ARCHITETTURA DI UN SISTEMA A MICROPROCESSORE

1. INTRODUZIONE

In questo capitolo viene presentata la struttura, sia interna che esterna, di un microprocessore generico riprendendo i concetti esposti nella prima presentazione e fornendo gli approfondimenti necessari per comprendere i meccanismi che stanno alla base del funzionamento di tale dispositivo.

Ricordiamo che per microprocessore si intende un componente a larghissima scala di integrazione in grado di svolgere le funzioni tipiche di una CPU all'interno di un sistema di elaborazione. Esso infatti è in grado di eseguire operazioni aritmetiche, logiche, di rotazione e di scorrimento.

1. INTRODUZIONE

- **I vantaggi dei sistemi basati sui microprocessori rispetto a quelli a logica cablata sono i seguenti:**

  basso costo dovuto all'elevato numero di chip in grado di sviluppare funzioni generali (general purpose) che possono essere prodotti e che possono essere impiegati per le applicazioni più disparate

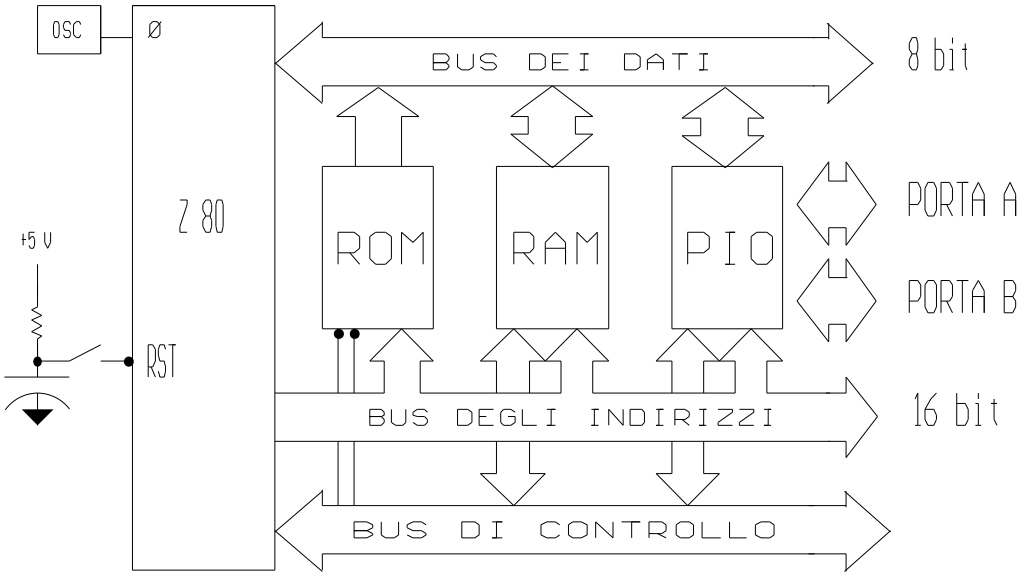
  basso numero di componenti ed elevata affidabilità

  elevata versatilità

  Per contro esiste una diminuzione della velocità di risposta rispetto alla logica cablata dovuta alla necessità di dover eseguire delle istruzioni prima di generare i segnali di uscita

2. SCHEMA A BLOCCHI Di UN SISTEMA A MICROPROCESSORE

- Un sistema a microprocessore può essere rappresentato facendo ricorso allo schema a blocchi standard riportato in fig.
- In esso sono presenti gli elementi tipici che caratterizzano il sistema e che ne permettono il corretto funzionamento



2. SCHEMA A BLOCCHI Di UN SISTEMA A MICROPROCESSORE

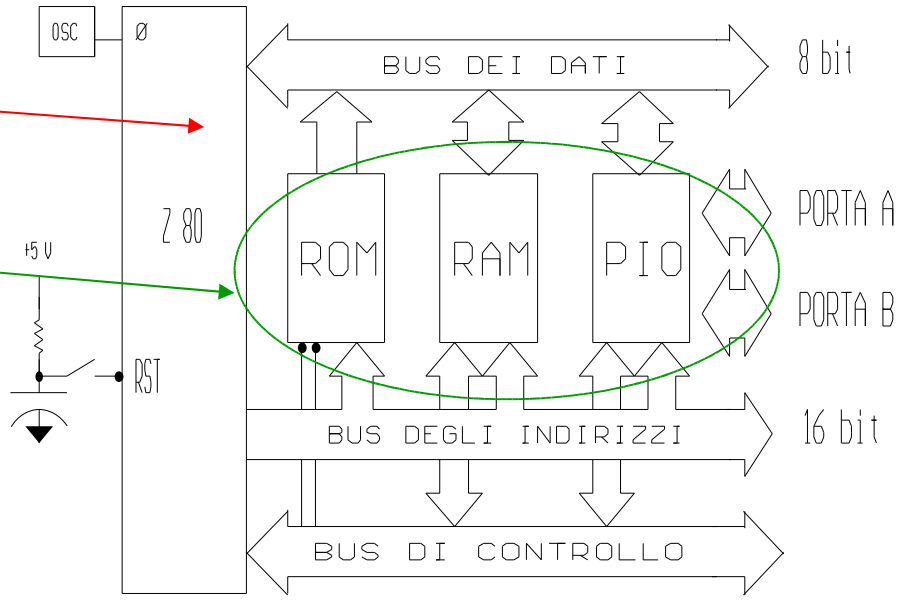
Si possono distinguere:

microprocessore (CPU): in esso sono compresi la logica di controllo, l'unità logico aritmetica e i registri interni

memoria: ancora suddivisibile in:

memoria RAM: memoria ad accesso casuale impiegata per la memorizzazione dei programmi utente

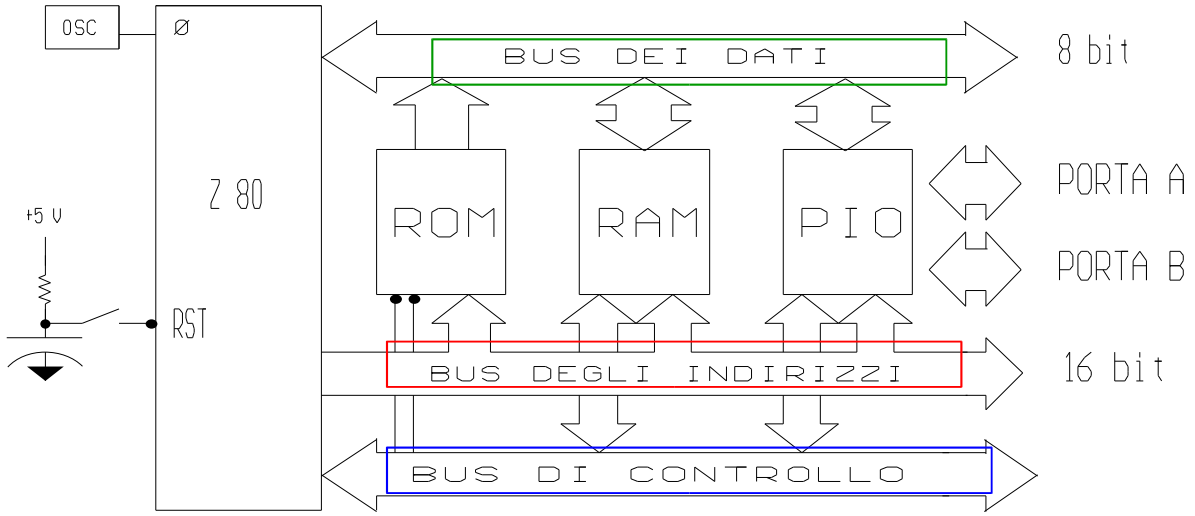
memoria ROM: memoria a sola lettura che contiene i programmi che costituiscono il sistema operativo necessario per l'avviamento e la gestione del sistema



dispositivi di ingresso-uscita (I10): consentono di immettere dall'esterno i dati da elaborare e di fornire in uscita i risultati dell'elaborazione.

2. SCHEMA A BLOCCHI DI UN SISTEMA A MICROPROCESSORE

In fig. oltre agli elementi già citati, è possibile notare la presenza dei tre canali di comunicazione costituiti dal bus dati, dal bus indirizzi e dal bus controlli



2. SCHEMA A BLOCCHI Di UN SISTEMA A MICROPROCESSORE

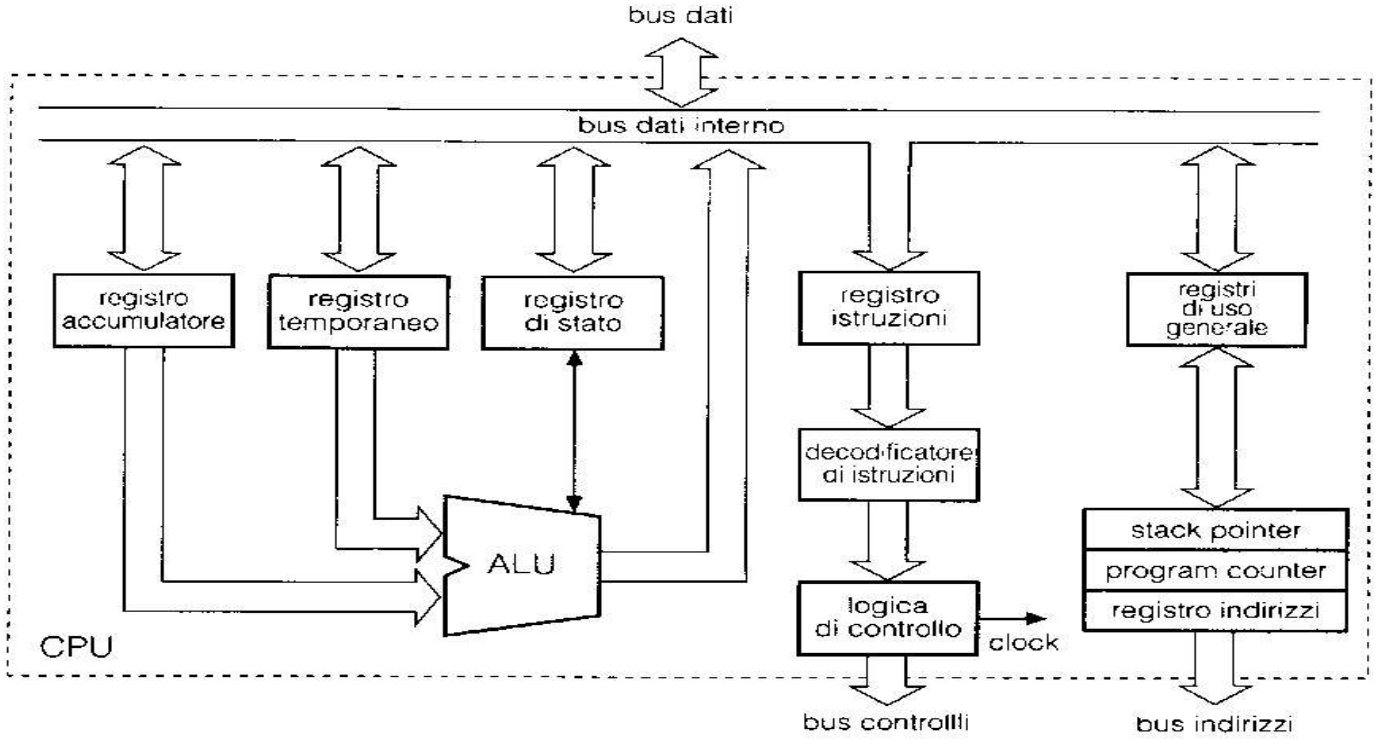
Bus Dati: consente lo scambio di dati tra i vari dispositivi che compongono il sistema, è bidirezionale e la sua dimensione è pari al numero di bit che caratterizza la parola del microprocessore.

Bus indirizzi: rappresenta il canale attraverso il quale viene fornito da parte della CPU l'indirizzo della locazione di memoria o del dispositivo di 1/0 interessato dalla trasmissione. t unidirezionale ed essendo l'indirizzo costituito da almeno 2 byte sono necessari come rnininio 16 fili

Bus controlli: consente di trasmettere l'insieme dei segnali necessari alla corretta gestione dell'intero sistema. In particolare permette la sincronizzazione dei vari dispositivi componenti Il bus controlli ha un numero di linee che differisce a seconda del tipo di microprocessore utilizzato ed è dimensionato in funzione dei controlli che si intendono effettuare.

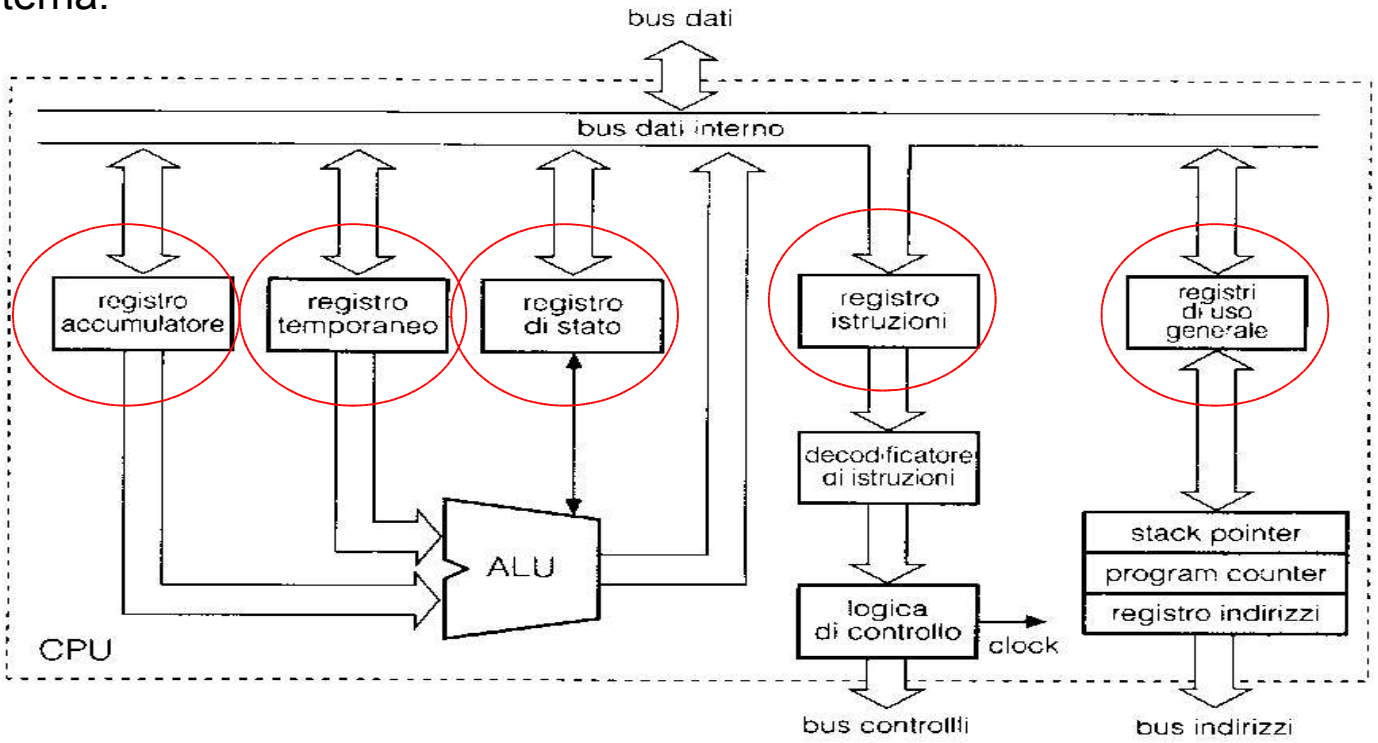
3. STRUTTURA INTERNA DI UN MICROPROCESSORE

- Per descrivere la struttura interna di un microprocessore è utile fare riferimento ad uno schema generico nel quale si possono individuare alcuni tra gli elementi più tipici, quali ad esempio: registri di uso speciale e generale, unità aritmetico-logica ^(ALU) e logica di controllo (fig.).

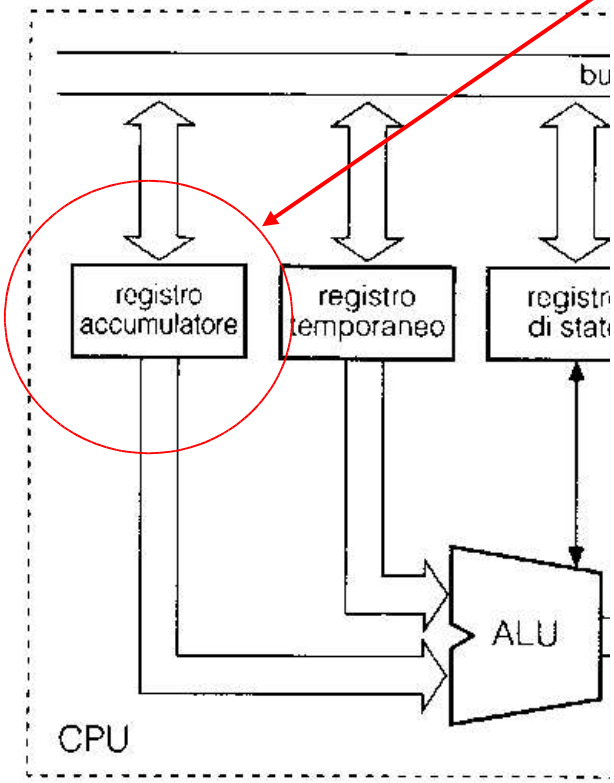


3. STRUTTURA INTERNA DI UN MICROPROCESSORE

I registri: sono particolari dispositivi di memoria, di capacità ridotta (uno o due byte), posti all'interno del microprocessore, con i quali si possono manipolare temporaneamente dati e indirizzi. Il loro utilizzo permette di semplificare le operazioni e renderle più veloci, evitando l'intervento continuo della memoria esterna.



3.1 Registri di uso Speciale



a) Accumulatore (registro di uso generale)

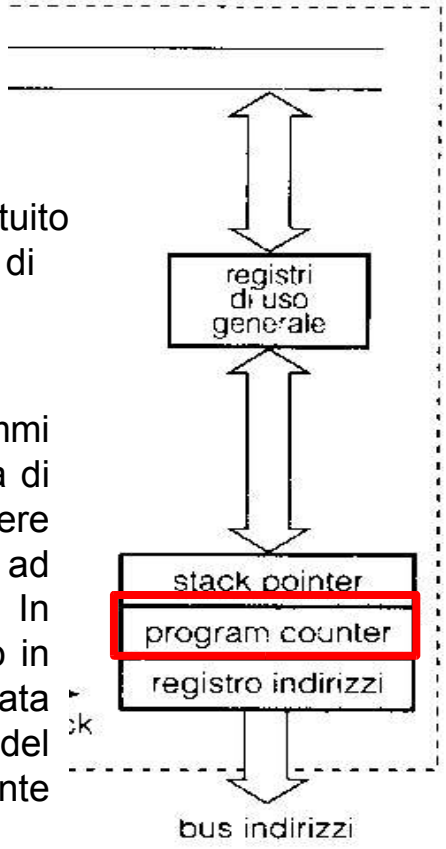
E' senza dubbio uno dei registri più utilizzati in quanto in esso vengono memorizzati i risultati delle varie operazioni effettuate dalla ALU ed è interessato anche dalle operazioni di ricezione e di invio dei dati. L'accumulatore è quindi un registro dotato di ampie capacità di comunicazione con i vari blocchi che costituiscono il sistema e le sue dimensioni generalmente corrispondono alla lunghezza della parola trattata.

In alcuni microprocessori sono presenti più registri di questo tipo e quindi sono possibili sia operazioni tra gli accumulatori che con altri dispositivi. Naturalmente in questi casi risulta più complessa la logica di controllo ed è quindi necessario poter disporre di un set di istruzioni più completo per gestire il sistema; ad esempio dovrà essere presente un'istruzione che permetta di individuare l'accumulatore a cui deve essere inviato il risultato di un'operazione svolta dalla ALU.

3.1 Registri di uso Speciale

b) Contatore di programma (Program Counter: PC)

La funzione di questo registro è quella di scandire la sequenzialità delle varie operazioni. Infatti esso memorizza l'indirizzo della locazione di memoria in cui è contenuta l'istruzione che deve essere eseguita e per questo motivo le sue dimensioni devono essere pari a quelle del bus indirizzi.



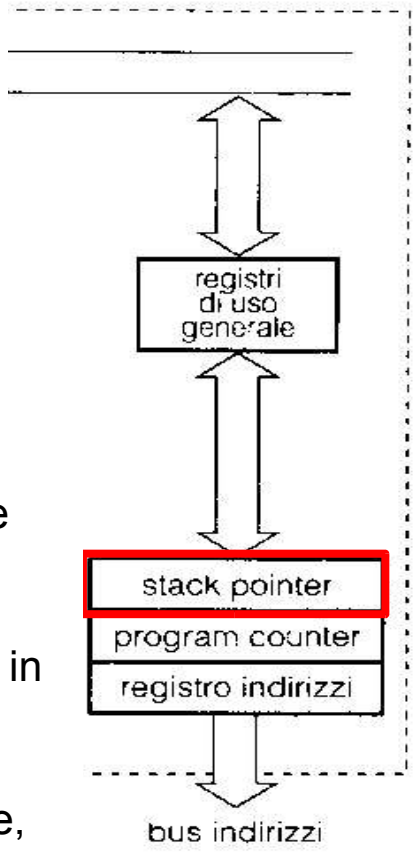
Durante la fase di prelievo il contenuto del contatore di programma è costituito dall'indirizzo dell'istruzione da eseguire e viene aggiornato durante la fase di esecuzione, predisponendosi in questo modo ad indirizzare la cella di memoria che contiene l'istruzione successiva.

Nel caso in cui nel programma siano presenti dei salti a sottoprogrammi oppure venga presentata da parte di un dispositivo esterno una richiesta di interruzione la sequenza operativa precedentemente descritta deve essere parzialmente modificata in quanto il contatore di programma oltre ad indirizzare la locazione prevista deve anche coinvolgere altri dispositivi. In particolare viene interessato il registro di Stack Pointer, di cui parleremo in seguito, che consente di indirizzare un'opportuna area di memoria utilizzata per memorizzare il contenuto di PC. Al termine dell'esecuzione del sottoprogramma viene posto in PC l'indirizzo della cella immediatamente successiva a quella contenente l'ultima istruzione prima del salto.

3.1 Registri di uso Speciale

e) Puntatore di stack (Stack Pointer: SP)

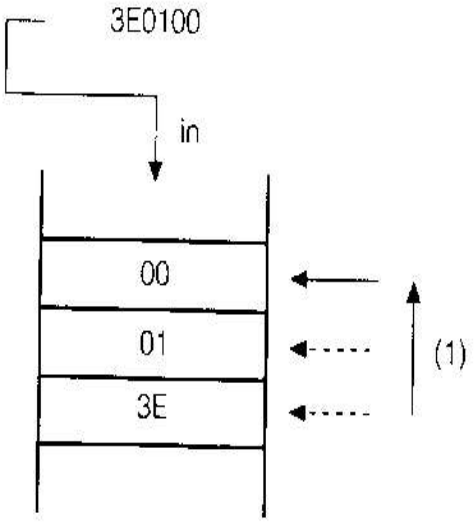
Questo registro, presente in quasi tutti i tipi di microprocessore, viene utilizzato per indirizzare, secondo le istruzioni del programma, una opportuna zona della memoria ad accesso casuale (RAM), denominata area di stack. Tale zona di memoria è organizzata come un registro LIFO (Last Input First Output), cioè i dati vengono prelevati a partire dal dato introdotto per ultimo.

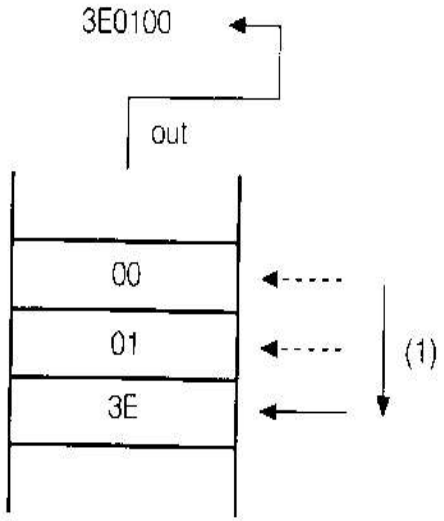


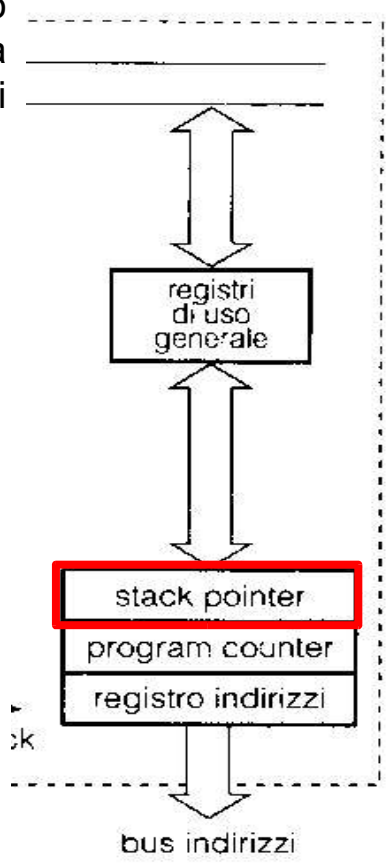
L'utilità del registro puntatore di stack si manifesta in particolare nel caso in cui si debbano utilizzare dei sottoprogrammi o gestire delle interruzioni. In questa eventualità infatti il PC deve puntare alla locazione in cui è memorizzata la routine di gestione dell'interruzione per cui si avrebbe la perdita del suo contenuto attuale e quindi l'impossibilità di un ritorno al programma principale. Si può eliminare tale inconveniente utilizzando, per memorizzare l'indirizzo contenuto in PC, l'area di stack, che deve essere definita a priori dal programmatore. Infatti il puntatore di stack, durante l'esecuzione dell'ultima istruzione del sottoprogramma di gestione dell'interruzione, individua tale indirizzo permettendo in questo modo il trasferimento del suo contenuto in PC.

3.1 Registri di uso Speciale

Ad esempio in fig. è riportato il modo con cui l'istruzione 3E0100 viene memorizzata e successivamente prelevata dall'area di stack. Come si può facilmente notare il byte meno significativo (00), che nella fase di scrittura viene introdotto per ultimo, è invece prelevato per primo durante la fase di lettura.



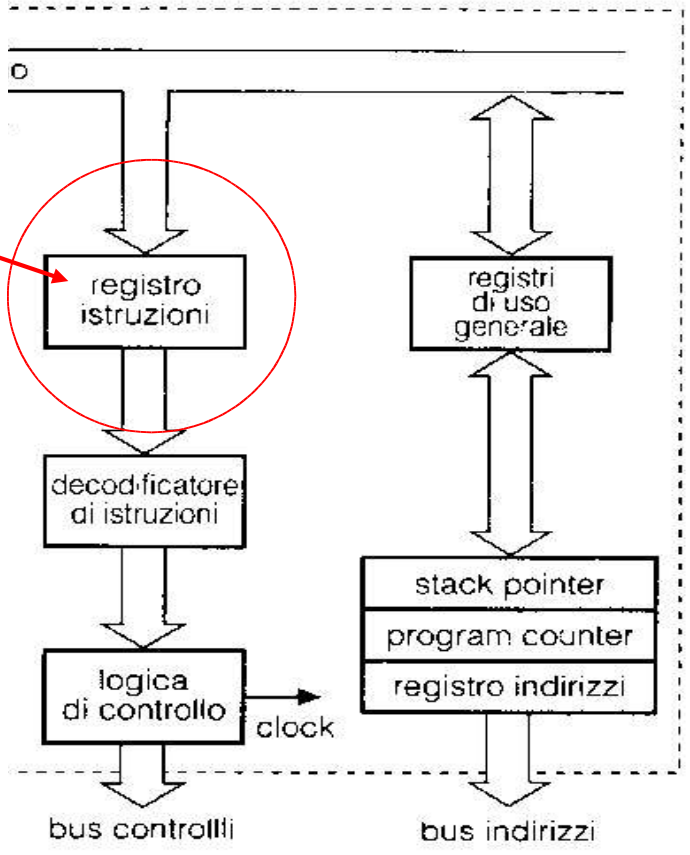




3.1 Registri di uso Speciale

d) Registro istruzioni (Instruction Register: IR)

- La funzione di questo registro è quello di memorizzare l'istruzione che deve essere eseguita; tale istruzione è espressa in forma codificata per cui la sua esecuzione può avvenire solo dopo la sua traduzione in linguaggio macchina. Questa operazione avviene in un decodificatore, collegato direttamente al registro istruzioni, che a sua volta attiva la logica di controllo.
- In conclusione quindi il registro istruzioni, collegato unidirezionalmente al circuito di decodifica, mantiene memorizzata l'istruzione per il tempo necessario alla sua esecuzione. Le dimensioni di questo registro sono solitamente uguali alla lunghezza della parola trattata.

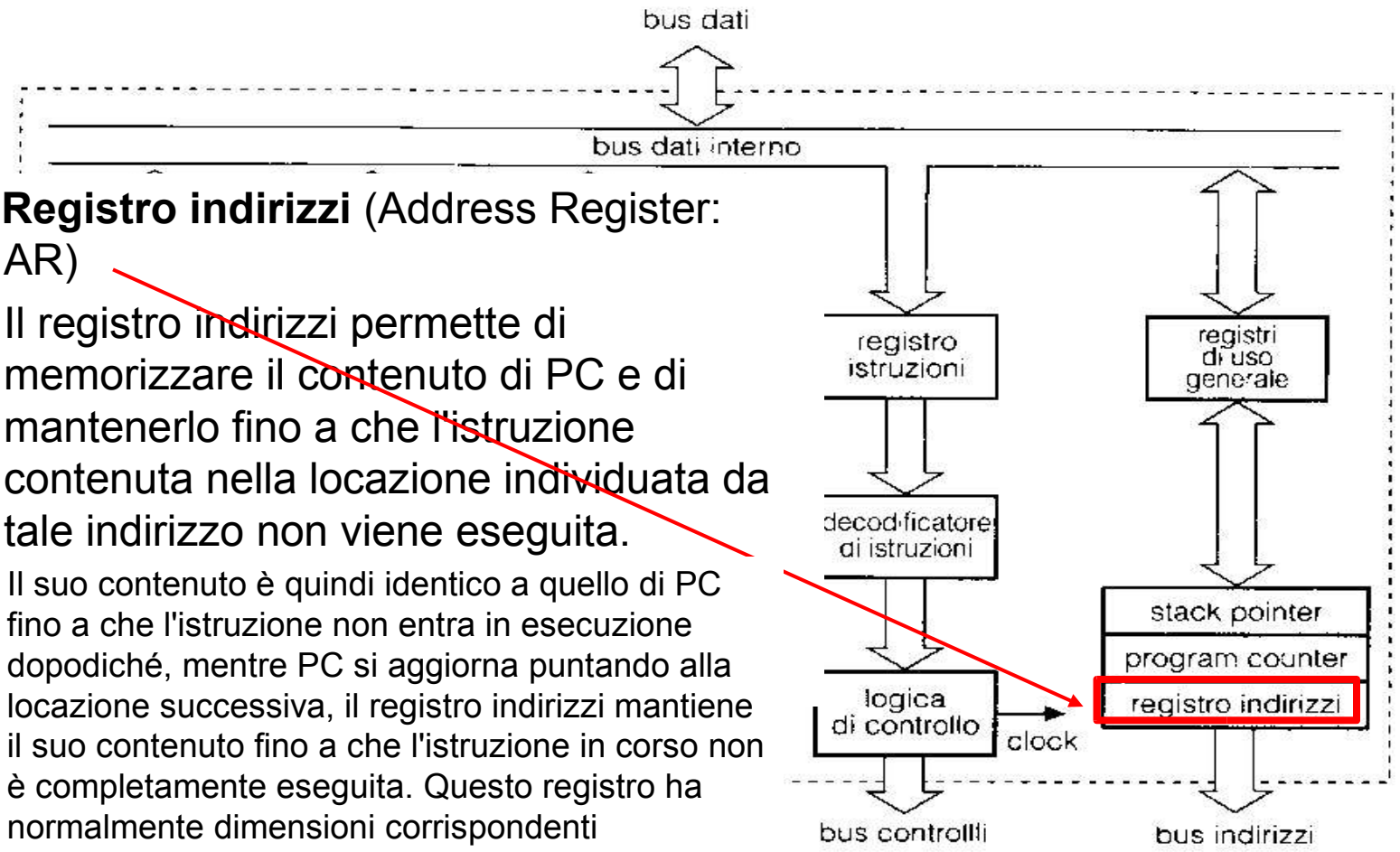


3.1 Registri di uso Speciale

e) Registro indirizzi (Address Register: AR)

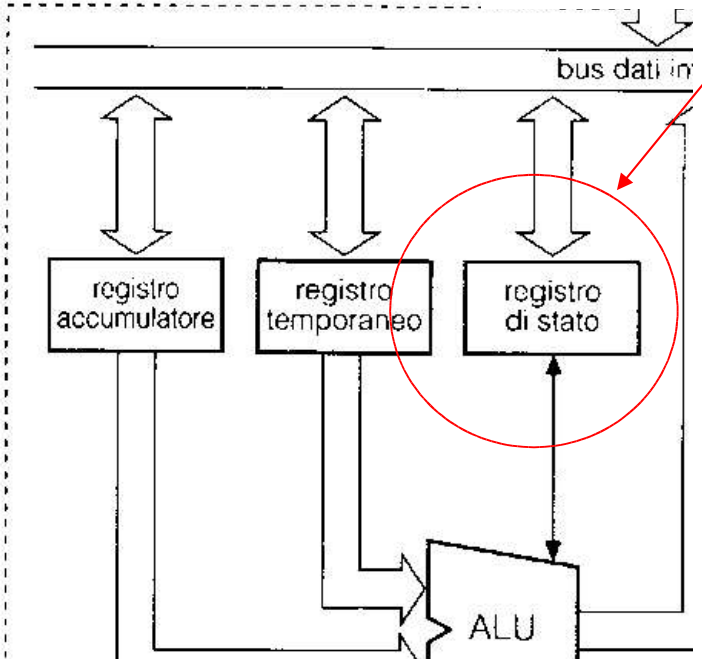
- Il registro indirizzi permette di memorizzare il contenuto di PC e di mantenerlo fino a che l'istruzione contenuta nella locazione individuata da tale indirizzo non viene eseguita.

  Il suo contenuto è quindi identico a quello di PC fino a che l'istruzione non entra in esecuzione dopodiché, mentre PC si aggiorna puntando alla locazione successiva, il registro indirizzi mantiene il suo contenuto fino a che l'istruzione in corso non è completamente eseguita. Questo registro ha normalmente dimensioni corrispondenti all'indirizzo generico di memoria.



3.1 Registri di uso Speciale

f) **Registro di stato** (Flags)



- Il registro di stato può essere scomposto in più celle elementari (flags) che consentono di ricavare una serie di informazioni relative allo stato delle unità interne al microprocessore.
- Infatti il loro contenuto si modifica, passando da " 1 " a " 0 " e viceversa, in conseguenza dell'esecuzione di determinate operazioni permettendo una verifica dei risultati ottenuti. 1 flags possono fornire indicazioni diverse a seconda del tipo di microprocessore utilizzato, ma comunque quelli normalmente presenti sono:

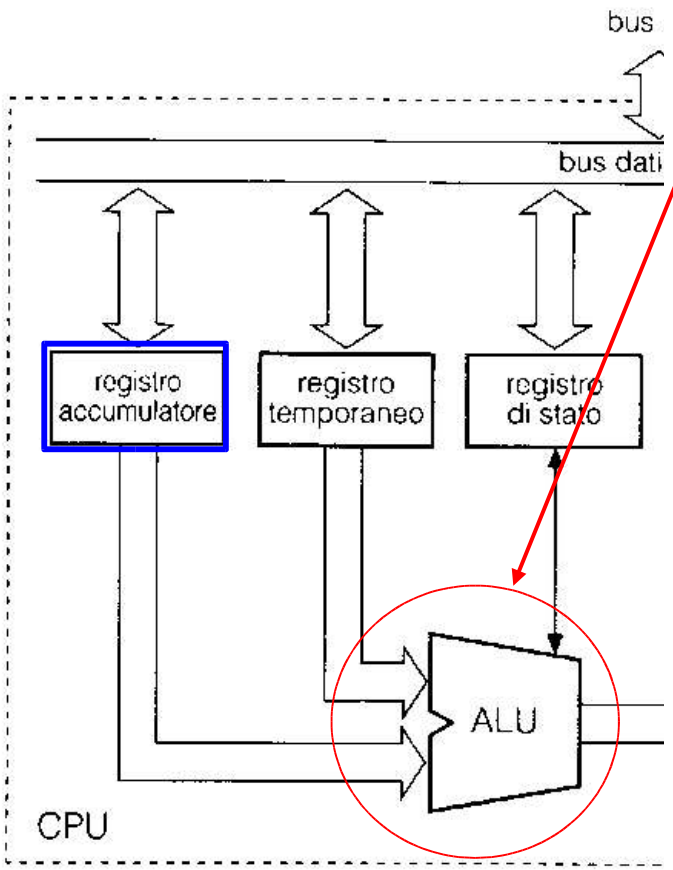
flag C = carry (presenza di riporti)

flag Z = zero (se il risultato è uguale a 0

flag V = overflow (se esiste un traboccamento nell'esecuzione di somme aritmetiche (V)

flag S = segno (se il segno del risultato è positivo)

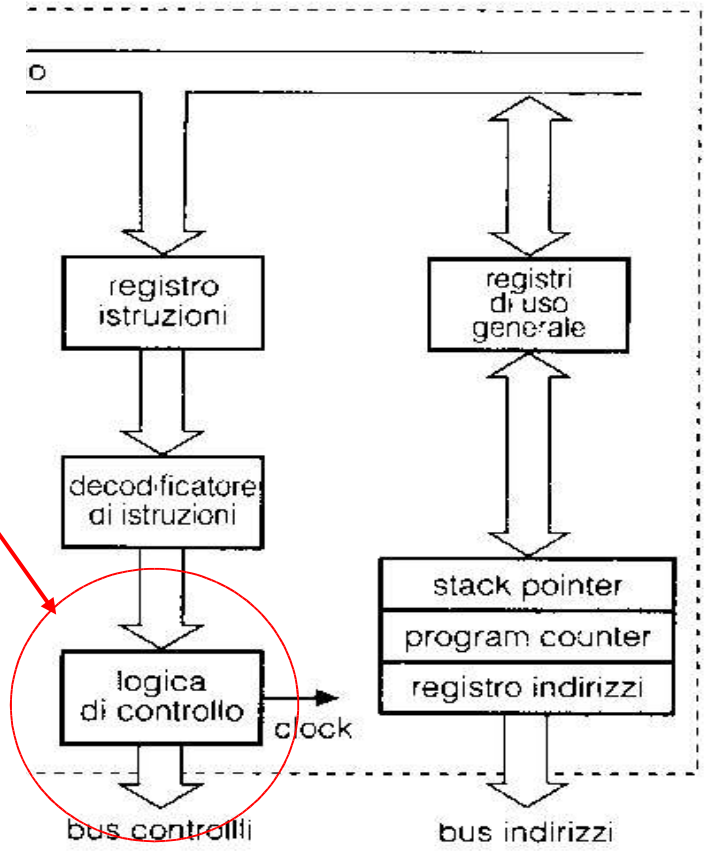
3.2 Unita Logico-aritmetica (Aritmetic Logic Unity: ALU)



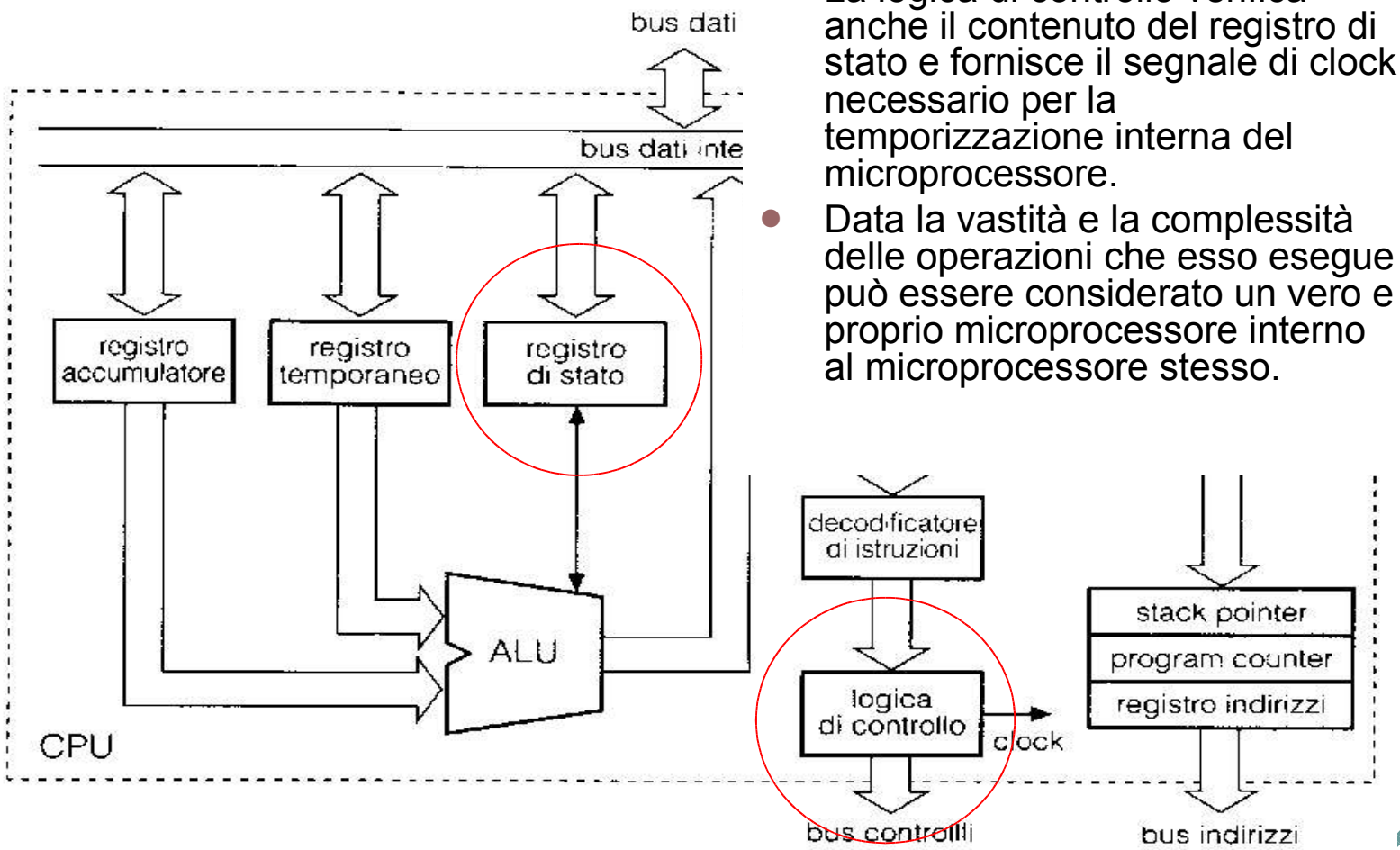
- **Aritmetic Logic Unity: ALU**
- Questa unità, realizzata con tecniche circuitali particolarmente sofisticate, si occupa dell'esecuzione delle operazioni aritmetiche e logiche. Le operazioni aritmetiche si riducono alla somma e alla sottrazione, mentre per quanto riguarda le operazioni logiche la ALU può eseguire:
- - le operazioni logiche AND, OR, XOR e NOT;
- - gli scorrimentì a destra e sinistra di un dato;
- - le rotazioni verso destra e verso sinistra dei dati.
- Siccome la ALU non è in grado di eseguire in modo immediato tutte le operazioni che ad essa vengono affidate è necessario che gli ingressi siano *bufferizzati* e ciò avviene attraverso appositi registri di transito i quali hanno la funzione di memorizzare, per il tempo strettamente necessario, i dati su cui dovrà operare la ALU
- Il risultato delle operazioni è invece normalmente scritto nel registro accumulatore.

3.3 Logica di Controllo

- **Logica di controllo**
- Questo blocco ha il compito di gestire il microprocessore fornendo tutti i segnali necessari al suo funzionamento. Dal decodificatore di istruzioni a cui è collegato riceve un'informazione in codice, la interpreta e genera i segnali necessari all'esecuzione dell'istruzione stessa.
- La logica di controllo verifica anche il contenuto del registro di stato e fornisce il segnale di clock necessario per la temporizzazione interna del microprocessore.
- Data la vastità e la com-plessità delle operazioni che esso esegue può essere considerato un vero e proprio microprocessore interno al microprocessore stesso.



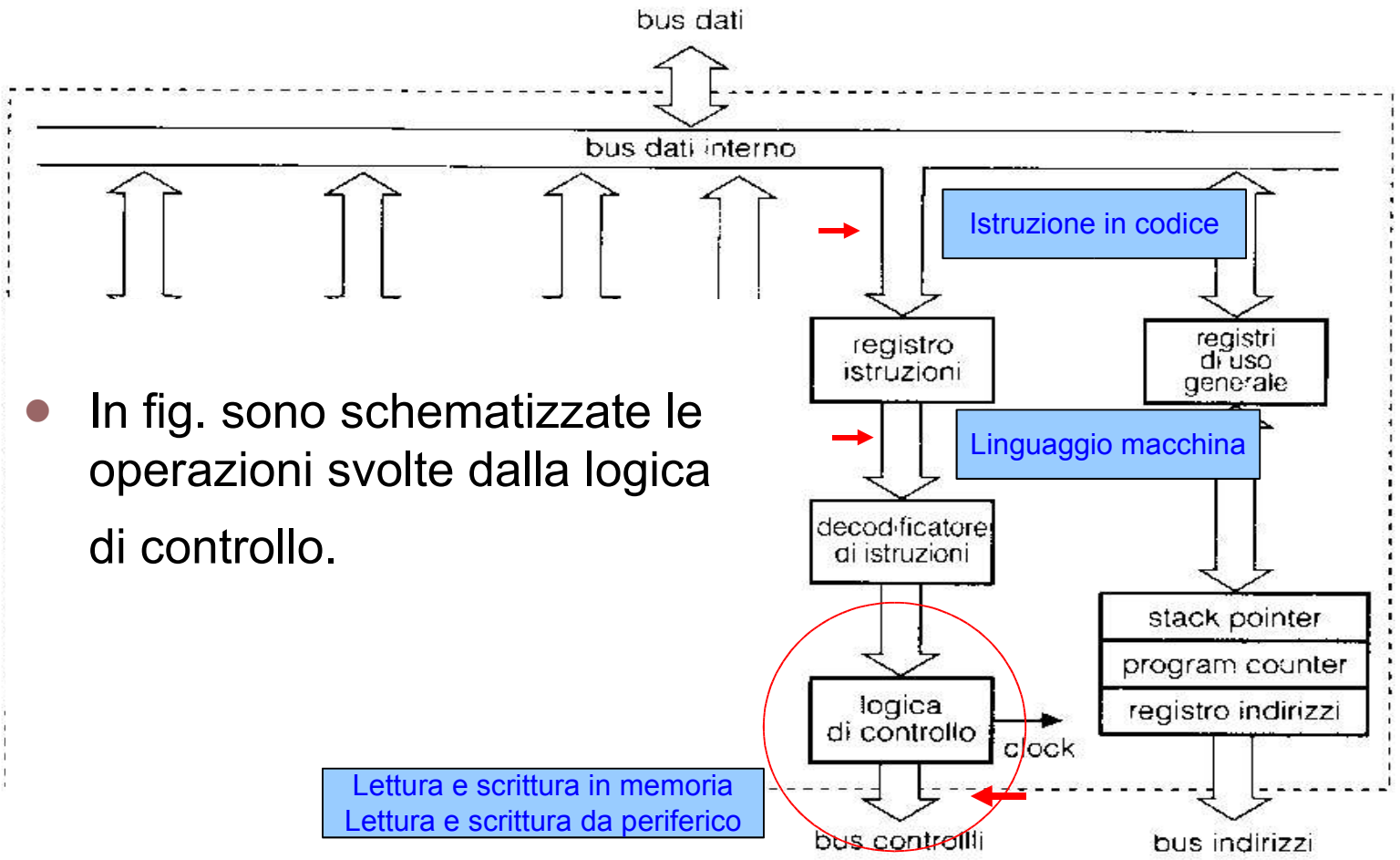
3.3 Logica di Controllo



- La logica di controllo verifica anche il contenuto del registro di stato e fornisce il segnale di clock necessario per la temporizzazione interna del microprocessore.
- Data la vastità e la complessità delle operazioni che esso esegue può essere considerato un vero e proprio microprocessore interno al microprocessore stesso.

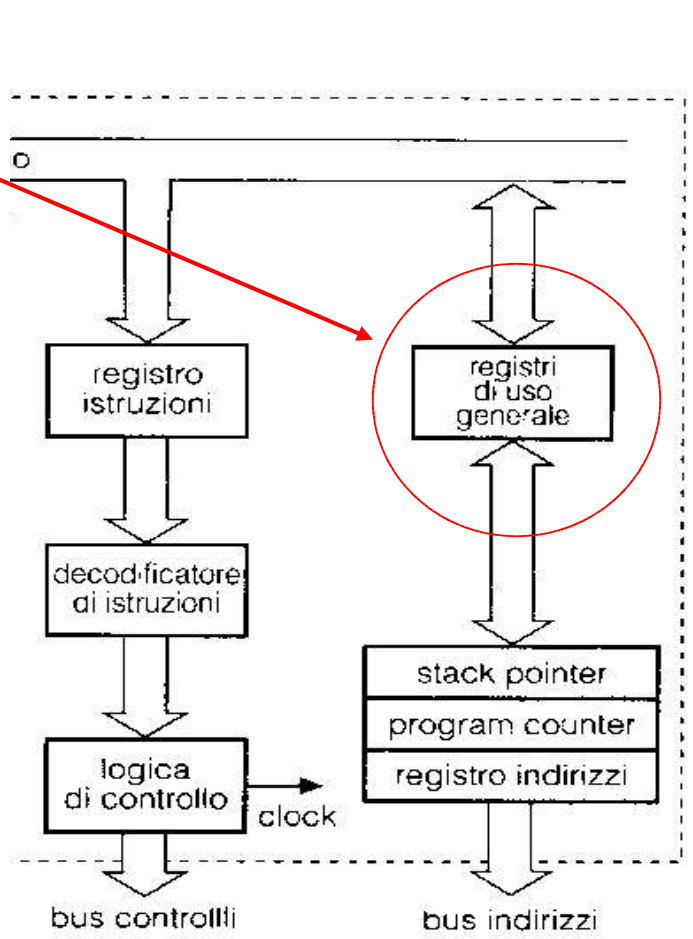
3.3 Logica di Controllo

- In fig. sono schematizzate le operazioni svolte dalla logica di controllo.



3.4 Registri di uso Generale

- **Registri di uso generale**
- Si tratta dell'insierne dei registri che, pur non svolgendo compiti particolari all'interno del microprocessore, risultano particolarmente utili in quanto consentono di memorizzare temporaneamente informazioni, dati o indirizzi all'interno del microprocessore non occupando inutilmente la memoria RAM esterna e riducendo nel contempo i tempi di esecuzione delle istruzioni.

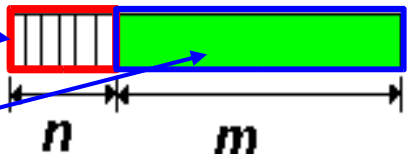


4. MODALITA DI ESECUZIONE Di UNA ISTRUZIONE

Dopo aver esaminato le caratteristiche degli elementi che compongono un generico sistema a microprocessore vediamo ora come essi vengono coinvolti nell'esecuzione di una istruzione.

Ricordiamo che una istruzione è un insieme di caratteri che definiscono, da soli o con altre informazioni, una serie specificata di azioni e che, indipendentemente dal numero di byte che la compongono, è formata da due parti:

- ***Codice operativo**:* rappresenta l'azione che il microprocessore deve eseguire in relazione alla codifica adottata;



- ***Codice operando o campo dati**:*numero su cui si deve eseguire l'azione individuata dal codice operativo. Ad esempio dovendo eseguire la somma di due numeri il campo dati è costituito dai due numeri che devono essere sommati. Il campo dati potrebbe anche mancare ed essere sostituito dall'indirizzo della cella che contiene il dato stesso.

4. MODALITA DI ESECUZIONE Di UNA ISTRUZIONE

- Per chiarire le modalità di esecuzione di una istruzione facciamo riferimento ad una parte di programma, codificata in Assembler Z80, che permette di scrivere il numero esadecimale 30H nel registro accumulatore e successivamente di incrementare il suo contenuto di una unità.

0110	LD A,30_H	3E	I^a istruzione
0111		30	
0112	INC A	3C	II^a istruzione

Il codice operativo della prima istruzione è 3E mentre il campo dati è rappresentato dal numero 30 e nell'insieme la prima istruzione è formata da due byte.

La seconda istruzione è invece una istruzione ad un byte e quindi è individuata unicamente dal codice operativo 3C, Il campo dati risulta costituito dal contenuto del registro accumulatore.

4. MODALITA DI ESECUZIONE Di UNA ISTRUZIONE

L'esecuzione di una istruzione comporta comunque la presenza di due fasi distinte.

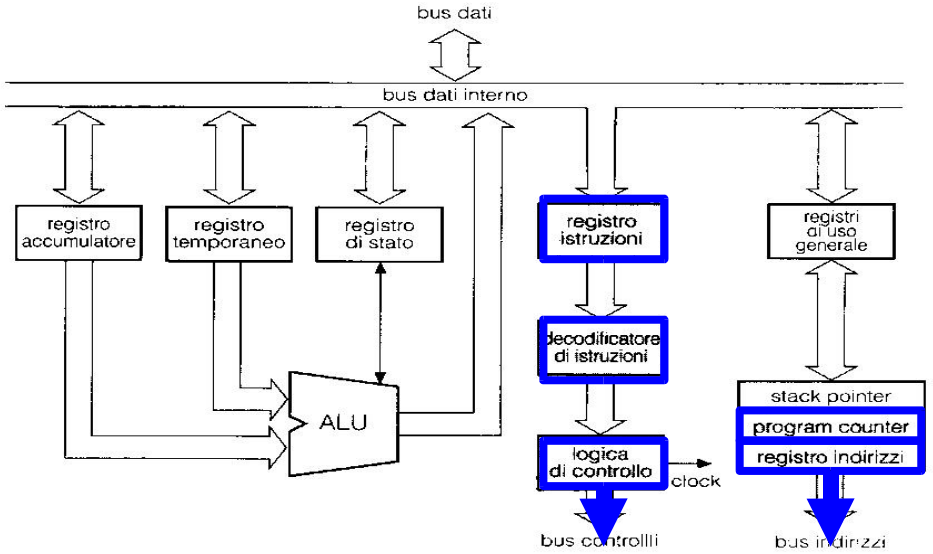
La prima, detta di **prelievo** (fetch), permette l'acquisizione del codice operativo dell‘ istruzione e la sua interpretazione mentre durante la seconda, detta di **esecuzione** (execute), si eseguono effettivamente le operazioni previste dall'istruzione.
Con riferimento alla parte di programma esaminata in precedenza vediamo di individuare le fasi di prelievo e di esecuzione delle istruzioni che lo compongono.

I° istruzione LD A,30

4. MODALITA DI ESECUZIONE Di UNA ISTRUZIONE

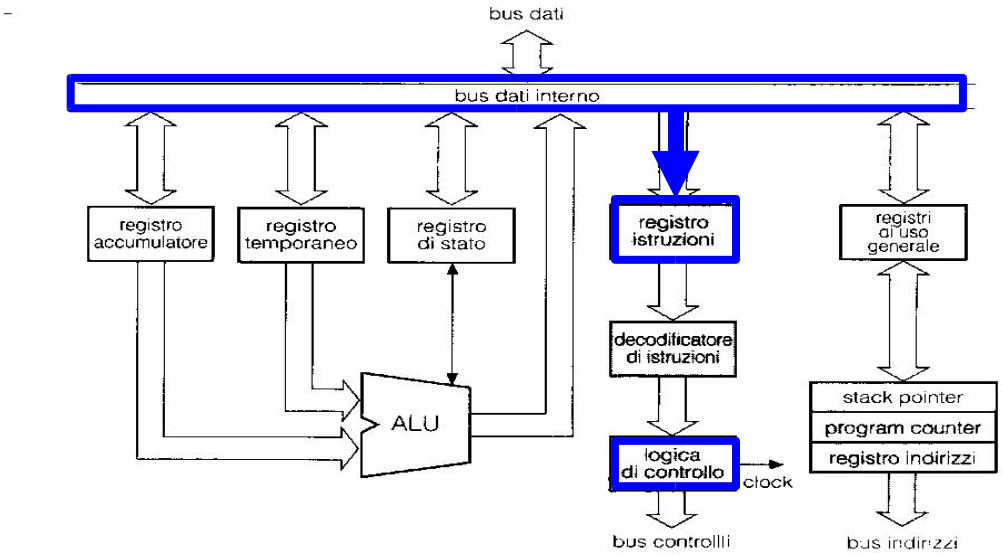
- ***Fase di prelievo***

  La fase di prelievo è quella in cui la CPU acquisisce il codice operativo dell'istruzione e può essere sviluppata in due passi. In un primo tempo il contenuto di PC (0110.) viene posto sul bus indirizzi permettendo alla CPU di individuare la cella contenente l'informazione in codice che, essendo rappresentata dal primo byte dell'istruzione, viene interpretata come codice operativo. Nella fig. sono indicati i dispositivi coinvolti in questa fase.



4. MODALITA DI ESECUZIONE Di UNA ISTRUZIONE

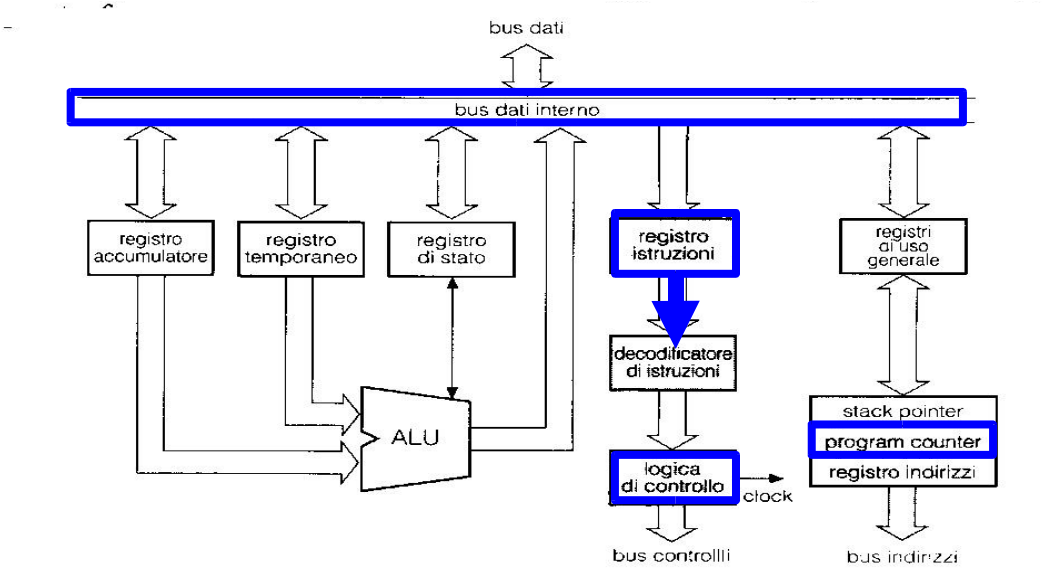
Nel passo successivo il contenuto della cella di indirizzo 0110. viene posto sul bus dati e trasferito al registro istruzioni della CPU.
In fig. è possibile individuare i dispositivi coinvolti.



Il decodificatore di istruzioni provvede quindi alla decodifica permettendo alla logica di controllo di generare i segnali necessari per eseguire effettivamente l'istruzione.

4. MODALITA DI ESECUZIONE Di UNA ISTRUZIONE

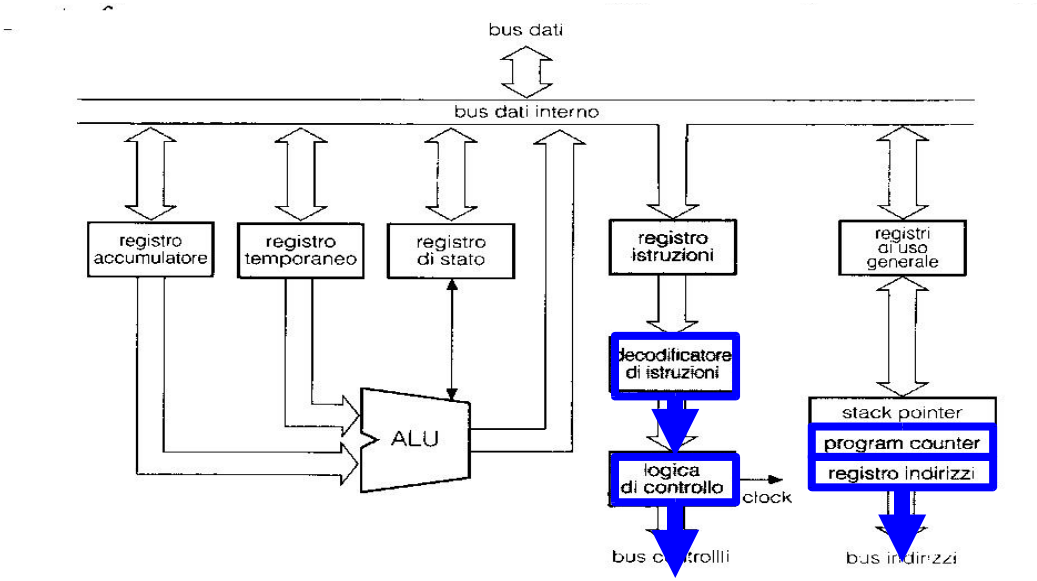
Durante questa fase si ha l'aggiornamento del contenuto di PC che viene incrementato di una unità, vedi fig.



4. MODALITA DI ESECUZIONE Di UNA ISTRUZIONE

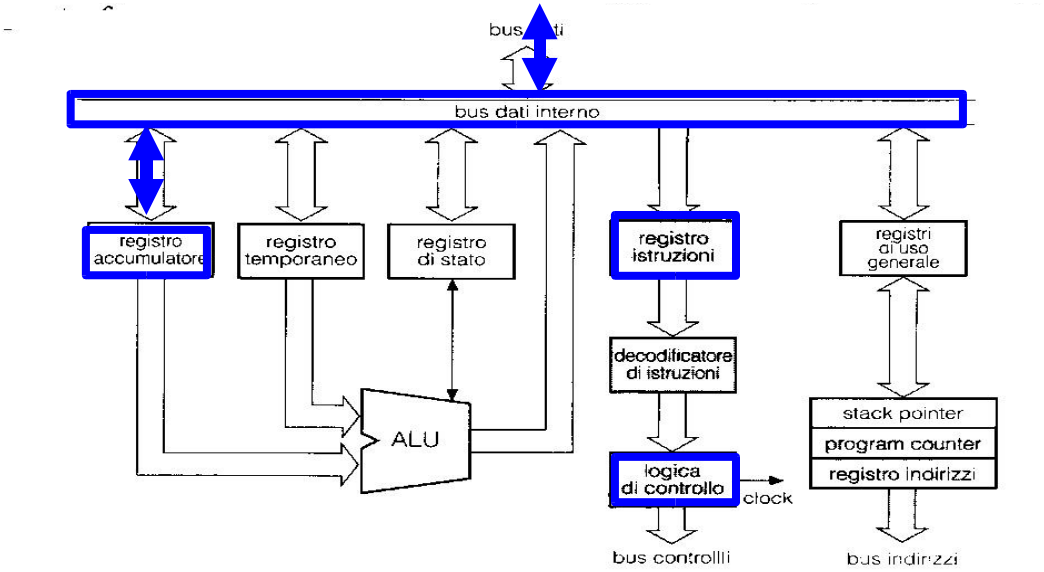
Fase di esecuzione

Tramite i segnali emessi dalla logica di controllo viene posto sul bus indirizzi il nuovo contenuto di PC (0111,) puntando quindi alla cella che contiene l'operando.
I dispositivi interessati sono mostrati in fig.



4. MODALITA DI ESECUZIONE Di UNA ISTRUZIONE

Successivamente il contenuto della cella 0111. viene posto sul bus dati e trasferito nel registro accumulatore completando quindi l'istruzione.
La fig. mette in evidenza i dispositivi che risultano interessati alla suddetta operazione .



4. MODALITA DI ESECUZIONE Di UNA ISTRUZIONE

II° istruzione INC A

Fase di prelievo

In questa fase il contenuto di PC (0112.) viene posto sul bus indirizzi e di qui attraverso il bus dati si provvede al suo trasferimento al registro istruzioni. Nel frattempo il contenuto di PC viene aggiornato e diventa 0113...

Fase di esecuzione

L'esecuzione dell'istruzione comporta dapprima il trasferimento del contenuto dell'accumulatore (30.) nella ALU dove si provvederà ad effettuare l'incremento di una unità previsto dall'istruzione ed al successivo trasferimento del risultato in accumulatore. Ad operazione eseguita il contenuto del registro sarà:

A←A + 1 = 31 mentre essendo l'istruzione a un solo byte il contenuto di PC non viene incrementato mantenendo quindi il valore 0113. corrispondente all'inizio dell'istruzione successiva.