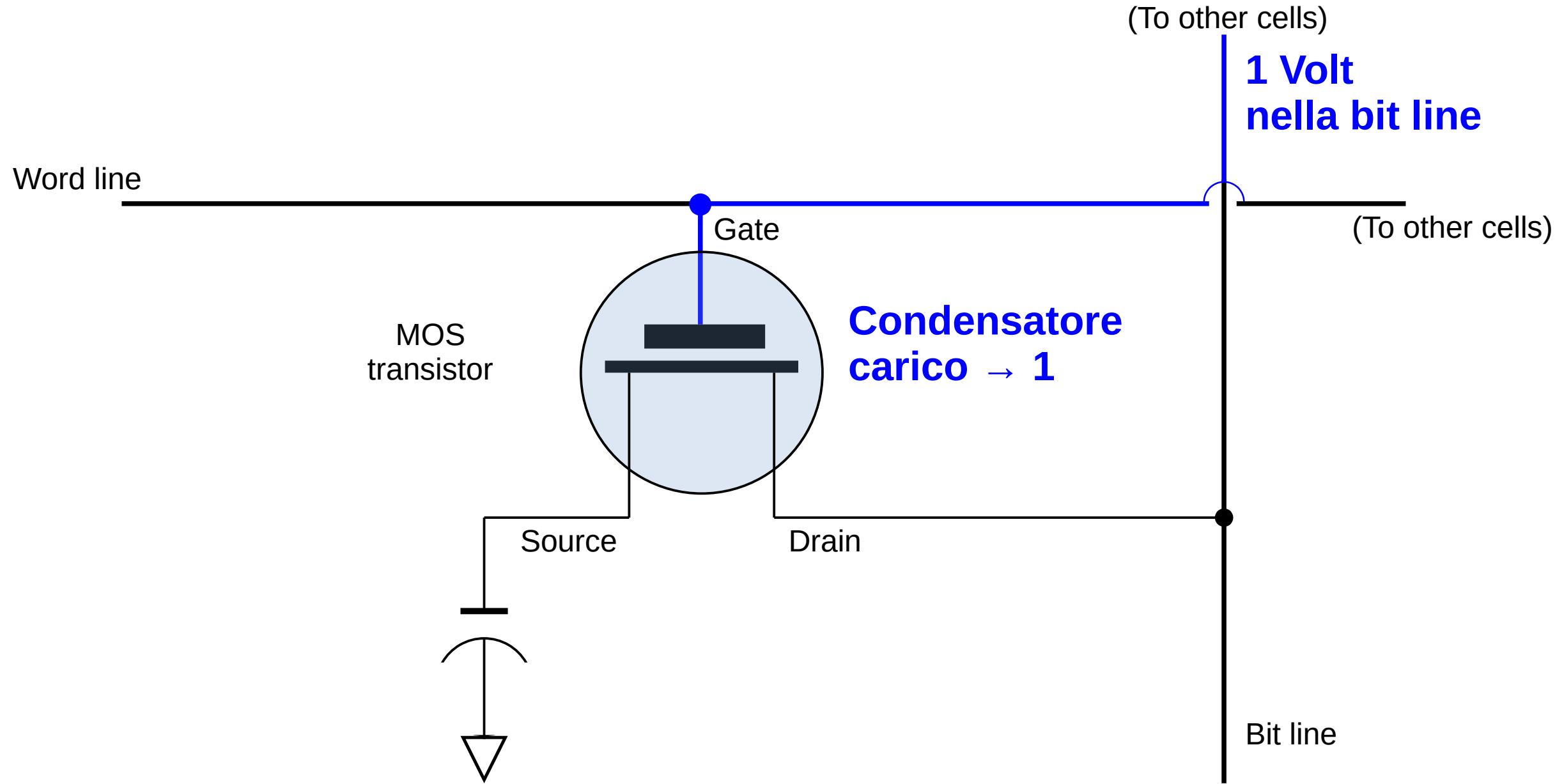
# Lettura di un bit da una cella DRAM: 1



# Scrittura di un bit da una cella DRAM: 0



# Scrittura di un bit da una cella DRAM: 1



# Tipi di memoria

- **ROM** (Read-Only Memory): sola lettura, non volatile, contiene le istruzioni per l'avvio del sistema, ROM della **BIOS** (Basic Input Output System) che fa parte della scheda madre (*firmware*)
- **DRAM** (Dynamic Random Access Memory)
- **SRAM** (Static Random Access Memory)

# SRAM (Static Random Access Memory)

- detta anche *cache*
- 6 transistor / bit ma nessun condensatore
- *refresh* non necessario
- molto veloce: tempo di accesso da parte della CPU  $\leq 0.25$  ns
- tiene il passo con un processore da 4 GHz
- in **R**:  
$$> 1 / (4 * 10^9)$$
  
[1]  $2.5 \times 10^{-10} = 0.25 \times 10^{-9}$
- ma meno densa e più costosa
- *cache hit*: i dati a cui la CPU deve accedere erano già stati copiati dalla memoria principale (DRAM) alla cache ad opera del *memory controller*
- *cache miss*: i dati non sono nella cache → la CPU deve attendere senza far alcunché (*wait states*)

# SRAM (Static Random Access Memory)

- es.: CPU e cache girano a 3.6 GHz → un ciclo ogni  
in R:

```
> 1 / (3.6 * 10^9)
```

```
[1] 2.777778e-10 ≈ 0.28e-9
```

**0.28 ns**; se la DRAM gira a 1.33 GHz:

```
> 1 / (1.333 * 10^9)
```

```
[1] 7.501875e-10 ≈ 0.75e-9
```

**0.75 ns**

•

## Chip della CPU

- **Circuito integrato:** circuito elettronico miniaturizzato formato da vari transistors
- **Chip:** supporto che contiene gli elementi che costituiscono il circuito
- Tutte le CPU moderne sono contenute in un unico chip
- La CPU comunica con la memoria e con i dispositivi di input-output (I/O) attraverso una serie di **pin** (connettori):
  - pin di indirizzi
  - pin di dati
  - pin di controlli
- **Bus:** insieme di cavi paralleli che collegano i pin della CPU con quelli della memoria e dei dispositivi di I/O

# Come la CPU accede ad un'istruzione (che può essere inclusa in un programma)

La CPU...

- imposta l'indirizzo dell'istruzione nei suoi pin di indirizzo
- asserisce (porta ad 1) una o più linee di controllo per informare la memoria che intende leggere una parola

La memoria...

- spedisce la parola richiesta sui pin dei dati della CPU
- asserisce un segnale che notifica alla CPU che l'operazione è stata completata

Quando la CPU vede questo segnale accetta la parola ed esegue l'istruzione

L'intero processo viene ripetuto per ogni parola

# Da cosa dipendono le prestazioni della CPU?

Le prestazioni della CPU dipendono da:

- **N° di pin d'indirizzo:**
- un chip con  $m$  pin d'indirizzo ( $m = 16, 20, 32, 64$ , corrispondenti ad altrettante linee – in ogni linea viene inviato un bit) può indirizzare  $2^m$  indirizzi o locazioni di memoria [N° di oggetti – 2 (0 e 1) elevato a quanti se ne prendono per volta combinandoli fra loro – disposizioni con ripetizione ()]
- **N° di pin di dati:** un chip con  $n$  pin di dati può leggere o scrivere in un'unica operazione una parola di  $n$  bit ( $n = 8, 32, 64$ )
  - **Quanti pin di dati ha il mio pc?**

```
piero@piero-XPS-9320:~$ uname -a
```

```
Linux piero-XPS-9320 6.1.0-1026-oem #26-Ubuntu SMP PREEMPT_DYNAMIC Wed  
Nov 1 14:06:23 UTC 2023 x86_64 x86_64 x86_64 GNU/Linux
```

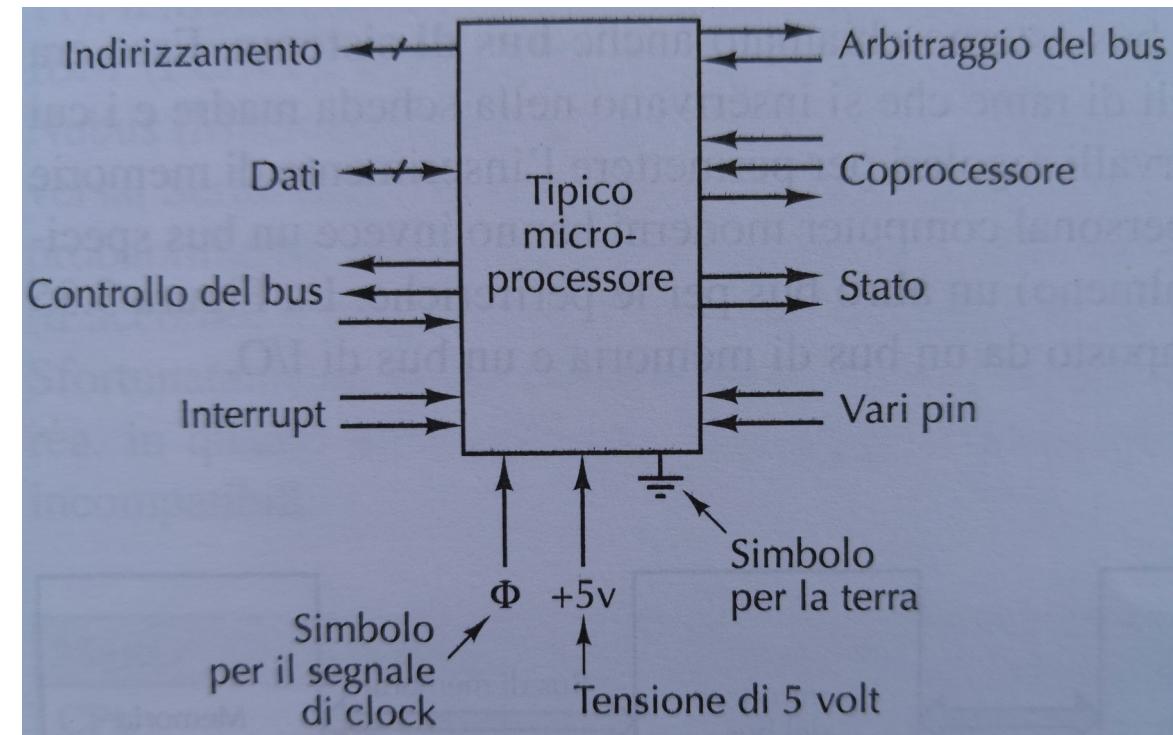
## Altri pin della CPU

- **pin di controllo**: regolano il flusso e la temporizzazione dei dati da e verso la CPU
- **pin per l'alimentazione (+1,2÷+1,5 V)**
- **pin per la messa a terra**
- **pin per il segnale di clock** (un'onda quadra con una determinata frequenza)

# I pin di controllo

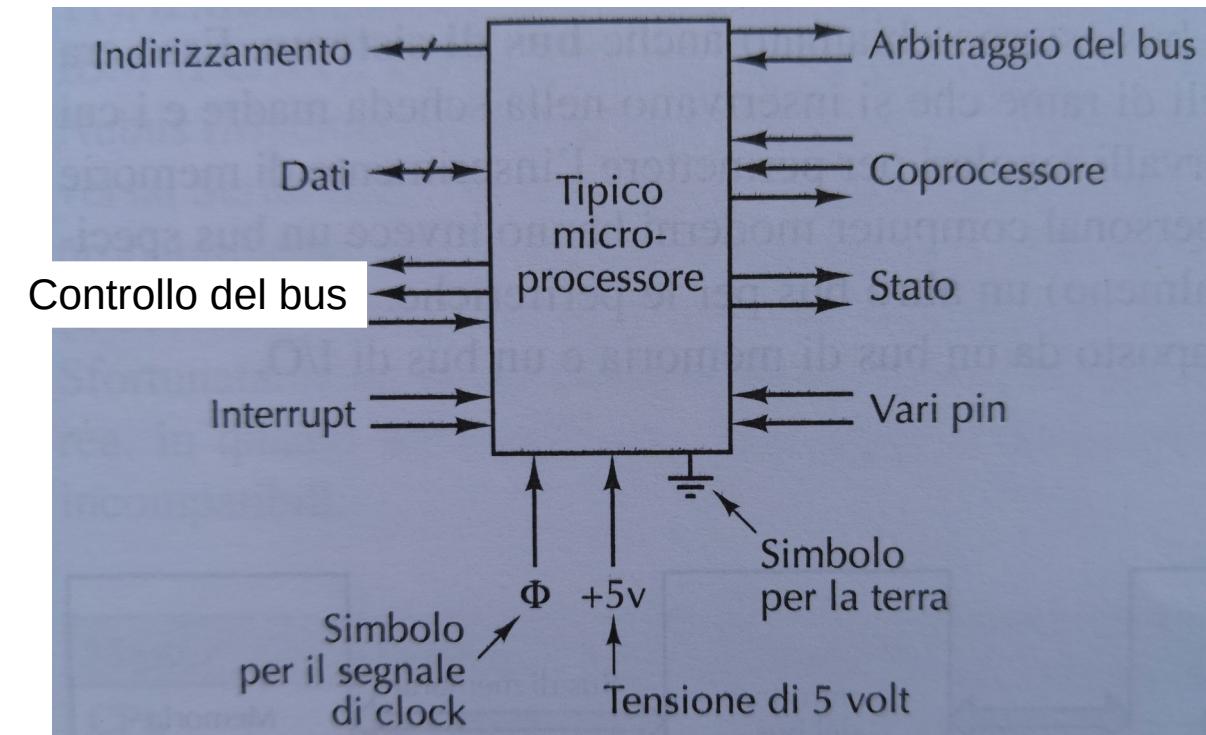
Possono essere pin di:

- Controllo del bus
- Interrupt
- Arbitraggio del bus
- Comunicazione con il coprocessore
- Stato
- Altro



# I pin di controllo del bus

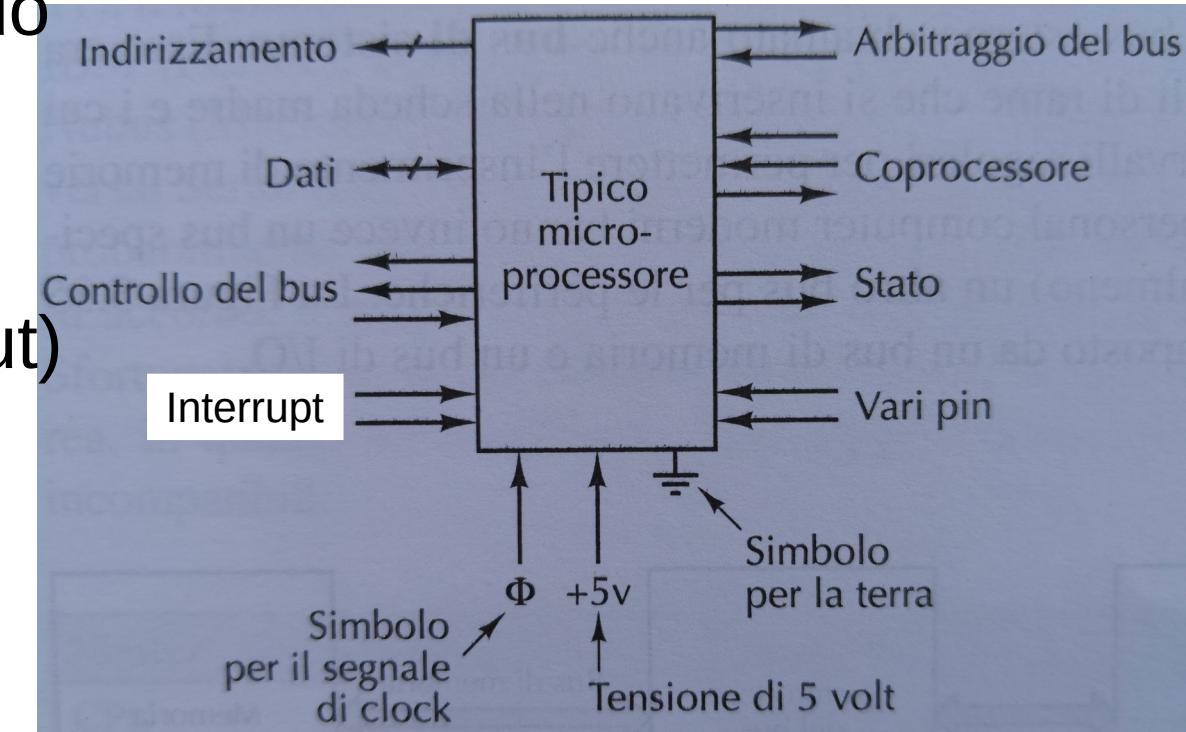
Inviano segnali dalla CPU alla memoria o ai chip di I/O per notificare l'azione che la CPU stessa vuole compiere (es. lettura o scrittura della memoria)



# I pin di interrupt

Ricevono input dalle periferiche

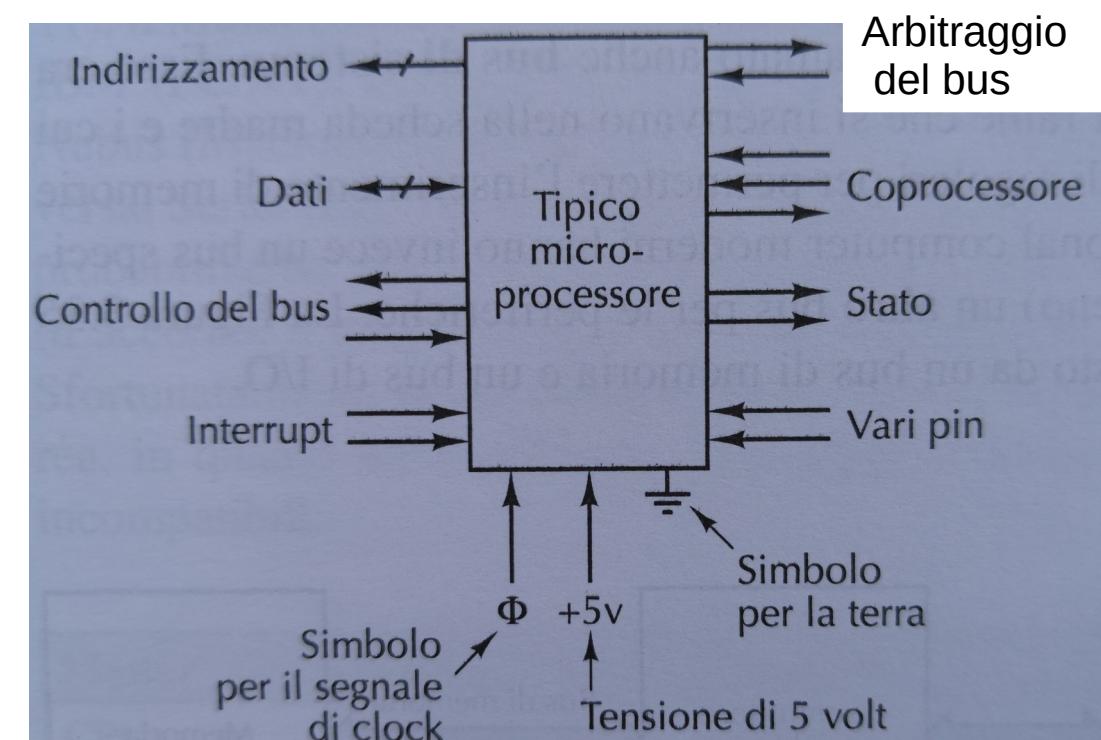
- La CPU comunica ad un dispositivo di I/O (attraverso i pin di controllo del bus) l'inizio di un'operazione
- Mentre la periferica esegue il lavoro assegnato, la CPU compie qualche altra operazione
- Quando il dispositivo di I/O ha finito asserisce un segnale su un pin di interrupt:
  - il lavoro che la CPU stava svolgendo si interrompe
  - la CPU invia al dispositivo una conferma in risposta all'**Interrupt Request** (attraverso un pin di output)
  - la CPU inizia a ricevere dati dal dispositivo



# I pin di arbitraggio del bus

Regolano il traffico sul bus per evitare che due dispositivi cerchino di usarlo nello stesso momento

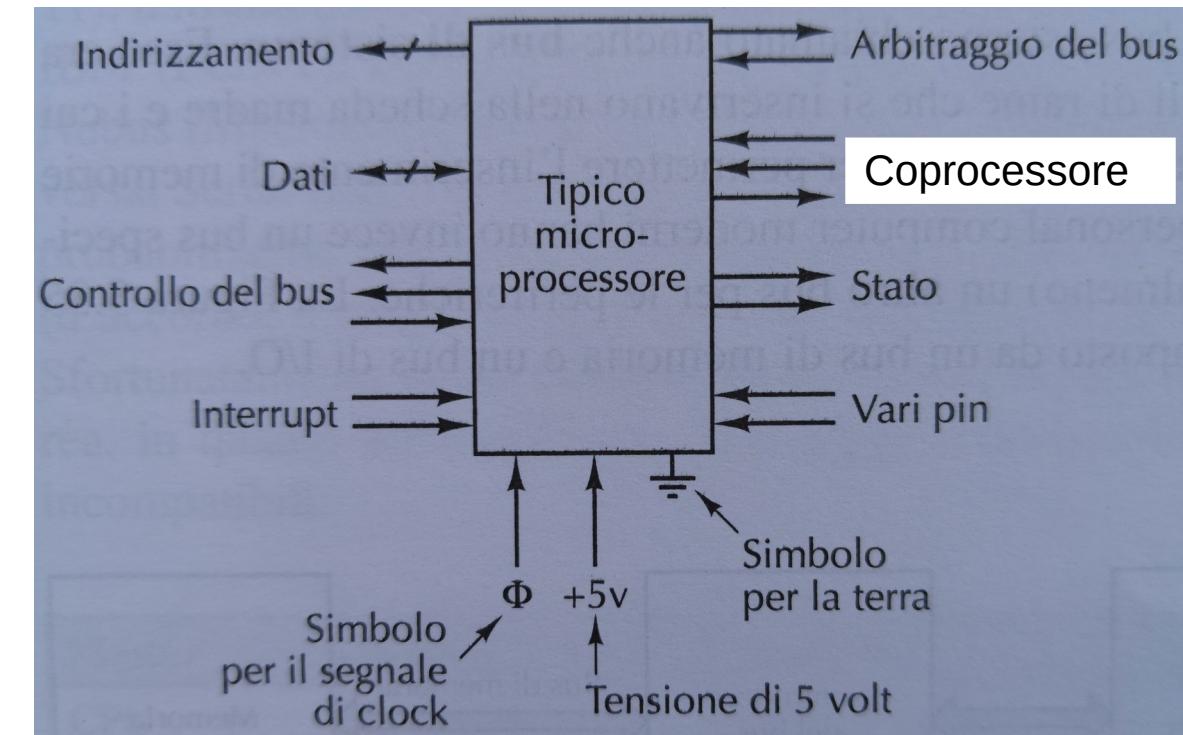
- La CPU conta quanto qualsiasi altro dispositivo e anch'essa deve fare esplicita richiesta di utilizzo del bus



# I pin del coprocessore

Esistono due tipi di coprocessori:

- chip per i calcoli in virgola mobile (**floating-point operations**)
- coprocessori grafici (**Graphical Processing Unit o GPU**)



# Il coprocessore matematico: la rappresentazione dei numeri nei calcolatori

Scriviamo il numero  $85_{10}$  in formato binario

In R:

```
> library(GA)
```

Caricamento del pacchetto richiesto: foreach

Caricamento del pacchetto richiesto: iterators

```
 _____|_____\n / \ | / \\ Genetic\n | | - / - \ Algorithms\n | | | | / \ \\\n \_ / / \ \ \ version 3.2.4
```

Type 'citation("GA")' for citing this R package in publications.

Caricamento pacchetto: 'GA'

Il seguente oggetto è mascherato da 'package:utils':

de

```
> decimal2binary(85,10)
```

```
[1] 0 0 0 1 0 1 0 1 0 1
```

```
>
```

# Il coprocessore matematico: la rappresentazione dei numeri nei calcolatori

Il numero  $85_{10}$  in formato binario è 1010101

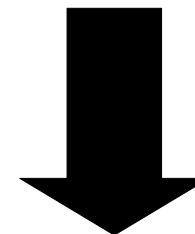
$$1010101_2 = 1*2^0 + 0*2^1 + 1*2^2 + 0*2^3 + 1*2^4 + 0*2^5 + 1*2^6 = 1+4+16+64=85$$

**5 bit**

<b>1</b>	<b>1</b>	<b>1</b>	<b>1</b>	<b>1</b>
----------	----------	----------	----------	----------

$2^4 \quad 2^3 \quad 2^2 \quad 2^1 \quad 2^0$   
16      8      4      2      1

$$16+8+4+2+1=31=2^5-1=\mathbf{31}_{10}$$



Numero massimo che può essere rappresentato con 5 bit

**5 bit**

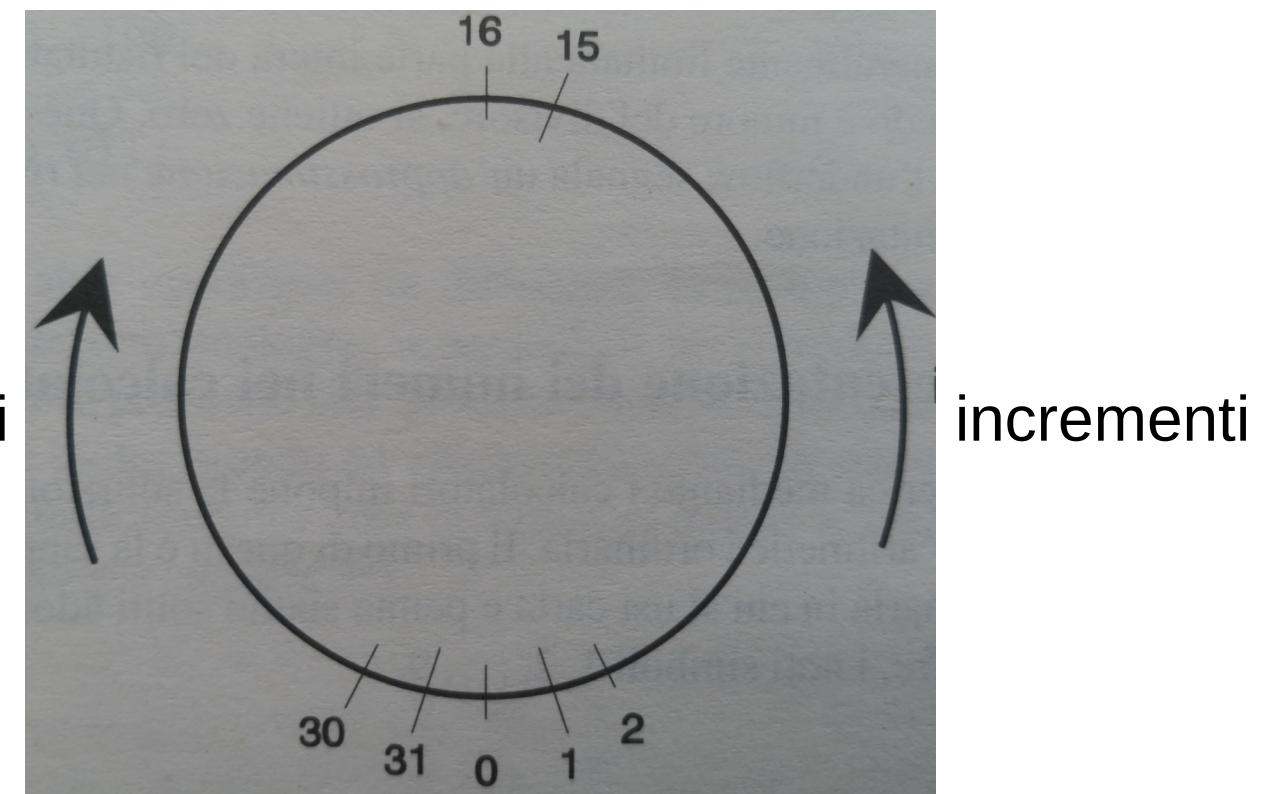
$$\begin{array}{r} 11111 \\ + 00001 \\ \hline \end{array}$$

**1 0 0 0 0 0**

$2^5 \ 2^4 \ 2^3 \ 2^2 \ 2^1 \ 2^0$

$= 0_{10}$

decrementi

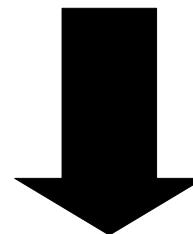


# 5 bit: rappresentazione in modulo e segno

<b>1</b>	<b>1</b>	<b>1</b>	<b>1</b>	<b>1</b>
----------	----------	----------	----------	----------

-       $2^3$      $2^2$      $2^1$      $2^0$   
      8        4        2        1

$$-(8+4+2+1) = -15_{10}$$



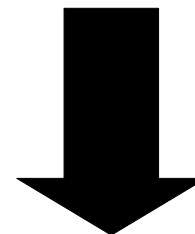
Numero negativo di modulo massimo  
che può essere rappresentato con 5 bit

## 5 bit: rappresentazione in modulo e segno

0	1	1	1	1
---	---	---	---	---

$$\begin{array}{r} + \quad 2^3 \quad 2^2 \quad 2^1 \quad 2^0 \\ \quad 8 \quad \quad 4 \quad \quad 2 \quad \quad 1 \end{array}$$

$$+(8+4+2+1)=+15_{10}$$



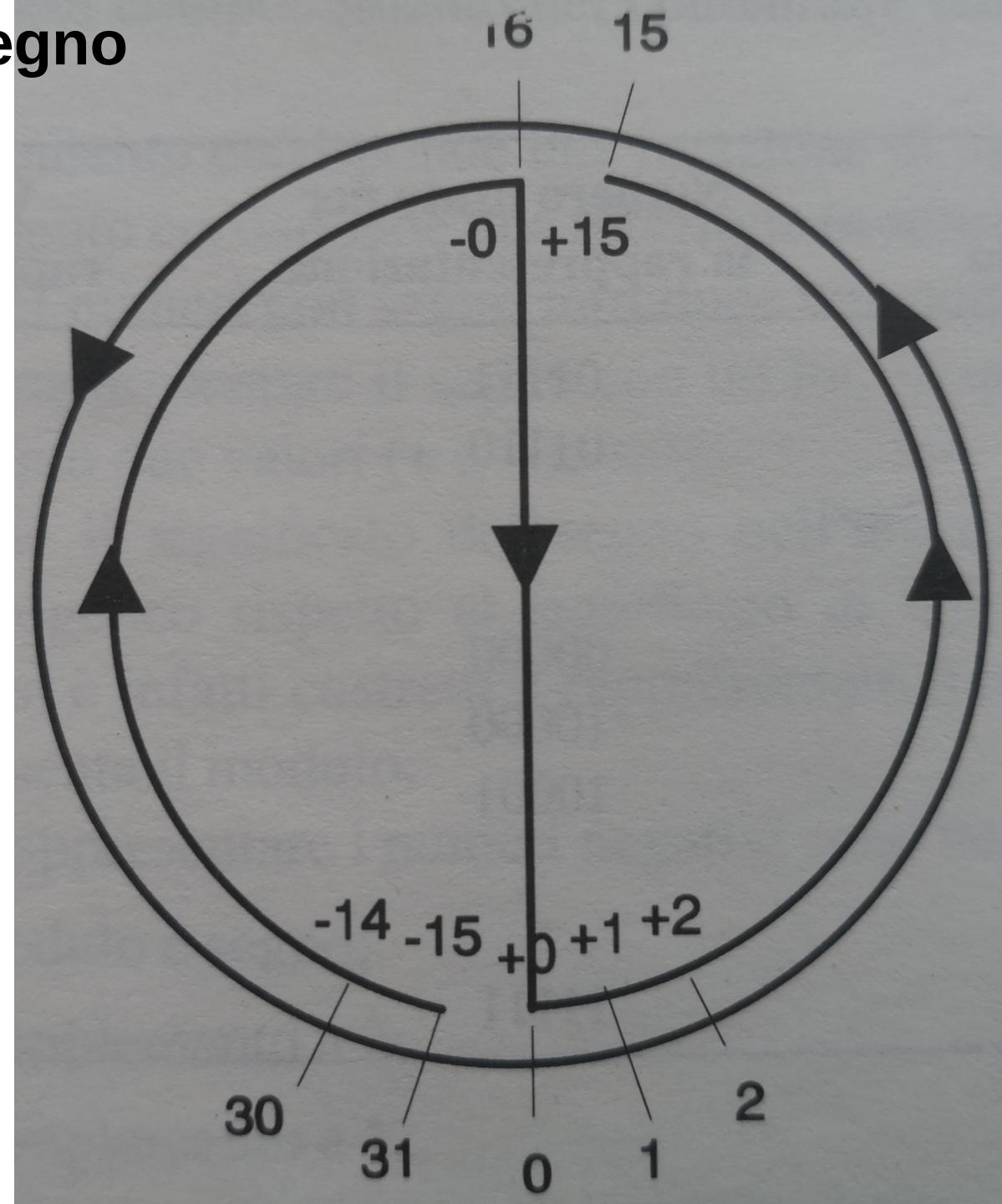
Numero positivo di modulo massimo  
che può essere rappresentato con 5 bit

## 5 bit: rappresentazione in modulo e segno

<b>1</b>	<b>1</b>	<b>1</b>	<b>1</b>	<b>1</b>
-	$2^3$	$2^2$	$2^1$	$2^0$
	8	4	2	1

$$-(8+4+2+1) = -15_{10}$$

$$\begin{array}{r} 11111 + -15_{10} \\ 00001 = \\ \hline 100000 + 0_{10} \end{array}$$



## 5 bit: rappresentazione in modulo e segno

1	0	0	0	0
---	---	---	---	---

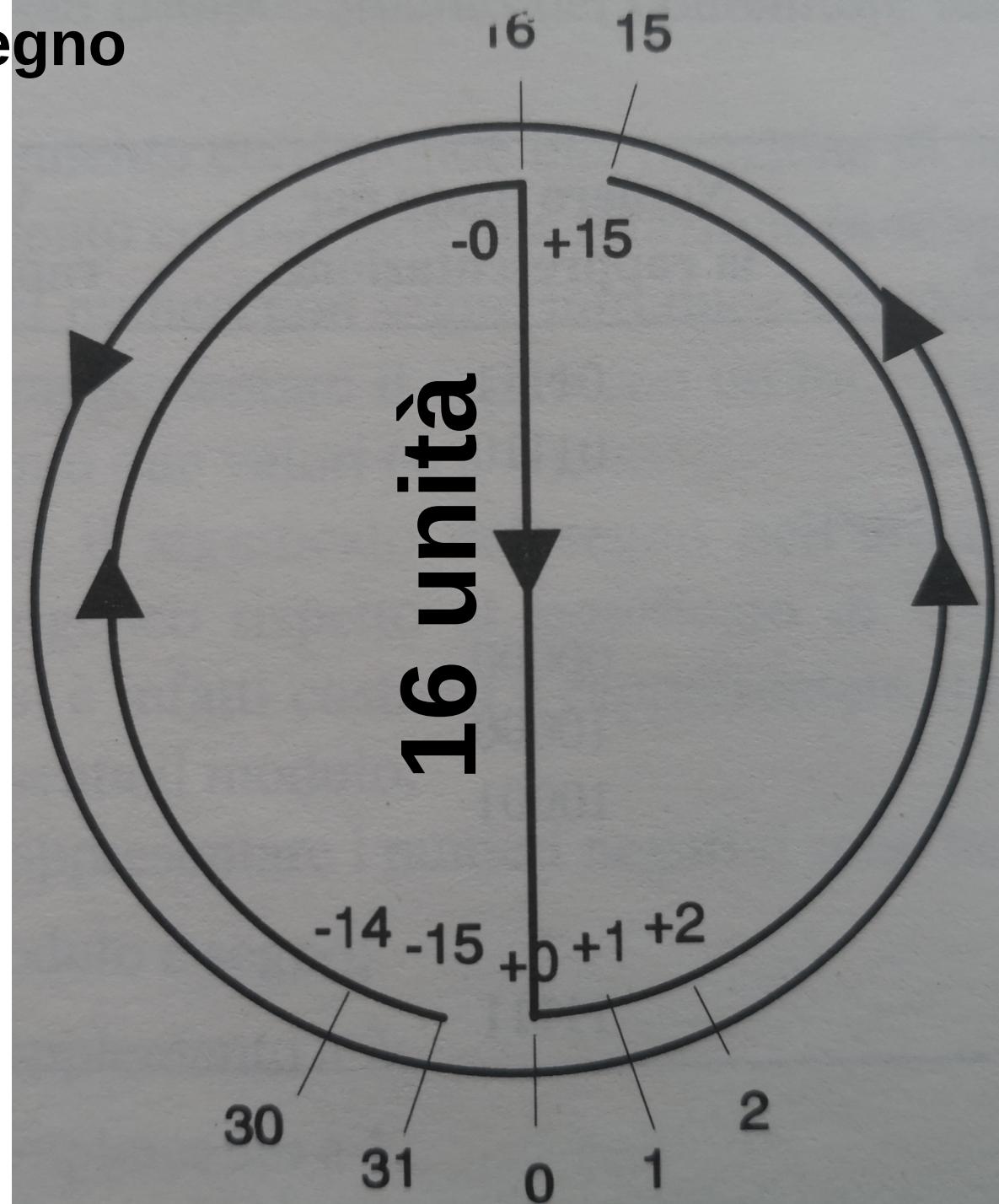
$$\begin{array}{ccccc} - & 2^3 & 2^2 & 2^1 & 2^0 \\ & 0 & 0 & 0 & 0 \end{array}$$

$$-(0+0+0+0) = -0_{10}$$

$$\begin{array}{r} 10000 + -0_{10} \\ 10000 = \end{array}$$

---

$$\begin{array}{r} 100000 + 0_{10} \\ \hline 100000 \end{array}$$



## 5 bit: rappresentazione in modulo e segno

1	0	0	0	1
---	---	---	---	---

$$\begin{array}{r} - \\ 2^3 \quad 2^2 \quad 2^1 \quad 2^0 \\ 0 \quad 0 \quad 0 \quad 1 \\ \hline -(0+0+0+1) = -1_{10} \end{array}$$

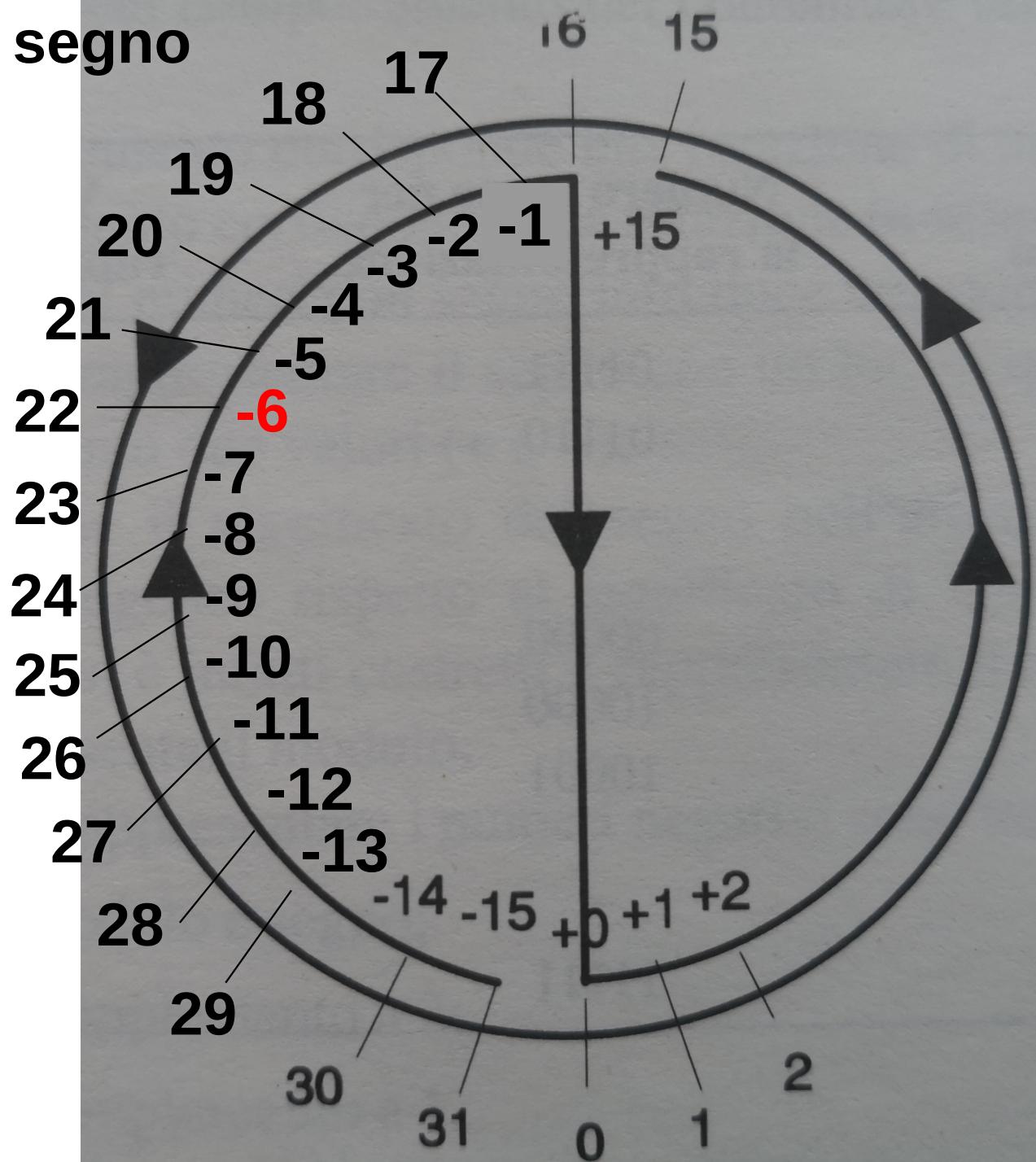
$$\text{N}^{\circ} \text{ rappresentante} = 17_{10}$$

$$\begin{array}{r} 10001 + \\ 00101 = \end{array} -1_{10}$$

$$\begin{array}{r} \\ \\ \\ \\ \hline \end{array} +5_{10}$$

$$\text{N}^{\circ} \text{ rappresentante} = 22_{10}$$

$$\begin{array}{r} 10110 \\ -6_{10} \end{array}$$



# Complemento di un numero

Dato un  $X$  in base  $r$  di  $n$  cifre nella parte intera, il complemento ad  $r$  è:

$$r^n - X$$

- Es.1)  $X = 64$ ,  $r = 10$ ,  $n = 2$ : il complemento a 10 è  $10^2 - 64 = 36$
- Es.2)  $X = 1630$ ,  $r = 10$ ,  $n = 4$ ; il complemento a 10 è

$$\begin{array}{r} 10000 - \\ 1630 = \\ \hline 8370 \end{array}$$

Quindi...

*Il complemento a 10 si trova analizzando le cifre a partire da destra: gli zeri fino alla prima cifra significativa si riportano tali e quali, della prima cifra significativa si fa il complemento a 10, di tutte le altre si fa il complemento a 9*

# Complemento di un numero

- Es.3)  $X = 01011$ ,  $r = 2$ ,  $n = 5$ ; il complemento a 2 sarà:

$$2^5 - 01011 =$$

$$100000 -$$

$$01011 =$$

---

**10101**

In pratica, si fa il complemento a 2 della prima cifra significativa da  $d_x$  e poi il complemento a 1 delle altre

- Es.4)  $X = 1011000$ ,  $r = 2$ ,  $n = 7$ ; il complemento a 2 sarà:

$$2^7 - 1011000 =$$

$$10000000 -$$

$$1011000 =$$

---

**0101000**

# Complemento alla base – 1 (o alla base diminuita)

Dato un numero  $X$  in base  $r$  di  $n$  cifre nella parte intera, il complemento alla base diminuita è:

$$(r^n - 1) - X$$

- Es.1)  $X = 64$ ,  $r = 10$ ,  $n = 2$ , il complemento a 9 è:  $(10^2 - 1) - 64 = 99 - 64 = 35$
- Es.2)  $X = 01011$ ,  $r = 2$ ,  $n = 5$ , il complemento a 1 è:  
 $(2^5 - 1) - 01011 = (100000 - 1) - 01011 = 11111 - 01011 = \mathbf{10100}$   
che si ottiene *complementando a 1 ogni singolo bit*

In **R**:

```
> 2^5 - 1
[1] 31
> decimal2binary(31,10)
[1] 0 0 0 0 0 1 1 1 1 1
>
```

# Complemento alla base – 1 (o alla base diminuita)

In Python:

```
piero@piero-XPS-9320:~$ python3
Python 3.11.4 (main, Dec 7 2023, 15:43:41) [GCC 12.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print(bin(31))
0b11111
>>>
```

*Il complemento alla base può essere ricavato dal complemento alla base diminuita sommando a quest'ultimo un 1 nella posizione meno significativa (più a destra)*

# Rappresentazione in complemento a 2 (dei numeri negativi) con un numero fisso di cifre

$X$  = numero intero senza segno

$$-X = 0 - X$$

Per  $n = 5$ ,  $2^n = 2^5 = 32 = \textcolor{red}{1}00000$ , quindi 0 e  $2^n$  sono congruenti modulo  $2^n$ ; quindi:

$$0 - X = 2^n - X$$

Complemento a 2 di  $X$

*I numeri positivi sono rappresentati dal loro modulo e hanno il bit più significativo (segno) uguale a zero; i numeri negativi sono rappresentati dal complemento a 2 del corrispondente numero positivo, segno compreso.*

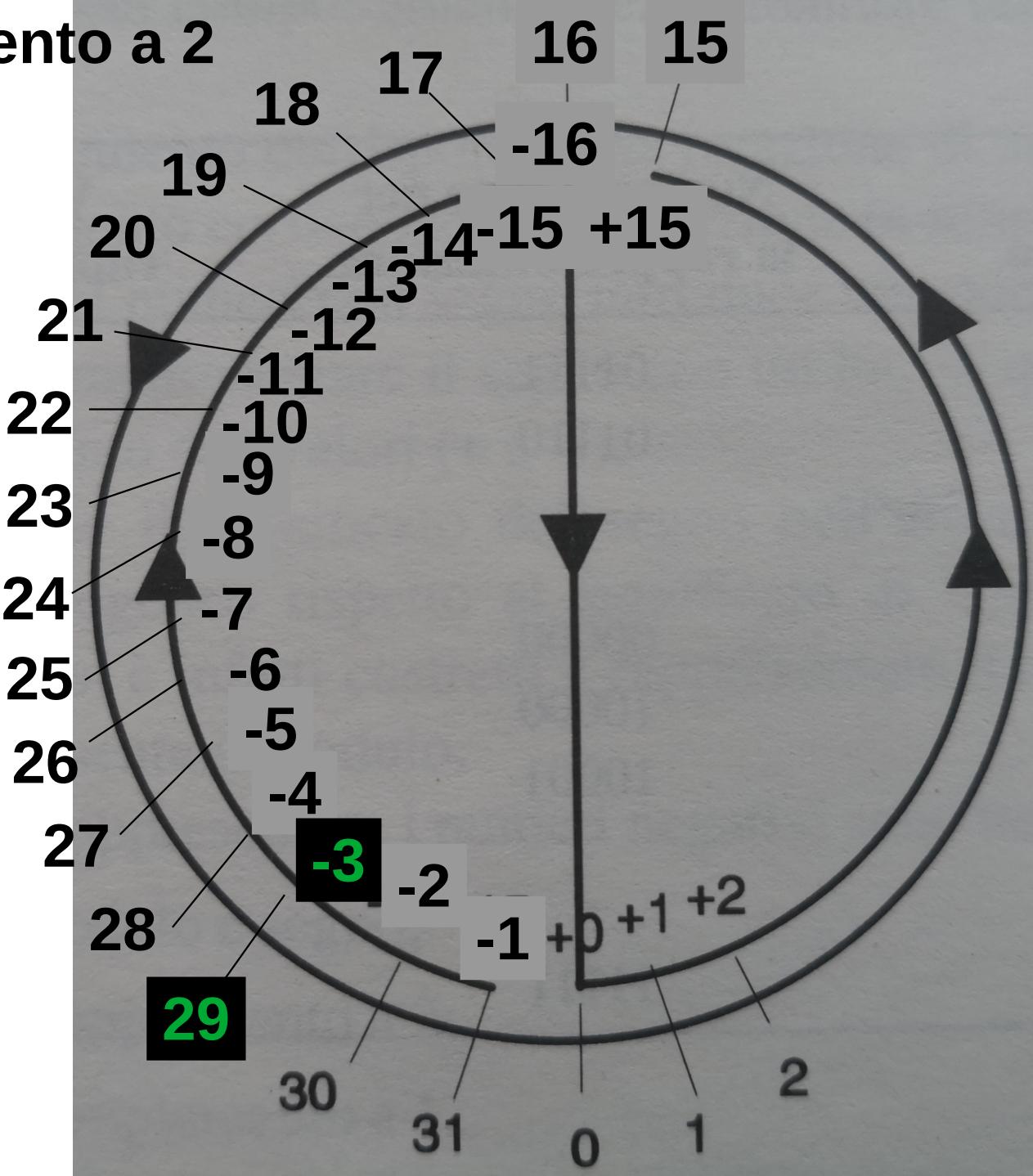
- Es.  $+3 \rightarrow 00011_2$
- $-3 \rightarrow 11101_2$

Complemento a 2 di  $+3$

In Python:

```
>>> print(0b11101)
29
>>>
```

# 5 bit: rappresentazione in complemento a 2



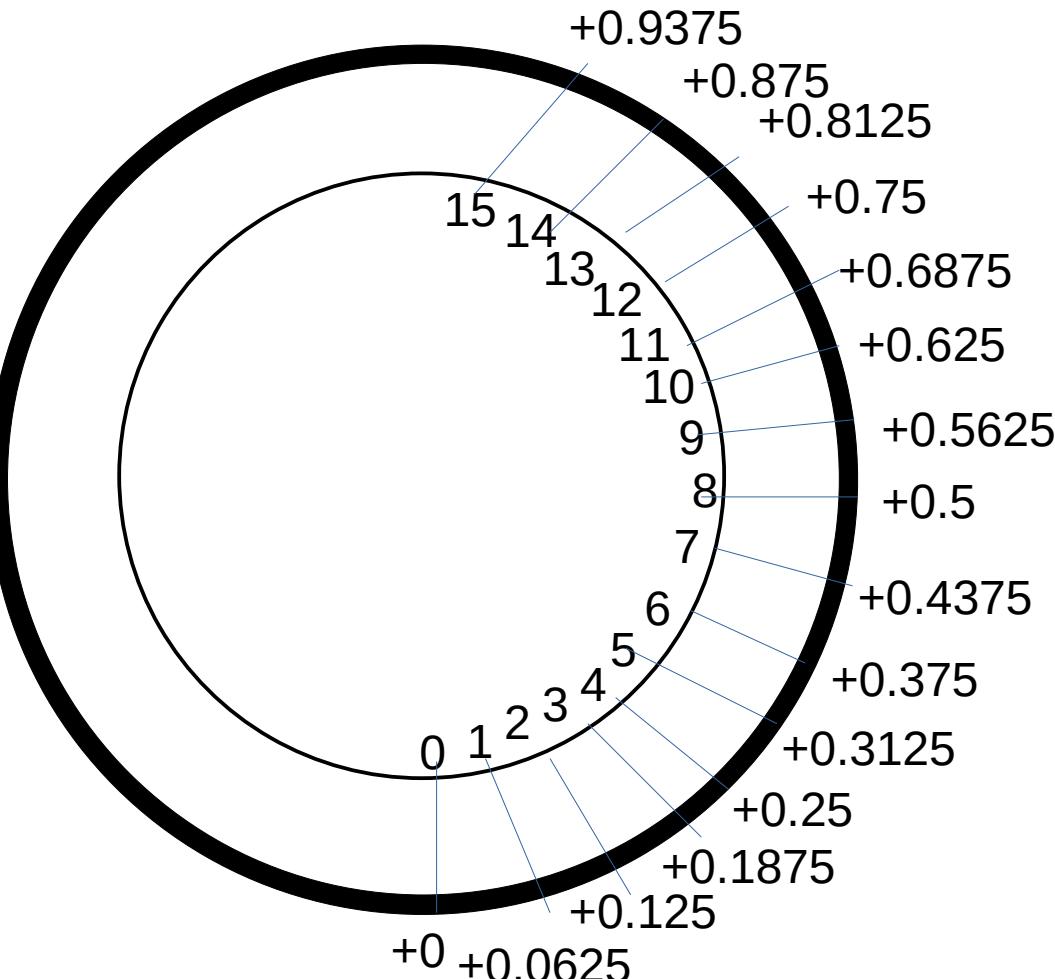
# Rappresentazione dei numeri positivi con parte frazionaria

$$y_0 = 0$$

0	1	1	1	1
$2^0$	$2^{-1}$	$2^{-2}$	$2^{-3}$	$2^{-4}$

Punto decimale  
 $0 + 0.5 + 0.25 + 0.125 +$   
 $+ 0.0625 = 0.9375$

```
>>> print(0b1111)  
15  
>>>
```



0	1111	0.9375	15
0	1110	0.875	14
0	1101	0.8125	13
0	1100	0.75	12
0	1011	0.6875	11
0	1010	0.625	10
0	1001	0.5625	9
0	1000	0.5	8
0	0111	0.4375	7
0	0110	0.375	6
0	0101	0.3125	5
0	0100	0.25	4
0	0011	0.1875	3
0	0010	0.125	2
0	0001	0.0625	1
0	0000	0.000	0

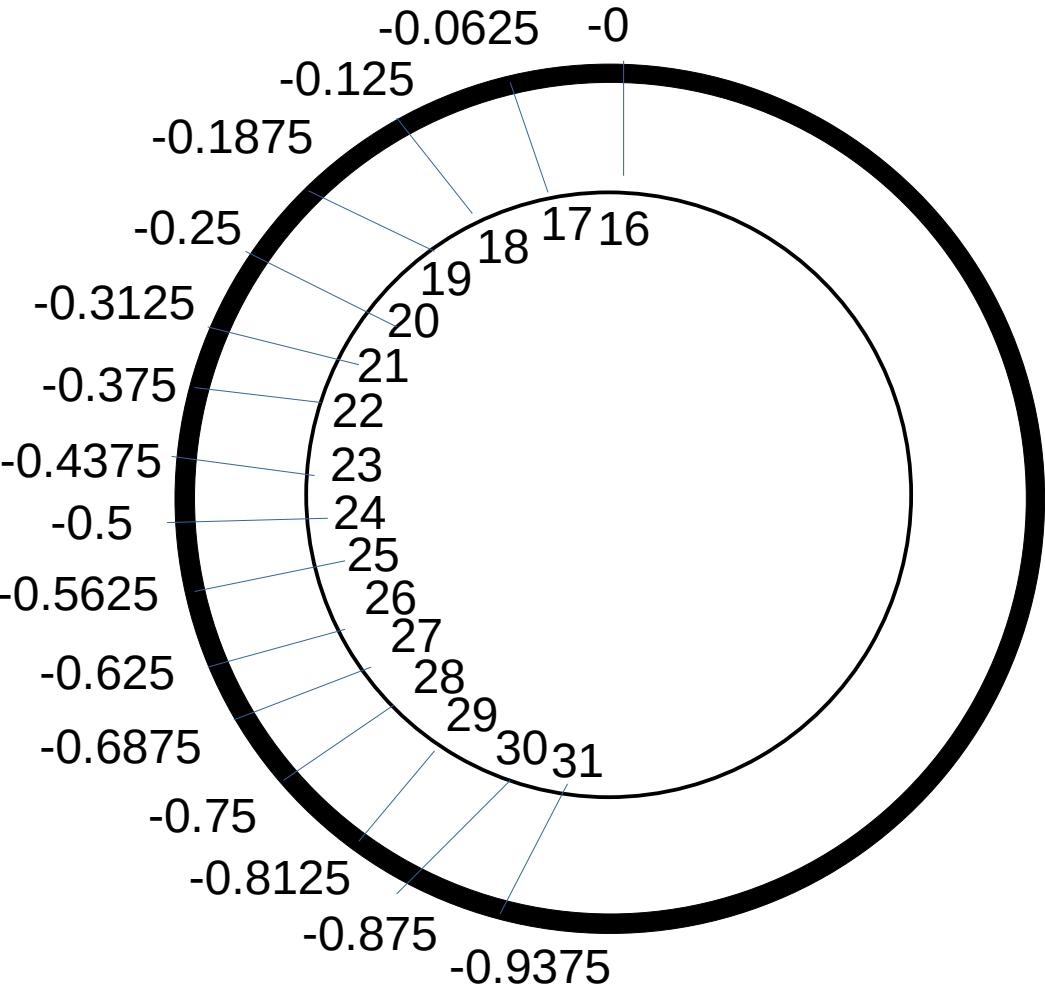
# Rappresentazione dei numeri negativi con parte frazionaria

$$y_0 = 1$$

1	1	1	1	1
$2^0$	$2^{-1}$	$2^{-2}$	$2^{-3}$	$2^{-4}$

Punto decimale  
 $1 + 0.5 + 0.25 + 0.125 +$   
 $+ 0.0625 = 1.9375$

```
>>> print(0b10000)  
16  
>>>
```



1	1111	1.9375	31
1	1110	1.875	30
1	1101	1.8125	29
1	1100	1.75	28
1	1011	1.6875	27
1	1010	1.625	26
1	1001	1.5625	25
1	1000	1.5	24
1	0111	1.4375	23
1	0110	1.375	22
1	0101	1.3125	21
1	0100	1.25	20
1	0011	1.1875	19
1	0010	1.125	18
1	0001	1.0625	17
1	0000	1.000	16

# Rappresentazione dei numeri in virgola mobile

$$13.25_{10} = \mathbf{1101.01}_2$$

$$1*2^3 + 1*2^2 + 0*2^1 + 1*2^0 + 0*2^{-1} + 1*2^{-2} = 8 + 4 + 1 + 0.25 = 13.25$$

Mantissa      Esponente, in base 10, della forma normalizzata  
 $\mathbf{1101.01}_2 = \mathbf{1.10101}_2 \cdot \mathbf{2^3}$

# spostare la virgola di 3 posizioni verso sinistra significa dividere per  $2^3$   
# dividiamo e moltiplichiamo per  $2^3$  “normalizzando il numero”

# calcoliamo l'esponente sommando 127 all'esponente della forma normalizzata:

$$(3+127)_{10} = \mathbf{130}_{10} = \mathbf{10000010}_2$$

$$1*2^7 + 1*2^1 = 128 + 2 = 130$$

```
>>> print(bin(130))
```

```
0b10000010
```

```
>>>
```

# Rappresentazione dei numeri in virgola mobile: single precision

	SEGNO	ESPOENTE	MANTISSA	
	0	10000010	1010100000000000000000000	
bit	1	8	23	32

Il bit di segno è definito pari a:

- 0 per valori  $\geq 0$
- 1 per valori  $\leq 0$
- La mantissa è in forma normalizzata 1.xxxxx
- Essendo quell'1 sempre presente, solo la parte frazionaria del numero è memorizzata
- L'1 omesso viene detto bit nascosto (hidden bit)
- Abbiamo, così, un bit in più di precisione (23 bit memorizzati + 1 nascosto = 24 bit effettivi)

# Bus sincroni

- es.: clock a 100 MHz → un ciclo ogni **in R:**

$$> 1 / (10^8)$$

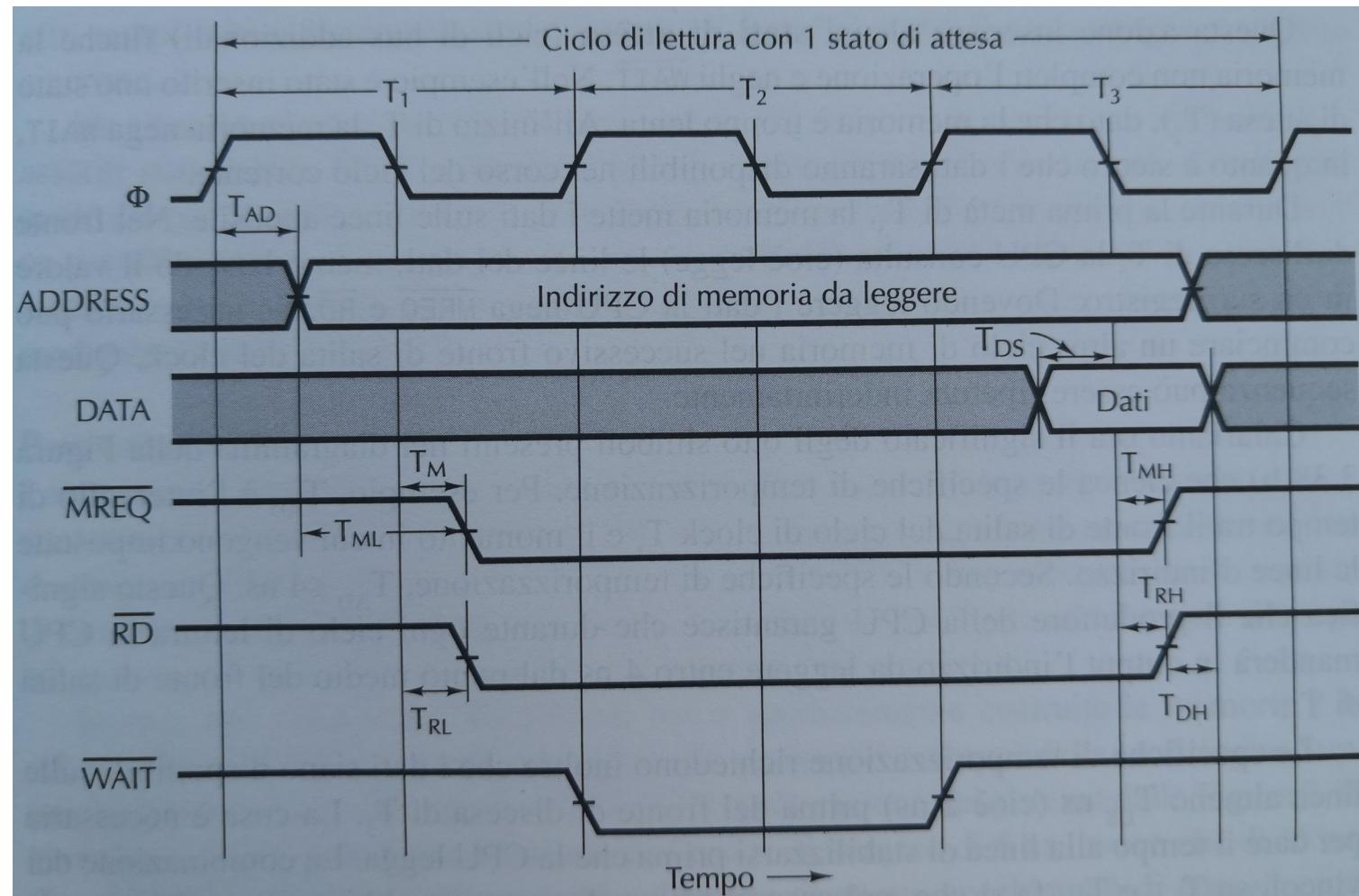
[1]  $1e-08$

>

$$10^{-8} = 10 * 10^{-9} = 10 \text{ ns}$$

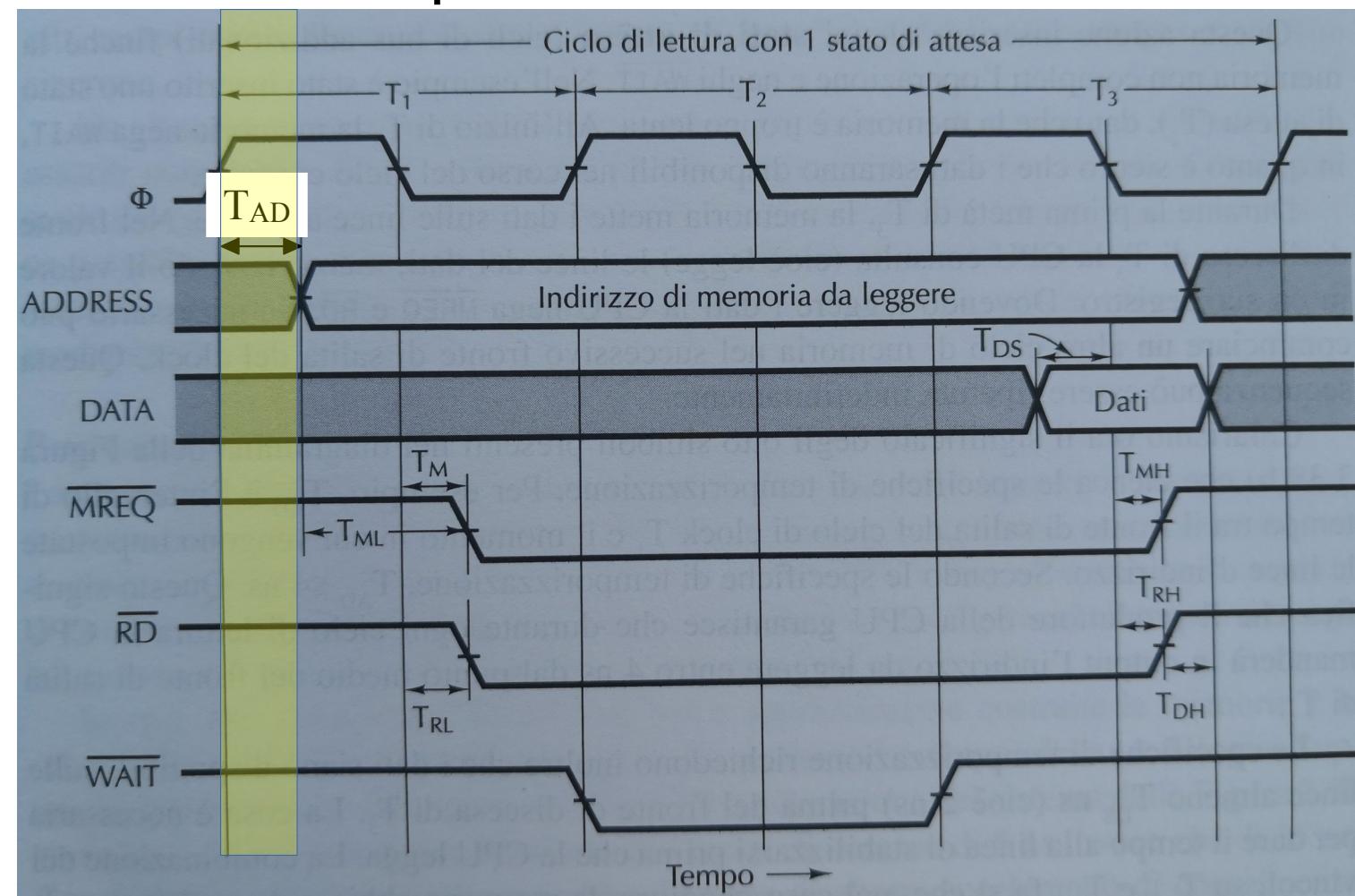
**10 ns;**

- Fronti di salita e discesa obliqui perché il segnale elettrico impiega un certo tempo a cambiare il proprio valore, per es. 1 ns



# Bus sincroni

- La CPU fornisce l'indirizzo della parola sulle linee ADDRESS, rappresentate da 2 linee che si incrociano quando l'indirizzo cambia
- $T_{AD}$  = ritardo dell'output dell'indirizzo (4 ns)



# Bus sincroni

- MREQ ed RD vanno a 0 (zero), ossia vengono “asserite”

