

Il modello di esecutore

Docente: Ing. Edoardo Fusella

Dipartimento di Ingegneria Elettrica e Tecnologie dell'Informazione

Via Claudio 21, 4° piano – laboratorio SECLAB

Università degli Studi di Napoli Federico II

e-mail: edoardo.fusella@unina.it

Cos'è un computer

- Un computer è un apparecchio elettronico
 - strutturalmente non ha niente di diverso da un televisore, uno stereo, un telefono cellulare o una calcolatrice elettronica
- progettato per eseguire autonomamente e velocemente attività diverse
 - Come tutte le macchine, non ha nessuna capacità decisionale o discrezionale, ma si limita a compiere determinate azioni *secondo procedure prestabilite*.
- Il computer è una macchina che in maniera automatica esegue operazioni “*elementari*” ad altissima velocità.
 - L'altissima velocità di elaborazione (ad es. milioni di istruzioni per secondo) fa sì che operazioni complicate
 - ... espresse mediante un gran numero di operazioni semplicisiano eseguite in tempi ragionevoli per l'ambiente esterno

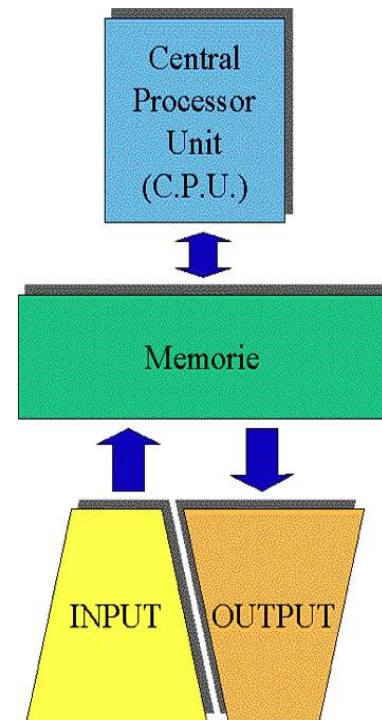
Algoritmo, Processo, Processore e Programma

- Informalmente, un *algoritmo* è la descrizione di un lavoro da far svolgere ad un *esecutore*.
 - ... se si vuole usare un computer bisogna non solo progettare preliminarmente un algoritmo, ma anche provvedere a comunicarglielo in modo che gli risulti comprensibile.
- L'esecuzione di un algoritmo da parte di un esecutore si traduce in una successione di azioni che vengono effettuate nel tempo.
- Si definisce *processo* il lavoro svolto eseguendo l'algoritmo, e *processore* il suo esecutore.
 - Il processo non è altro che l'elenco delle azioni effettivamente svolte come si susseguono nel tempo.
 - Ogni algoritmo è costituito da uno a più processi
- La descrizione di un algoritmo in un linguaggio comprensibile ad un calcolatore è detto Programma

Il modello di Von Neumann

- è uno schema di principio rappresentativo dei tradizionali computer.
- Prende il nome da Von Neumann, il primo ricercatore che lo propose nel 1945.

- La *Central Processing Unit (CPU)* coordina l'esecuzione delle operazioni fondamentali;
- La *memoria* contiene un programma che descrive le operazioni da eseguire e i dati su cui il programma stesso opera;
- i dispositivi di *input* e *output* sono le interfacce della CPU nei confronti del mondo esterno



Stored Procedure

- Il modello di Von Neumann si basa sul concetto di programma memorizzato:
 - la macchina immagazzina nella propria memoria i dati su cui lavorare e le istruzioni per il suo funzionamento.
- → flessibilità operativa
 - macchine nate per fare calcoli possono essere impiegate nella risoluzione di problemi di natura completamente diversa, come problemi di tipo amministrativo, gestionale e produttivo.
- Caratteristiche
 - schema di funzionamento semplice nelle sue linee generali
 - velocità e affidabilità nella esecuzione degli algoritmi;
 - ... milioni di istruzioni svolte dalla CPU in un secondo (*MIPS*: milioni di istruzioni per secondo)
 - un computer non commette mai errori di algoritmo poiché è un esecutore obbediente dell'algoritmo, la cui esecuzione gli è stata affidata
 - adeguata capacità di memoria
 - La capacità è la misura del numero di informazioni immagazzinabili nella memoria: oggi si misura in numero di byte
 - costo vantaggioso

Memorie

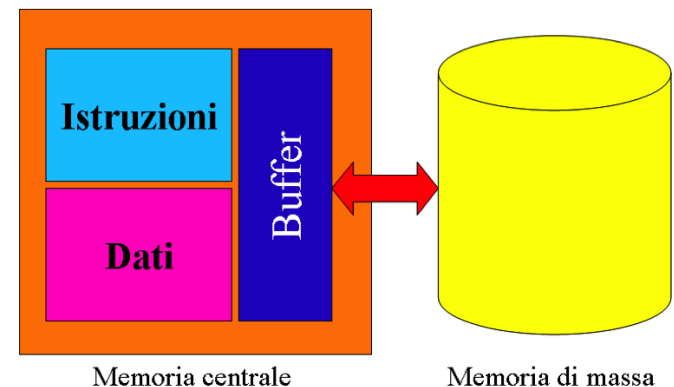
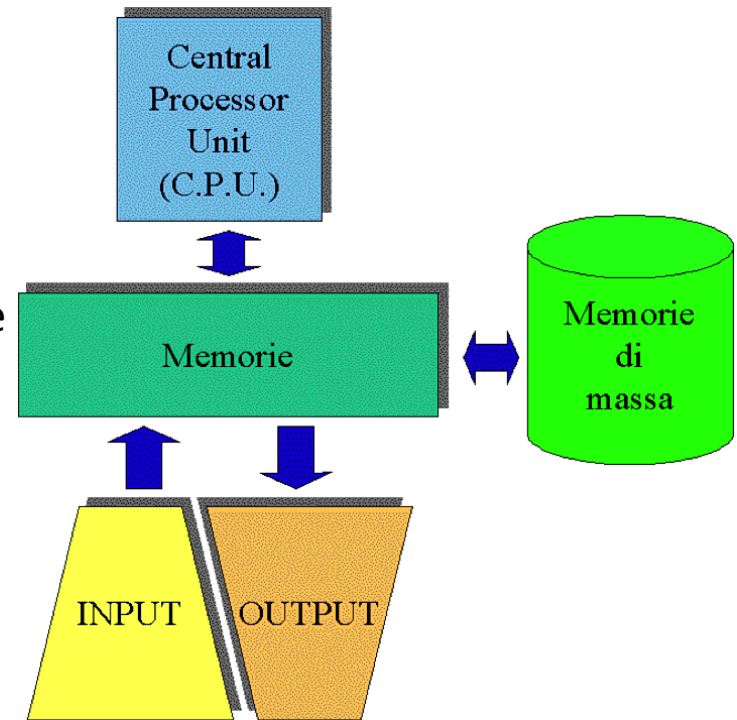
- In generale le memorie possono essere viste come un insieme di contenitori fisici, detti anche *registri*, di dimensioni finite e fissate a cui si può far riferimento mediante la posizione occupata nell'insieme detta *indirizzo* di memoria.
 - La dimensione di un registro si misura in numero di bit. Il bit è un dispositivo capace di assumere due sole condizioni:
 - nelle memorie di tipo elettronico sono circuiti detti flip-flop che mostrano un valore di tensione o uguale a 5 Volt o a 0 Volt;
 - nelle memorie di tipo magnetico è una sorta di calamita polarizzata o positivamente o negativamente;
 - nelle memorie di tipo ottico è una superficie con o senza un buco in modo da riflettere diversamente il raggio laser che la colpisce.
 - In ogni caso il dispositivo di lettura deve essere in grado di associare allo stato del bit il valore 1 (ad esempio tensione a 5 volt, polo positivo, assenza di buco) o il valore 0 (tensione a 0 volt, polo negativo, presenza di buco).
- Memorie con registri di otto bit sono dette a *byte o caratteri*; con più di otto (solitamente 16 o 32) vengono invece dette *a voce*.
 - I calcolatori moderni sono dotati di memorie a byte e le memorie a voce sono solo un ricordo del passato

Diversi tipi di memoria

- *Memoria ad accesso casuale*
 - ... il tempo di accesso non dipende dalla posizione - *R.A.M.* (Random Access Memory);
- *Memoria ad accesso sequenziale*
 - il tempo di accesso dipende dalla posizione, come avviene nei nastri magnetici.
- Alcune memorie vengono realizzate in modo che sia possibile una sola scrittura di informazioni. Tali memorie vengono dette a *sola lettura o ROM* (da *Read Only Memory*).
 - L'uso di queste memorie è necessario quando si desidera che alcune istruzioni o dati non siano mai alterati o persi.
- *Memorie volatili*
 - .. perdono le informazioni in esse registrate quando il sistema viene spento
 - ... memorie elettroniche;
- *Memorie permanenti*
 - memorie di tipo magnetico, ottico, a stato solido (SSD) e tutti i tipi di ROM.

Memorie di Massa

- Le memorie di massa sono memorie ausiliarie caratterizzate da una elevata capacità.
 - ... Le informazioni contenute nella memoria di massa devono essere dapprima trasferite nella memoria centrale e successivamente elaborate
 - ... le informazioni prodotte dalla CPU, viceversa, devono essere depositate in memoria centrale per poi essere conservate nelle memorie di massa.
- Le memorie di massa hanno tempi di accesso maggiori dovuti alle tecnologie impiegate per realizzarle.
- Per ovviare alla differenza di velocità tra i due dispositivi si impiegano tecniche che prevedono che la memoria centrale contenga anche *aree di accumulo dei dati in transito verso tutti i dispositivi esterni* (buffer).



Operazioni sulla memoria

- Load (lettura):
 - si preleva l'informazione contenuta nel registro senza però distruggerla
- Store (scrittura):
 - si inserisce una informazione nel registro eliminando quella precedente.
- Per comprendere il funzionamento di un registro di memoria si può pensare ad una lavagna il cui uso può essere così esemplificato:
 - leggere informazioni a patto che vi siano state scritte;
 - la lettura non cancella quanto scritto;
 - la scrittura di nuove informazioni obbliga a cancellare quelle precedenti che pertanto vengono perse.
- La memoria è un sistema che assolve al compito di conservare il dato, depositandolo in un registro nel caso di operazione di scrittura, e di fornire il dato conservato in un registro in caso contrario di operazione

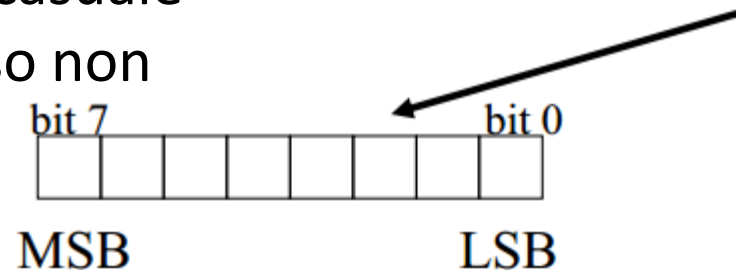
Funzionamento di una Memoria

- La CPU indica preventivamente l'indirizzo del registro interessato dall'operazione;
- La memoria decodifica tale indirizzo abilitando solo il registro ad esso corrispondente affinché:
 - per uno *store* copi il dato del buffer nel registro;
 - per un *load* copi il dato del registro nel buffer.
 - dove il *buffer* può essere vista come un'area di transito dei dati dalla CPU alla memoria e viceversa.
- Le operazioni di load e store richiedono tempi di attuazione che dipendono dalle tecnologie usate per la costruzione delle memorie e dalle modalità di accesso.
 - Nel caso di load, il tempo di accesso misura il tempo che trascorre tra la selezione del registro di memoria e la disponibilità del suo contenuto nel registro di buffer. I
 - Il tempo di accesso nel caso dello store misura invece il tempo necessario alla selezione del registro e il deposito del contenuto del registro di buffer in esso

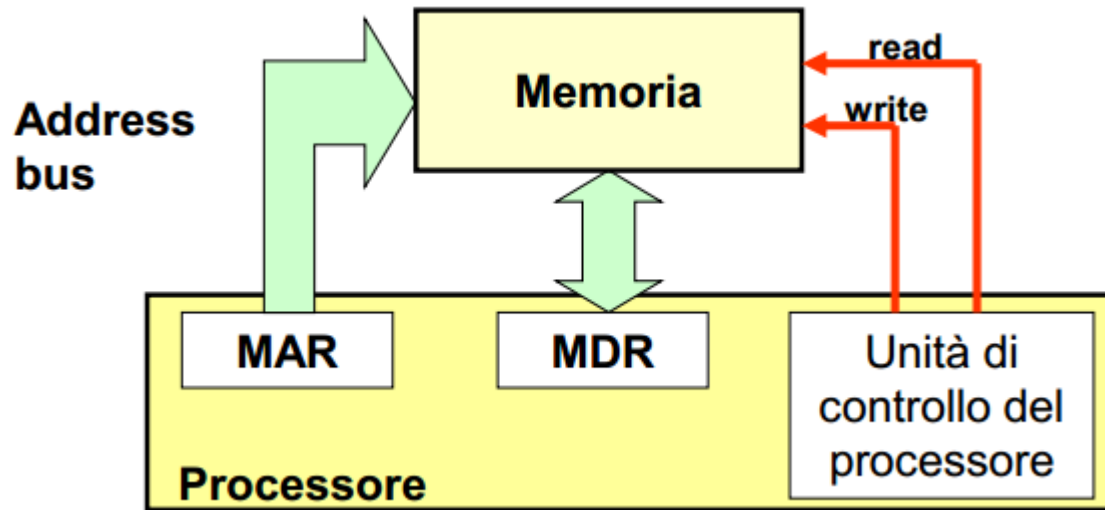
La memoria centrale

- k locazioni di memoria (con $k = 2^n$)
- Indirizzo: rappresentazione univoca della locazione di memoria (n bits)
- Contenuto: valore memorizzato nella locazione (m bits)
- Memoria con una capacità di k x m bits
- $[0, k-1]$ = Spazio di indirizzamento
- La memoria centrale è ad accesso casuale (RAM) in quanto il tempo di accesso non dipende dalla posizione del dato

0000	
0001	
0010	
0011	00101101
0100	
0101	
0110	
	⋮
1101	10100010
1110	
1111	

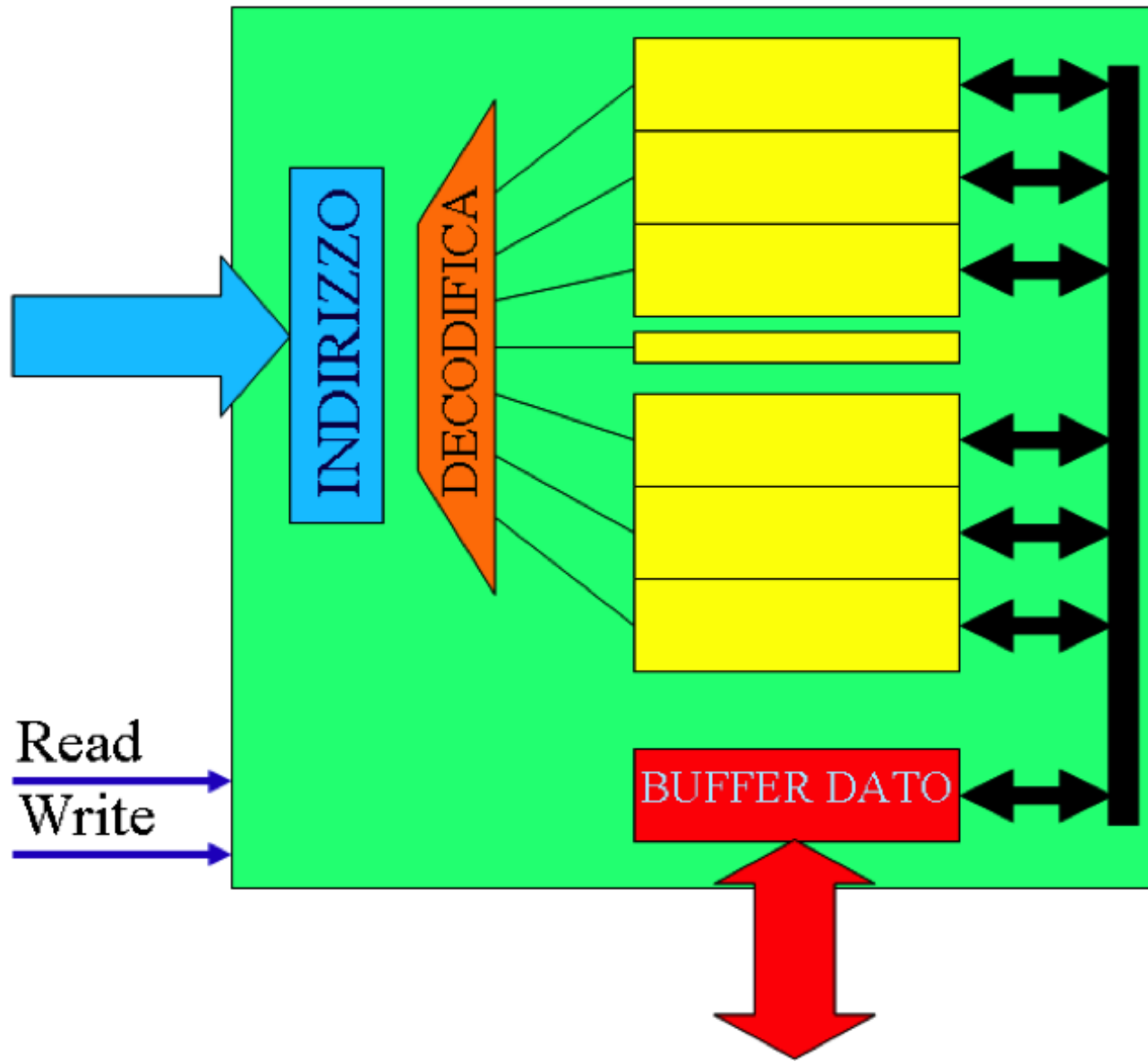


Interazione processore-memoria



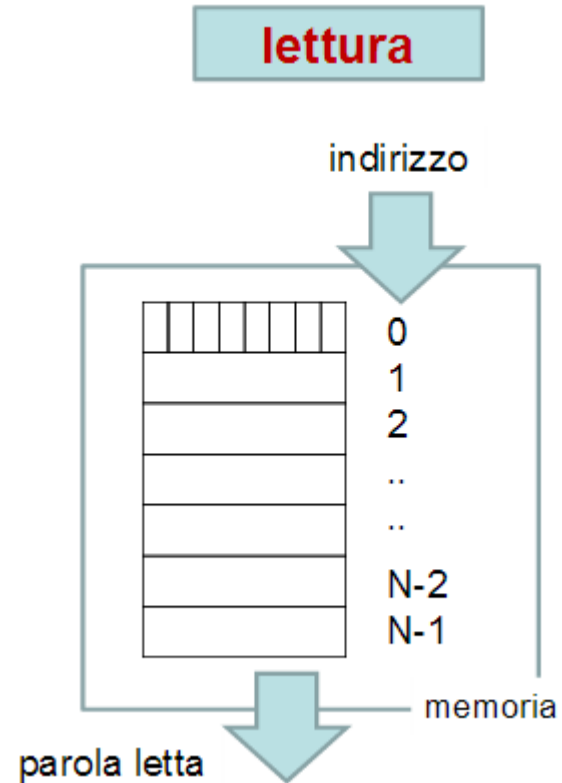
- Registro Memory Address (MAR)
 - fornisce alla memoria l'indirizzo del dato da leggere/scrivere
- Registro Memory Buffer o Memory Data (MBR/MDR)
 - contiene il dato da scrivere *verso*, o da leggere *dalla* memoria
- Segnali di controllo: *read* / *write*
 - ordinano alla memoria operazioni di lettura oppure di scrittura

Schema



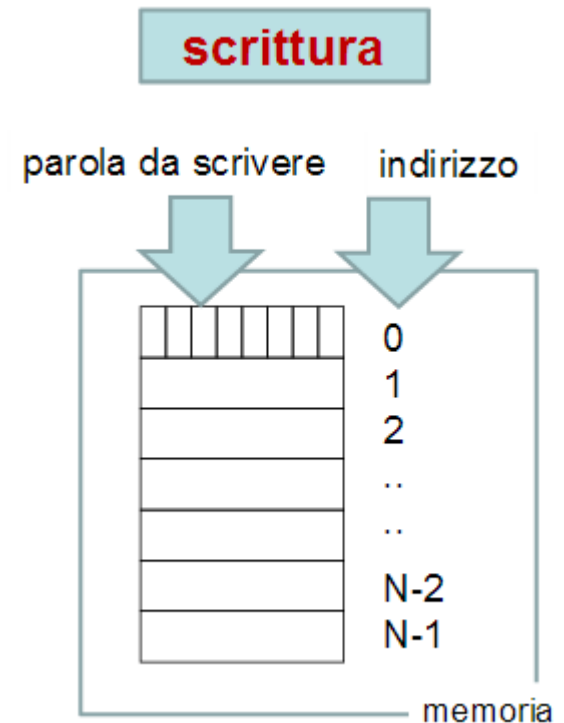
Operazione di lettura

- Le operazioni di **lettura** si effettuano fornendo alla memoria l'informazione che indica dove leggere la parola
 - viene ovvero fornito **l'indirizzo**
 - la memoria risponde fornendo il valore della parola contenuta a quell'indirizzo



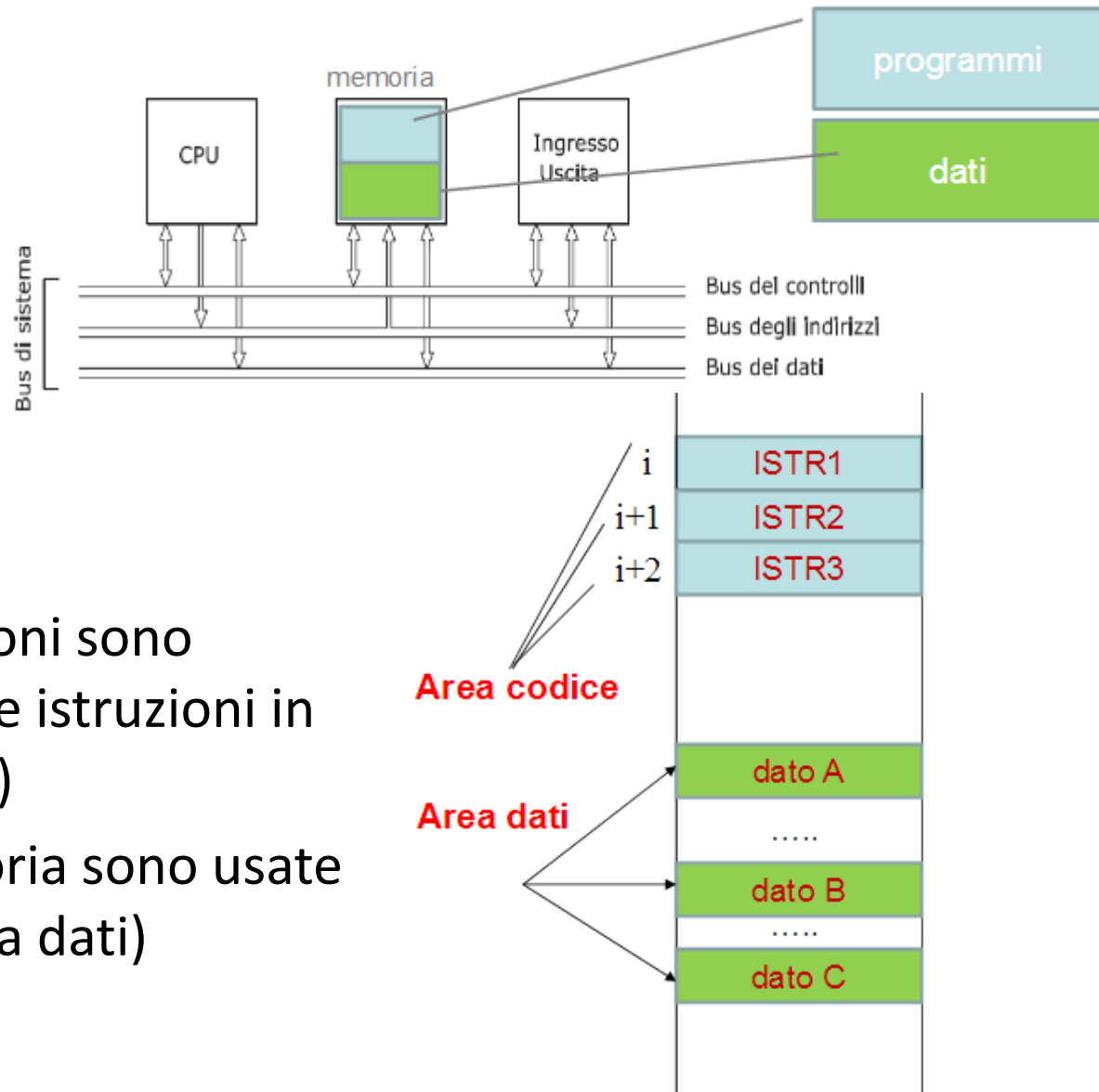
Operazione di scrittura

- Le operazioni di **scrittura** si effettuano fornendo alla memoria la parola da scrivere e l'informazione che indica dove scrivere la parola stessa
 - vengono ovvero forniti sia il **dato** che l'**indirizzo**
 - in questo caso la memoria non fornisce risposta

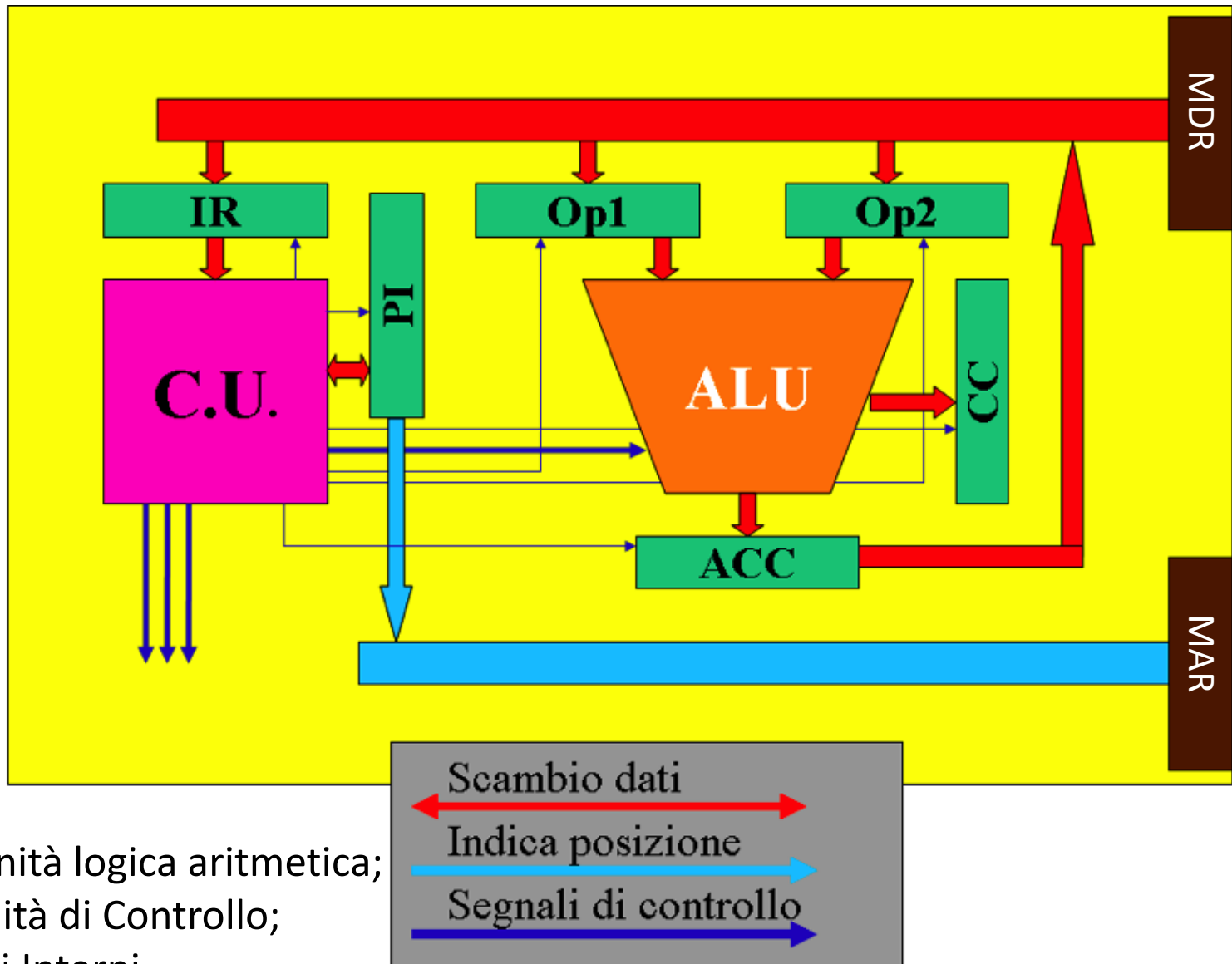


Memoria dati ed istruzioni

- La memoria è usata per ospitare sia le istruzioni (programmi) che il processore deve eseguire, sia i dati su cui deve lavorare
- Alcuni gruppi di locazioni sono utilizzati per ospitare le istruzioni in sequenza (area codice)
- Altre parti della memoria sono usate per ospitare i dati (area dati)



La C. P. U.



- ALU: unità logica aritmetica;
- CU: Unità di Controllo;
- Registri Interni

Control Unit

- È preposto
 - ... all'interpretazione delle singole istruzioni e all'attivazione di tutti i meccanismi necessari per eseguire le singole istruzioni.
- In particolare la CU ha il compito di:
 - prelevare ogni istruzione dalla memoria centrale e la decodifica,
 - prelevare i dati dalla memoria se servono all'istruzione,
 - eseguire l'istruzione.
 - Per esempio: se l'istruzione prelevata è di tipo aritmetico e richiede due operandi, la CU predispone dapprima il prelievo dalla memoria di tali operandi, attiva poi l'ALU perché esegua l'operazione desiderata, ed infine deposita il risultato di nuovo in memoria.
- Al termine dell'esecuzione di una istruzione la CU procede al prelievo dalla memoria della successiva istruzione secondo un ordine rigidamente sequenziale
 - l'esecuzione di una istruzione può avere inizio solo se la precedente è stata portata a termine.

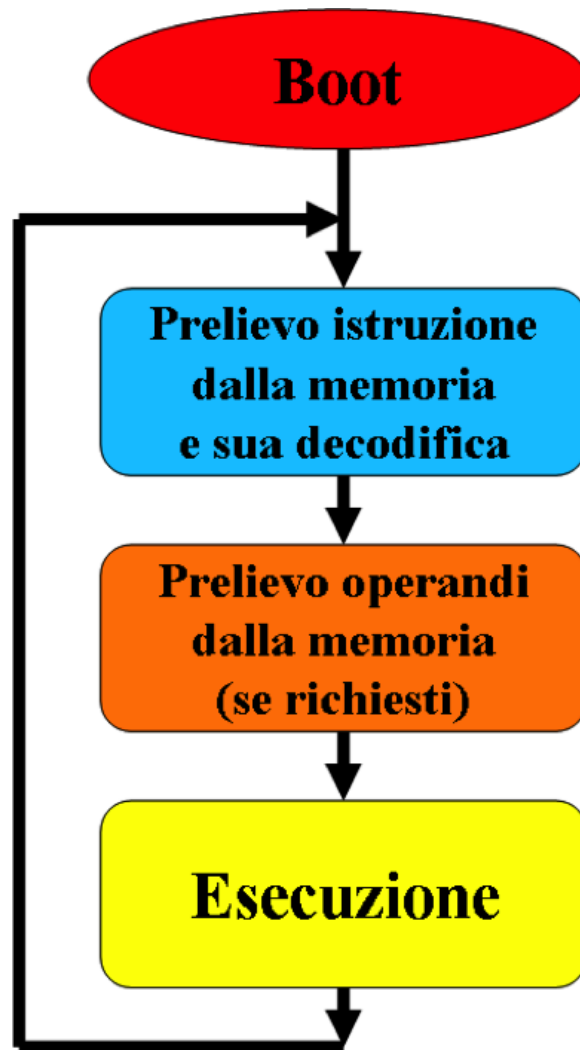
Aritmetic Logic Unit

- Esegue operazioni
 - aritmetiche, di confronto o bitwise
- sui dati della memoria centrale o dei registri interni.
- L'esito dei suoi calcoli viene segnalato da appositi bit in un registro chiamato *Condition Code*.
- A seconda dei processori l'ALU può essere molto complessa.
- Nei sistemi attuali l'ALU viene affiancata da processori dedicati alle operazioni sui numeri in virgola mobile detti *coprocessori matematici*.
- Durante le sue elaborazioni la CU può depositare informazioni nei suoi registri interni in quanto sono più facilmente individuabili e hanno tempi di accesso inferiori a quelli dei registri della memoria centrale.

Registri Interni speciali

- Il numero e tipo di tali registri varia a seconda dell'architettura della CPU.
- In molte CPU
 - *Instruction Register (IR)*
 - contiene l'istruzione prelevata dalla memoria e che l'unità di controllo sta eseguendo
 - *Prossima Istruzione (PI) o Program Counter (PC)*
 - ricorda alla CU la posizione in memoria della successiva istruzione da eseguire.
 - Nei casi in cui ogni registro di memoria contenga un'intera istruzione, e l'insieme delle istruzioni del programma sia disposto ad indirizzi consecutivi, la CU incrementa di uno il valore contenuto in PI dopo ogni prelievo di una istruzione dalla memoria
 - *Accumulatore (ACC)*
 - serve come deposito di dati da parte dell'ALU nel senso che contiene prima di un'operazione uno degli operandi, e al termine della stessa operazione il risultato calcolato.
 - In questo caso i registri *Op1* e *Op2* diventano interni all'ALU
 - *Condition Code (CC)*
 - indica le condizioni che si verificano durante l'elaborazione, quali risultato nullo, negativo e overflow

Il Ciclo del Processore



- Le tre fasi del ciclo vengono anche dette
 - *fetch*
 - *operand assembly*
 - *execute*.

Considerazioni

- Fase di Boot
 - Perchè il ciclo del processore possa avere inizio si deve predisporre in modo che il registro PI contenga l'indirizzo del registro di memoria contenente la prima istruzione da eseguire
- Una volta avviato, il ciclo del processore non termina mai e quindi ad ogni istruzione deve sempre seguirne un'altra da eseguire successivamente.
 - quando termina un'applicazione un elaboratore torna ad eseguire i programmi del sistema operativo.
 - Perché tutto ciò proceda nel rispetto del modello di Von Neumann, deve avvenire che in memoria siano sempre presenti i programmi e i dati del sistema

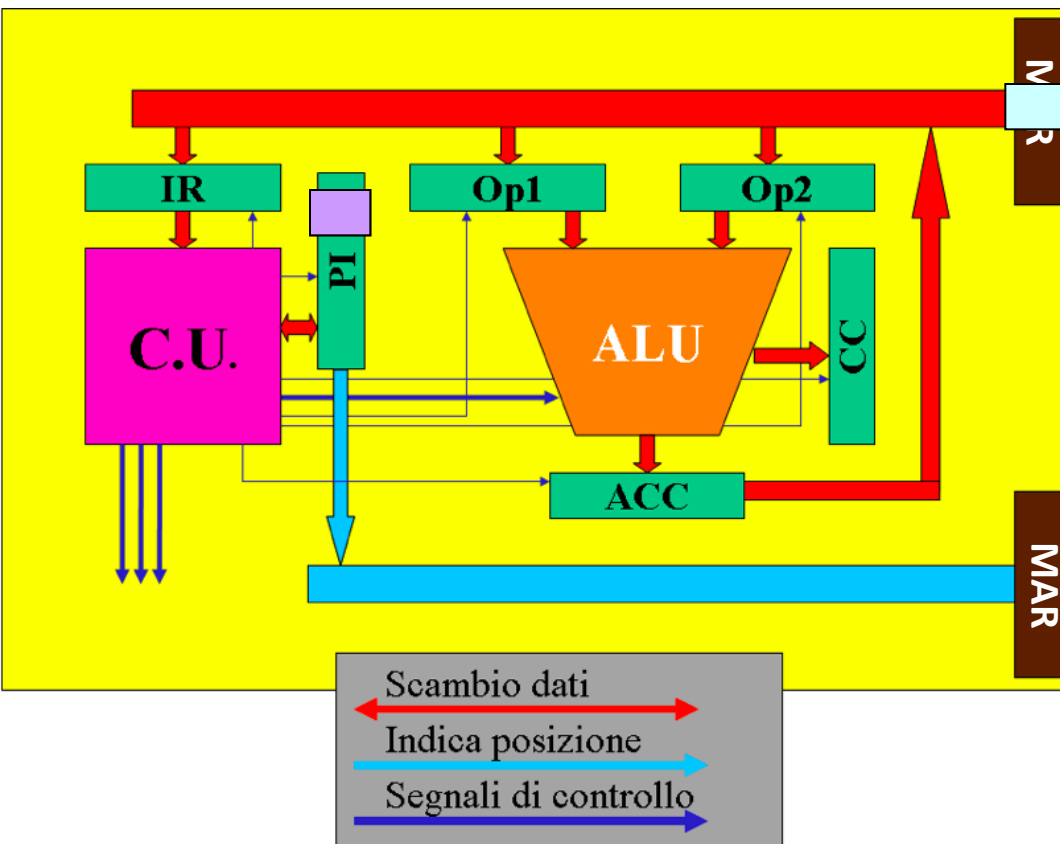
Ciclo del Processore: dettaglio

- La **fase fetch** inizia con il prelievo dell'istruzione dalla memoria.
 - Per farlo la UC comunica alla memoria il puntatore ad istruzione presente nel registro PI.
 - La risposta della memoria viene depositata nel registro IR così da consentire alla UC di:
 - interpretare il codice operativo dell'istruzione da eseguire;
 - conoscere i puntatori ai dati di input ed output;
 - ricevere il puntatore all'istruzione da eseguire successivamente.
- La **fase fetch si conclude** con l'aggiornamento del registro PI con il valore del puntatore all'istruzione successiva presente in IR.
- La **fase operand assembly** serve alla UC per predisporre gli operandi che servono al codice operativo.
 - Per farlo la UC usa i puntatori ai dati contenuti nel registro IR.
- La **fase execute** consiste nel mettere in essere le azioni richieste con il codice operativo presente nel registro di IR.
- Nel caso vengano prodotti risultati, ne verrà effettuata la memorizzazione negli indirizzi specificati dai puntatori ai dati di output presenti nel registro IR.

Fetch

Fase di fetch

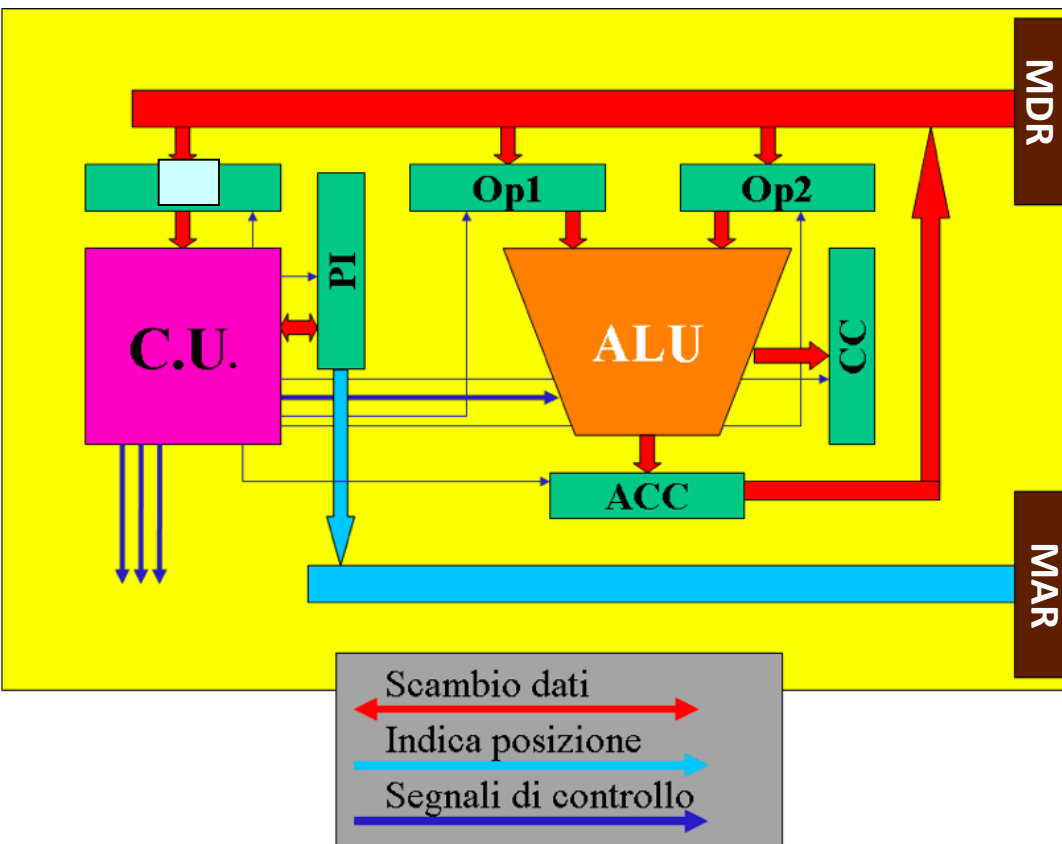
1. Per avviare un programma, residente nella memoria principale, bisogna inserire l'indirizzo della prima istruzione nel PI.
2. La CU ricopia il contenuto del PI nel MAR e viene effettuato un accesso in lettura a quell'indirizzo.
3. Il dato prelevato dall'indirizzo, cui si è acceduto, viene conservato in MDR.
4. Il contenuto di MDR viene ricopiato in IR. Nel frattempo PI si è, in maniera automatica, aggiornato alla locazione successiva.



Decode

Fase di decode

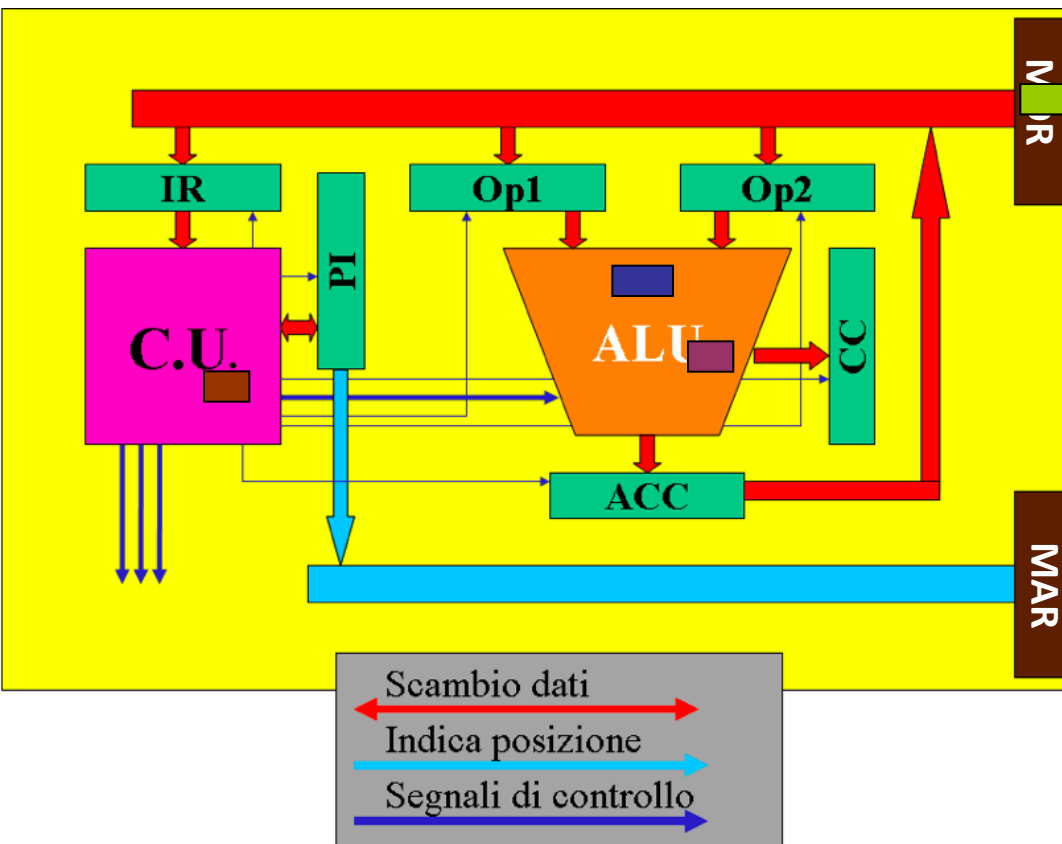
L'istruzione viene interpretata, prelevando eventualmente dalla memoria altri dati necessari. In ogni caso, alla fine, PI punterà all'istruzione da eseguire successivamente..



Execute

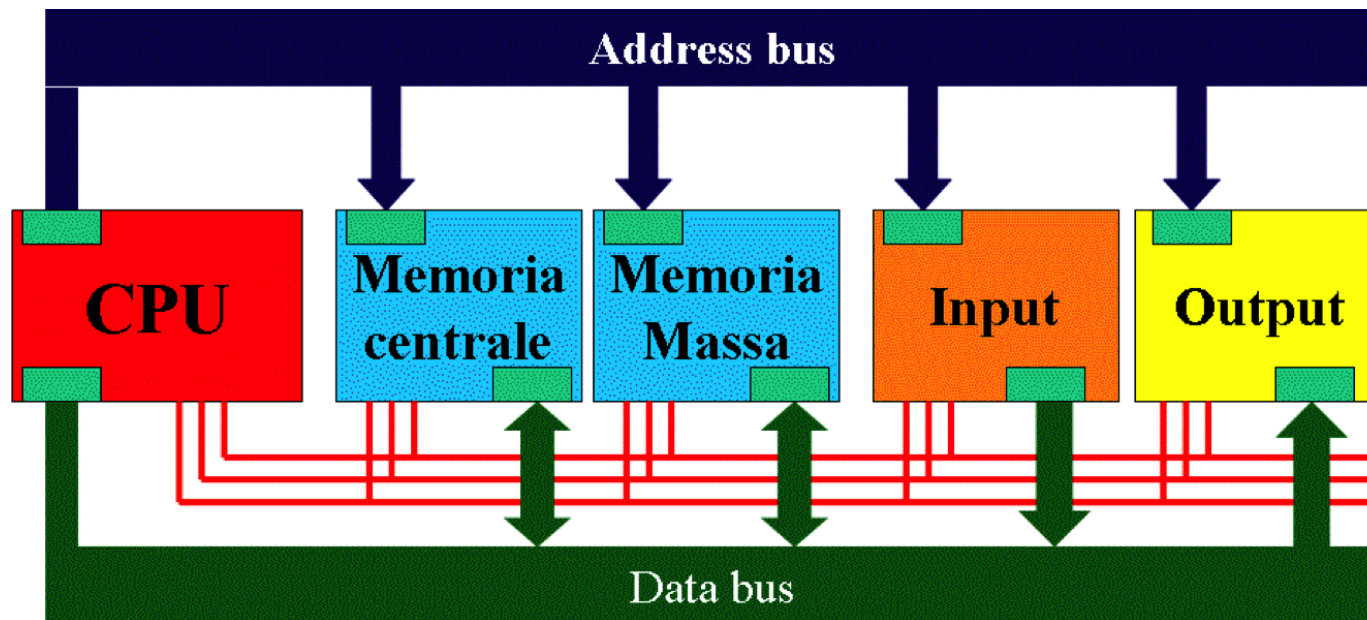
Fase di execute

Se, per esempio, l'istruzione da eseguire è una somma, i registri verranno caricati con i numeri da sommare e la CU invia alla ALU il comando per l'esecuzione della somma. La ALU esegue l'operazione: il risultato prodotto sarà conservato in uno dei registri e verrà aggiornata il CC.



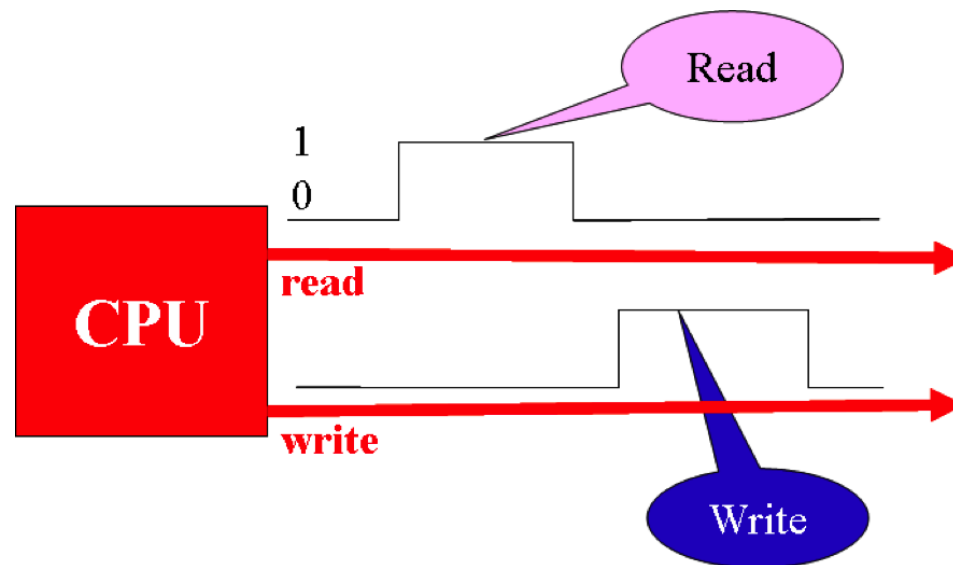
Il Bus

- E' un canale di comunicazione *condiviso* da più utilizzatori.
- E' fisicamente costituito da uno o più fili su cui possono transitare uno o più bit contemporaneamente.
- A seconda delle informazioni trasportate si hanno:
 - *bus dati (data bus)*
 - *bus indirizzi (address bus)*
 - *bus comandi o di controllo (command o control bus)*



Lettura e Scrittura

- Il control bus serve alla C.U. per indicare ai dispositivi cosa essi devono fare.
 - Tipici segnali del control bus sono quelli di *read* e *write* mediante i quali la CU indica ai dispositivi se devono leggere un dato dal data bus (*read*) o scriverlo su di esso (*write*).
 - Il data bus permette ai dati di fluire da CPU a registro di memoria selezionato per operazioni di *store* e viceversa per quelle di *load*



Address Bus

- L'address bus serve alla CU per comunicare l'indirizzo del dispositivo interessato da una operazione di lettura o scrittura.
 - In questa ottica anche i dispositivi di input /output sono identificati da un indirizzo alla stessa stregua dei registri di memoria.
- Tutti i componenti del sistema (memoria, input, output, memoria di massa, etc.) devono essere dotati della capacità di riconoscere sull'address bus il proprio indirizzo.
 - In altri termini attraverso l'address bus la CU effettua la selezione del dispositivo a cui sono rivolti i comandi e i dati.

Scambio di Informazioni

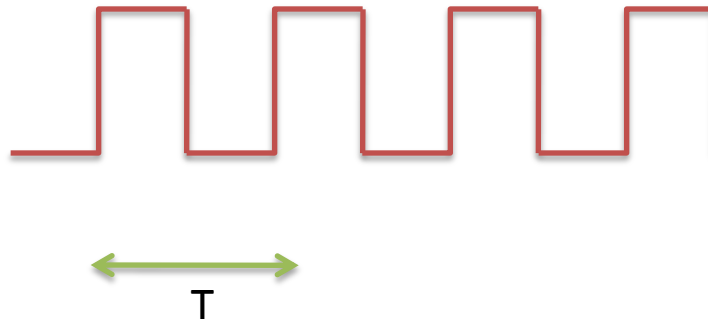
- La CPU è l'unico elemento che fornisce l'indirizzo all'address bus;
- Memorie e dispositivi di input ed output devono *ascoltare l'address bus* per attivarsi quando su di esso compare il proprio indirizzo identificativo;
 - nel caso della memoria l'attivazione avviene quando viene riconosciuto l'indirizzo corrispondente ad uno dei registri di cui essa è composta;
 - il dispositivo attivo deve interpretare i segnali del control bus per eseguire i comandi della CU;
 - le memorie prelevano dati dal data bus o immettono dati in esso in funzione del comando impartito dalla CU;
- i dispositivi di input possono solo immettere dati sul data bus;
- viceversa i dispositivi di output possono solo prelevare dati dal data bus.

Bus Seriale e Parallelo

- Un bus costituito da un solo filo è chiamato bus *seriale* e su di esso i bit transitano uno dietro l'altro.
- Un bus costituito da n fili è chiamato bus *parallelo* perché su di esso transitano n bit alla volta.
 - Tipici sono i bus a 8 e 32 fili sui quali si possono trasferire rispettivamente 8 e 32 bit (4 Byte) alla volta.
- L'address e il data bus sono paralleli e le loro dimensioni caratterizzano i sistemi di calcolo.
 - Il numero di bit dell'address bus indica la *capacità di indirizzamento* della CPU: ossia la sua capacità di gestire la dimensione della memoria centrale e il numero di dispositivi di input ed output.
 - Infatti un address bus con n bit consente di selezionare un registro tra 2^n .
 - La dimensione del data bus condiziona invece la velocità di scambio delle informazioni tra i diversi dispositivi in quanto con m fili solo m bit possono viaggiare contemporaneamente.

Il clock

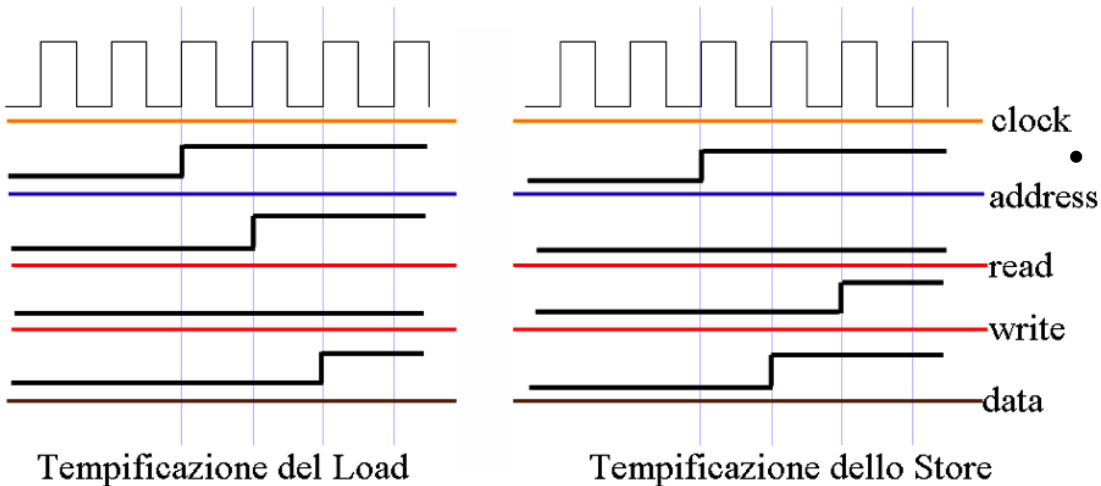
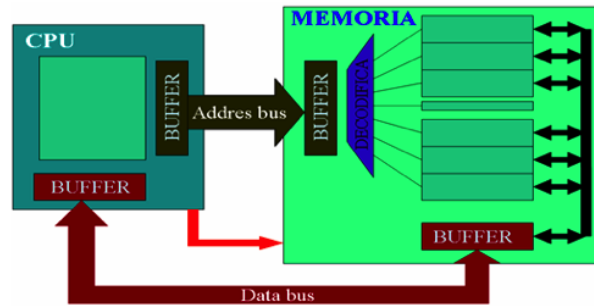
- Le attività di tutti i dispositivi vengono sincronizzate tra loro mediante un orologio interno che scandisce i ritmi di lavoro.
- Il *clock* è un segnale periodico di periodo fisso
 - un'onda quadra caratterizzata da un periodo T (detto ciclo) e da una frequenza f ($f=1/T$) misurata in Hertz (Hz).
 - Ad esempio un clock composto da 10 cicli al secondo ha la frequenza $f = 10 \text{ Hz}$ e il periodo $T = 0.1\text{s} = 100\text{ms}$.
 - La frequenza dei clock presenti nei moderni sistemi spazia dai MHz (1 MHz corrisponde a un milione di battiti al secondo) ai GHz (1 GHz corrisponde a un miliardo di battiti al secondo).



Clock e velocità di elaborazione

- La velocità di elaborazione di una CPU dipende dalla frequenza del suo clock
 - più accelerato è il battito del clock maggiore è la velocità di esecuzione.
- Alla frequenza del clock è legato il numero di operazioni elementari che vengono eseguite nell'unità di tempo dalla CU
 - Ad esempio, se si assume che ad ogni ciclo di clock corrisponde esattamente l'esecuzione di una sola operazione, allora la frequenza del clock indica il numero di operazioni che vengono eseguite nell'unità di tempo dalla UC.
 - ... con un clock a 3 GHz si ha che il processore è in grado di eseguire 3 miliardi di operazioni al secondo.
 - ...In realtà tale ipotesi non è sempre vera in quanto l'esecuzione di una operazione può richiedere più cicli di clock sia per la complessità delle operazioni che per la lentezza dei dispositivi collegati alla CPU.

Esempio di Tempificazione



- **LOAD**

- con il primo battito di clock si pone l'indirizzo del registro di memoria, di cui si vuole leggere il contenuto, sull'address bus;
- con il secondo battito si segnala alla memoria che si tratta di una operazione di read;
- con il terzo battito si prende il dato dal data bus, dove la memoria ha provveduto a depositarlo.

- **STORE**

- con il primo battito di clock si pone l'indirizzo del registro di memoria, in cui si vuole scrivere il contenuto, sull'address bus;
- con il secondo battito si deposita il dato sul data bus;
- con il terzo battito si segnala alla memoria che si tratta di una operazione di write e quindi che il dato è pronto per essere depositato nel registro selezionato.

Istruzioni

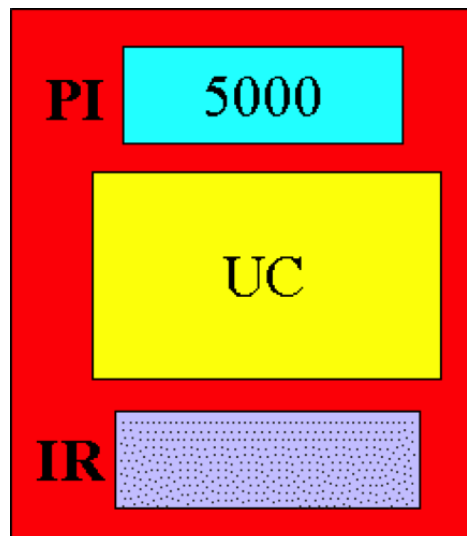
- Le istruzioni sono operazioni semplici:
 - trasferimento dati da un registro ad un altro
 - (da memoria a memoria,
 - da memoria a registri della CPU o viceversa,
 - da memoria a output, da input a memoria);
 - operazioni aritmetiche o logiche eseguite dall'ALU;
 - controllo di condizioni riportate dal registro CC o deducibili dal confronto di due registri.

Istruzione in Linguaggio Macchina

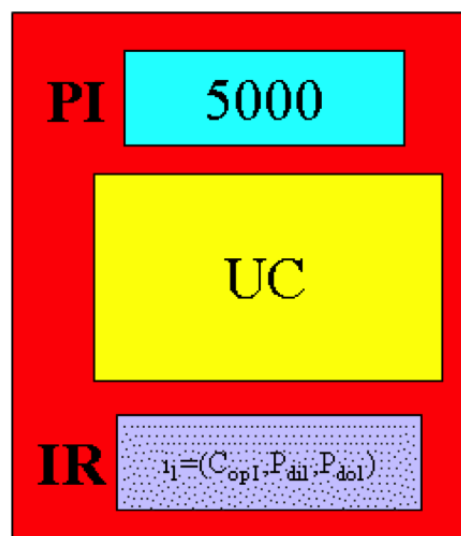
- Concettualmente è una quadrupla:
 - $i = (C_{op}, P_{di}, P_{do}, P_{is})$
 - in cui:
 - C_{op} è il codice operativo, ossia il codice che indica alla UC della CPU l'operazione da compiere;
 - l'insieme dei C_{op} prendere il nome di *repertorio di istruzioni* e dipende dalla specifica CPU;
 - P_{di} sono i puntatori ai dati che servono per svolgere l'operazione C_{op} detti anche di input;
 - esistono istruzioni che non hanno operandi di input;
 - P_{do} sono i puntatori ai dati prodotti dall'operazione C_{op} detti anche di output;
 - ... esistono istruzioni che non hanno operandi di output;
 - P_{is} è il puntatore all'istruzione da svolgere al termine dell'esecuzione di quella corrente.

Allocazione contigua di istruzioni

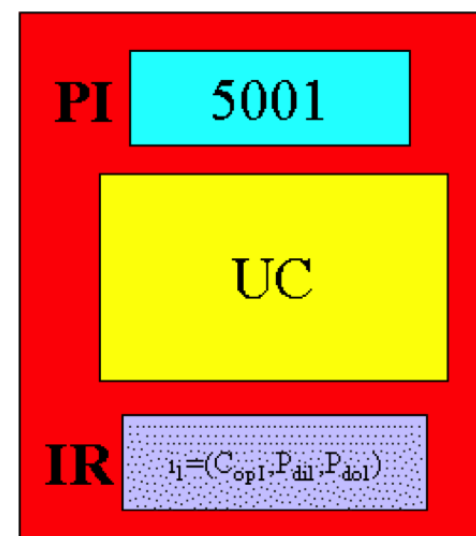
- Il programmatore dispone le istruzioni ad indirizzi consecutivi di memoria e facendo in modo che la UC nella fase fetch aggiorni il PI semplicemente incrementando il suo contenuto della lunghezza dell'istruzione.



T1: invio alla memoria del contenuto di PI



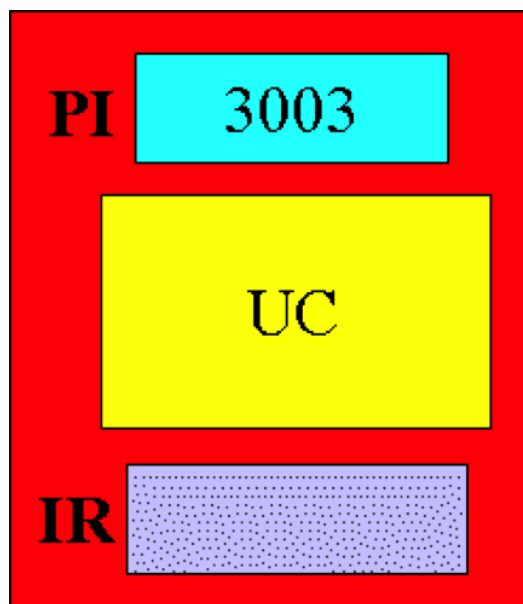
T2: istruzione inserita in IR



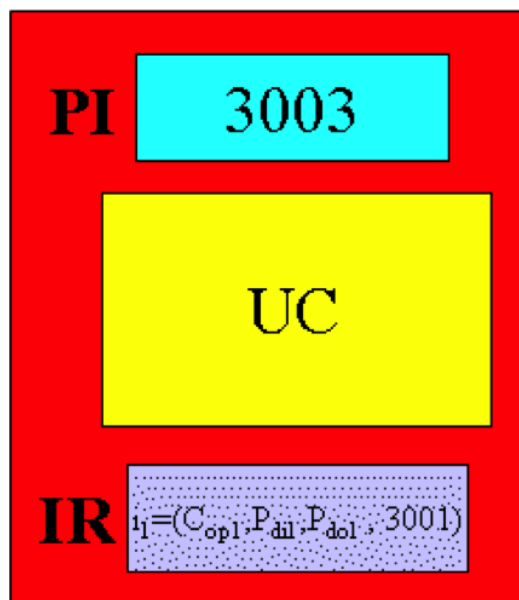
T3: aggiornamento di PI

Allocazione non contigua di istruzioni

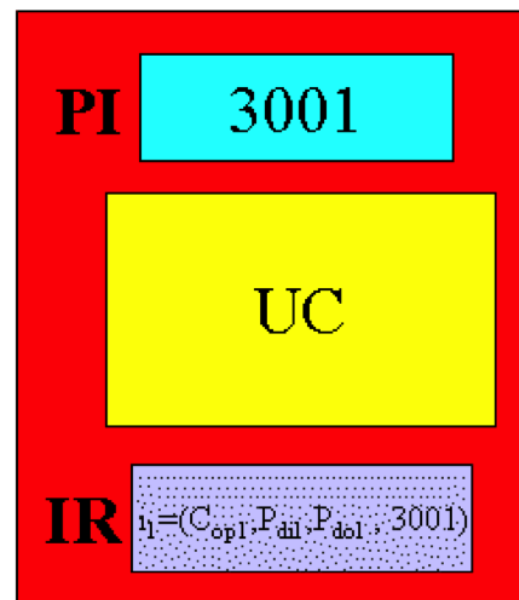
- Il programmatore dispone le istruzioni ad indirizzi non consecutivi di memoria. Il puntatore all'istruzione successiva è contenuto in quella precedente



T1: invio alla memoria
del contenuto di PI

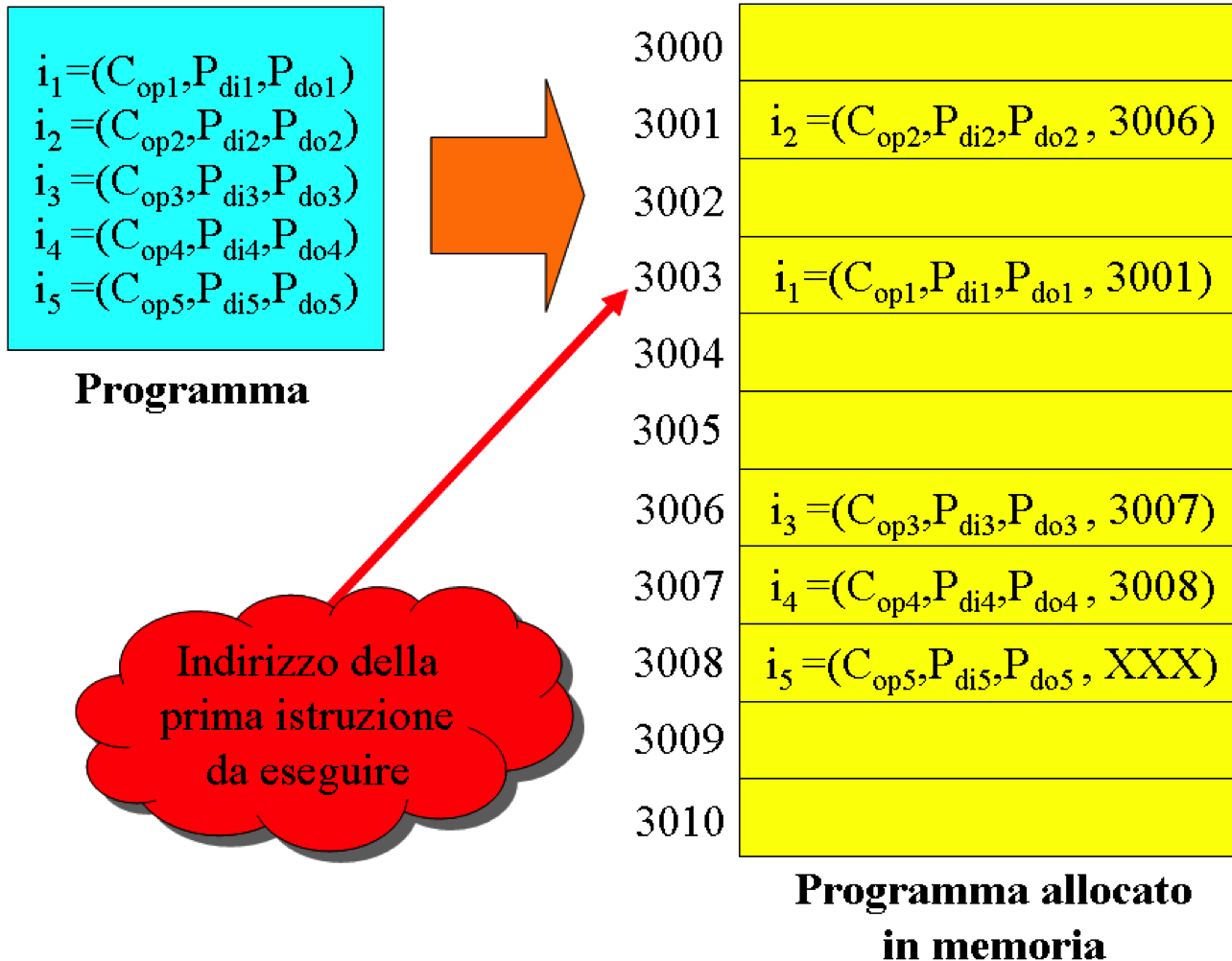


T2: istruzione inserita
in IR



T3: aggiornamento di
PI

Esempio



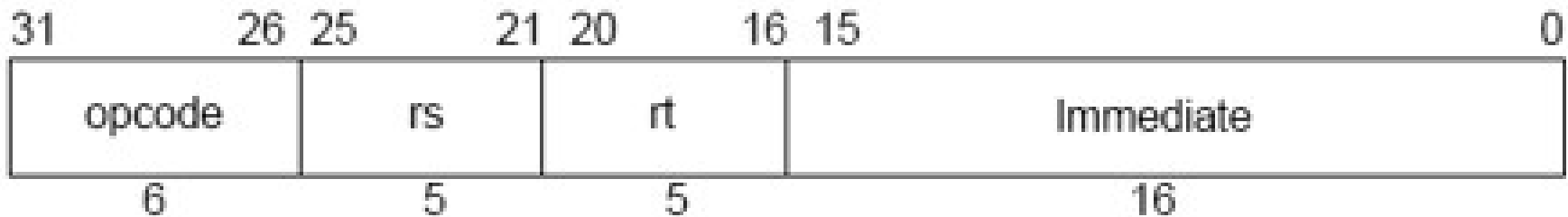
Formato Istruzione

Codice Operativo	Tecnica Indirizzamento	Operandi
------------------	------------------------	----------

- Codice operativo: identifica il tipo di istruzione
- Tecnica Indirizzamento: indica la modalità di composizione dell'indirizzo
- Operandi: indica gli operandi di ingresso e uscita dell'istruzione

Esempio Formato Istruzione

Formato I del processore MIPS (1/2)



- Istruzioni a 32 bit
- Codice operativo nei bit 31-26
- Registro di ingresso: rs
- Registro di uscita: rt
- Immediate: valore rappresentato in complemento

Esempio Formato Istruzione

Formato I del processore MIPS (2/2)

decimal representation:

8	22	21	-50
---	----	----	-----

binary representation:

001000	10110	10101	1111111111001110
--------	-------	-------	------------------

- Istruzione di somma con immediato: `addi $21,$22,-50`
- Codice operativo: 8
- Registro di ingresso: 22
- Registro di uscita: 21
- Immediate: -50

Modi di Indirizzamento (1/2)

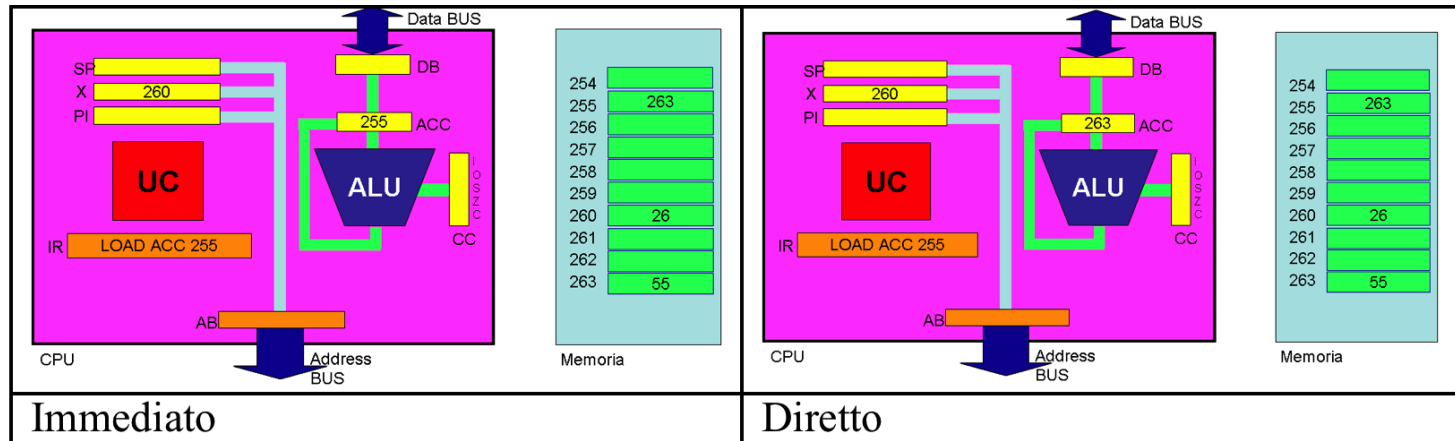
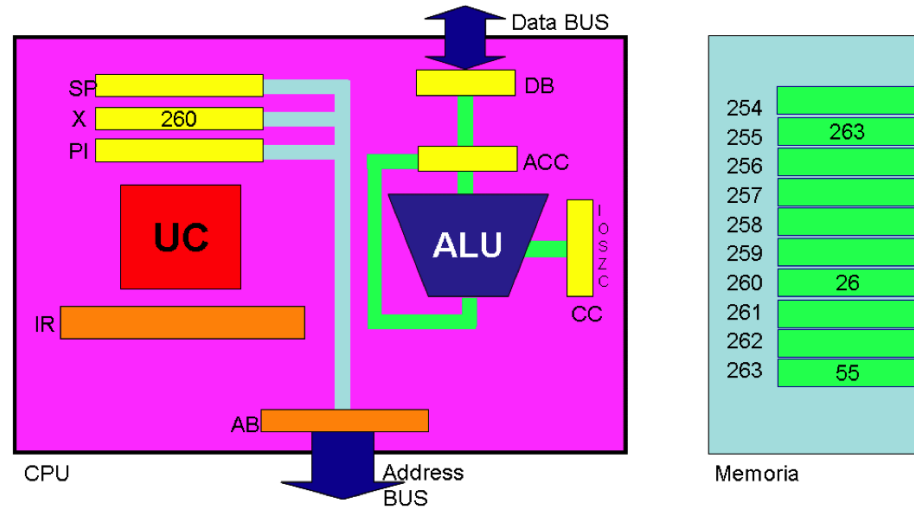
- Nella costruzione dell'indirizzo di un operando di una istruzione le tecniche di indirizzamento più diffuse sono:
 - indirizzamento immediato che indica che il valore è contenuto già nell'istruzione;
 - indirizzamento diretto con il quale viene riportato nell'istruzione l'indirizzo del registro di memoria che contiene il valore o nel quale depositare il valore;
 - indirizzamento indiretto che riporta nell'istruzione l'indirizzo del registro di memoria al cui interno è specificato l'indirizzo del registro dal quale prelevare un valore o nel quale depositare un valore;
 - indirizzamento relativo con il quale l'indirizzo del registro di memoria che contiene il valore o nel quale depositare il valore è specificato nel registro interno del processore detto indice.

Modi di indirizzamento (2/2)

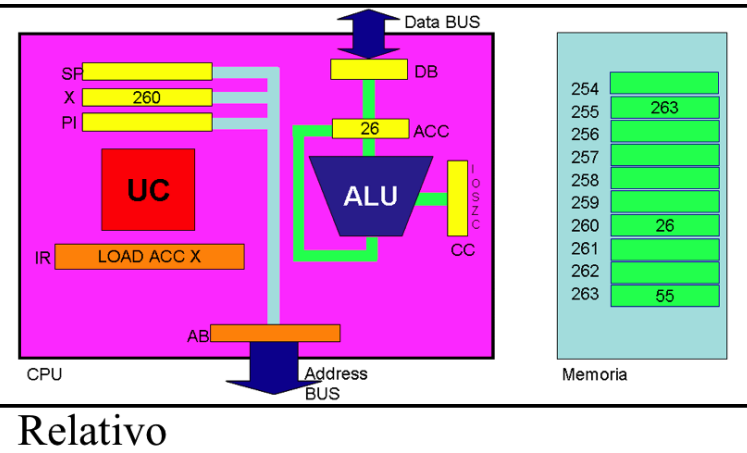
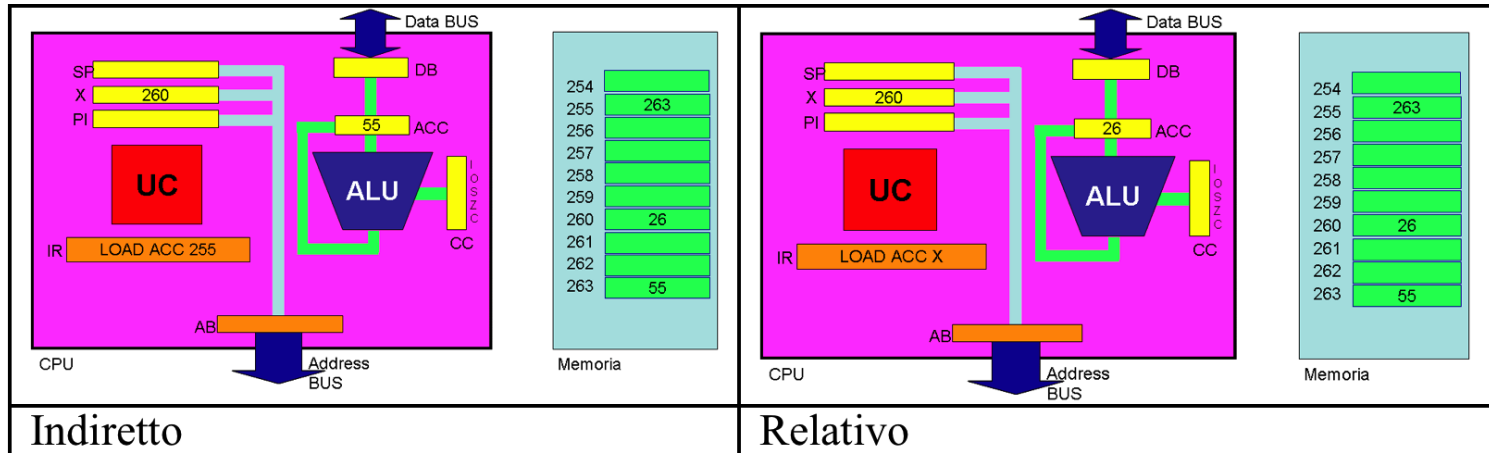
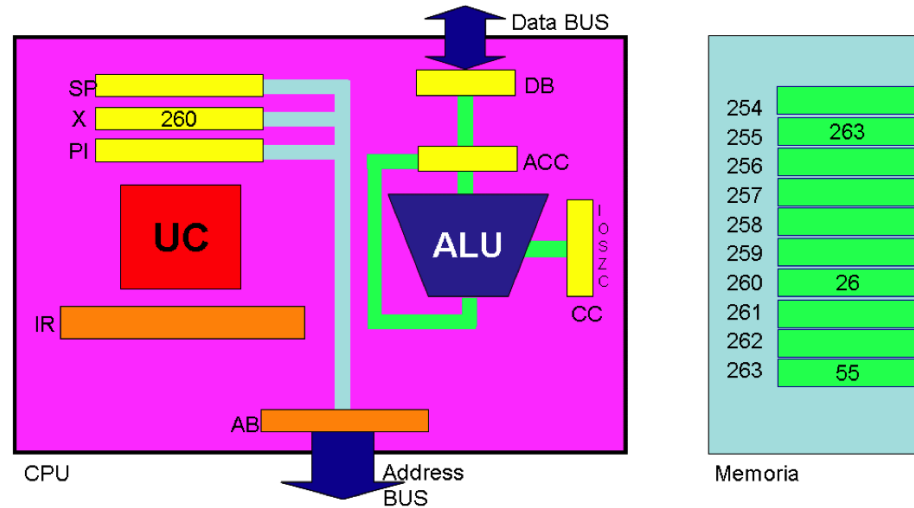
- Le diverse tecniche di indirizzamento vengono indicate nell'istruzione o diversificando il codice operativo o aggiungendo dei bit appositi il cui valore indica alla UC come costruire l'indirizzo.
 - Ad esempio per la semplice istruzione per il caricamento dell'accumulatore si potrebbero avere in linguaggio macchina (rappresentato in esadecimale) le quattro istruzioni di tabella:

Codice Operativo	Operando	Tecnica	Commento	Accessi in memoria nella fase Operand Assembly
60	00ff	Immediata	LOAD ACC con il valore 255	nessuno in quanto il dato è prelevato nella fase fetch con l'istruzione
61	00ff	Diretta	LOAD ACC con il contenuto del registro di memoria di indirizzo 255	un solo accesso in memoria
62	00ff	Indiretta	LOAD ACC con il contenuto del registro di memoria il cui indirizzo è contenuto nel registro di indirizzo 255	due accessi in memoria: il primo per prelevare l'indirizzo alla posizione indicata; il secondo per prelevare il dato
63		Relativa	LOAD ACC con il contenuto del registro di memoria il cui indirizzo è presente nel registro indice X	un solo accesso in memoria

Modi di indirizzamento: esempi 1/2



Modi di indirizzamento: esempi 2/2

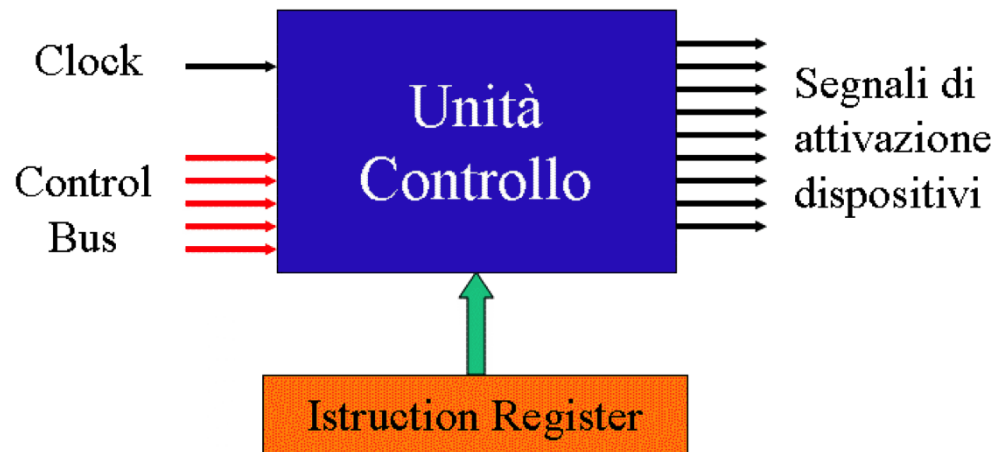


Repertorio Istruzioni (ISA)

- istruzione di lettura e modifica dei registri interni e di memoria;
- istruzioni di tipo aritmetico;
- istruzioni di tipo logico;
- istruzioni di salto;
- istruzioni di input ed output per la gestione della interazione con il mondo esterno.

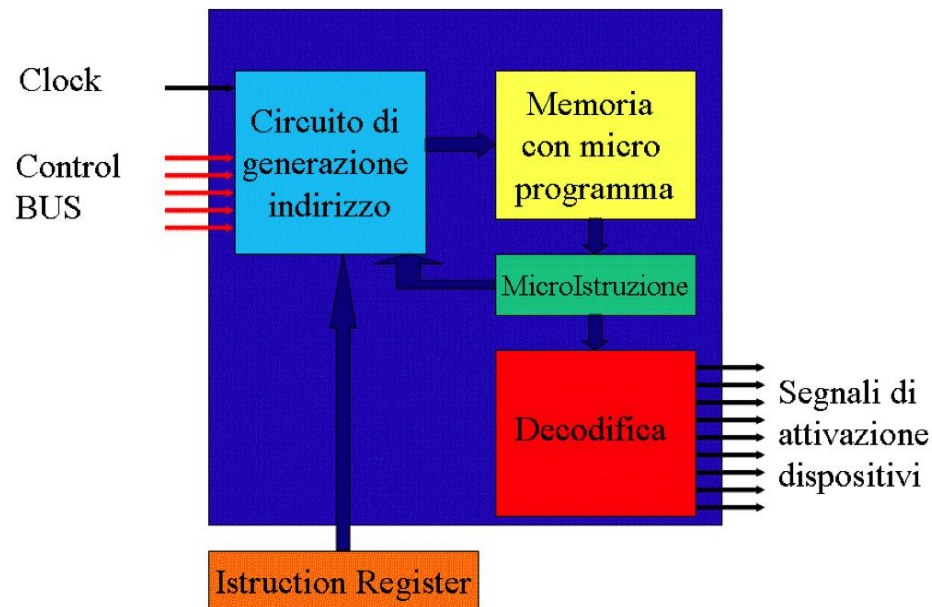
Logica cablata

- L'esecuzione di una istruzione da parte della UC consiste *nell'inoltro di una sequenza di abilitazioni* dei dispositivi il cui effetto corrisponde alla operazione richiesta.
- Le prime UC erano realizzate con circuiti, detti a *logica cablata*, che evolvevano in tanti modi diversi quante erano le istruzioni che essa era in grado di svolgere.



Logica Microprogrammata

- Nei microprocessori ad ogni istruzione corrisponde una sequenza di microistruzioni conservate in una memoria interna alla UC.
 - La sequenza di microistruzioni ha il compito di generare le abilitazioni necessarie alla attuazione della istruzione.
 - A tal fine un circuito interno alla UC provvede alla generazione di indirizzi per individuare una dopo l'altra le microistruzioni che un decodificatore trasforma in segnali di abilitazione.
 - L'istruzione nel registro IR determina la posizione della prima microistruzione.



Processori RISC e CISC

- CISC (Complex Instruction Set Computer)
 - inserimento nel linguaggio macchina di istruzioni con potenza espressiva prossima a quella dei linguaggi di programmazione di alto livello
 - sono caratterizzati quindi da un ampio repertorio di istruzioni
 - sviluppo di programmi più semplice
 - anche se molte di esse non risultano strettamente necessarie, potendosi ottenere con l'esecuzione di sequenze di istruzioni più semplici
 - maggiore complessità costruttiva
- RISC (Reduced Instruction Set Computer).
 - repertorio costituito da un ridotto ed essenziale insieme di istruzioni al fine di ottenere processori più veloci e di costo ridotto, data la minore complessità del loro progetto.
 - L'obiettivo fondamentale dell'approccio RISC è di disporre di un insieme fondamentale di istruzioni per ridurre al minimo il numero dei cicli di macchina (clock) necessari per loro esecuzione.
 - Tutte le istruzioni RISC fondamentali hanno la stessa durata (un ciclo macchina), la stessa lunghezza e lo stesso format

I Microprocessori

- Sono dispositivi elettronici in grado di contenere all'interno di un unico circuito integrato le funzioni di un'intera CPU.
- Il microprocessore interagisce con tutti gli altri dispositivi attraverso i collegamenti dei bus di dati (data bus), di indirizzo (address bus) e di controllo (control bus).

Bit Data BUS	Bit Address BUS	Capacità di indirizzamento
8	16	64 KByte
16	20-24	1-16 MByte
64	64	fino a circa 10^{19} Byte

Classificazione

- sulla base dei seguenti parametri:
 - **parallelismo** esterno espresso come numero di bit trasferiti o prelevati in un singolo accesso in memoria (8, 16, 32, 64,...) e caratterizzanti quindi il suo data bus;
 - **capacità di indirizzamento** legata alla dimensione in bit del suo address bus
 - numero, tipo e parallelismo dei registri interni;
 - **tecniche di indirizzamento** intese come la modalità con la quale costruire **l'indirizzo logico con il quale prelevare o salvare il valore dell'operando** di una istruzione;
 - gestione delle periferiche di **input ed output**;
 - **repertorio delle istruzioni** inteso come numero e tipo di istruzioni costituenti il suo linguaggio macchina;

Gestione Input Output

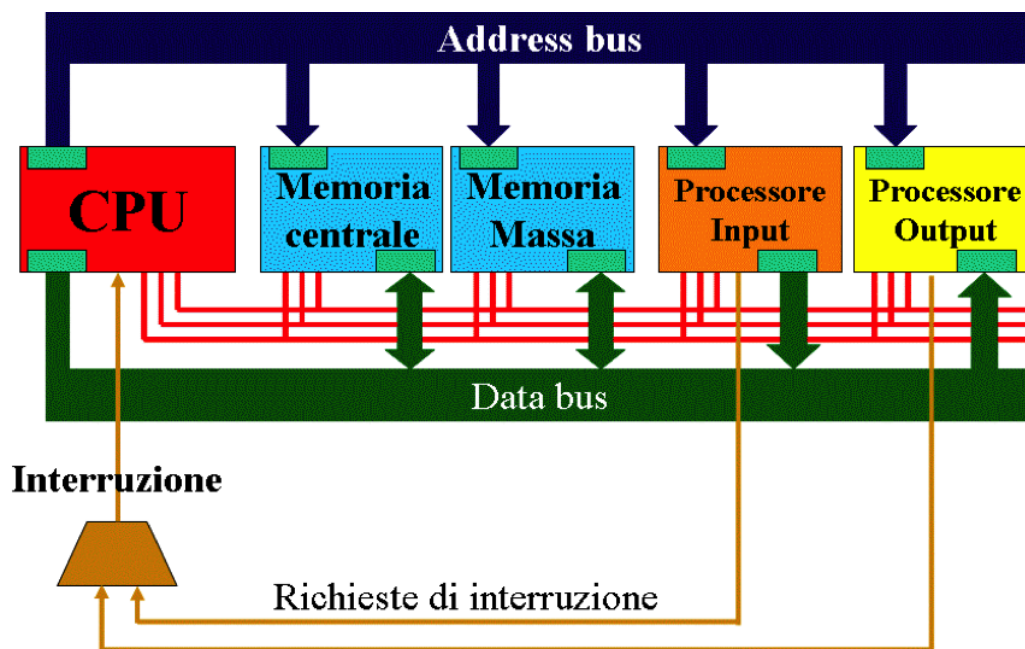
- Tecnica *memory-mapped*
 - L'UC usa le stesse istruzioni utilizzate per leggere e scrivere in memoria anche per accedere ai dispositivi di I/O.
 - I dispositivi di I/O hanno quindi dei propri indirizzi che devono essere riservati e non sovrapposti a quelli usati per la memoria. I dispositivi di I/O controllano il bus indirizzi e rispondono solo quando riconoscono un indirizzo a loro assegnato
 - Il vantaggio dell'uso del memory-mapped è che, non richiedendo da una parte hardware aggiuntivo per la gestione della periferia e dall'altra un insieme di istruzioni specifiche, consente la realizzazione di CPU con una complessità inferiore, più economiche, veloci e facili da costruire.
- Tecnica *I/O-mapped*.
 - Vengono invece usate istruzioni specifiche per l'esecuzione dell'input/output.
 - I dispositivi di I/O hanno uno spazio indirizzi separato da quello della memoria, e un segnale del control bus serve alla UC per specificare se si tratta di un accesso alla memoria o ad un dispositivo periferico.

Evoluzione del modello di Von Neumann: interruzioni

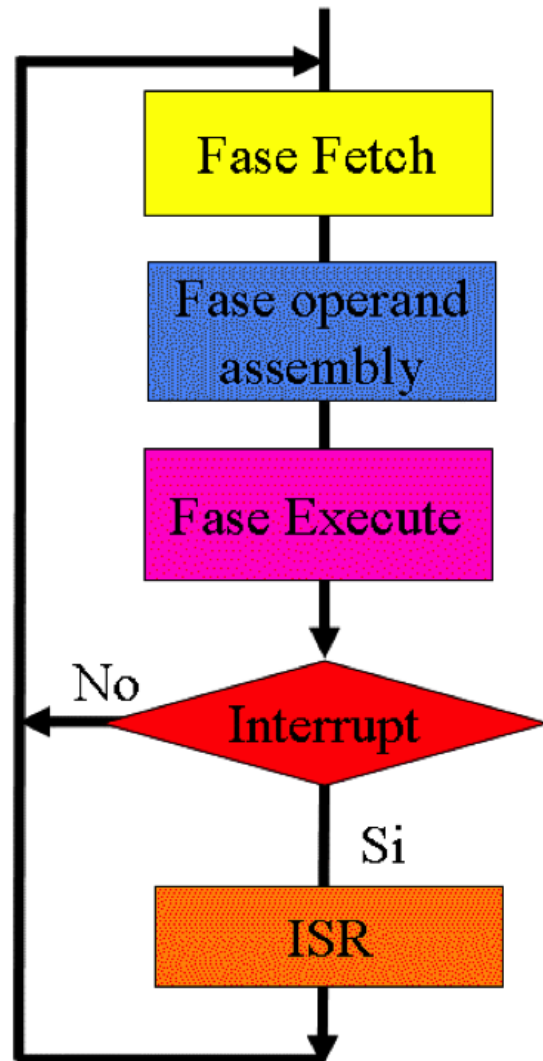
- Nel modello di Von Neumann, non era possibile sovrapporre i tempi delle operazioni di input con quelli dell'output.
- *Introduzione di sistemi dedicati (canali)* il cui compito è scaricare la CPU della gestione di attività specifiche.
 - I canali, con la loro autonomia possono lavorare anche contemporaneamente con la CPU.
 - Es. canali di input ed output, processori dedicati alla grafica, alle operazioni sui numeri reali, alla acquisizione di segnali analogici.
- Per rendere indipendenti i processori dedicati è stato introdotto nell'architettura hardware un segnale detto delle interruzioni mediante il quale una qualsiasi entità esterna alla CPU può richiederle attenzione.

Le interrupts

- Con la presenza del *segnale di interruzione* la CPU può attivare un processore periferico e disinteressarsi delle sue attività.
 - Quando un processore dedicato termina il suo compito, avanza una richiesta di interruzione al processore centrale e aspetta che gli venga rivolta attenzione.
 - Mentre i processori periferici lavorano, la CPU può lavorare anch'essa a meno che non sia indispensabile quanto richiesto allo specifico processore



ISR



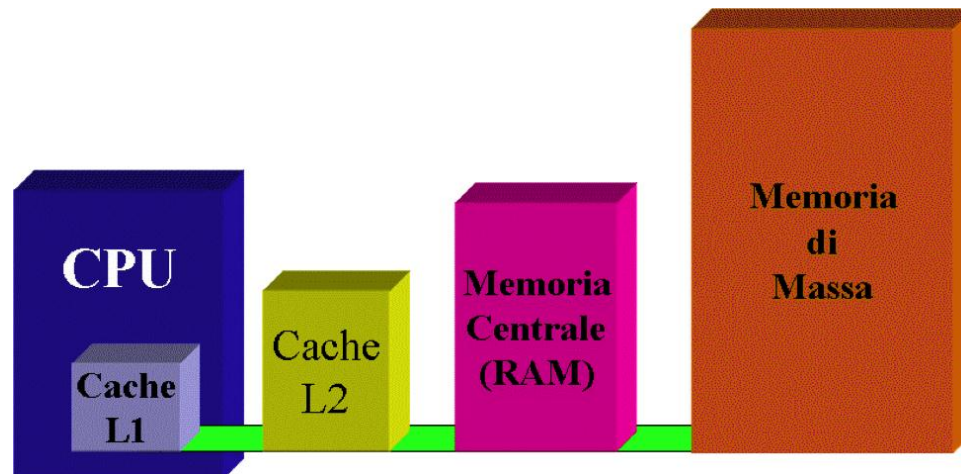
- Per consentire alla UC di accorgersi del verificarsi di una interruzione il registro di condizione CC è stato dotato di un bit che diventa uguale ad uno quando arriva una interruzione.
- La UC controlla il bit al termine delle esecuzione di ogni istruzione:
 - se è uguale zero procede normalmente con il prelievo dell'istruzione successiva;
 - in caso contrario comincia l'esecuzione di un programma del sistema operativo, detto ISR (interrupt service routine) che ha come compito primario di capire la causa della interruzione, ossia quale dispositivo ha avanzato la richiesta.
 - Nel caso si accorga della presenza di più richieste stabilisce quale servire per prima secondo criteri di importanza o priorità di intervento.

Evoluzione del modello di Von Neumann: le Cache

- Per ridurre i tempi di trasferimento dalla memoria centrale ai registri interni della CPU, viene replicata una porzione di memoria e posta tra memoria e CPU stessa. Tale memoria, molto veloce, viene chiamata *cache* e fa da buffer per il prelievo di informazioni dalla memoria centrale.
 - Con operazioni particolari istruzioni e dati vengono trasferiti dalla memoria centrale nella cache secondo la capacità di quest'ultima.
 - La UC procede nelle tre fasi del suo ciclo al prelievo di istruzioni e operandi dalla cache.
 - Quando la UC si accorge che il prelievo non può avvenire scatta un nuovo travaso dalla memoria centrale.
 - Se la cache è interna alla CPU viene detta di primo livello (L1);
 - Le cache di secondo livello (L2) sono invece esterne e solitamente un pò più lente di quelle di primo livello ma sempre più veloci della memoria centrale.
 - la cache L2 risulta 4 o 5 volte più lenta della cache L1 mentre la RAM lo è addirittura 20 o 30 volte. I due livelli possono coesistere.

Gerarchie di Memorie

- Consente di offrire ai programmi l'illusione di avere una memoria grande e veloce.
 - Nella gerarchia i livelli più prossimi alla CPU sono anche quelli più veloci, ma sono anche quelli con dimensioni più piccole visto il loro elevato costo.
 - I livelli più lontani sono quelli che mostrano una capacità massima ed anche tempi di accesso maggiori.



Linguaggi di programmazione



Linguaggio macchina e linguaggio assembly

- Una **istruzione macchina** è una sequenza di 0 e 1 suddivisa in campi, diversi nel formato e nel significato a seconda dell'istruzione

001000	10110	10101	1111111111001110
--------	-------	-------	------------------

- Nel riferirsi alle istruzioni macchina si preferisce la rappresentazione simbolica a quella numerica, costituita dal **linguaggio assembly**

```
addi    $21, $22, -50
```

Assemblatore

- L'assemblatore è un programma che esegue la traduzione di un programma, scritto in linguaggio assembler, in linguaggio macchina.
- E' il più semplice traduttore di linguaggi presente in informatica
 - fa corrispondere ai codici mnemonici del codice operativo il rispettivo codice binario;
 - converte da decimale a binario indirizzi e valori dei dati;
 - determina gli indirizzi delle etichette associate alle istruzioni;
 - converte dati alfanumerici nella loro rappresentazione binaria.

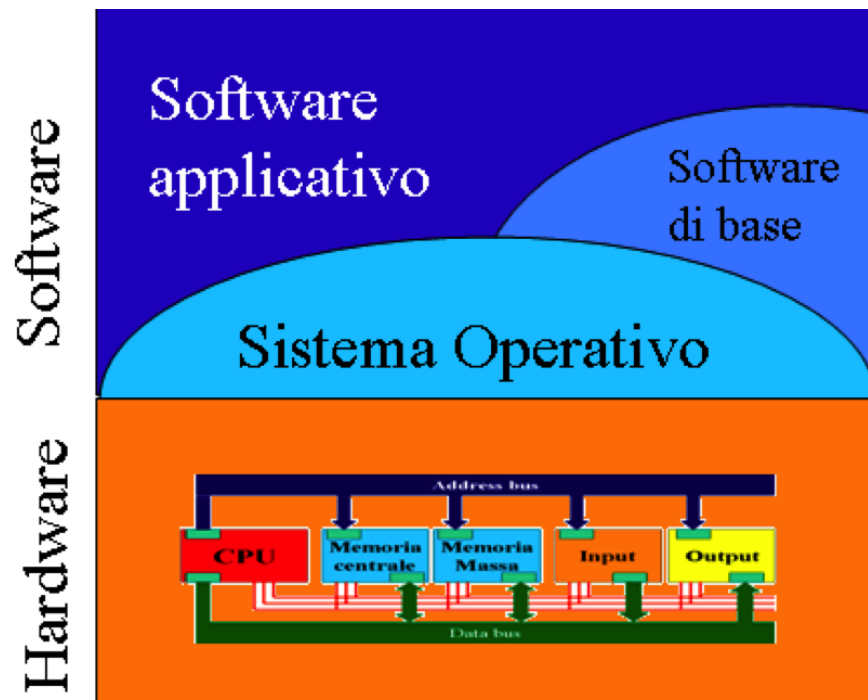
Firmware e Software

- L'insieme dei micro programmi composti dalle microistruzioni memorizzate nella memoria interna alla UC prende il nome di *firmware*
- L'insieme di tutte le applicazioni del computer, quindi di tutti i programmi per computer, prende il nome di *software*.
 - In una accezione più ampia il termine software può essere inteso come tutto quanto può essere preteso dall'hardware: basta infatti inserire in memoria un programma diverso perché il sistema cambi le sue attività.
 - Tra tutte le macchine automatiche il computer è un sistema polifunzionale in quanto può eseguire infinite funzioni sempre che venga progettato un programma per ogni applicazione.



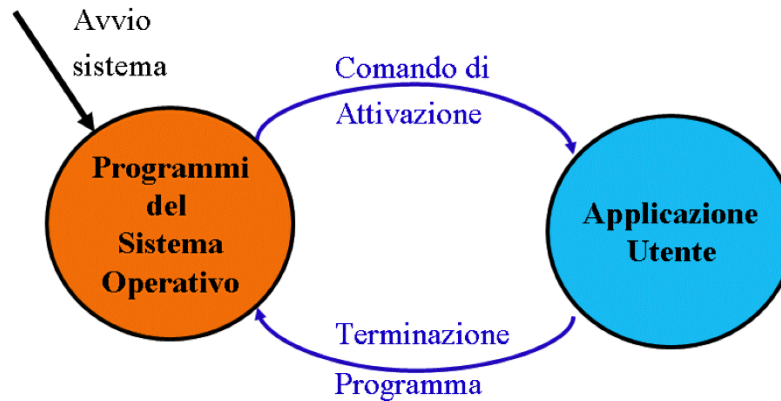
Hardware vs. Software

- programmi che servono a tutti gli utenti del sistema
 - classificati come *software di base*
 - sistemi operativi e traduttori dei linguaggi di programmazione
- programmi che risolvono problemi specifici
 - *software applicativo*.



Sistema Operativo

- Il sistema operativo è un insieme di programmi che deve garantire la gestione delle risorse hardware in modo semplice ed efficiente a tutti gli utenti del sistema
 - ... persone oppure altre applicazioni.
 - I primi calcolatori non avevano il sistema operativo.
 - In essi il programmatore doveva prevedere tutto
 - Al termine dell'esecuzione del programma il programmatore o l'operatore del sistema doveva provvedere ad un nuovo caricamento in memoria ed ad una successiva attivazione.
 - Con il sistema operativo il passaggio da una applicazione ad un'altra è svolto in automatico
 - La CPU si trova così ad eseguire i programmi del sistema operativo in alternanza con quelli applicativi.



Middleware

- E' il software che fornisce *un'astrazione di programmazione* che maschera l'eterogeneità di elementi sottostanti
 - reti, hardware, sistemi operativi, linguaggi di programmazione
- Il middleware definisce una macchina generalizzata fissandone modalità di interazione con le applicazioni.

