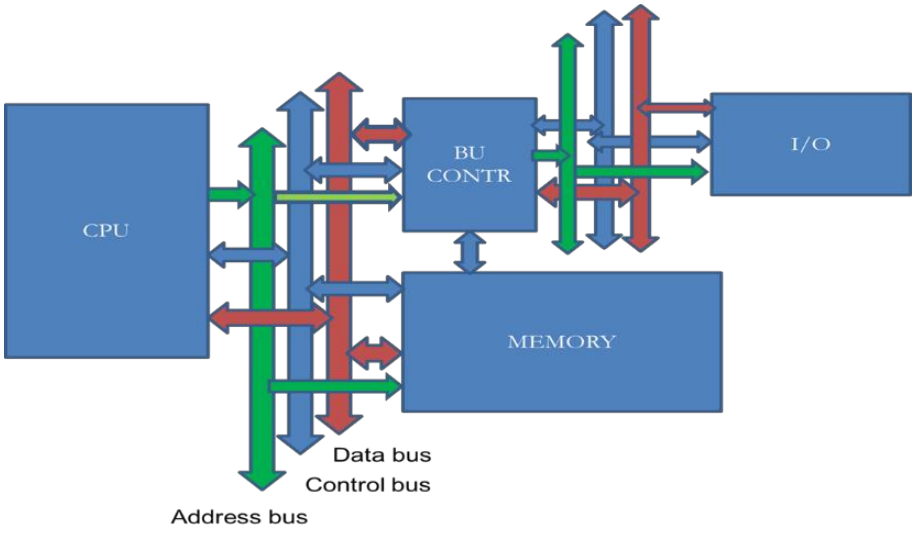
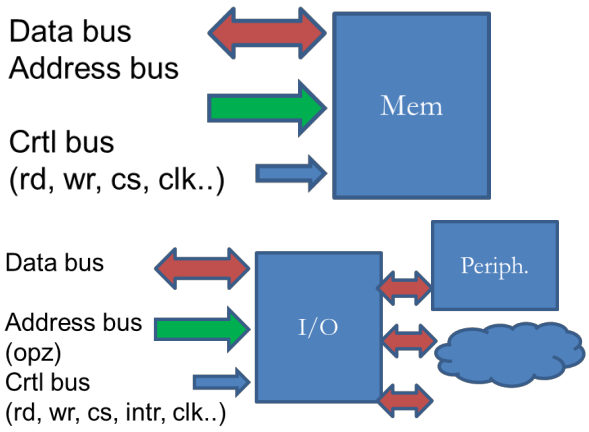
ELEMENTI DI ARCHTETTURA DEI CALCOLATORI



Descrizione strutturale del calcolatore → ***l'unione di reti logiche nella maggior parte sequenziali***.

LA MEMORIA:

C'è da ricordare ***che la memoria*** o più propriamente la **memoria centrale,** *è l'unità di memorizzazione temporanea di dati ed istruzioni che normalmente risiedono in una memoria di massa a sua volta parte dell'I/O.* La memoria centrale ***è di tipo volatile,*** ossia capace di mantenere i dati solo nel periodo in cui è alimentata. È un sistema passivo ora di solito sincrono che se abilitato riceve indirizzi e memorizza dati (in scrittura) o fornisce dati (in lettura), costituente la interfaccia di riferimento con la CPU per quel che riguarda il mantenimento temporaneo di dati ed istruzioni, a cui al CPU accede con indirizzamento diretto e casuale. È vista dalla CPU indipendentemente dalla sua implementazione come una unità logica che contiene **M** locazioni di memoria elementari (celle) ad un byte che possono essere raggruppati in n/8 byte consecutivi per formare una parola di n bit. **Nel caso ideale M= 2^naddr.**



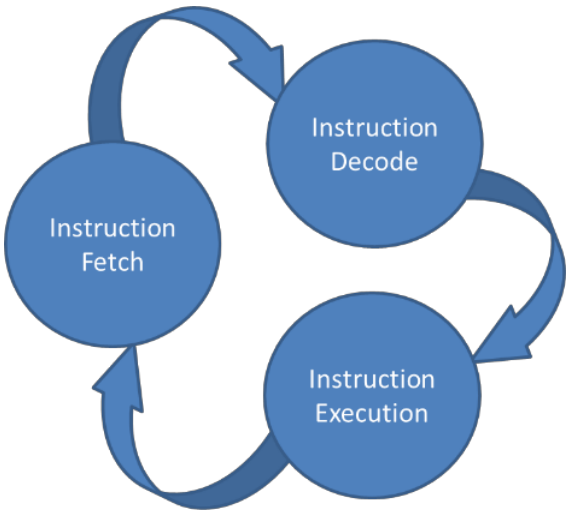
L'INPUT/OUTPUT: è il ***sottosistema composto da periferiche e sistemi di controllo delle periferiche*** (dette appunto "**controller**") di diversa natura che si interfacciano con il resto del calcolatore in modo omogeneo, definito da standard di comunicazione, **ricevendo indirizzi o segnali di abilitazione (chip select), segnali di controllo e di configurazione e trasferendo dati in lettura, scrittura o in modo bidirezionale.**

I BUS: sono ***i sistemi di collegamento multi-punto*** composti da un insieme di segnali omogenei dal punto di vista funzionale, genericamente bidirezionali, trasmessi in ogni istante da una singola unità (**Driver**) ricevuti da più unità.
L'architettura di bus come sappiamo prevede sempre tre tipi di bus:

- Un **bus di dati** bidirezionali che trasferisce dati da e verso le varie unità.
- Un **bus di indirizzi o address bus** che trasferisce gli indirizzi che servono primariamente alla memoria ma anche ai moduli di I/O.
- Un **bus di controllo** che contiene tutti i segnali di controllo da e per la CPU.

IL CICLO DI ISTRUZIONE

Concetto su cui si basa il funzionamento del calcolatore. La CPU esegue indefinitivamente il ciclo di istruzione composto da ***3 fasi: fetch dell'istruzione dalla memoria, decodifica dell'istruzione ed esecuzione dell'istruzione.***



La CPU durante la fase di Instruction fetch (o reperimento) legge in memoria le istruzioni da eseguire, poi le decodifica e le esegue.

Molto sinteticamente si può dire **che durante la fase di esecuzione la CPU esegue uno dei tre tipi di istruzioni possibili:**

1. **Istruzioni di ALU:** sono le istruzioni che genericamente lavorano con operandi all'interno della CPU per eseguire calcoli sui dati impiegando le unità funzionali di calcolo come la ALU → sono le classiche operazioni aritmetiche o logiche
2. **Istruzioni di trasferimento dei dati (data transfer instruction), o load-store, o di memoria** → necessitano nell'istruzione dell'informazione dell'indirizzo di memoria da dove o dove trasferire.
3. **Istruzioni di controllo → ex. di salto o di salto incondizionato**

Il tipo, la quantità ed il formato di tali istruzioni dipende dall'ISA. L'istruzione a seconda dell'ISA può avere diversi formati:

a) **Un formato a due registri: <opcode>, <sorg>, <dest>** in cui un registro funge sia da sorgente che da destinazione
b) **Un formato a tre registri: <opcode>, <dest>, <sorg1>, <sorg2>** come nelle macchine RISC

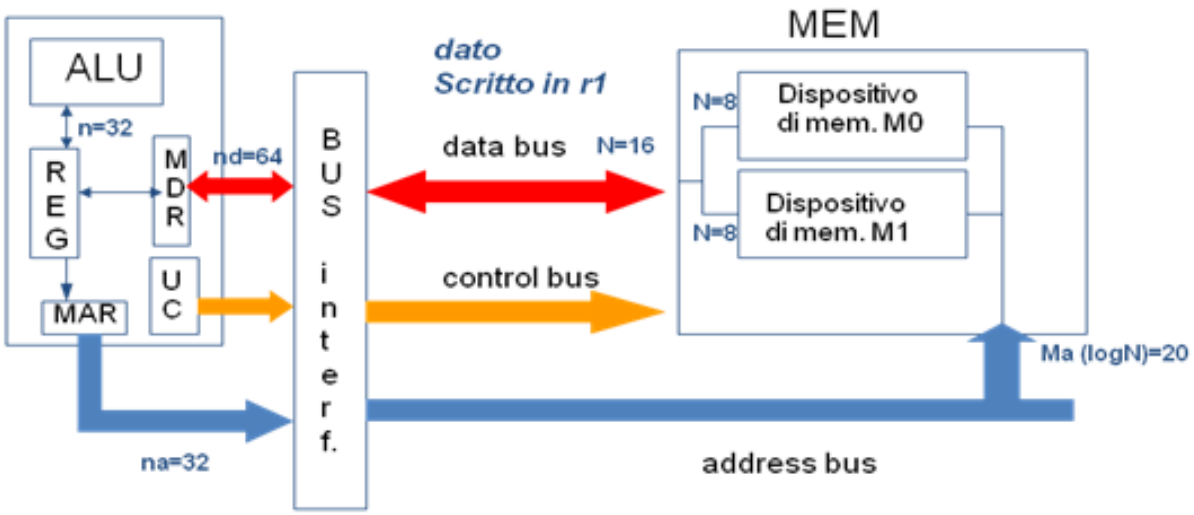
Osservando solo la memoria si può affermare che la CPU comunica con la memoria per tre motivi:

a. Leggere un operando **Load** → quando i dati o le istruzioni arrivano alla CPU
b. Scrivere un operando **store** → i dati escono dalla CPU per essere trasferiti verso memorie o I/O
c. Leggere o reperire l'istruzione (fase di fetch).

ELEMENTI ISA

L'Instruction Set Architecture (ISA) di ogni processore ne definisce alcuni parametri fondamentali quali:

- Parallelismo di parola e ordinamento
- La dimensione dei registri
- Formato e tipo delle istruzioni

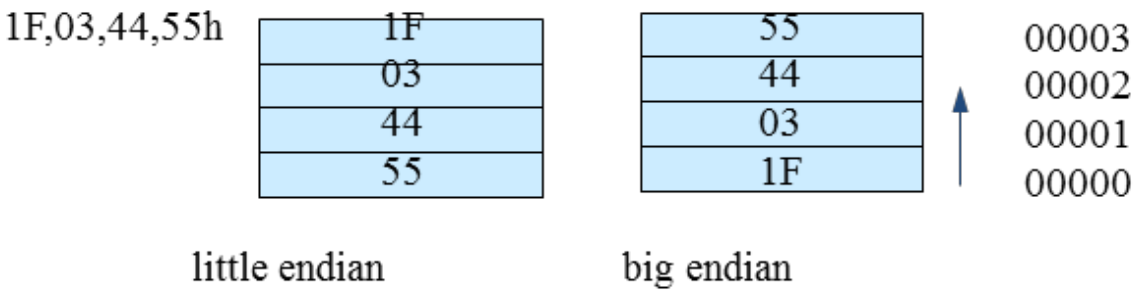


Nella CPU l'ISA definisce il **parallelismo interno n**, **la dimensione n dei registri**, **la dimensione delle istruzioni Nistr (in byte)** e **lo spazio di indirizzamento** (il numero di bit di indirizzo **naddr=logNi**) **ed i bit massimi di indirizzo naddr.** *La microarchitettura della CPU e la organizzazione del calcolatore definisce l'interfaccia del* **Data Bus (*nd* con cui comunica con l'esterno) e del bus di indirzzi *na* (bus dati riportato alla memoria).** Il progetto del calcolatore e l'organizzazione delle memorie definisce il parallelismo N delle memorie. **Per avere la massima efficienza nel trasferimento converrebbe avere n=nd=N** ma non è sempre così.

Oss:

Si chiama **parola** l'unità di elaborazione della CPU. La dimensione di parola dipende dalla CPU. Quando le parole sono di più byte bisogna capire come vengono memorizzate, esistono due **modelli di ordinamento**:

- **Little endian:** byte meno significativo all'indirizzo più basso
- **Big endian:** byte meno significativo all'indirizzo più grande



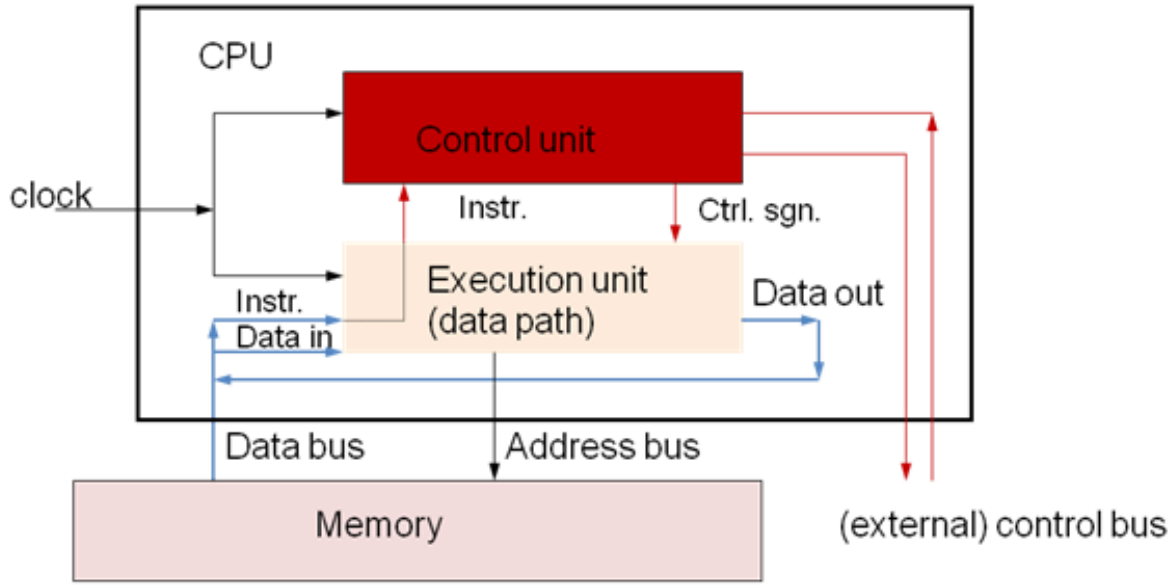
ELEMENTI MICROARCHITETTURA

La **microarchitettura** implementa le scelte dell'ISA e come detto è diversa da CPU a CPU, impiegando le reti logiche e i componenti di base. In questo momento si definiscono solo gli elementi di base sempre presenti in ogni CPU, almeno per implementare il ciclo di istruzione semplice come descritto.

Dal punto di vista funzionale, la microarchitettura della CPU prevede:

Data path il percorso dei dati o Unità operativa (UO) o **unità di elaborazione:** acquisisce istruzioni e dati, passa le istruzioni alla unità di controllo, esegue le operazioni di ALU, genera i risultati, calcola il prossimo indirizzo.

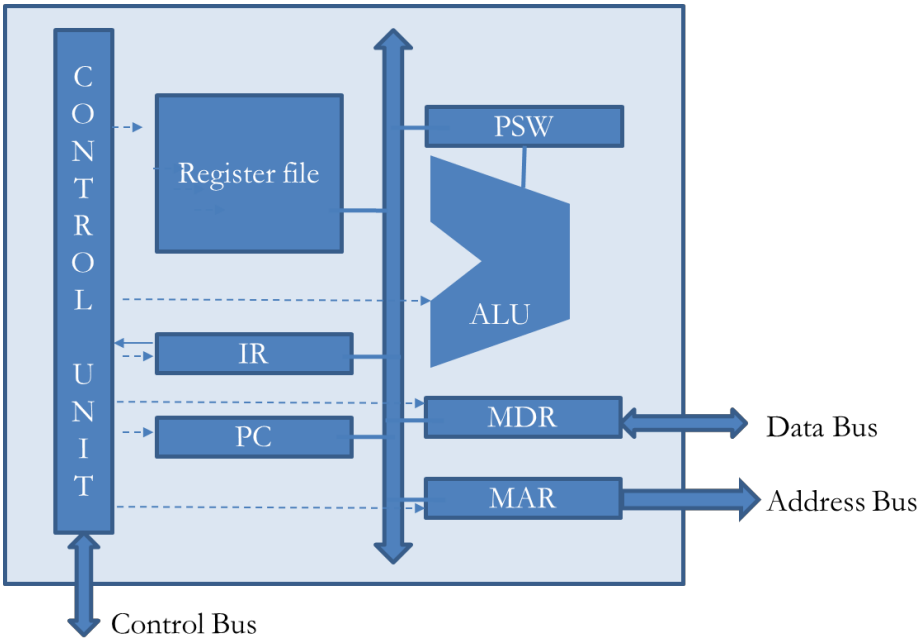
Control unit (l'unità di controllo UC) l'unità di controllo che gestisce l'unità di elaborazione e gestisce i segnali con l'esterno per tutto il calcolatore → **L'unità di controllo** deve decodificare le istruzioni ed eseguire tutte le microoperazioni necessarie.



I componenti interni minimi di una CPU dal punto di vista delle reti logiche sono Registri, reti combinatorie (ALU) e reti sequenziali nell'unità di controllo. Tra essi sono sempre presenti:

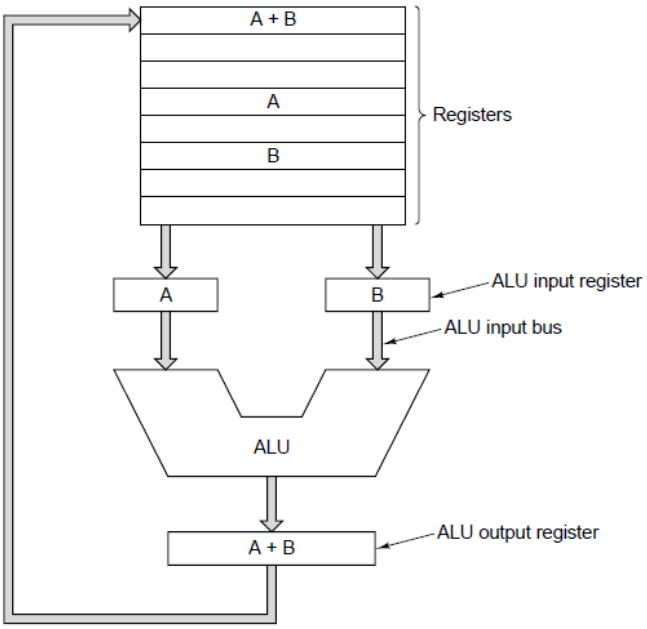
- **PC (Program Counter)** contiene l'indirizzo della prossima istruzione da eseguire → ***ha la dimensione di un indirizzo di memoria naddr.***
- **IR (Instruction Register)** contiene l'istruzione che è in fase di esecuzione → ***se l'ISA prevede istruzioni di lunghezza diversa l'IR avrà la lunghezza dell'istruzione più lunga.***

- **MAR (Memory Address Register)** è una porta di uscita per la CPU che contiene l'indirizzo della locazione di memoria da leggere o scrivere → ***ha tanti bit quanti ci sono nell'interfaccia esterna nel bus di indirizzo (na)*** e determina il massimo spazio di indirizzamento.
- **MDR (Memory Data Register)** è la porta di passaggio del dato che viene scambiato tra CPU e memoria o I/O; il dato può essere o istruzione o operando → ***ha tanti bit quanto il bus dei dati nd.***



- **Register FILE** o R insieme dei registri interni che mantengono gli operandi, e che sono usati dall'ISA. ***Hanno la dimensione n della parola***. Sono visibili dal programmatore nelle istruzioni assembly.
- **La ALU (Arithmetic logic unit):** è l'unità di elaborazione dei dati di tipo intero ha la dimensione del parallelismo n della CPU. È sempre collegata ad un registro di stato **PSW Processor status world o FLAG** che contiene i bit di stato modificati durante il funzionamento della ALU e bit di controllo.

A tali unità si sono aggiunte nel tempo molte altre unità come l'unità **FP unit** per l'elaborazione in virgola mobile.



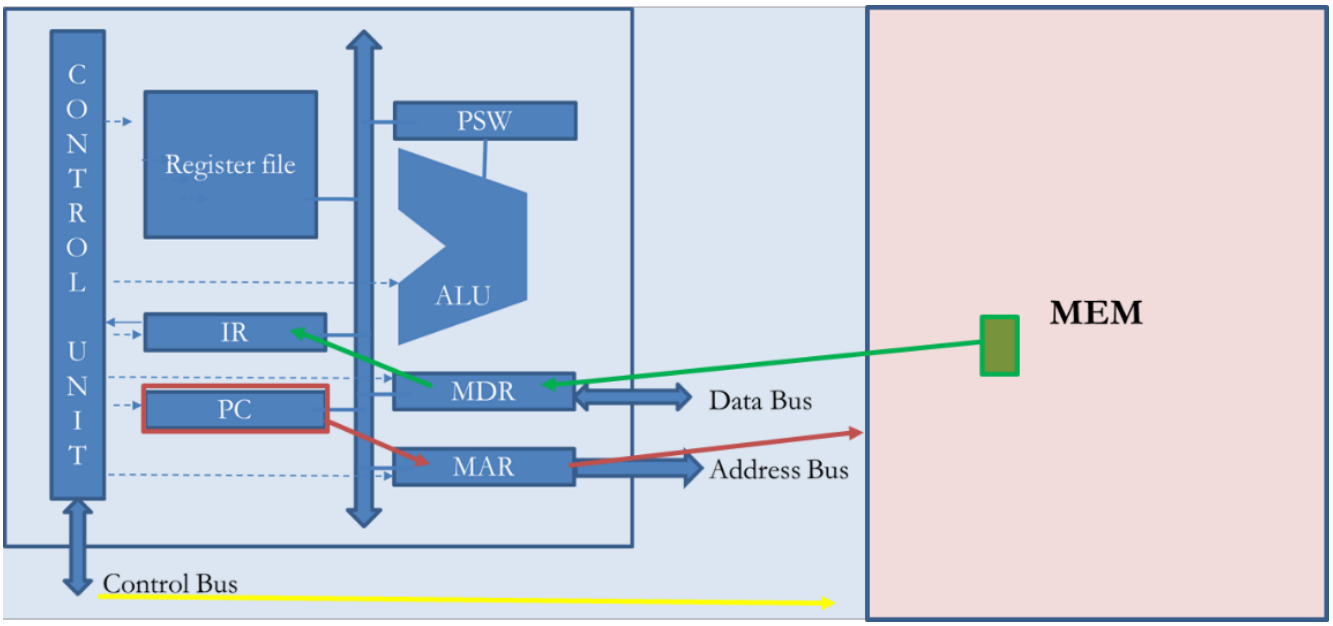
La ALU è collegata ai registri attraverso porte (normalmente chiamate A, B e AluOut) collegate a loro volta a multiplexer e decoder. Durante le fasi di esecuzione dell'istruzione si usano tutti i registri interni. **La CPU esegue una serie di micro-operazioni** durante la fase di esecuzione dell'istruzione.

1) Fase di FETCH

Nella fase di prelievo dell'istruzione si hanno le seguenti micro-operazioni:

MAR ← PC; **MDR ← M[MAR];** **IR←MDR,PC←PC+Nistr;**

Dove *Nistr è nbyte_istruzione.* Viene letta l'istruzione attraverso il MAR che riceve il contenuto di PC e attraverso il MDR l'istruzione che si trova in memoria all'indirizzo indicato nel mar (M[MAR]) viene caricata nell'IR. Mentre il registro IR viene caricato, contemporaneamente Il PC viene incrementato (di tanti byte quanti sono i byte dell'istruzione).



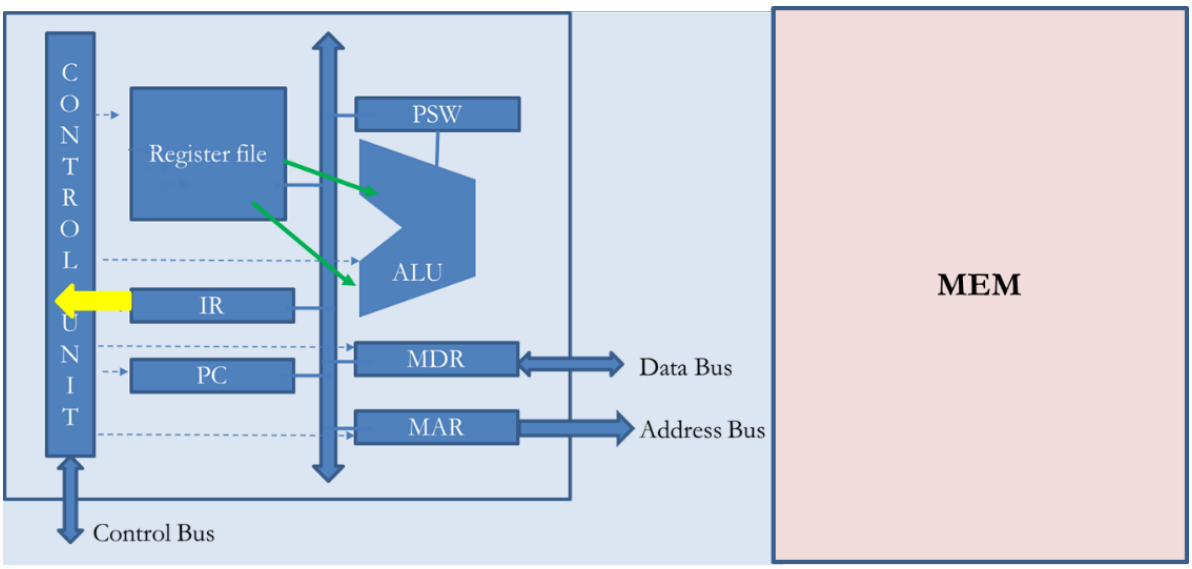
2) Fase di DECODE

Decode(IR(opcode)), Fetch(operandi);

Mentre nella Control Unit viene decodificato il codice operativo dell'istruzione, l'indirizzo degli operandi viene impiegato per portare i dati alla ALU se l'istruzione è

di ALUop (operazione di fetch degli operandi dai registri). Il Fetch operandi è il seguente:

A ←R[IF(sorg1)]; B←[IR(sorg2)];



Questa è un'operazione speculativa: viene eseguita prima della decodifica e potrebbe essere una operazione inutile, se l'istruzione non è una operazione di ALU.

3) Fase di EXECUTE

Viene eseguita l'istruzione con micro-operazioni diverse a seconda del tipo di istruzione. Qui di seguito si indicano i casi più comuni:

LETTURA: se l'istruzione prevede la lettura verso un registro (reg) dell'indirizzo A0 (es mov EAX, A0) si avrà un trasferimento del tipo

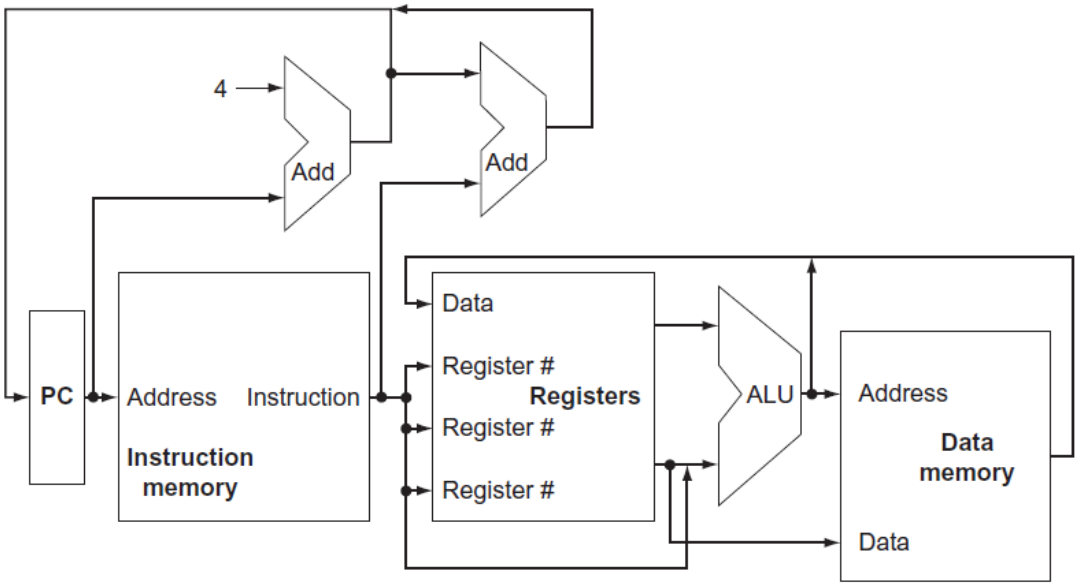
MAR ← IR(indirizzoA0); MDR←M[MAR]; R[IR(regdest)]←MDR;

La notazione R[x] indica il registro x-esimo del register file che viene riempito con il contenuto di MDR che ha ricevuto dalla memoria il dato che si trovava all'indirizzo indicato da MAR ossia A0.

SCRITTURA: il caso della scrittura è simile ma in ordine opposto.

MAR<←IR(indirizzoA0); MDR←R[IR(regdest)]; M[MAR]←MDR;

LA MICROARCHITETTURA DI UN PROCESSORE RISC-V



Fase di Fetch:

IR← IM[PC]; update (PC)

I registri MDR e MAR non sono mostrati e sono impliciti se la memoria è integrata nel chip e le due memorie sono divise. Il *PC, viene aggiornato* (update(PC)) *ad ogni istruzione* o con il contenuto del PC precedente +4 (Nistr) o con l'indirizzo di salto, se l'istruzione conteneva un salto. ***Il contenuto dell'istruzione (a parte l'OPCODE non mostrato) serve:***

- **per ALUop:** per indirizzare i registri sorgente e destinazione (architettura a 3 registri) da usarsi nelle istruzioni di ALU
- **per load/store:** per essere l'ingresso di una ALU che costruisce l'indirizzo per andare in memoria in caso di load o di store
- **per istruzioni di controllo**: per essere l'ingresso di una ALU che somma il valore al PC in caso di salto.

Fase di decode:

il codice operativo viene decodificato e viene fatto il fetch degli operandi

Fase di execute:

- e l'istruzione è una ALUop, la ALU somma (o fa l'operazione su) i registri e mette il risultato nel registro destinazione;

- in caso di scrittura o lettura in memoria le informazioni caricate sui registri o eventualmente sommate nella ALU sono indirizzi per la Data Memory e i dati escono dai registri(scrittura) o entrano (lettura).
- in caso di istruzione di controllo, viene modificato il PC.

EVOLUZIONE DELL'ARCHITETTURA

La **legge di Moore** è stata sempre un punto di riferimento: *stabilisce* ***che il numero dei transistor e la potenza di calcolo dei processori è destinata a raddoppiare ogni 18 mesi.*** → Per molti anni con miglioramenti tecnologi ed architetturali si è riusciti ad ottenere in raddoppio di prestazioni ogni anno e mezzo.

Dobbiamo considerare:

1) **Evoluzione dell'Architettura del calcolatore:**

   - Single CPU
   - Multiple CPU (architetture parallele)
   - CPU specificche (GPU, Neuralchips..)

2) **Evoluzione della single CPU:**

   - ISA Expansion
   - Pipelining
   - Parallelismo (Instruction Level Parallelism)
   - Esecuzione concorrente e parallela di processi e di unità di codice (thread)
   - Multi-core, many core

3) **Evoluzione delle memorie:**

   - Evoluzione della tecnologia delle memorie centrali
   - Gerarchia di memorie
   - Miglioramento delle prestazioni nella gerarchia
   - Cache coherency nei multiprocessori

4) **Evoluzione dell'I/O:**

   - Aumento del parallelismo e della frequenza
   - Disaccoppiamento tra controllo, trasferimento ed esecuzione
   - Evoluzione verso I/O processor