ARCHITETTURA DEGLI ELABORATORI

OBIETTIVO

Si occupa della **struttura** e del **funzionamento** dei **primi 2 livelli** del modello ISO-OSI:

- **Livello 1** (**Fisico**) di **trasmissione dei dati** tramite i **canali** di **comunicazione**:
  - **Fisico**: Cavi e Fibra Ottica: **segnali elettrici** o **ottici**
  - **Etere**: **Wireless**: **onde elettromagnetiche**
- **Livello 2 (Collegamento): interfaccia** con i **canali**, **multiplazione**, **framing**, **controllo** e **correzione** degli **errori**.

Comprensione della **struttura fisica dell'Hardware** e del **Software** di **basso livello** che si interfaccia all'HW.

ARGOMENTI DEL CORSO

- Memorie, circuiti sequenziali e combinatori, macchine a stati
- Architettura delle CPU
- Programmazione Assembly
- Periferiche e Processi gestiti da CPU
- BUS e comunicazione logica e fisica tra le varie periferiche HW
- Organizzazione e Accesso in memoria
- Multiprocessi, Multithread e CPU con multicore paralleli.

STORIA DEI CALCOLATORI

1642 Calcolatrice di **Pascal** a **8 cifre** per **Somme Algebriche**

1670 Leibniz Calcolatrice **meccanica** per **Somme** e **Prodotti**

1834 Babbage -> Calcolatrice meccanica **Programmabile**

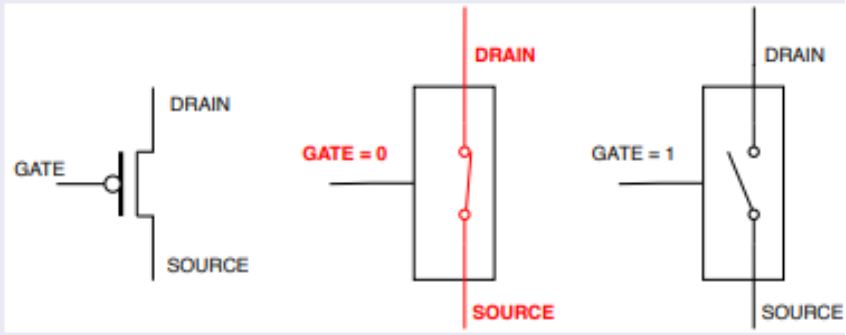
1930 Zuse -> Calcolatrice a **Relè** (**valvole elettromeccaniche**).

1944 Aiken in Gran Bretagna crea la **Mark I**, **macchina** programmabile **con crittografia** usata per la **decodifica** dei **messaggi segreti** dei **nazisti** durante la **WW2**

1946 USA Eokert e **Mauchly** creano la **ENIAC**, calcolatrice con **TRIODI**

1951 Von Neumann crea la prima macchina con una architettura che è **in uso ancora oggi**. In **memoria risiedono** sia i **programmi** che i **dati**, **separatamente**.

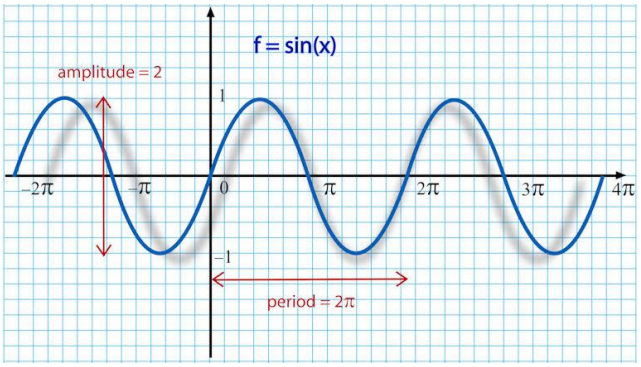
1956 TX-0 prima **calcolatrice** a **Transistors.** Questi componenti sono tuttora in uso e hanno dato origine alla **legge di Moore**: “la complessità di un circuito, misurata in numero di transistor per chip (microcircuito integrato), quadruplica ogni 3 anni”



SEGNALI ELETTRICI

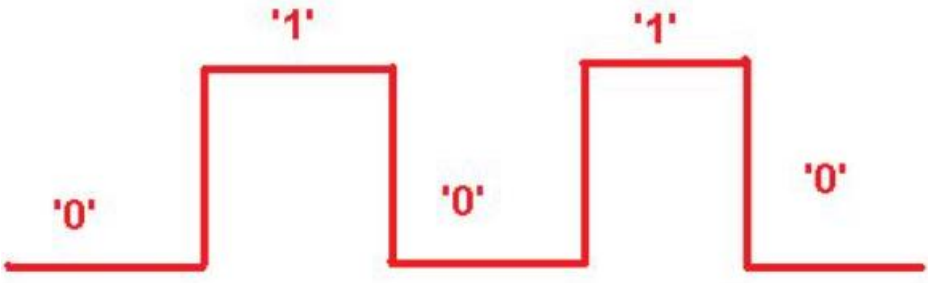
All’interno di un **calcolatore viaggiano segnali** di **natura elettrica**. All’interno di questi segnali è racchiuso il contenuto informativo da trasmettere.

- **Analogici**: variano in analogia con la grandezza fisica (Corrente o Tensione), Possiedono un **numero infinito** di **valori all’interno dell’intervallo** di Tensione Vengono rappresentati da **un’onda sinusoidale**.



V(t) = Vp * sin (2π* F * t)

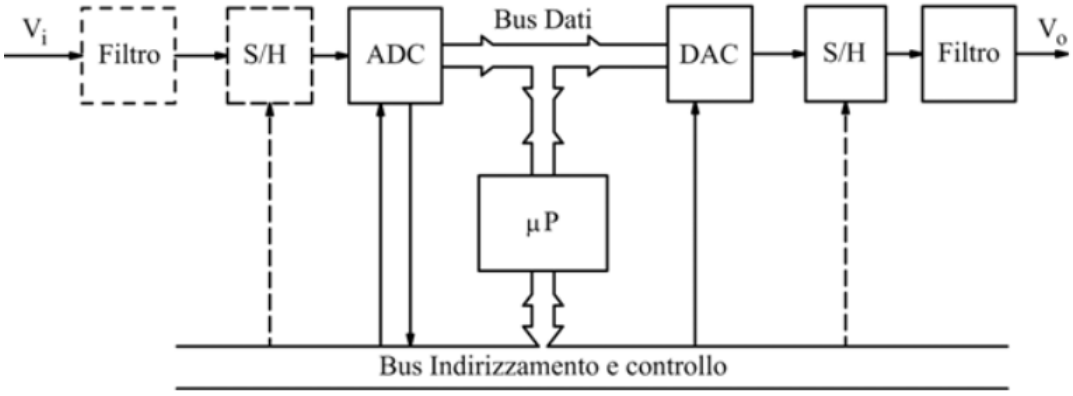
- **Digitali**: possiedono un **numero finito** di **valori** nell’intervallo, espressi in cifre (“digit”). Vengono rappresentati da **onde quadre.**



I valori digitali possono essere espressi da più cifre, e il numero di Stati (M) = 2^nbit. Nbit è il numero di bit con cui si vuole codificare i valori del segnale.

I **segnali digitali** sono **maggiormente immuni** ai **rumori**, infatti il valore di tensione **dell'interferenza talvolta non** è **sufficiente** a **superare** la **soglia intermedia** tra due valori digitali.

ADC = Analog to Digital Conversion



- **Filtro anti Aliasing:** la frequenza di campionamento deve rispettare il teorema di Shannon: Fc >= 2*Fmax Fmax = frequenza massima dello spettro del S analogico
- **S/H campionamento:** in base alla Fc viene letto il valore analogico e mantenuto nel tempo. Si da origine a un segnale discreto nel tempo ma non nelle ampiezze.
- **ADC:** ogni valore campionato viene associato a un intervallo di valori digitali affinché possa essere rappresentato correttamente

CIRCUITI LOGICI

Lo scopo principale è quello di **produrre** degli **output** (risultati) **in base** agli **input inseriti**. Dati dei valori di **input fissi** anche gli **output** devono essere **fissi**.

- **Sequenziali**: **l'output dipende solo dall'input**
- **Combinatori**: possiedono un **sistema** di **memoria**, perciò un **output dipende anche** dai **valori** di input/output **precedenti**

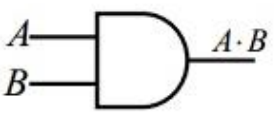
Questi **circuiti** sono **realizzati combinando** tra loro diverse **porte logiche.**

PORTE LOGICHE

Sono dei particolari circuiti che hanno delle **regole fisse** per **produrre** gli **output**.

2 INPUT -> 1 OUTPUT

AND

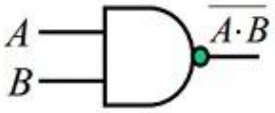


A	B	$A \cdot B$
0	0	0
0	1	0
1	0	0
1	1	1

OR

A	B	$A+B$
0	0	0
0	1	1
1	0	1
1	1	1

NAND

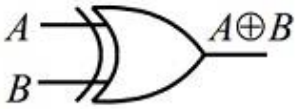


A	B	$\overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

NOR

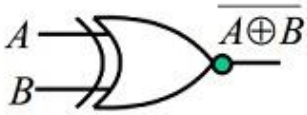
A	B	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

EXOR



A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

EXNOR



A	B	$\overline{A \oplus B}$
0	0	1
0	1	0
1	0	0
1	1	1

Le **PORTE LOGICHE** possono essere **realizzate** anche mediante dei circuiti fisici, usando dei **transistor**.

PROPRIETÀ LOGICHE

- **DUALITÀ: un'espressione logica si dice duale quando invertendo AND con OR e 0 con 1 si ottiene lo stesso OUTPUT**
- **DE MORGAN:** **not(A) OR not(B) = not(A AND B)**

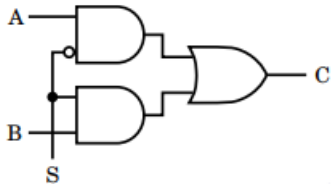
  not(A OR B) = not(A) AND not(B)

Elemento nullo	$0 + A = A$
Identità	$1 + A = 1$
Idempotenza	$A + A = A$
Inverso	$A + \overline{A} = 1$
Commutatività	$A + B = B + A$
Associatività	$(A + B) + C = A + (B + C)$
Distributività	$A(B + C) = AB + AC$
Assorbimento	$A + (A \cdot B) = A$
De Morgan	$\overline{A + B} = \overline{A} \cdot \overline{B}$
Negazione	$\overline{\overline{A}} = A$

Identità	$1 \cdot A = A$
Elemento nullo	$0 \cdot A = 0$
Idempotenza	$A \cdot A = A$
Inverso	$A \cdot \overline{A} = 0$
Commutatività	$A \cdot B = B \cdot A$
Associatività	$(A \cdot B) \cdot C = A \cdot (B \cdot C)$
Distributività	$A + BC = (A + B)(A + C)$
Assorbimento	$A \cdot (A + B) = A$
De Morgan	$\overline{A \cdot B} = \overline{A} + \overline{B}$
Negazione	$\overline{\overline{A}} = A$

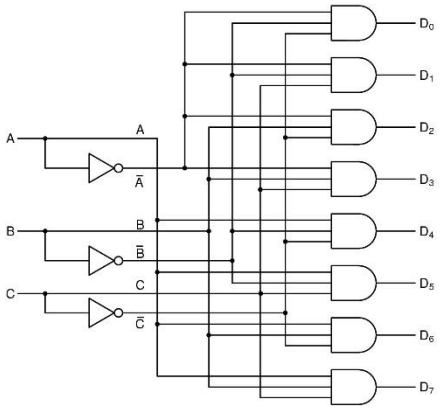
CIRCUITI LOGICI DI BASE

- **MULTIPLEXER: N segnali di controllo (S), 2^N segnali di ingresso, 1 uscita segnale. Lo scopo è di selezionare tramite S quale segnale in ingresso mandare in OUTPUT.**
- **DEMULTIPLEXER: N segnali controllo (S), 2^N uscite di segnale, 1 segnale ingresso: Si sceglie in quale delle uscite mandare il segnale di ingresso-**



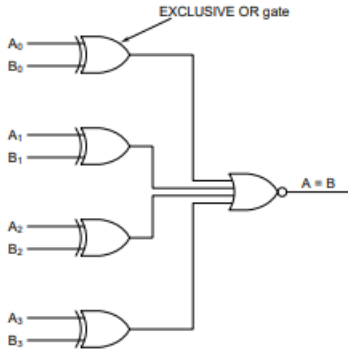
Quando S = 1 la AND prende il valore dell'ingresso

- **DECODER: N ingressi di controllo, 2^N uscite**
  Si attiva l'uscita corrispondente alla codifica in binario degli ingressi di controllo.



- **ENCODER: N uscite, 2^N ingressi di segnale**
  Codifica in binario il numero dell'ingresso ATTIVO

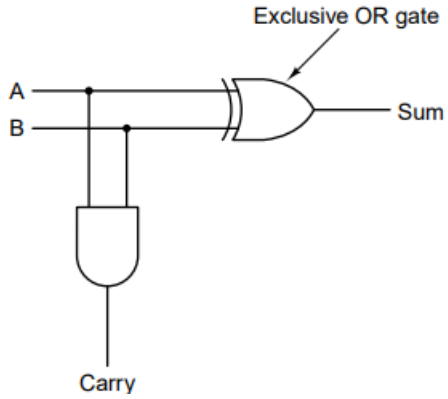
- **COMPARATORE:** serve a **verificare** che delle **sequenze di N bit siano uguali**



La **porta finale** è una **NOR**, perciò **tutti** i suoi **ingressi** devono essere a **0 affinché produca 1.**
Se **A0** e **B0** sono **uguali** la **XOR** produce **0, altrimenti 1.**
Avendo **1** la **NOR** darà in **uscita 0**, **perciò** le due **sequenze** sono **diverse**.

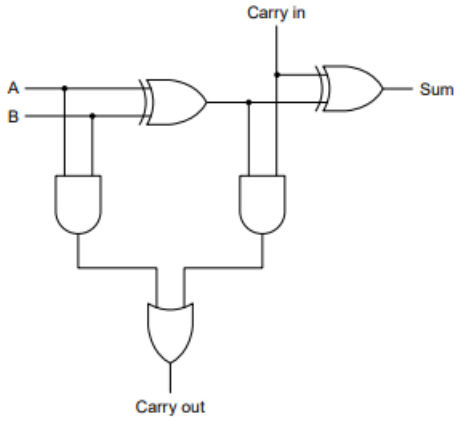
- **MEZZO SOMMATORE: effettua la somma tra due numeri, riportando risultato e eventuale riporto**

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

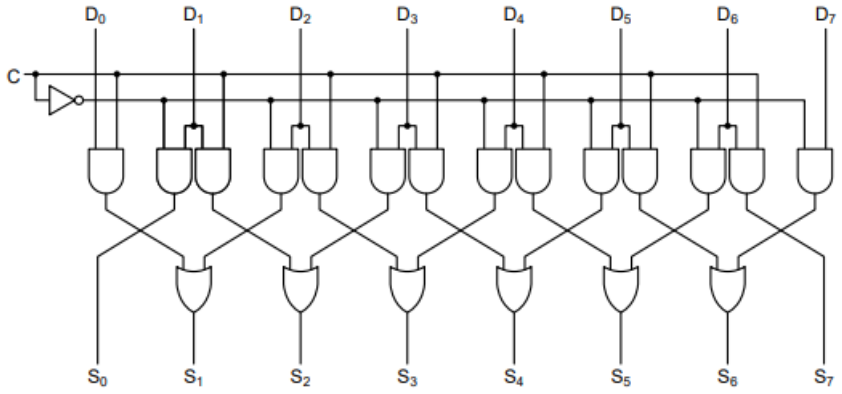


- **SOMMATORE COMPLETO: effettua la somma contando anche dell'eventuale riporto precedente**

A	B	Carry in	Sum	Carry out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



- **SHIFTER: serve a traslare di 1 posizione verso DX (C=1) o SX (C=0)**



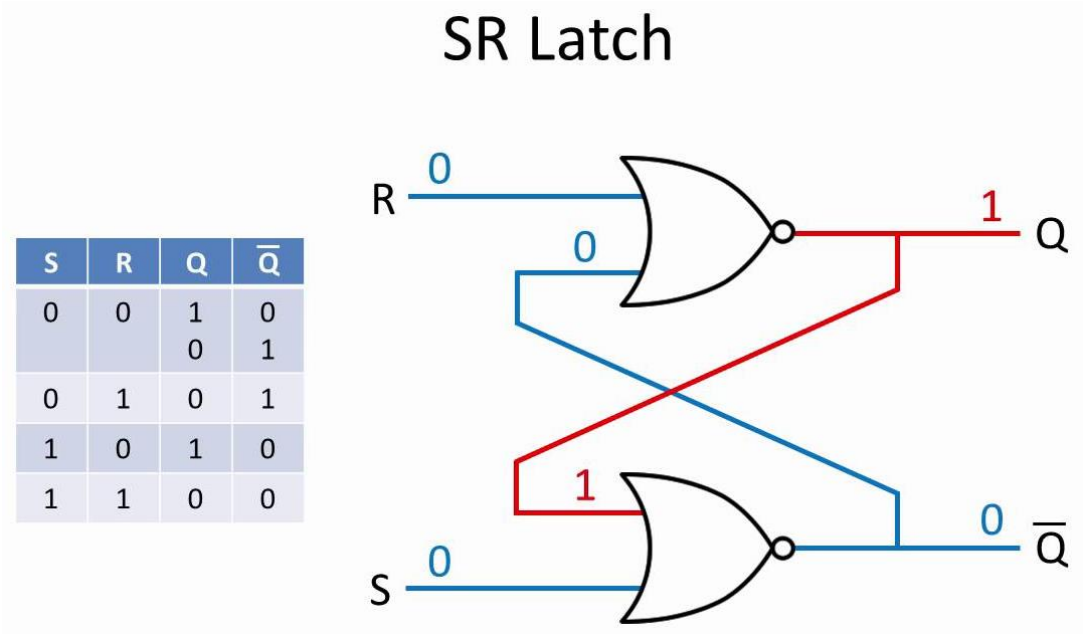
RITARDI DEI CIRCUITI

I **circuiti logici possiedono** un lieve **ritardo** nel **calcolo** dei risultati, nell'ordine di **0,1ns** = 10^(-10)s. Può sembrare un numero insignificante, ma è **molto influente**, infatti il **PERIODO** di **CLOCK** della **CPU** si aggira **attorno** a **1ns** (nel caso di F = 1GHz). Con **frequenze maggiori** il **periodo diminuisce**, **aumentando l'effetto** del **ritardo**.

CIRCUITI COMBINATORI (CON MEMORIA)

Questi circuiti producono un risultato in base al valore precedentemente calcolato.

Retroazione: l'output viene riusato come input



S	R	Q	$\overline{Q}$
0	0	1 0	0 1
0	1	0	1
1	0	1	0
1	1	0	0

S=SET: se **S=1** imposta il valore di **Q = 1**

R=RESET se **R=1** imposta il valore di **Q = 0**

Q = valore di **output** che andrà successivamente letto

Quando **S=R=0** si è in una condizione di **LATCH**, cioè il valore di **Q dipende dal valore precedente**; infatti, in una NOR il valore 0 determina che nell'uscita ci sia il valore dell'altro ingresso.

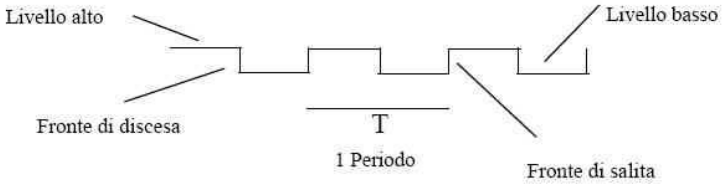
Quando S=R=1 si è in una condizione non accettabile; infatti, Q e not(Q) non possono avere lo stesso valore.

1 LATCH MEMORIZZA 1 BIT

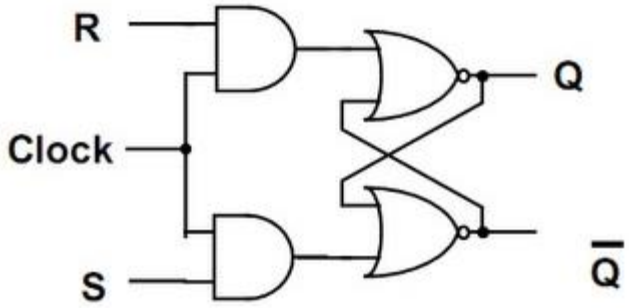
CLOCK

Rappresenta il segnale di sincronismo, solitamente onda quadra, che fa da riferimento per tutte le operazioni che avvengono all'interno del PC.

Il clock è la frequenza della CPU, cioè quante volte al secondo è in grado di eseguire le sue operazioni principali (Fetch, Decode e Execute).

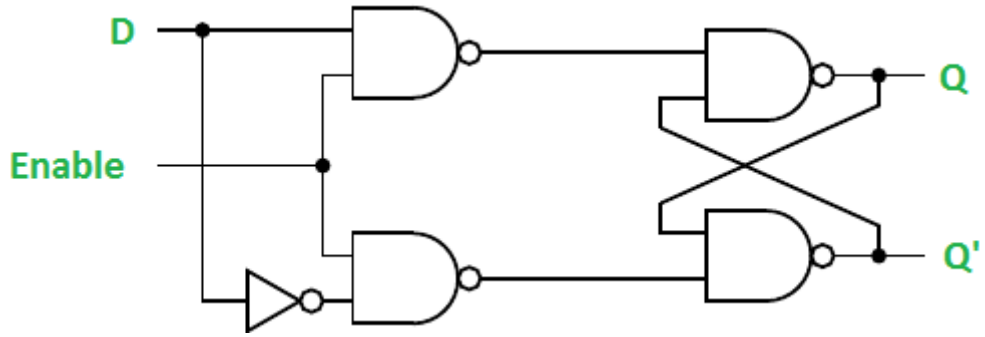


LATCH SR SINCRONO



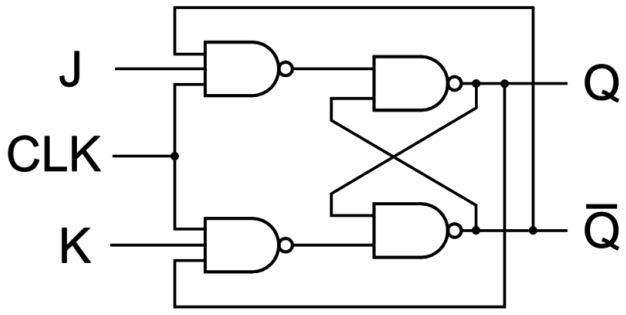
Si comporta come il circuito precedente; l'unica differenza sta nella presenza di un segnale di abilitazione, rappresentato dal CLK. Quando il CLK è alto (=1) abilita la AND di S o R, in base a quale dei 2 viene attivato.

LATCH D



La variabile D sostituisce S e R, e elimina il caso di S=R, in modo che le uniche operazioni che possono avvenire siano il Set e il Reset.

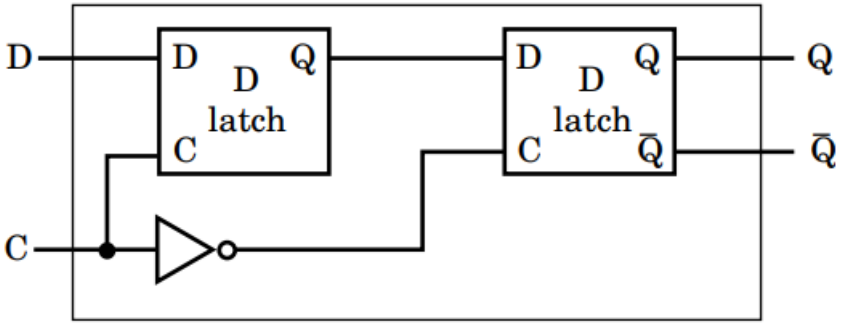
FLIP – FLOP



A differenza del Latch, il FLIP-FLOP si attiva quando il CLK è in uno dei due fronti, salita o discesa. La logica di funzionamento resta invariata.

- LATCH : level-triggered (azionato dal valore del livello, alto o basso)
- FLIP-FLOP: edge-triggered (azionato dal fronte di salita o discesa)

ARCHITETTURA MASTER – SLAVE



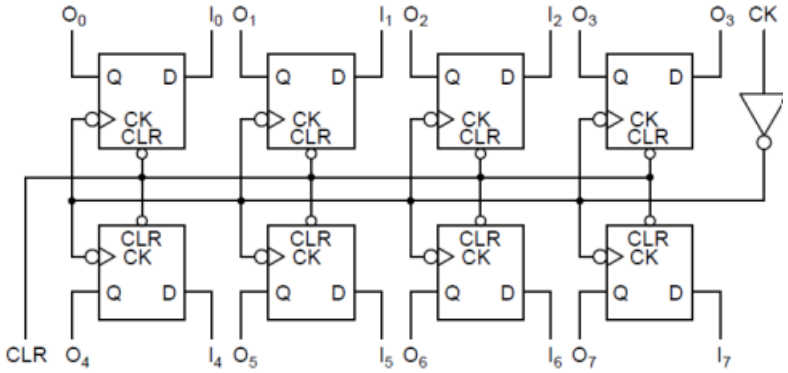
Il latch più a SX rappresenta il MASTER, e il suo valore di uscita aziona lo SLAVE.

Come si nota il CLK viene negato nello SLAVE, in modo che i due circuiti si attivino separatamente.

REGISTRO

Insieme di FLIP-FLOP / LATCHES che possono memorizzare sequenze di bit.

Un registro contiene 1 bit



MACCHINA A STATI FINITI

In ogni istante la macchina si trova in una determinata situazione, descritta da:

- Configurazione dello **stato**
- **Valore** delle variabili di **input**

Di conseguenza a ogni ciclo di CLOCK può variare:

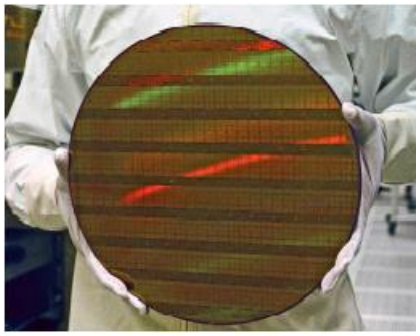
- **Stato**
- Valore di **output**

Perciò **a ogni ciclo** lo **stato** della macchina **si evolve** in ogni caso, **anche se** non **cambia l'output**.

CIRCUITI INTEGRATI (CHIP)

Sono delle unità fisicamente separate dal resto e possiedono uno scopo ben preciso.

Vengono fatti interagire con altri CHIP o parti dell'HW di un calcolatore.



- Sono delle **piastrine** di **Silicio** su cui sono **incisi transistor, condensatori** e **resistori**.
- I **componenti** elettronici sono **ottenuti esponendo** il **Silicio** ad altri agenti, quali **Boro**, **Arsenico** o **Fosforo** (fenomeno della **drogatura**)
- I **collegamenti fisici** sono ottenuti mediante **Rame**

FOTOLITOGRAFIA: fenomeno tramite cui il circuito viene coperto da materiali fotosensibili e successivamente esposti a fonti luminose… la parte di circuito illuminata si solidifica.

Successivamente vengono inseriti all'interno di celle frigorifere a bassissime temperature, attorno a -270°C, circa 0 Kelvin, in modo che tutti i componenti si solidificano e si resettano. A quelle temperature la corrente non passa.

CHIP DI MEMORIA

I **chip** di **memoria** possono **contenere moltissimi registri**, ma essi **non** sono **accessibili individualmente**. I **registri** sono **raggruppati** in **locazioni** di **memoria**, **accessibili tramite l'indirizzo** di memoria.

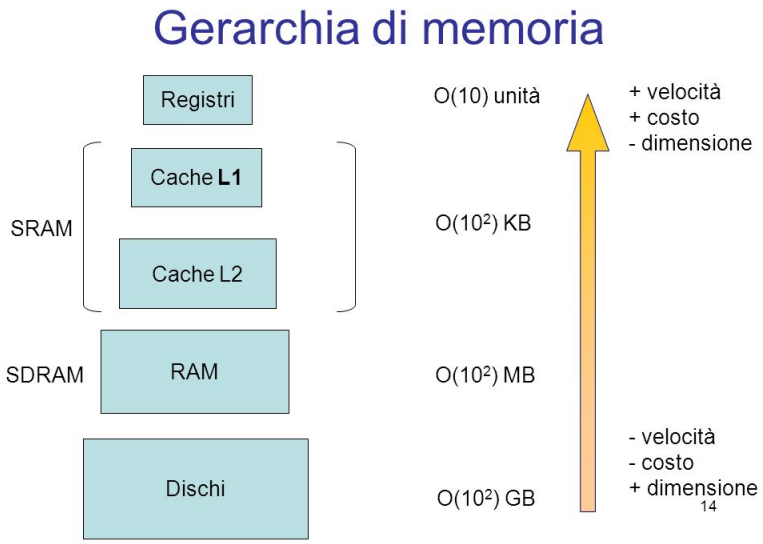
- **Indirizzo di memoria** che identifica una locazione -> **BUS INDIRIZZI**
- **Input:** dato da SCRIVERE in memoria -> **BUS DATI**
- **Output:** dato LETTO dalla memoria -> **BUS DATI**
- **Segnali di controllo** provenienti dal -> **BUS CONTROLLO**
    - **Chip Select:** seleziona quale chip di memoria contattare
    - **Write Enable:** concede il permesso di scrittura in memoria
    - **READ or WRITE:** specifica quale operazione andare ad effettuare.
        - In base alla operazione il **BUS DATI** si comporterà come **input** o **output**

MEMORIA RAM (RANDOM ACCESS MEMORY)

La memoria RAM è la memoria principale del PC, sul quale la CPU va a cercare la maggior parte dei dati o programmi che gli servono in ogni istante.

Esiste una gerarchia di memoria all'interno di un PC.

È **volatile**, cioè perde il contenuto informativo se non c'è alimentazione.



Nei livelli più alti ci si avvicina alla CPU.

La CPU contatta prima i registri interni per cercare il dato occorrente.

In caso di mancanza del dato comincia a scendere nella piramide.

Cache L1 = interna alla CPU

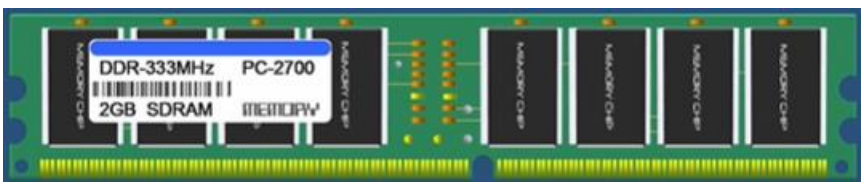
Cache L2 = interna alla scheda madre

RAM = memoria alta velocità e capienza

Dischi = HDD o SSD = alta capacità ma poca velocità e tanta latenza

La RAM si divide in:

- **Statica SRAM**
- **Dinamica DRAM**



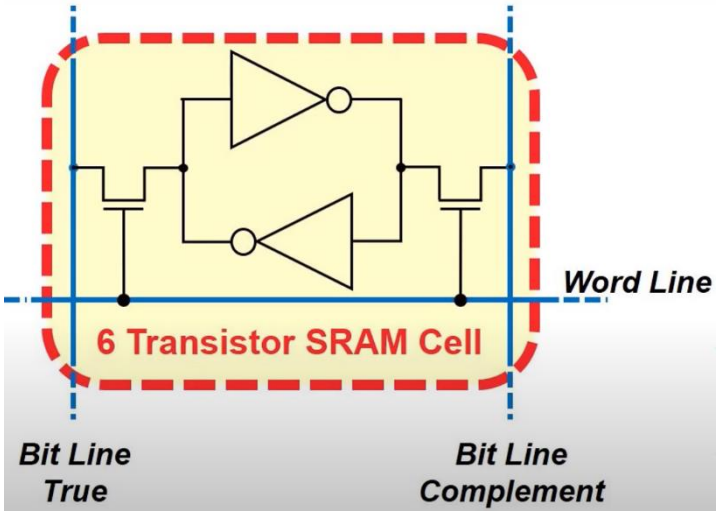
Il numero dopo DDR indica la frequenza di operatività, quello dopo PC indica il larghezza di Banda = bitrate

SRAM

Una cella di memoria è composta da 2 transistor e 1 LATCH/FLIP-FLOP (4 transistor).

Ogni cella memorizza 1 BIT.

È molto dispendioso dal punto di vista economico.

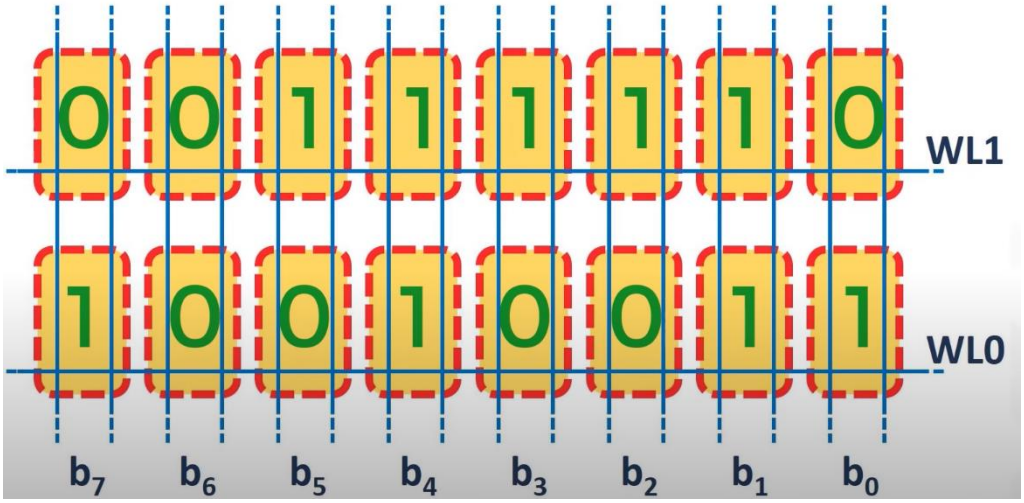


WORD LINE: si attiva quando viene decodificato l'indirizzo di memoria, corrispondente a una locazione.

BIT LINE: contiene il contenuto informativo da LEGGERE o SCRIVERE in memoria.

Quando la WL = 1 i transistor si chiudono.

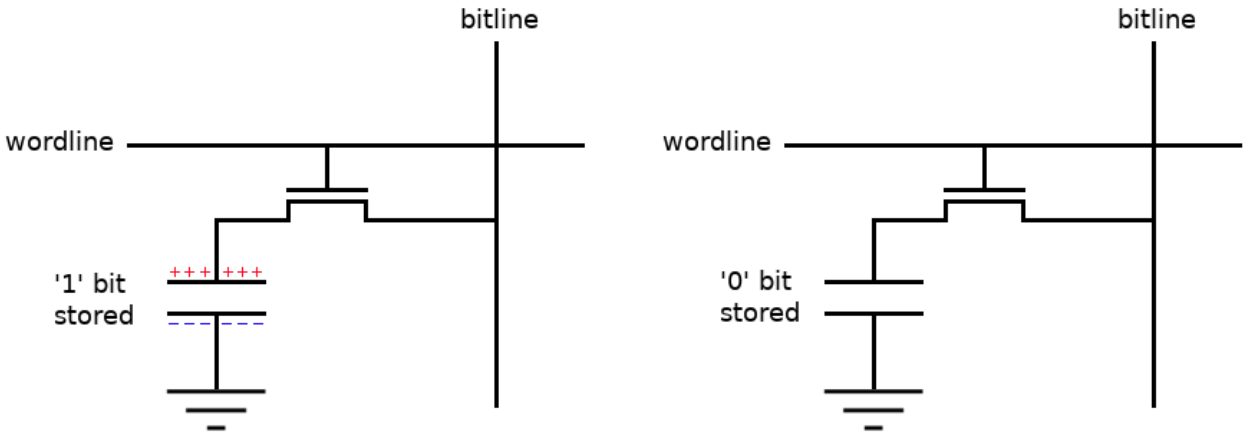
- **Lettura:** la BL riceve il valore contenuto nella cella
- **Scrittura:** la BL invia il valore nella cella



Le operazioni di lettura e scrittura possono avvenire all'infinito senza causare danni ai circuiti interni. Tempo Lettura = Tempo Scrittura

- **Synchronous RAM:** architettura basata sulle pipeline, cioè in parallelo.
  - **BUS** grandi con **trasferimenti** fino a **Gb/s**
  - Molto **costose** per essere sviluppate a causa del **grande numero** di **componenti** richiesto (6 transistor = 1 bit)
- **Asynchronus RAM:**
  - **velocità** di trasferimento **modesta**
  - **Risparmio energetico**

DRAM



Come nelle SRAM sono presenti la WORD LINE e la BIT LINE, e hanno la stessa funzione.

In questo caso **1 bit** viene **salvato** all'interno di un **condensatore**, che se è **carico = 1**, **scarico = 0**.

Un condensatore però non mantiene la sua carica all'infinito, ma si scarica in un tempo di $5T$ = 5 * (R*C).

Esso può scaricarsi sulla BL in caso di LETTURA, oppure sul transistor quando esso è aperto, quindi la WL = 0. Ciò vuol dire che non è stata selezionata la locazione.

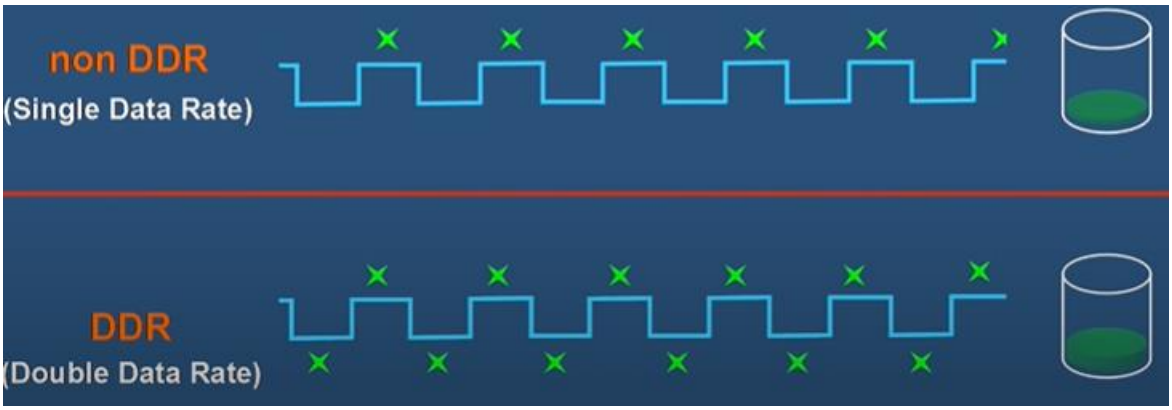
Se si scarica si perde il contenuto informatico, perciò è necessaria una operazione di REFRESH circa 16 volte al secondo = 16Hz -> 1 / 16 = 64ms

VANTAGGI E SVANTAGGI DRAM

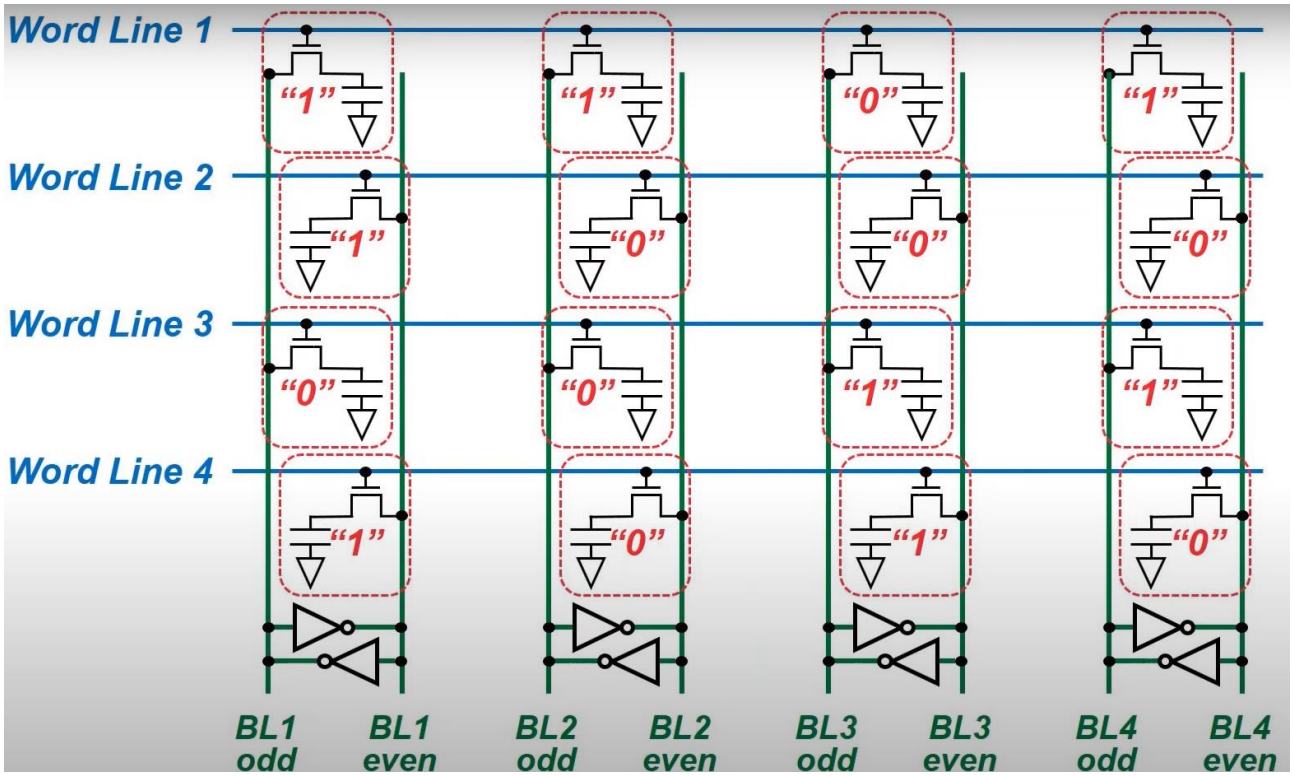
- **Più lente** delle **SRAM**
- **Più economiche**, richiedono **meno componenti**: 1 transistor e 1 Condensatore
- **Complessità** della **conservazione** dello stato: **operazione** di **REFRESH onerosa**

TIPOLOGIE DI DRAM

- **DDR = Double Data Rate:** i dati vengono trasmessi anche quando CLK è basso



REFRESH



Ci sono **2 WORD LINE**, una per le **locazioni DISPARI** e una per quelle **PARI**.

Il circuito sotto è un **Inverter LOOP**.

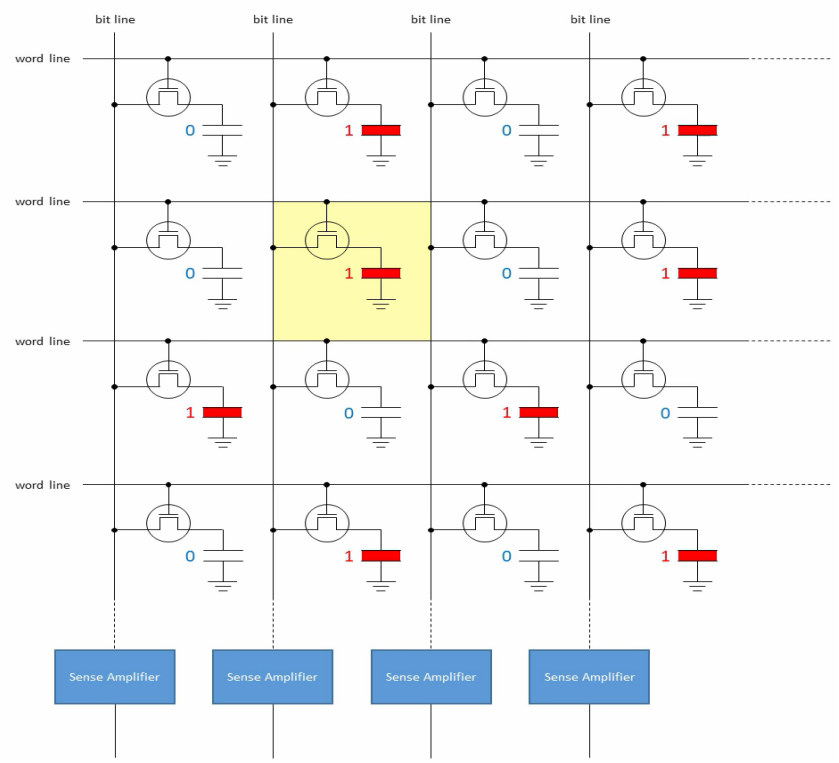
Durante l'operazione di **REFRESH** i due **inverter** sono a una **tensione intermedia tra** i valori **0 e 1**, circa 0.5V. Se **BLodd = BLeven** si è in una **condizione** di **equilibrio**.

- Se **C = 1**, la sua tensione sarà maggiore di 0.5V, perciò **comincerà** ad **aumentare** quella sulla **BITLINE**. Il **LOOP incrementa** sempre di più **la tensione** fino ad arrivare a una condizione in cui una BL vale 1V e l'altra 0V.

Quando la **tensione** sulla **BL = 1V ricaricherà** il **C** che nel mentre si stava scaricando, **ripristinando il valore logico 1**, presente in precedenza.

- Se **C = 0**, la tensione sulla **BL = 0V**, perciò la tensione viene **subito ripristinata**.

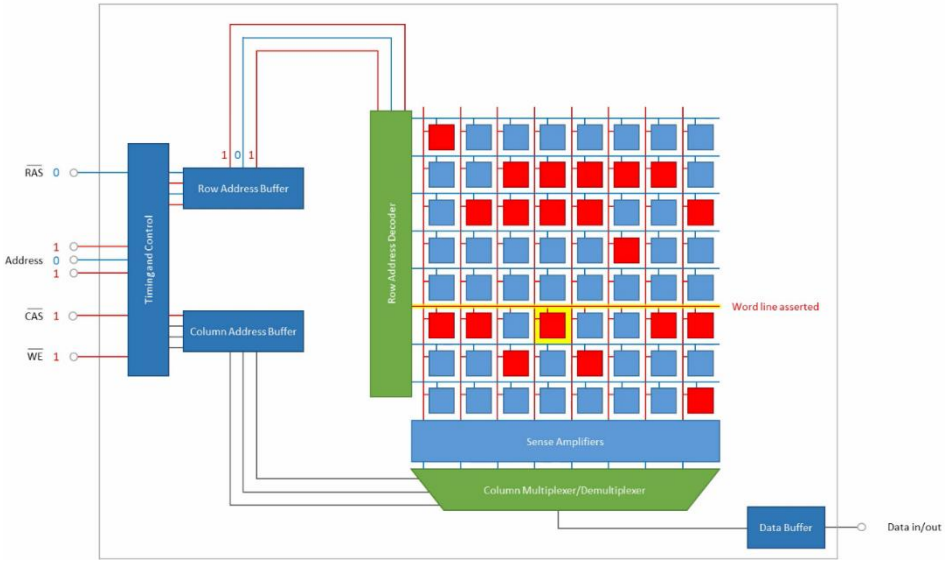
ACCESSO IN MEMORIA



I **SENSE AMPLIFIER** sono dei **circuiti logici contenenti** dei **LATCH** che **memorizzano temporaneamente** il **dato** presente nella cella.

Dopo ogni operazione di **LETTURA** il **dato** viene **perso**, perciò è **necessaria** una operazione di **REFRESH** immediata.

LETTURA E SCRITTURA IN DRAM



not(RAS) = flag che indica se è il momento di effettuare la fase RAS. Attivo BASSO

not(CAS) = flag che indica se è il momento di effettuare la fase CAS. Attivo BASSO

not(WE) = flag che permette o meno la scrittura in memoria. Attivo BASSO

ADDRESS = numero di bit che servono a codificare un indirizzo di memoria

N bit -> 2^N allocazioni possibili

La fase di accesso in memoria si divide in 2 fasi principali:

- **RAS (Raw Address Strobe)** : individua la RIGA (WORDLINE) da attivare
- **CAS (Coloumn Address Strobe)** : individua la cella nella Locazione

Il modulo di controllo determina le fasi dal punto di vista tempistico e logico, in base ai flag attivi.

- **Indirizzi** entrano nel Modulo. Quando **RAS è attivo parte** la **fase di RAS**
- Nel **Raw Address Decoder** vengono **inseriti n bit** per la **codifica dell'indirizzo**.
  - **DECODER** = **n ingressi, 2^n uscite** -> **attiva l'uscita** corrispondente alla **codifica in binario degli ingressi**
  - Viene **attivata la RIGA** (WORDLINE) corrispondente alla codifica
  - **Tutta la LOCAZIONE** viene **salvata** all'interno dei **SENSE AMPLIFIER**
  - La **LOCAZIONE** viene **inserita** nel **Coloumn Address DE/Multiplexer**
- **LETTURA**:
  - **CAM**: viene **mandato** in **uscita** l'**ingresso corrispondente** alla **codifica** in binario **degli** ingressi **selettori** (n bit degli indirizzi)
- **SCRITTURA**
  - **CAD**: viene **mandato** in **una** delle 2^n **uscite**, **selezionata** dai **selettori**, il **segnale** di **ingresso**.

MEMORIE PERMANENTI

Sono memorie non volatili, che mantengono i dati anche in caso di mancanza di alimentazione.

- **ROM (Read Only Memory)**: sono scritte nel momento in cui sono fabbricate e non possono essere modificate. Solitamente le ROM contengono le informazioni utili durante il BOOT della macchina.
  - **PROM (Programmable ROM)**: programmabili 1 volta
  - **EPROM (Eresable PROM)**: possono essere cancellate e riscritte
  - **EEPROM (Electrically EPROM)**: cancellabili elettricamente
  - **FLASH**

- **DISCHI RIGIDI**: usati per le memorie di massa ad alta capacità
    - **HDD Hard Disk Drive**: Alta capacità ma bassa velocità
    - **SSD Solid State Drive**: minore capacità ma alta velocità

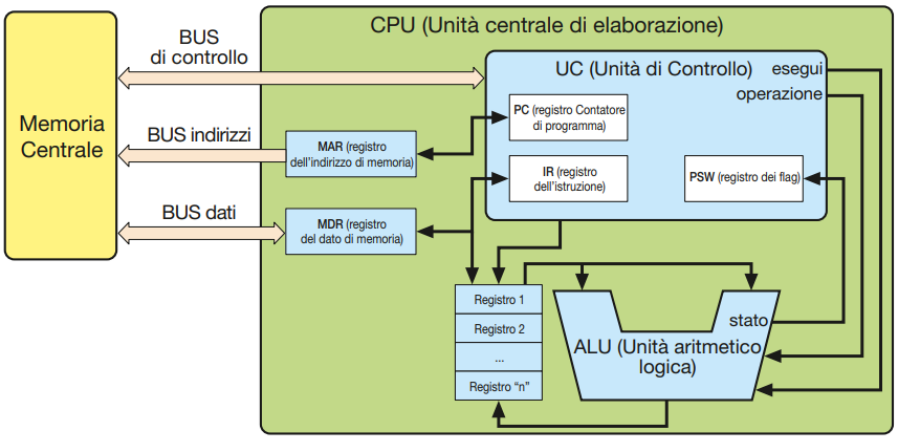
CPU (Central Processing Unit)

Il Processore è quel componente responsabile di eseguire le operazioni e regolare il traffico dati all'interno del PC.

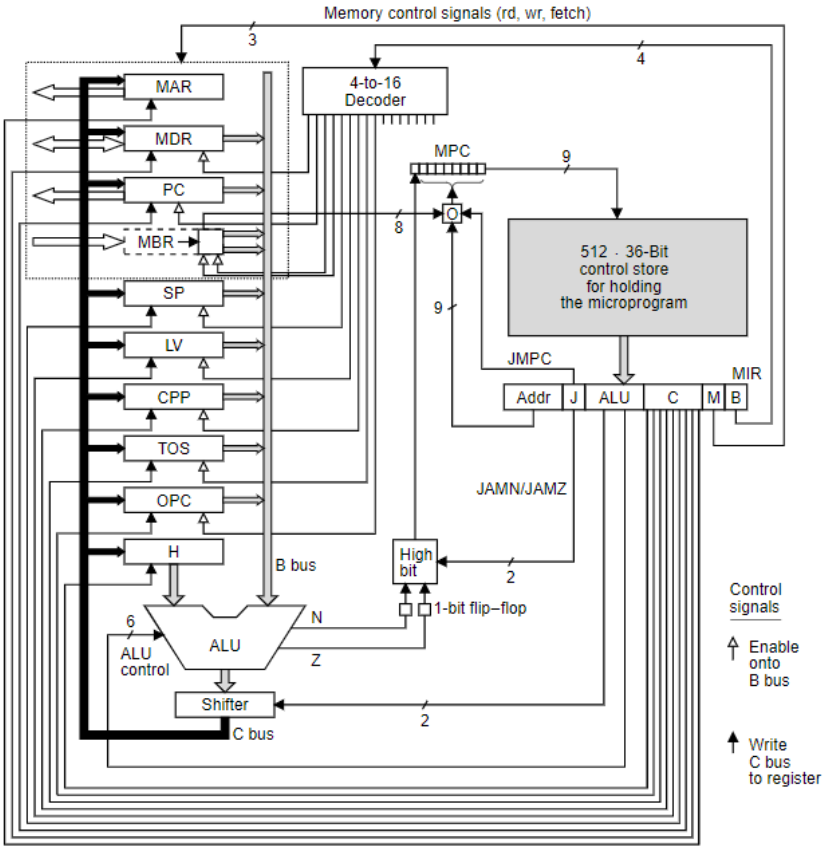
Viene definito **"General Purpose"** perché offre una grande versatilità di utilizzo.

"Special Purpose" sono i dispositivi adatti a compiere un obiettivo ben preciso e definito, come i Router, che si occupano dell'indirizzamento.

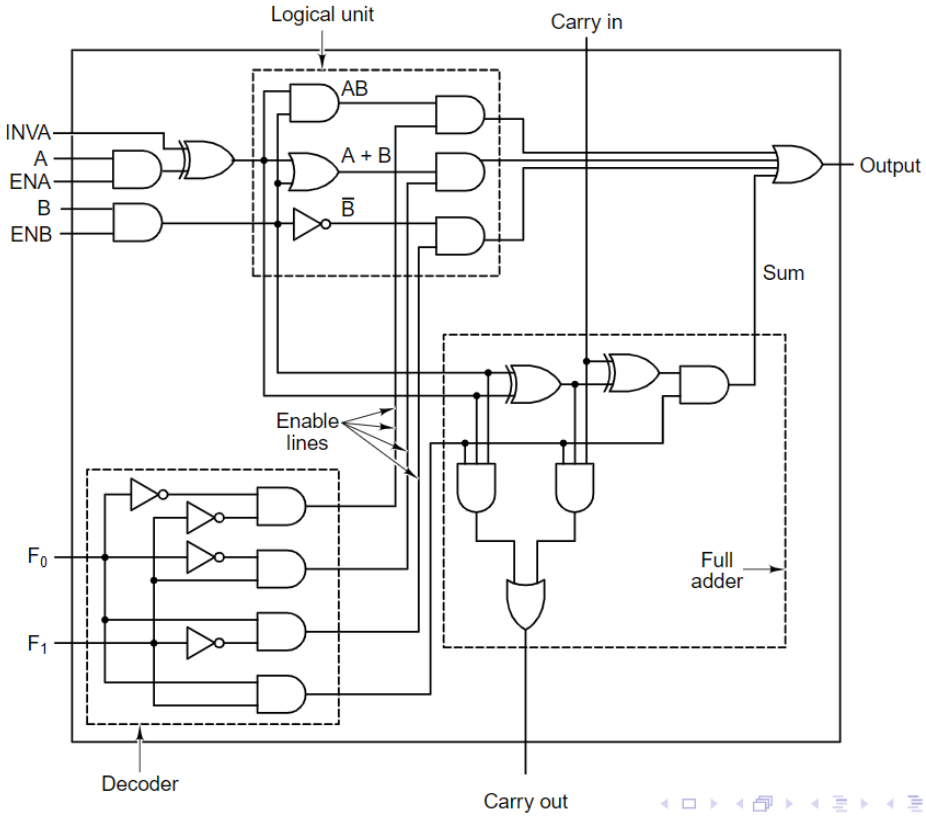
STRUTTURA INTERNA DELLA CPU



- **Unità di Controllo (CU)**: preleva istruzioni e dati dalla memoria centrale (RAM o HDD/SSD). Invia i dati alla ALU e ne governa l'esecuzione. Gestisce i risultati, salvandoli in memoria. Gestisce anche gli eventuali salti di istruzione, agendo sullo stack PC, aumentando o diminuendo il IP a piacimento, in base alla prossima istruzione da eseguire



- **ALU(Aritmetic Logical Unit)**: ha il compito di eseguire le operazioni aritmetico logiche ai dati forniti dalla CU



F0 e F1 sono dei segnali di controllo, e attivano le linee di Enable, che attiveranno rispettivamente il circuito Logico o Aritmetico

F0 = F1 = 0, si attiva la E0 che attiva l'operazione AND

F0 = 0, F1 = 1 si attiva la E1 per l'operazione OR

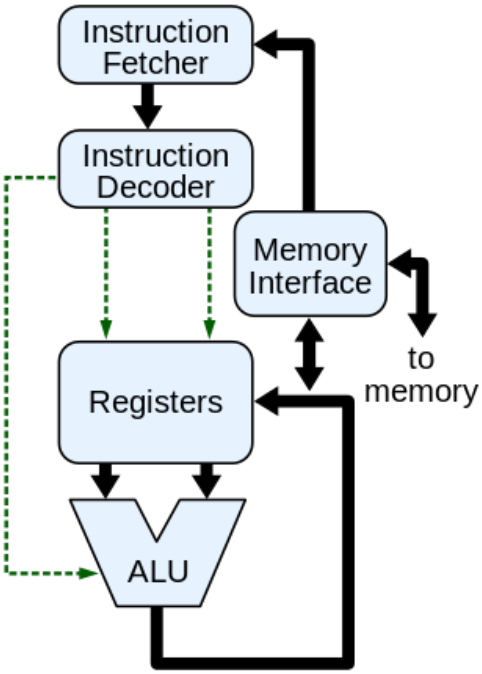
F0 = 1, F1 = 0 si attiva E2, per operazione NOT

F0 = F1 = 1 si attiva E3, che attiva il circuito aritmetico Full Adder

INVA = è un segnale di controllo che abilita o meno l'inversione di A

Carry IN = eventuale riporto dell'operazione precedente. Da carry OUT passa a carryIN

- **Shifter**: si occupa di eseguire operazioni di moltiplicazione e divisione per potenze del 2
- **Registri interni alla CPU**: hanno un tempo di accesso molto veloce, più della RAM, ma a discapito della capienza. Il valore complessivo dei dati nel registro totale identifica lo stato della CPU



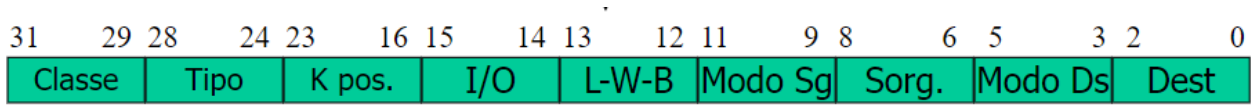
  - **REGISTRI DELLE ISTRUZIONI**
  - **PC(Program Counter) o IP(Instruction Point):** contiene l'indirizzo di memoria della prossima istruzione (linguaggio macchina) da eseguire. La CPU legge al suo interno l'indirizzo da contattare per

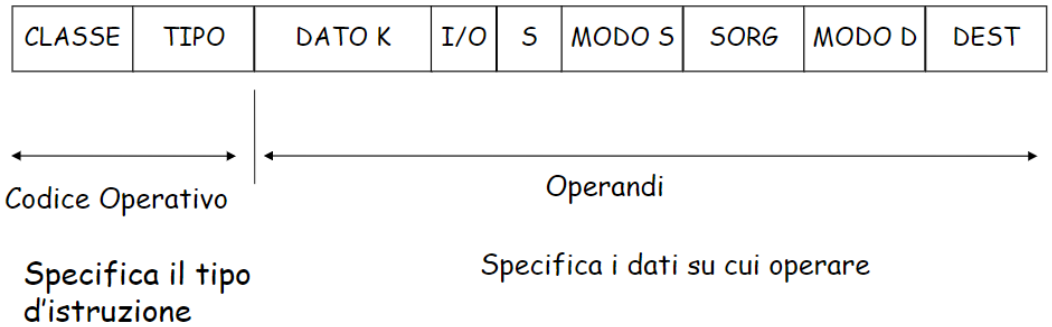
trovare la operazione successiva. Una volta prelevato l'indirizzo il PC viene shiftato. Questo registro ha una logica LIFO (Last Input Fist Output).

- Questo registro è un puntatore, infatti contiene l'indirizzo della cella a cui si vuole accedere. Dopo ogni prelievo il PC viene incrementato di 1, per puntare all'indirizzo successivo

- **IR Instruction Register:** contiene l'istruzione intera prelevata e decodificata dalla memoria che andrà eseguita della fase di execute

- **ISA (Instruction Set Architecture) :** insieme delle istruzioni di basso livello che operano nel livello HW
  - **Operative: le operazioni vengono eseguite dalla ALU**
    - **Operazione aritmetica**
    - **Operazione logica**
  - **Trasferimento dei dati da MEM a CPU e viceversa**
    - **Load word:** carica i dati dalla RAM ai registri
    - **Store word:** carica i dati dai registri alla RAM
  - **Controllo:** variazione dell'ordine di esecuzione di istruzioni. SALTI
    - **Rotazione o SHIFT**
    - **Aggiornamento flag di stato (PSW)**
    - **Controllo del flusso del programma**
    - **Controllo della macchina**
    - **INPUT/OUTPUT**
- **Formato istruzione 32bit**





OPERATION CODE , DEST, SRC1, SRC2: SRC indica il dato su cui operare e DEST indica il registro in cui salvare il risultato

CLASSE: 3 bit che identificano la classe dell'op code

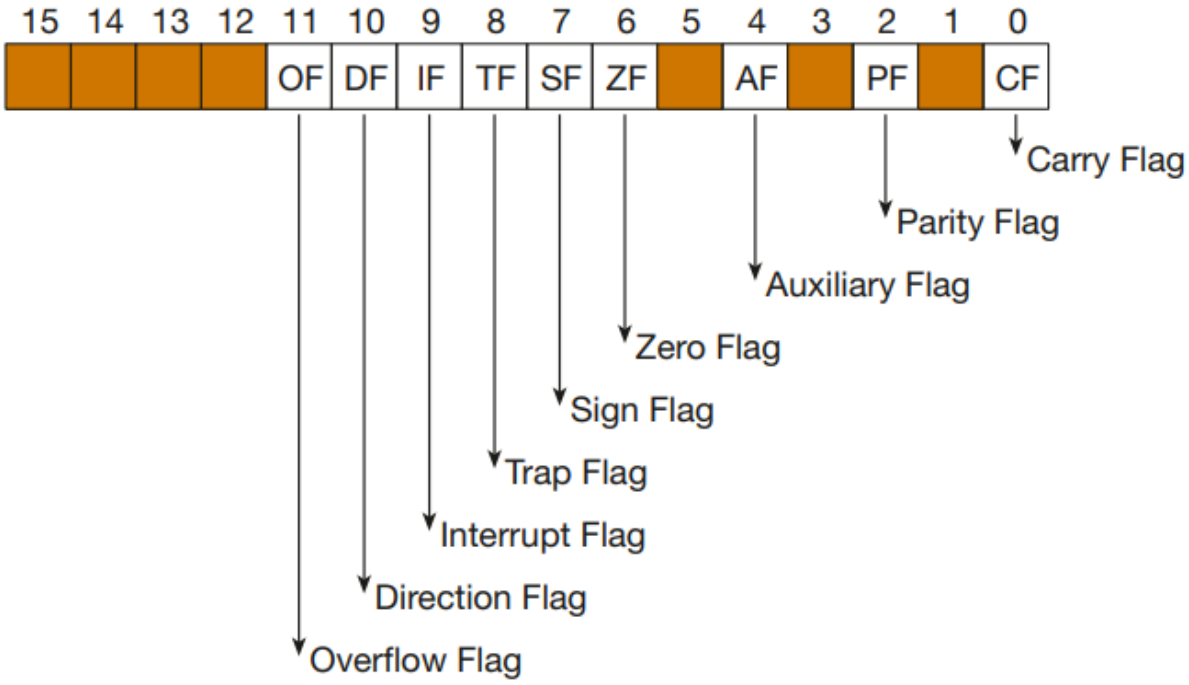
- 000 = movimento dati
- 001 = Operazione aritmetica
- 010 = Operazione logica
- 011 = Rotazione e Shift
- 100 = Operazione sui flag di stato PSW
- 101 = Controllo del flusso del programma. SALTI, LOOP, Branches
- 110 = Controllo della macchina
- 111 = I/O

Campo	n°.Bits	Commento
Classe	3	Indica la classe di istruzione (movimento dati, rotazione e shift, aritmetiche...)
Tipo	5	All'interno della classe viene indicato quale tra le operazioni disponibili deve essere eseguita
dato k	8	Campo contenente un dato utilizzato nelle istruzioni di rotazione e shift e nelle istruzioni di I/O
I/O	2	Campo utilizzato nelle istruzioni di I/O, codifica il modo con cui l'indirizzo del device può essere recuperato
s	2	Indica il formato del dato che deve essere trattato dall'operazione.
modo s	3	Indica il modo di indirizzamento dell'operando sorgente.
sorg	3	In caso di indirizzamento diretto a registro, indiretto a registro, con predecremento e con postincremento indica uno degli otto registri di uso generale R0-R7
modo d	3	Indica il modo di indirizzamento dell'operando destinazione.
dest	3	In caso di indirizzamento diretto a registro, indiretto a registro, con predecremento e con postincremento indica uno degli otto registri di uso generale R0-R7

Modi di indirizzamento

- Modi diretti
  - Diretto con registro
  - Immediato
  - Assoluto
- Modi indiretti
  - Indiretto con registro
  - Indiretto con spiazzamento
  - Relativo
  - Indiretto con predecremento
  - Indiretto con postdecremento

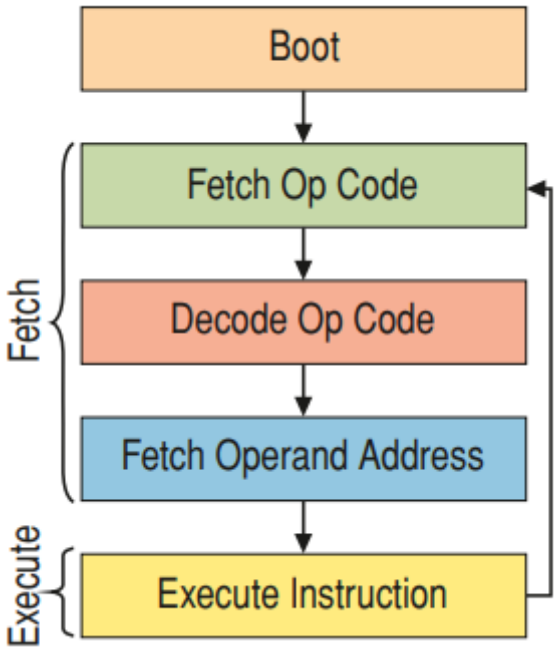
- **Registro dei Flags. FLAG = 1bit di controllo.** Determinano degli stati della CPU. = CCR (Condition Code Register)
  = PSW (Program Status Word)



  - **Overflow:** segnala il verificarsi di un overflow al seguito di una operazione aritmetica
  - **Zero:** segnala se il risultato dell'operazione è 0
- **REGISTRI DEI DATI**
  - **MDR (Memory Data Register):** Collegato al **BUS DATI** tramite **buffer bidirezionale a 3 stati.** Contiene i dati ricevuti o da inviare alla memoria.
    Da MEM a CPU i dati finiscono nei registri interni
    Da CPU a MEM i dati partono dai registri e arrivano in RAM

  - **MAR (Memoria Address Register):** Nella fase di fetch contiene l'indirizzo di memoria dell'istruzione, che poi verrà inserito nel IR. Nella fase di execute contiene l'indirizzo di memoria del dato da cercare (operando dell'istruzione).
    Collegato al **BUS ADDR**

CICLO MACCHINA (Fetch, Decode, Execute)

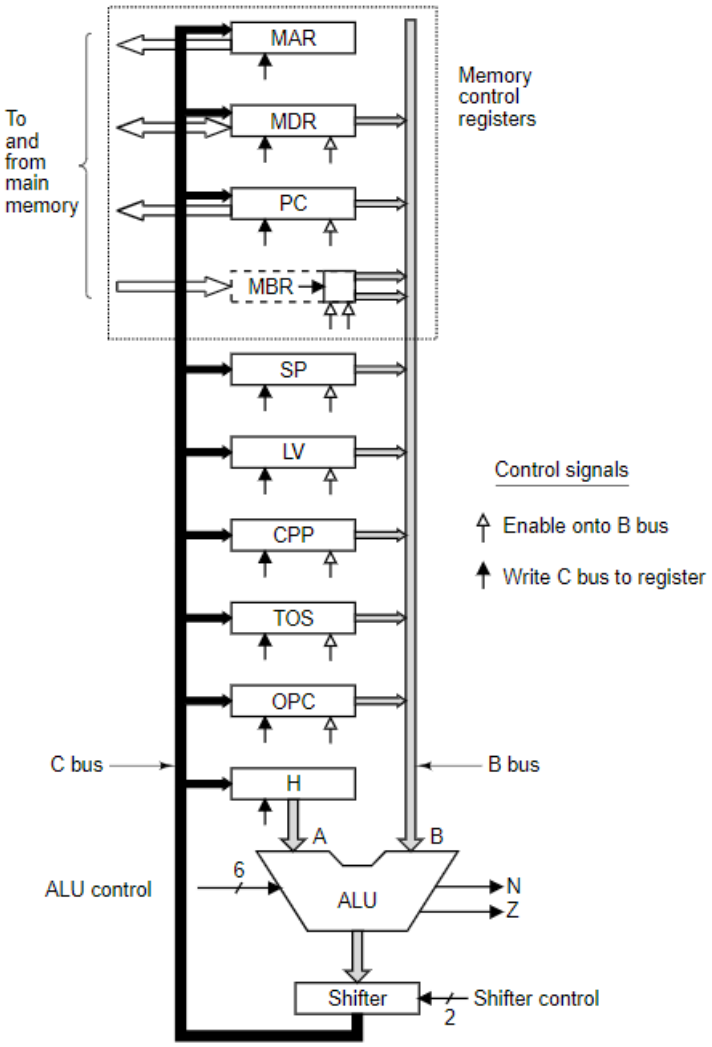


- **Fetch dell'ISTRUZIONE:** la CPU deve comunicare con la RAM. All'interno dell'IP/PC è presente l'indirizzo di memoria dell'istruzione da eseguire. Una volta prelevata l'istruzione il PC viene incrementato per puntare all'operazione successiva. L'indirizzo dell'istruzione viene copiato del MAR e successivamente nel BUS ADDR. Dopo un'opportuna decodifica dell'indirizzo si trova la cella di memoria, e si preleva l'istruzione. L'istruzione prelevata viene inserita nel BUS DATI e finirà nel MDR.
- **Decode istruzione:** L'istruzione entra nell'IR(Instruction Register) e viene interpretata dalla CU. Vengono preparate le risorse necessarie. L'istruzione

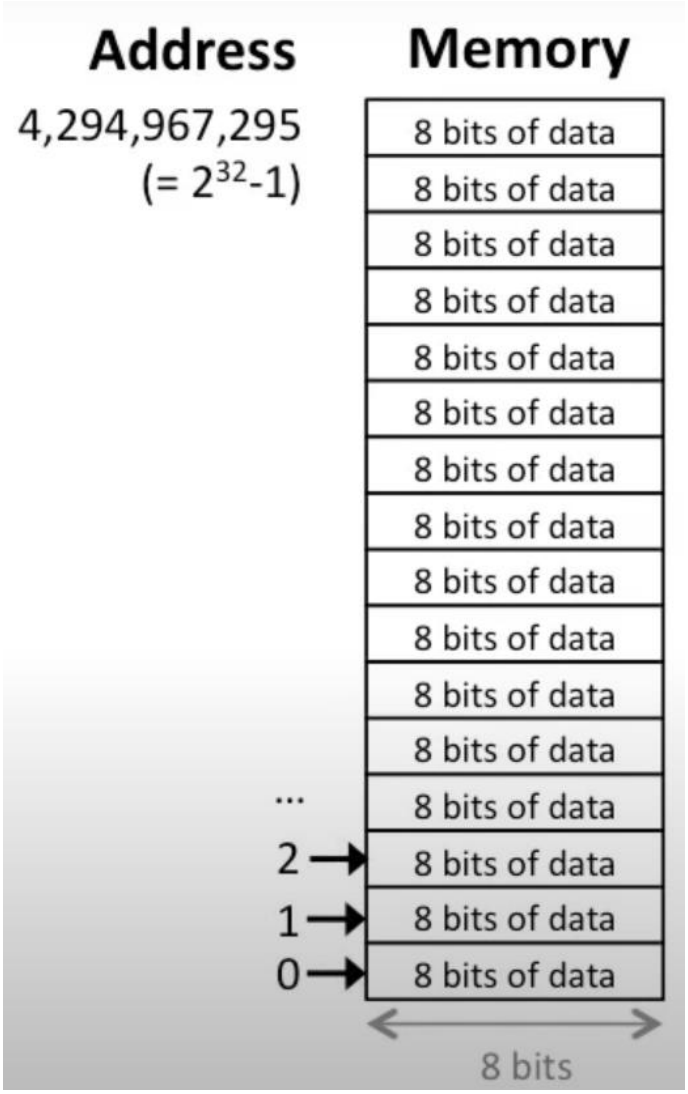
deve interfacciarsi direttamente con l'Hardware in base a delle regole definite dalla logica in uso:
    - **Logica cablata:** l'HW è predisposto a compiere l'istruzione seguendo i passaggi determinati dalla struttura fisica dei componenti. Più complicato e costoso da realizzare, e meno flessibile. Offre prestazioni migliori
    - **Logica microprogrammata:** l'insieme di microistruzioni da compiere per effettuare l'operazione fisicamente è storata all'interno di una ROM adatta. La CU è a sua volta una microarchitettura capace di eseguire delle istruzioni programmate. Meno costoso e più versatile, ma meno efficiente
- **Fetch degli operandi:** dopo la decodifica dell'istruzione si cercano gli eventuali operandi, tramite gli indirizzi di memoria espressi nell'istruzione decodificata. Ancora una volta entra in gioco in BUS ADDR e il MAR per l'indirizzo di memoria del dato da cercare, e una volta trovato il BUS DATI lo mette nel MDR.Affinchè l'operazione possa essere eseguita gli operandi devono trovarsi all'interno dei registri della CPU
    - 4 metodi di indirizzamento:
        - **Immediato:** il dato su cui operare è gia presente nell'istruzione, quindi può essere una costante numerica o letterale
        - **Diretto:** viene fornito l'indirizzo di memoria in cui andare a cercare il dato su cui operare
        - **Indiretto:** fornire l'indirizzo della cella di memoria contenente a sua volta un indirizzo di memoria che punta alla cella effettiva in cui si trova il dato
        - **Base + Offset:** all'interno dell'istruzione è presente un registro GPR(General Purpose Register) che conterrà l'indirizzo base. A questa base viene sommato un offset in modo da trovare la cella corretta.

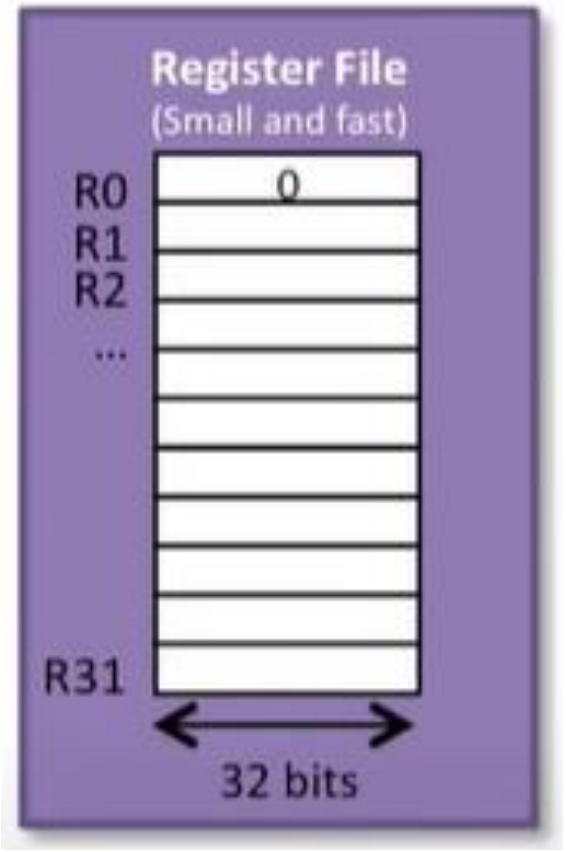
- **Execute operazione:** se l'operazione è un calcolo viene eseguita dalla ALU, altrimenti se è una lettura in memoria oppure un salto viene gestita diversamente. Il risultato viene salvato in un registro, che poi sarà opportunamente trasferito in memoria
- **Write Back:** vengono ripristinati i registri usati dall'operazione, e in particolare viene aggiornato il PSW

DATA PATH


MAR
MDR
PC
MBR
SP
LV
CPP
TOS
OPC
H
To and from main memory
Memory control registers
Control signals
Enable onto B bus
Write C bus to register
C bus
B bus
A
B
6
ALU control
ALU
N
Z
Shifter
Shifter control
2

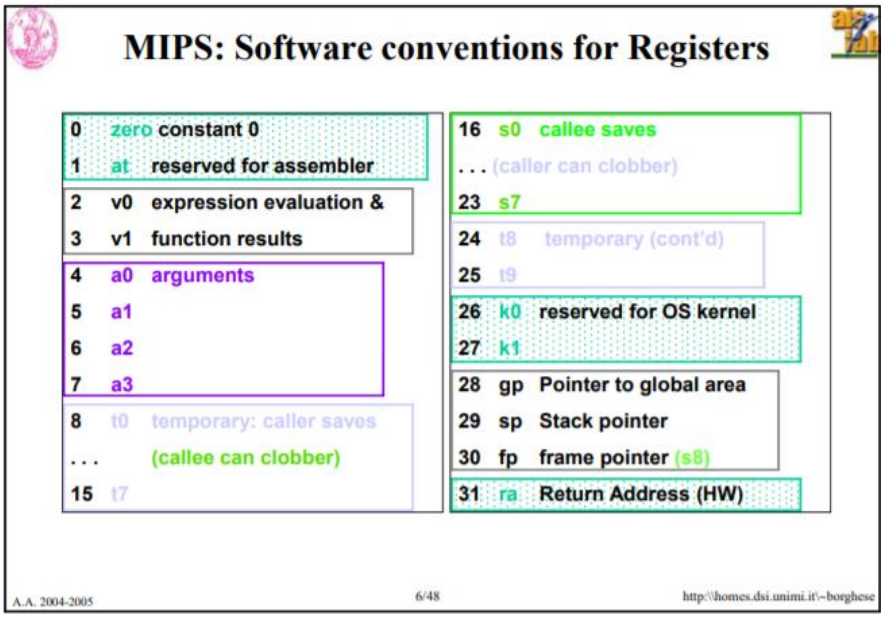
REGISTRI -> MEMORIA

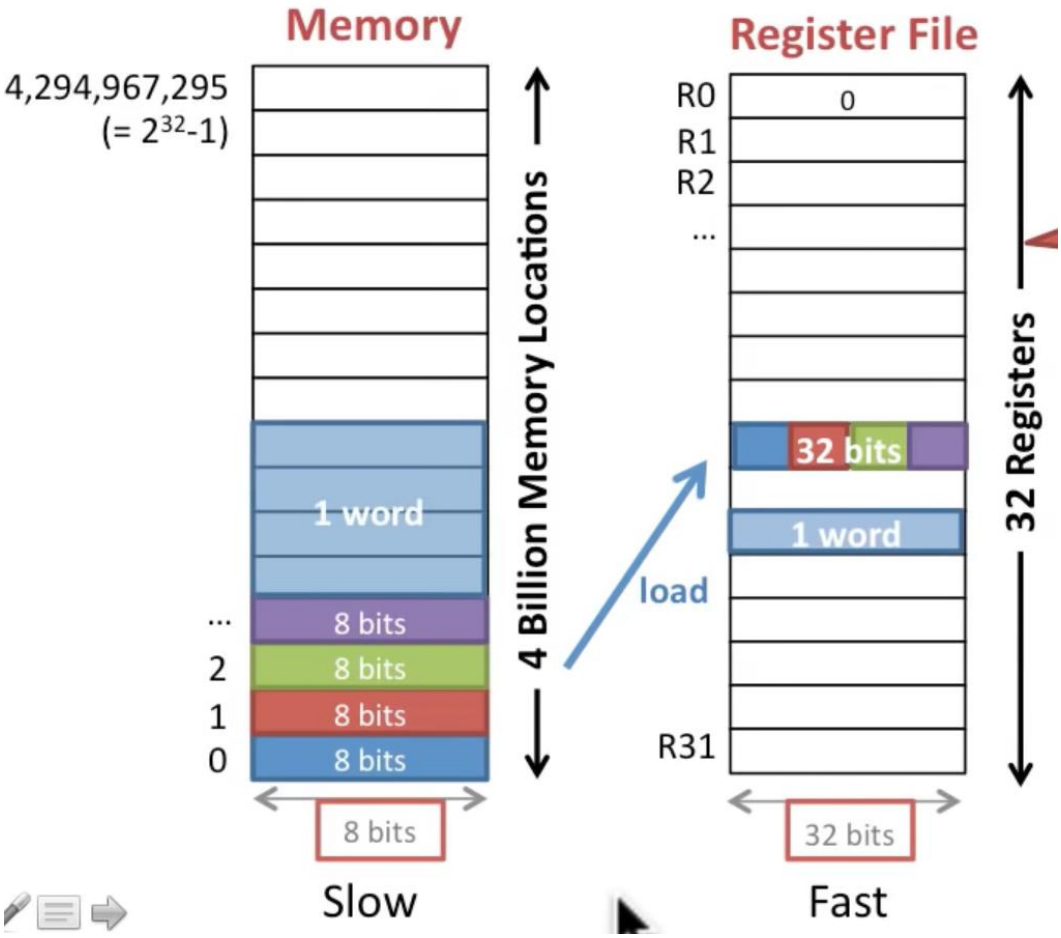




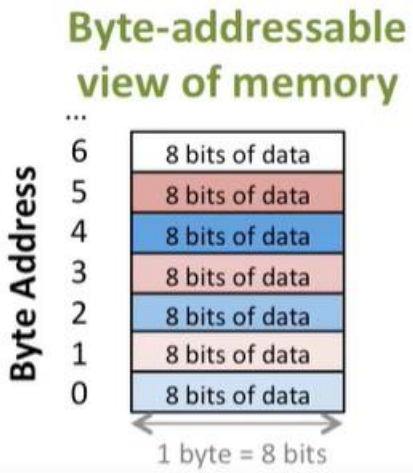
Registri: 32 bit di larghezza, 32 registri totali.

Memoria: 8 bit larghezza, 2^nbit nbit=bit BUS ADDR

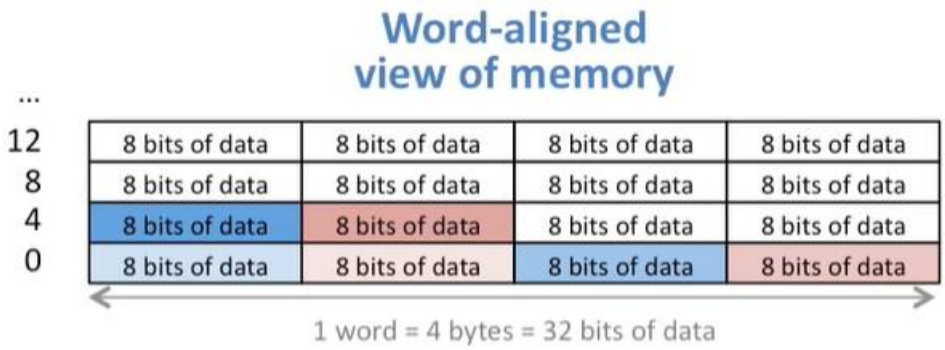




4 celle di memoria vengono salvate in 1 registro



- Most data in MIPS is handled in words not bytes
  - A word is 32 bits or 4 bytes
  - A word is the size of a register



Ogni indirizzo identifica 4 celle di memoria, che verranno tutte trasferite ai registri.

Essendo multipli di 4, gli indirizzi termineranno sempre con 00

ACCESSO ALLA RAM

- Allineato: viene prelevata l'intera riga di celle, quindi l'indirizzo sarà sempre multiplo di 4
- Non allineato: viene indicato un indirizzo intermedio. Vengono sempre prelevate 4 celle, anche appartenenti alla riga successiva. Essendo più complicato da realizzare solitamente si prelevano le 2 righe intere e poi vengono frammentate nelle celle interessate