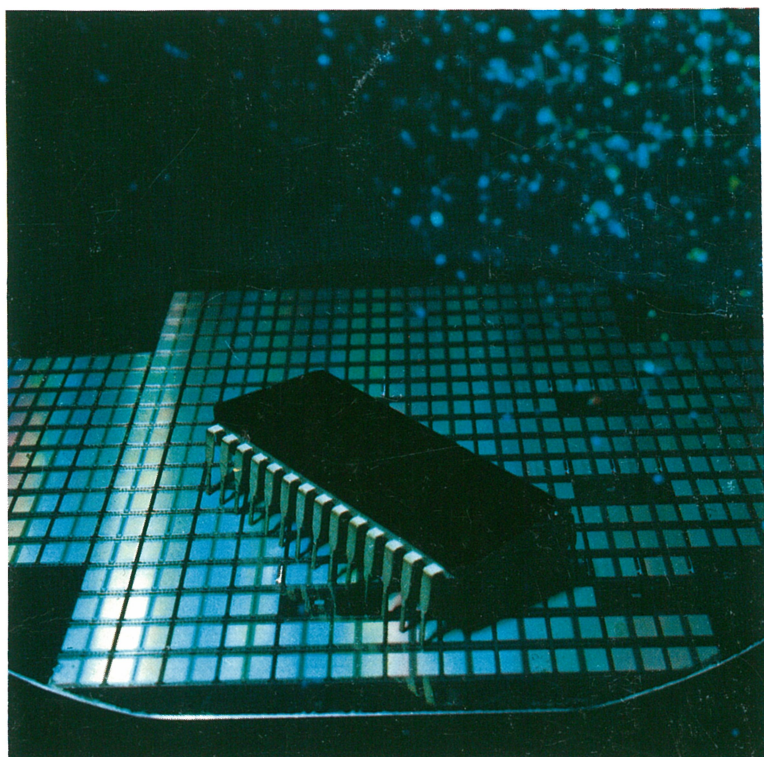


# ELETRONICA HANDBOOKS

## MICROPROCESSORI



GRUPPO EDITORIALE  
**JACKSON**



**ELETTRONICA**  
**HANDBOOKS**

# **MICROPROCESSORI**



**GRUPPO EDITORIALE**  
**JACKSON**

**Direttore responsabile :**

Paolo Reina

**Direttore di divisione :**

Roberto Pancaldi

**Supervisore dell'opera :**

Fosco Bellomo

**Coordinamento editoriale :**

Renata Rossi

**Copertina :**

Sergio Mazzali

**Fotolito :**

3C - Milano

**Stampa :**

GRAFICA 85 - Rodano Millepini

**Distribuzione :**

Sodip - Milano

Tutti i diritti di riproduzione e pubblicazione di disegni , fotografie e testi sono riservati.

© Gruppo Editoriale Jackson - 1988

Aut. alla pubblicazione n° 793 del 30/11/1987

(autorizzazione della Direzione Provinciale delle PPTT di Milano)



# INDICE

## *Capitolo 1*

5     Introduzione

## *Capitolo 2*

9     Cos'è un microprocessore

## *Capitolo 3*

21    Tecnologie costruttive

## *Capitolo 4*

35    Microprocessori a 8 bit più comuni (I)

## *Capitolo 5*

85    Microprocessori a 8 bit più comuni (II)

## *Capitolo 6*

121   Microprocessori a 8 bit più comuni (III)

## *Capitolo 7*

163   Microprocessori a 16 bit

## *Capitolo 8*

---

201 Conclusioni

---

## INTRODUZIONE

**A**

nche se il tempo trascorso dalla comparsa dei microprocessori è relativamente breve, il loro impatto è stato notevole, e le persone attente alle innovazioni tecnologiche hanno compreso la rivoluzione apportata nel mondo dell'elettronica.

In effetti, i microprocessori aprono nuove strade e semplificano il progetto di sistemi complessi, in quanto dispongono di una notevole flessibilità e rendono possibile l'introduzione di nuove funzioni nei sistemi già esistenti.

Tra i molti vantaggi, è rilevante la diminuzione dei costi per innumerevoli applicazioni, nelle quali decine di circuiti integrati possono essere sostituiti da qualche *chip*. Il ridotto cablaggio e la miniaturizzazione del circuito consentono una maggior affidabilità, un minor consumo di potenza e, inoltre, una elevata facilità di diagnosi dei guasti, di riparazione, e di manutenzione degli apparecchi. I microprocessori hanno una notevole influenza sulla concezione del progetto, sul tempo e sui costi di progettazione di nuovi apparecchi e, di conseguenza, sui costi di fabbricazione.

I dispositivi a microprocessore sono utilizzati in numerose applicazioni, che comprendono un'ampia varietà di apparecchi, dalle comunicazioni all'industria, ai prodotti di consumo, agli strumenti di misura ed agli elaboratori. In tale gamma si possono citare sistemi svariati, come il controllo di processo, il controllo numerico, i terminali intelligenti, i giochi elettronici, gli strumenti autocalibranti, le apparecchiature biomediche, le applicazioni militari, il controllo di veicoli, l'elaborazione di dati, gli elettrodomestici, ecc.

Nello sviluppo di sistemi basati sui microprocessori intervengono tre fasi molto complesse: la fase di progetto, l'uso di un adeguato supporto software per lo sviluppo, e, infine, le tecniche per la creazione del software applicativo.

## *LSI: tecnologia che ha permesso di produrre i microprocessori*

Tutti conoscono il transistor, un componente molto piccolo, ma visibile; si immagini però lo spazio che occuperebbero cinquantamila transistori.

Fortunatamente, negli ultimi anni, la tecnologia ha reso possibile la realizzazione di dispositivi molto complessi, con dimensioni molto ridotte.

Questi dispositivi, che per il colore e la forma hanno ricevuto il soprannome di *ragnetti*, sono i circuiti integrati (CI), costituiti da pastiglie, sia in plastica che in ceramica, contenenti una piccola piastra di silicio, materiale semiconduttore, con cui si costruiscono i transistori. Il vantaggio dei C.I. sta nel fatto che, invece di utilizzare dei transistori sciolti e collegarli esternamente, sfruttano una piastrina che contiene tutti i transistori e i relativi collegamenti.

Nel progetto dei circuiti integrati, delle apparecchiature speciali convertono gli schemi logici, rappresentati su carta con elementi AND, OR e NOT, negli equivalenti schemi elettronici a transistori, e questi ultimi in tracciati adatti che permettono di costruire il chip di silicio mediante un processo fotolitografico. Il vantaggio è enorme, dato che lo spazio occupato dagli elementi attivi si riduce, fotograficamente, a dimensioni dell'ordine dei micron, rendendo possibile l'integrazione di migliaia di funzioni logiche.

Per porte semplici, si usa l'integrazione a bassa scala (SSI = Small Scale Integration), dato che i circuiti da costruire sul chip non sono molti. In un personal computer, le porte logiche sono solitamente circuiti integrati a 14 piedini che si trovano sulla scheda principale.

Per funzioni più complicate si utilizza una integrazione a media scala (MSI).

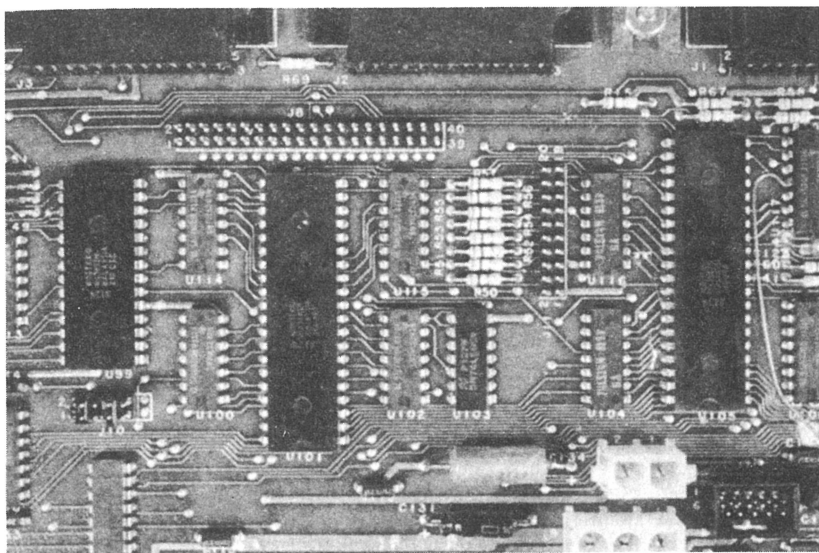
Infine gli integrati più complessi, grandi e costosi, come i microprocessori, che costituiscono la CPU di un personal computer, si costruiscono con tecniche di integrazione a larga scala (LSI e VLSI).

Un fatto che occorre tenere presente è che le dimensioni del contenitore non dipendono tanto dalla complessità del chip, quanto dal numero di funzioni che occorre fornirgli dall'esterno, in ingresso e uscita.

Un microprocessore è solitamente inserito in un contenitore con un minimo di 40 piedini, poichè il numero di segnali che deve controllare è molto elevato. In realtà, il circuito integrato, normalmente, non occupa una superficie maggiore di 25 mm<sup>2</sup>.

## *Hardware di un personal computer*

La scheda principale, a volte l'unica esistente e necessaria per il completo funzionamento del computer, presenta, di solito, una disposizione dei com-



*Foto 1.-Aspetto della scheda madre di un personal computer basato su un microprocessore, cuore dell'apparecchio.*

ponenti che indica la diversità delle funzioni dei chips utilizzati. Prima di tutto, il *capofamiglia*, il microprocessore, sarà sempre facilmente identificabile: è un circuito integrato grande, con un contenitore solitamente a 40 piedini, come nel caso del 6502 (microprocessore dell'Apple), o dell'8088 (microprocessore dell'IBM-PC). Negli apparecchi più moderni, come il Mac Apple, il  $\mu P$  è inserito in un contenitore a 64 piedini: si tratta del 68000, molto più potente degli altri due, che, di conseguenza, necessita di più terminali.

Più avanti si vedrà come è fatto e come funziona un microprocessore.

Per ora, basta ricordare che il microprocessore viene definito anche Unità Centrale di Processo (o CPU), poichè il suo compito è di dirigere l'esecuzione dei programmi e il processo dei dati inseriti dall'utente.

Attorno alla CPU si trovano altri circuiti integrati che risultano normalmente più piccoli. Una buona parte di essi costituiscono la memoria, che funziona, relativamente alla CPU, come una specie di magazzino per i dati e gli ordini in corso di esecuzione. Infine, altri circuiti, noti come *integrati di ingresso/uscita*, permettono il collegamento della scheda principale con altre schede esterne, e il collegamento di quelli citati precedentemente tra di loro. Sulla scheda esiste un vasta gamma di altri integrati più piccoli, la cui funzio-

ne è di adattare i segnali che fluiscono tra i circuiti integrati principali, facilitandone il collegamento.

La necessità di questi piccoli circuiti ausiliari deriva dal fatto che, sia il microprocessore, sia la memoria, che i dispositivi di ingresso/uscita, sono progettati in modo che possano formare configurazioni molto diverse per vari scopi.

Questa versatilità obbliga ad aggiungere, per ogni tipo di configurazione, un certo numero di tali componenti ausiliari.

## *COS'E' UN MICROPROCESSORE*



n questo capitolo verrà definito il microprocessore e ne verranno descritte le caratteristiche principali.

### *Definizione di microprocessore*

Il microprocessore è un circuito integrato capace di eseguire un programma e di controllare i dispositivi necessari a tale esecuzione. Le principali applicazioni del microprocessore sono:

- la sostituzione di circuiti logici, utilizzabili per eseguire un unico compito, con circuiti programmabili in grado di risolvere diversi problemi con programmi distinti.
- il funzionamento come unità centrale di processo di un microelaboratore.

### *Caratteristiche di un microprocessore*

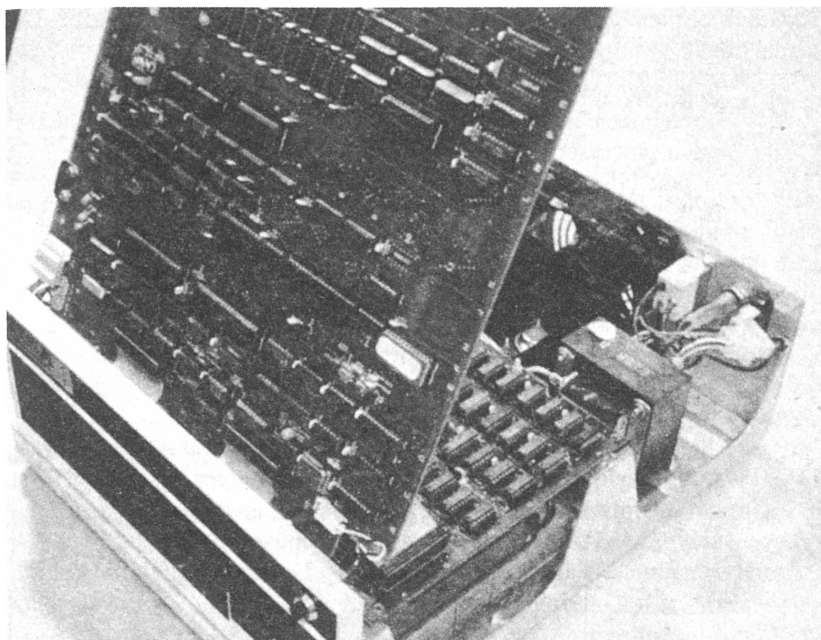
I microelaboratori hanno caratteristiche strettamente dipendenti dal microprocessore su cui si basano, poichè sia la loro potenza, sia le altre prestazioni, sono condizionate dalle caratteristiche della loro CPU, costituita da un microprocessore. Le prestazioni caratteristiche di un microprocessore sono le seguenti:

- lunghezza della parola processata: le lunghezze più comuni della

parola nei microprocessori attuali sono 8 o 16 bit, anche se ne esistono alcuni che operano con parole di 4 o 32 bit. La precisione di calcolo del microprocessore e la sua capacità di indirizzamento aumenta con l'aumentare della lunghezza della parola elaborata;

- capacità di memoria: questa caratteristica è normalmente correlata con la lunghezza della parola elaborata. La capacità massima della memoria utilizzabile da un microprocessore è stabilita dalle sue capacità di indirizzamento. Tuttavia, microprocessori con parole della stessa lunghezza possono avere memorie diverse nella configurazione iniziale;
- velocità di esecuzione delle istruzioni: si definisce ciclo di istruzione, il tempo impiegato dal microprocessore per eseguire completamente un'istruzione. Tale caratteristica determina la velocità di esecuzione di un microprocessore. Il fattore da considerare nell'adottare questa misura come dato caratteristico, è che il ciclo è diverso in funzione del tipo di istruzione eseguita. Un'altra misura più omogenea è il ciclo macchina, cioè il tempo impiegato dal microprocessore per eseguire un'operazione elementare, che costituisce parte di una qualsiasi istruzione completa;
- registri speciali: un'altra caratteristica importante dei microprocessori è il numero dei registri speciali che possiedono. La maggioranza dispone di un unico accumulatore nell'unità aritmetico-logica, ma esistono microprocessori che incorporano due accumulatori, fatto che amplia la loro potenza e velocità di elaborazione. Analogamente, esistono due tendenze per gli altri registri interni: la prima consiste nell'utilizzare parte della memoria RAM come registri propri del microprocessore, la seconda è di inserire vari registri di lavoro all'interno del microprocessore;
- capacità di interrupt: l'esecuzione di un programma può essere interrotta in alcuni casi. Una caratteristica fondamentale del microprocessore è la capacità di recepire e gestire un determinato numero di interruzioni. Mediante tali interruzioni si possono stabilire le comunicazioni necessarie, sia con l'utente che con altri dispositivi del microelaboratore, senza che ciò influisca sulla corretta esecuzione del programma in corso;
- famiglia di circuiti complementari: la necessità di completare l'operatività del microprocessore, esige l'impiego di una serie di circuiti integrati adattabili allo stesso. In questo modo nascono





*Foto 1.-Nonostante il volume minimo, il microprocessore funziona come cervello di potenti microelaboratori. Ad esempio, il microprocessore Z80 costituisce la CPU del sistema multiutente ALTOS-8010 che è visibile nella fotografia.*

diverse famiglie di circuiti complementari. Ad esempio, si può citare la famiglia del 6800 o dell'8080, riferendosi non solo ai microprocessori MOTOROLA-6800 o INTEL- 8080, ma anche ai circuiti a questi adattabili.

### *Programmazione di un microprocessore*

Quanto precedentemente esposto sui microprocessori porta a concludere che, a livello fisico (hardware), hanno la stessa configurazione dell'unità centrale di processo di un elaboratore. Questo vale non solo per la parte fisica, ma anche dal punto di vista logico (software), poichè il loro funzionamento è analogo. Il microprocessore funziona direttamente in linguaggio macchina, tuttavia, disponendo del corrispondente traduttore, si possono uti-

lizzare determinati linguaggi di programmazione più evoluti. La classificazione dei linguaggi utilizzati è la seguente:

- linguaggio macchina: le istruzioni del programma si scrivono direttamente in binario e sono immediatamente interpretabili dal microprocessore. In alcuni casi, invece di un linguaggio binario si può utilizzare una codificazione ottale o esadecimale. Un programma codificato direttamente in linguaggio macchina viene definito programma oggetto;
- linguaggi simbolici: invece di ricorrere a codificazioni binarie, come nel caso del linguaggio macchina, si può programmare con istruzioni simboliche. Questo tipo di programma viene definito sorgente, e deve essere tradotto in programma oggetto da un compilatore;
- linguaggi ad alto livello: rappresentano un passo avanti verso l'utilizzazione di linguaggi simili a quelli convenzionali e facili da interpretare dall'utente. Allo stesso modo dei linguaggi simbolici, i programmi redatti in linguaggio ad alto livello vengono definiti sorgente e, prima di essere eseguiti, devono essere tradotti in linguaggio macchina.

## *L'informazione nel microprocessore*

Un microprocessore elabora due tipi di informazioni: quelle che verranno interpretate dall'unità di controllo, e i dati, di cui si occupa l'unità aritmetico-logica in accordo con quanto indicato nelle istruzioni. Si analizzerà, nei paragrafi successivi, il flusso di informazioni che si stabilisce nel microprocessore, quando questo esegue le tre istruzioni fondamentali di caricamento, memorizzazione, e biforcazione.

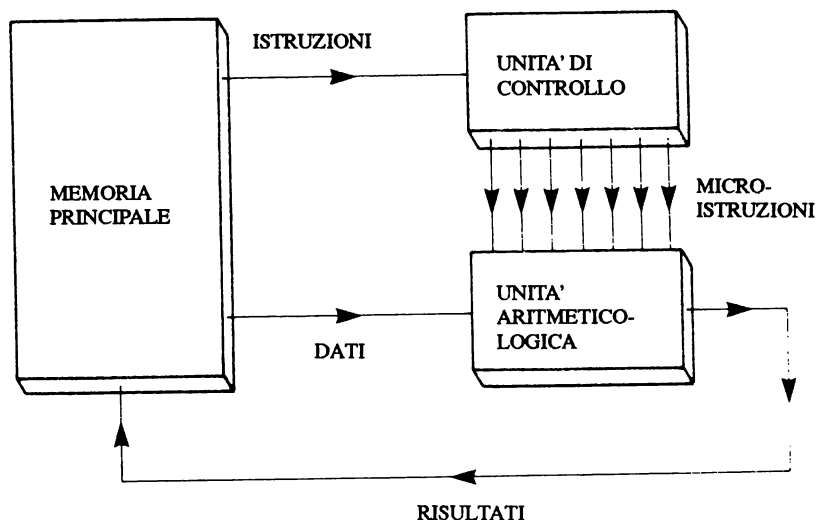
## *Esecuzione sequenziale*

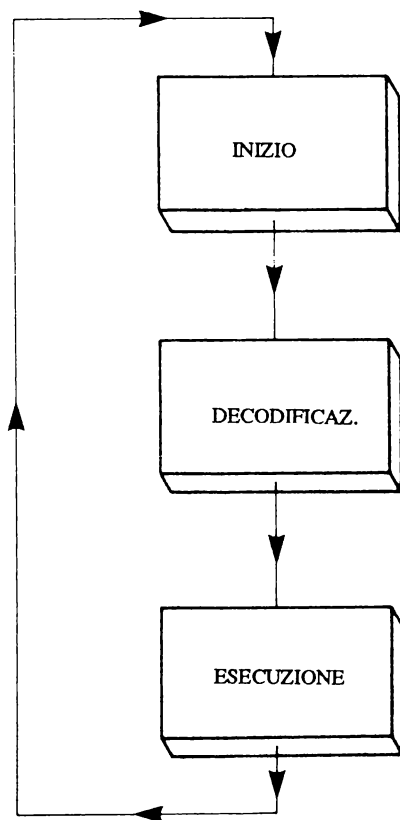
Le istruzioni che devono essere eseguite dal microprocessore sono costituite da due parti: codice di operazione e operando (uno o più). Ad esempio, nell'operazione di somma di due numeri ( $A + B$ ), A e B sono gli operandi (in questo caso gli operandi coincidono con i dati reali) e il segno + il codice dell'operazione. L'elaborazione di ciascuna delle parti che costituiscono l'istruzione è svolta da diverse unità interne del microprocessore. I codici dell'operazione sono elaborati dall'unità di controllo, e gli operandi dall'uni-

tà aritmetico- logica.

Per inviare in modo sequenziale le parole binarie dell'informazione verso l'unità di controllo, o verso l'unità aritmetico-logica, ed ottenere la corretta esecuzione del programma, è necessario conoscere perfettamente le posizioni della memoria in cui risiede tale informazione. Le diverse posizioni della memoria sono identificate da uno specifico indirizzo. Per accedere in qualsiasi momento alla posizione di memoria necessaria, il microprocessore conta, mediante un registro specializzato per questo compito, definito *contatore di programma* o *contatore di indirizzi*. In ogni istante, il contenuto di tale registro coincide con l'indirizzo corrispondente alla posizione della memoria alla quale accederà il microprocessore, sia per leggere che per memorizzare in essa una informazione. Per il momento interessa conoscere il metodo seguito dal microprocessore per eseguire un programma, o un insieme di istruzioni costituite da codici di operazione (ordini) e operandi (dati o indirizzi corrispondenti alla posizione nella memoria dei dati che interessano).

Il processo di esecuzione inizia con la lettura da parte del microprocessore, del contenuto di una locazione di memoria, che, per quanto visto, è indicata dal contatore di programma. Perchè l'esecuzione avvenga automaticamente da questo istante, è necessario che la prima parola letta nella memoria, che può essere sia un'istruzione che un dato, coincida con un *codice di operazione*, in quanto, essendo un comando, indicherà al microprocessore qual'è la natura (codice di operazione o dato) della parola che sarà letta successivamente.





*Fig. 1.-a) Il grafico indica il flusso dell'informazione all'interno di un sistema basato su microprocessore. b) Le tre fasi che costituiscono il trattamento dell'istruzione da parte del microprocessore*

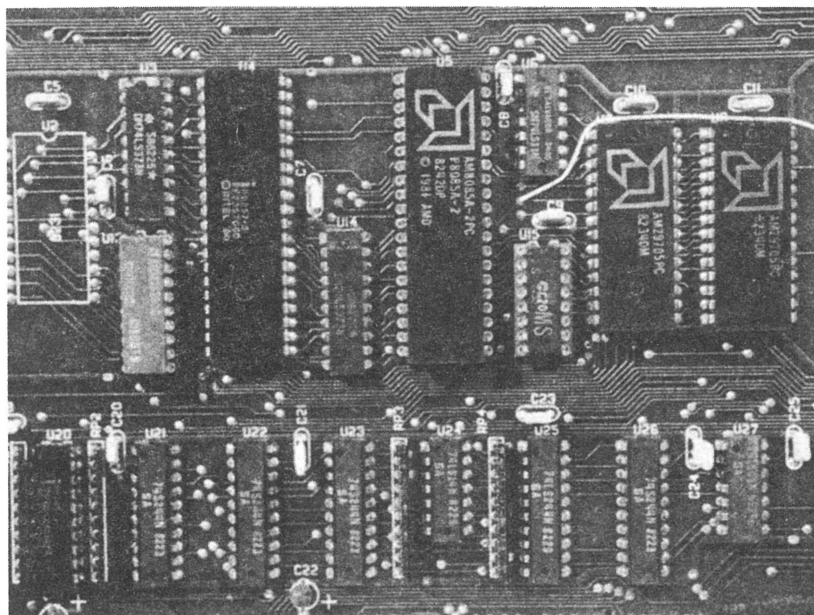
Questa prima parola dell'informazione che si legge nella memoria, raggiunge il microprocessore attraverso il *bus dei dati* e viene inviata ad un registro interno detto *registro delle istruzioni*. Fatto ciò, l'unità di controllo interpreta tale parola che, com'è stato detto, coincide con un codice operativo o ordine da eseguire. Dopo averla interpretata, l'unità di controllo conoscerà l'operazione da eseguire, e la natura della parola che deve leggere nella memoria. Se la nuova parola è un codice di operazione, verrà interpretata analogamente alla precedente; se si tratta invece di un operando, lo invierà all'unità aritmetico-logica e genererà gli ordini necessari perchè venga elaborato adeguatamente.

Il flusso delle informazioni si stabilisce attraverso tre vie:

- il bus dei dati, che ha il compito di leggere e memorizzare le informazioni nella memoria;
- il bus degli indirizzi, che indica la posizione in memoria dalla quale leggere o nella quale scrivere la parola dell'informazione;
- il bus di controllo, che trasmette gli ordini (microistruzioni) generati in seguito all'interpretazione dei codici operativi delle istruzioni.

### *Esecuzione di un'istruzione tipica*

Il processo di esecuzione inizia con l'invio, al bus degli indirizzi, del contenuto del registro contatore degli indirizzi, in modo da puntare alla posizione di memoria che contiene la prima parte dell'istruzione: il codice operativo.



*Foto 2.-Il microprocessore non è solo in grado di elaborare i dati che gli vengono forniti, ma tra le sue funzioni vi è anche quella di controllare le unità interessate nell'elaborazione.*

Tramite il bus di controllo, il microprocessore indicherà all'unità di memoria che vuole *leggere* il contenuto della posizione indirizzata; la parola dell'informazione passerà nel bus dei dati e da questo all'interno del registro delle istruzioni del microprocessore. Questa prima fase del trattamento dell'istruzione è definita *fase iniziale o inizio*.

Nella seconda fase di *decodificazione*, l'unità di controllo decodifica e interpreta il codice dell'operazione depositato nel registro delle istruzioni.

Infine, l'unità di controllo genererà i comandi adeguati nella fase definita di *esecuzione*.

Se l'esecuzione dell'istruzione necessita di più bit, l'unità di controllo verrà incaricata di fornirli e, in seguito, si passerà ad eseguire una nuova istruzione. Ogni volta che si accede alla memoria (fase iniziale) inizia un *ciclo macchina*, che contiene da 2 a 5 stati interni, ognuno dei quali corrisponde ad una microistruzione.

### *Esecuzione di una istruzione particolare*

Per studiare il flusso dell'informazione e i segnali generati dall'unità di controllo del microprocessore, si descrive il processo di esecuzione di un'istruzione di caricamento. Per semplificare la descrizione si suppone che il microprocessore sia da 8 bit, con un bus degli indirizzi a 16 bit.

#### *Istruzione di caricamento o lettura*

Legge il contenuto della locazione di memoria indirizzata e lo carica nell'accumulatore. Questa istruzione si completa normalmente in quattro cicli di macchina.

- Primo ciclo di macchina:

ha il compito di riconoscere e decodificare il codice dell'operazione. Con il primo impulso di clock, il contenuto del contatore di programma è inviato al bus degli indirizzi.

Con il secondo impulso di clock si invia attraverso il bus di controllo un ordine di lettura e si incrementa di un'unità il contatore di programma.

Con il terzo impulso di clock, il contenuto della locazione di memoria è inviato al registro delle istruzioni tramite il bus dei dati.

Dopo il quarto impulso di clock, il contenuto del registro delle istruzioni verrà decodificato. La configurazione binaria del codice operativo indica che le due locazioni di memoria seguenti contengono l'operando (a 16 bit), che coincide con l'indirizzo in cui si trova il dato interessato all'esecuzione dell'istruzione.

- Secondo ciclo di macchina:

carica i primi otto bit dell'operando (indirizzo del dato) nella prima parte del registro di indirizzamento.

Si esegue nuovamente un'operazione di lettura in memoria e si deposita la parola binaria letta, nella prima parte del registro di indirizzamento.

- Terzo ciclo di macchina:

carica il secondo byte dell'operando (indirizzo del dato) nella seconda parte del registro di indirizzamento.

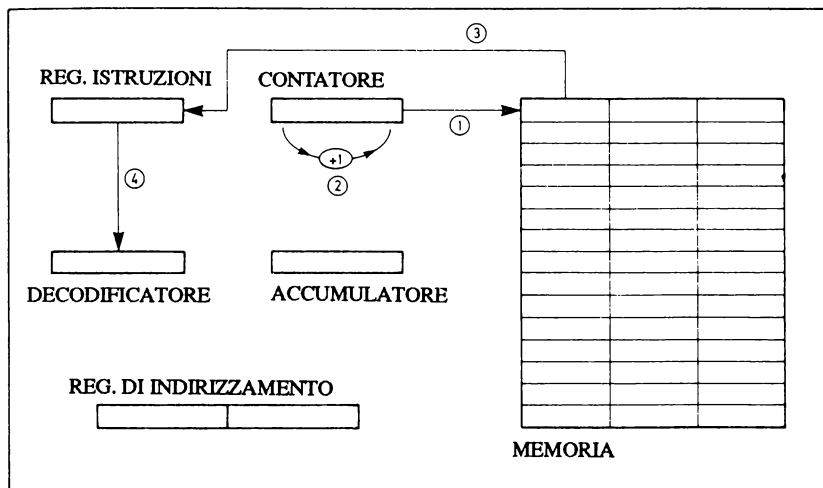
Si eseguono le operazioni analoghe a quelle del secondo ciclo macchina, eccetto che la parola binaria viene caricata nella seconda parte del registro di indirizzamento.

- Quarto ciclo di macchina:

in tale ciclo si effettua il caricamento del dato reale (il cui indirizzo era nell'operando), nell'accumulatore.

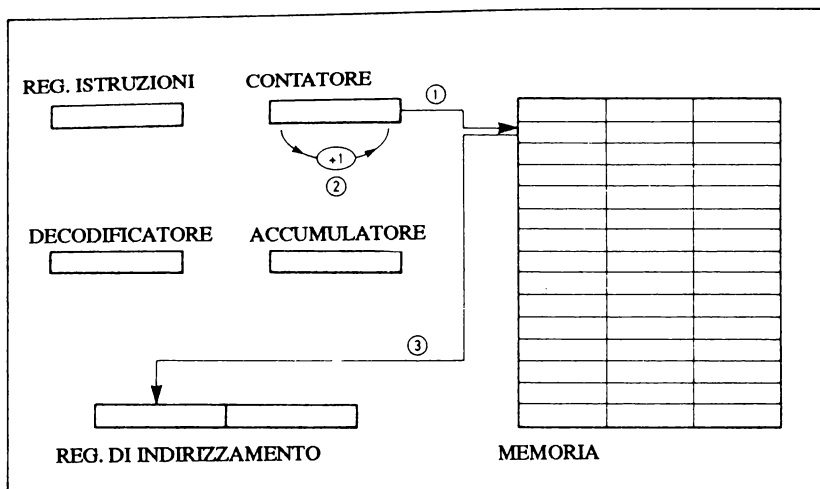
Con il primo impulso di clock si invia il contenuto del registro degli indirizzi.

Il secondo impulso coincide con la lettura del dato e la sua trasmissione al bus dei dati.



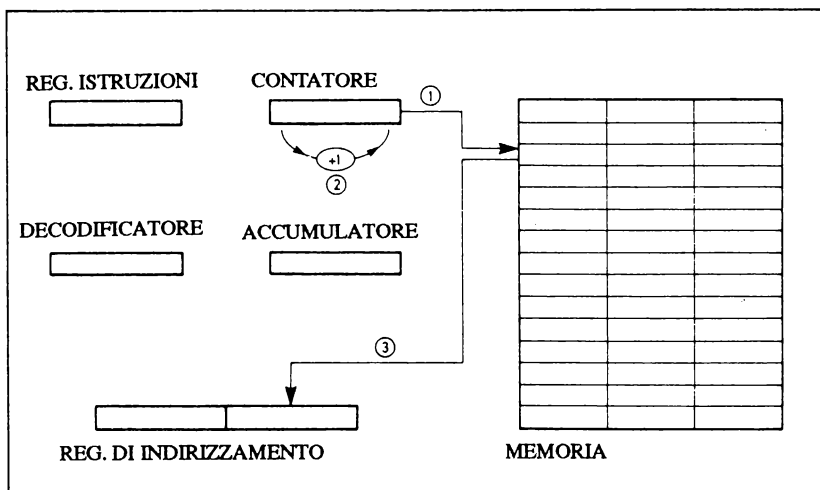
PRIMO CICLO ISTRUZIONI DI CARICAMENTO

| IMPULSO DI CLOCK | AZIONE                                 |
|------------------|--|
| ①                | INDIRIZZAMENTO                         |
| ②                | INCREMENTO CONTATORE                   |
| ③                | CARICAMENTO NEL REGISTRO DI ISTRUZIONE |
| ④                | DECODIFICA                             |



SECONDO CICLO ISTRUZIONI DI CARICAMENTO

| IMPULSO DI CLOCK | AZIONE   |
|------------------|--|
| ①                | INDIRIZZAMENTO                                     |
| ②                | INCREMENTO CONTATORE                               |
| ③                | CARICAMENTO NELLA 1ª PARTE DEL REG. INDIRIZZAMENTO |



TERZO CICLO ISTRUZIONI DI CARICAMENTO

| IMPULSO DI CLOCK | AZIONE   |
|------------------|--|
| ①                | INDIRIZZAMENTO                                     |
| ②                | INCREMENTO CONTATORE                               |
| ③                | CARICAMENTO NELLA 2ª PARTE DEL REG. INDIRIZZAMENTO |



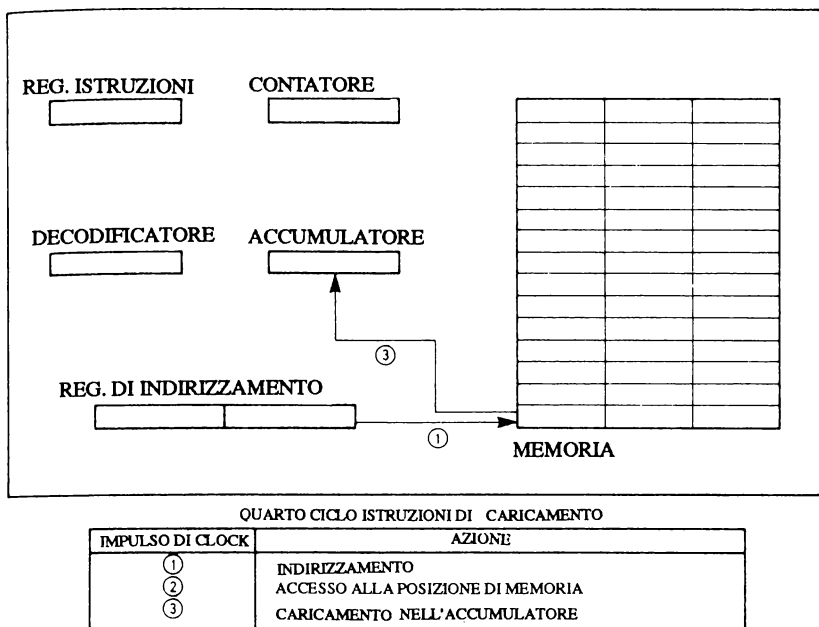


Fig. 2.-a) Primo ciclo dell'esecuzione di un'istruzione di caricamento. b) Secondo ciclo dell'istruzione. c) Operazioni eseguite durante il terzo ciclo. d) Quarto ciclo dell'istruzione.

Con il terzo impulso di clock, l'informazione contenuta nel bus dei dati viene trasferita nell'accumulatore, e nello stesso tempo si azzerà il registro delle istruzioni, che rimane libero per ricevere l'istruzione successiva.



**I**

n questo capitolo, vengono descritte le diverse tecnologie utilizzate per la costruzione dei microprocessori integrati. Inoltre, sono evidenziate le varie tecniche che tendono a migliorare le caratteristiche elettriche di questi sottosistemi.

### *Introduzione*

I microprocessori ( $\mu P$ ) sono sottosistemi digitali relativamente complessi e, pertanto, adatti ad essere costruiti con la tecnologia di integrazione a larga scala (LSI).

Il primo  $\mu P$  integrato venne messo in commercio nel 1972. Nonostante il breve tempo trascorso, la ricerca nel campo dei semiconduttori è stata talmente intensa, da poter dire di essere giunti attualmente alla terza generazione. Senza dubbio, nei prossimi anni, a causa della grande concorrenza, verranno prodotte versioni migliori (maggior capacità, maggior rapidità, minor consumo, software più semplice, minor costo, ecc.).

Anche se si possono progettare versioni a componenti discreti più flessibili, ma più costose, con porte logiche standard, saranno descritte solo le tecnologie utilizzate attualmente per le versioni integrate. Saranno inoltre commentate le tendenze future prevedibili.

Nella tabella I sono riassunte le attuali tecnologie dei  $\mu P$ , evidenziando alcuni interessanti dati; nella tabella II sono invece riassunti i dati più interessanti relativi ai microprocessori.

Dall'esame comparativo delle due, si può ricavare una valida informazione sulla progettazione di questi componenti.

**TABELLA I. ATTUALI TECNOLOGIE DEI  $\mu P$**

| TECNOLOGIA | FAMIGLIE   | RITARDO<br>PROPAGAZIONE<br>FORTE | RITARDO *<br>DISSIPAZIONE | TEMPO DI<br>CICLO<br>ISTRUZIONE | 4 PORTE    |                                  |          |            |
|------------|--|----------------------------------|---------------------------|---------------------------------|------------|----------------------------------|----------|------------|
|            |  |                                  |                           |                                 | COMPONENTI | SUPERFICIE<br>(mm <sup>2</sup> ) | MASCHERE | DIFFUSIONI |
| MOS        | CANALE P<br>1972                                   | < 1 $\mu s$ (tip 100 ns)         | 50                        | 1,25 - 62,5 $\mu s$             | 2          | 10,6                             | 4        | 1          |
|            | CANALE N<br>PORTAL<br>SILICIO 1973                 | 1 - 100 ns                       | 1                         | 0,3 - 13 $\mu s$                | 2          | 5,6                              | 7        | 3          |
|            | CMOS<br>1974                                       | < 100 ns (tip 50)                | 4                         | 0,5 - 5 $\mu s$                 | 3          | 49,8                             | 6        | 3          |
| BIPOLARE   | TTL-SCHOTTKY<br>BASSA DISSIPAZIONE<br>(LSTTL) 1974 | 10-25 ns                         | 10-20                     | 100-200 ns                      | 3          | 19,9                             | 7        | 4          |
|            | I <sup>2</sup> L<br>IMPIANTAZIONE<br>IONICA 1975   | 10-50 ns                         | 0,1-1                     | 100 - 1000 ns                   | 1          | 4,8                              | 4        | 2          |
|            | BCL<br>1975  | 1 10 ns                          | 50                        | 55 ns                           | 3          | 31                               | 7        | 4          |

\* 1 mil 25,4  $\mu$

**TABELLA II. MICROPROCESSORI**

| <b>FABBRICANTE<br/>MODELLO</b>   | <b>DATI<br/>TECNOLOGICI</b>  | <b>BITS IN<br/>PARALLELO</b>   | <b>CAPACITA'<br/>INDIRIZZAMENTO</b>  | <b>TEMPO DEL<br/>CICLO DI<br/>ISTRUZIONE<br/>(minimo)</b>  | <b>ALIMENTAZIONE</b>  |
|--|--|--|--|--|---|
| Intel<br>4040<br>8080<br>8085<br>80386<br>Signetics<br>2650<br>Motorola<br>6800<br>68000<br>Rockwell<br>6502<br>National<br>PACE<br>SC/MP<br>Zilog<br>Z-80<br>Z 8000 | PMOS<br>NMOS<br>NMOS<br>HMOS<br><br>NMOS<br><br>NMOS<br>NMOS<br><br>NMOS<br><br>PMOS<br>PMOS<br><br>NMOS<br>NMOS | 4<br>8<br>8<br>16<br><br>8<br><br>8<br>16<br><br>8<br><br>16<br>8<br><br>8<br>16 | 4 K<br>64 K<br>64 K<br>1 M<br><br>32 K<br><br>64 K<br>16 M<br><br>64 K<br><br>64 K<br>64 K<br><br>64 K<br>48 M | 10,8 $\mu$ s<br>1,5 $\mu$ s<br>1,3 $\mu$ s<br>0,4 $\mu$ s<br><br>2,4 $\mu$ s<br><br>1 $\mu$ s<br>0,5 $\mu$ s<br><br>2 $\mu$ s<br><br>2,5 $\mu$ s<br>5 $\mu$ s<br><br>1 $\mu$ s<br>0,75 | 15,6-10 y 5 V<br>-5,5 y 12 V<br>5 V<br>5 V<br><br>5 V, 525 mW<br><br>5 V<br>5 V<br><br>5 V<br><br>-12,5 y 8 V<br>-7, +5 V<br><br>5 V<br>5 V |

## Riassunto delle famiglie utilizzate nei $\mu P$ attuali

In questo paragrafo sono riassunte le varie famiglie utilizzate nei  $\mu P$  integrati.

### PMOS

#### a) Porta statica

In Fig. 1 è visibile una porta NAND (logica negativa) con transistori MOS a canale P.

Tutte le porte MOS digitali sono costituite da transistori del tipo ad accumulazione. Il motivo è duplice: per permettere l'accoppiamento diretto tra stadi, e per minimizzare il consumo (se  $V_{GS}=0$ ,  $I_D=0$ ).

In tutte le famiglie MOS si utilizza come carico un transistorore MOS che funziona come resistenza. In tal modo si riduce la superficie richiesta per l'integrazione. Il terminale di gate (G) del transistorore di carico, perchè funzioni come tale, è solitamente connesso ad una tensione fissa ( $V_{GG}$ ). Se  $V_{GG}=V_{PDD}$ , il transistorore di carico è polarizzato nella sua regione piana di saturazione (il MOS è in saturazione se  $|V_{DS}| > |V_{GS}-V_{TH}|$ ), il che non è adatto per la carica rapida delle capacità. Se  $V_{GG}$  raggiunge un potenziale fisso di grandezza superiore a  $V_{DD}$ , il transistorore di carico può funzionare nella regione lineare, a minor impedenza. La connessione dipende dalla particolare applicazione.

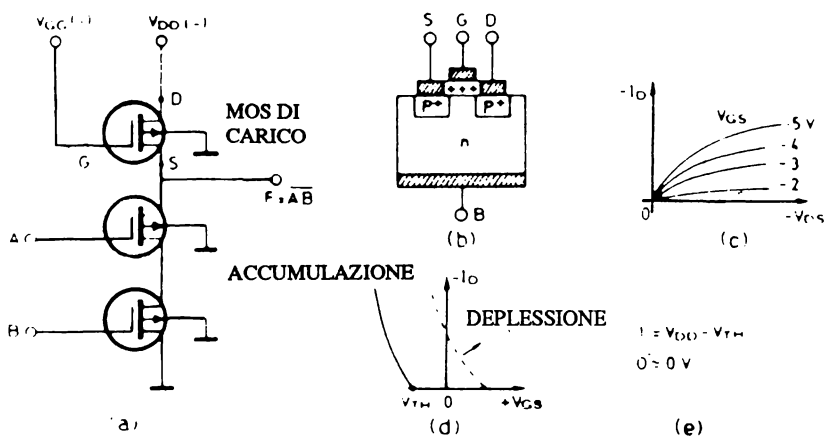


Fig. 1.-a) Porta NAND. b) Sezione trasversale. c) Caratteristiche del drain. d) Caratteristica di trasferimento. e) Livelli logici.

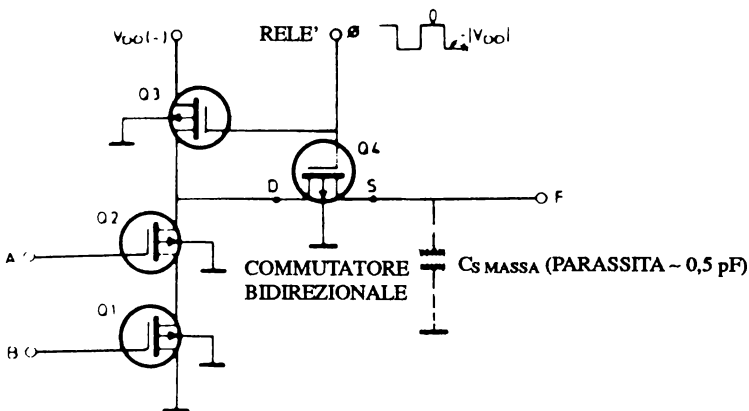


Fig. 2.-Porta MOS dinamica NAND ad una fase.

#### b) Porta dinamica

Le porte MOS dinamiche immagazzinano temporaneamente l'informazione in una capacità parassita. Il loro funzionamento è simile a quello dei circuiti di campionamento e mantenimento. Mediante un segnale di clock, l'immagazzinamento citato può essere reso permanente (operazione di refresh). Poichè le fughe sono molto piccole, le costanti di tempo di scarica delle capacità parassite sono, di solito, dell'ordine dei millisecondi e quindi il periodo del segnale di clock deve essere, tipicamente, inferiore a 1 msec (frequenza superiore a 1 KHz), per mantenere i dati permanentemente memorizzati.

In Fig. 2 si vede una porta MOS dinamica NAND.

Quando  $\Phi=0$ , Q3 e Q4 sono interdetti, poichè non si forma canale, e rendono indipendente l'uscita (F) dal circuito di eccitazione, evitando inoltre il consumo della sorgente di alimentazione ( $V_{DD}$ ).

Se  $\Phi=1$ , Q3 e Q4 sono in conduzione e all'uscita si ottiene la funzione NAND.

Le porte dinamiche possono utilizzare più fasi (vari clock).

Il vantaggio dei circuiti dinamici rispetto a quelli statici è il minor consumo e costo.

Tra i vantaggi della famiglia PMOS si possono evidenziare:

- costruzione semplice (1 diffusione - 4 maschere);
- media densità di integrazione.

Gli svantaggi sono:

- grande lentezza;
- incompatibilità con altre famiglie logiche (necessitano interfacce).

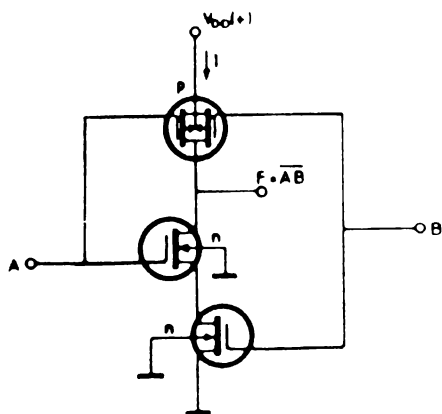
Anche se la famiglia PMOS è stata la prima, sembra che il suo sviluppo futuro sarà scarso.

## NMOS

Le porte fondamentali NMOS sono totalmente simili alle PMOS, varia solo la polarità dell'alimentazione.

I vantaggi degli NMOS rispetto ai PMOS sta semplicemente nella maggior velocità intrinseca di commutazione. Dato che i processi fisici sono dovuti al trascinamento dei portatori maggioritari e la mobilità degli elettroni è superiore a quella delle lacune, le risposte (tempi di commutazione) sono più rapide.

Anche se alcuni costruttori pensano che entro pochi anni i produttori di NMOS passeranno alla tecnologia  $I^2L$ , essendo questa meno costosa, attualmente lo sviluppo degli NMOS è molto elevato, grazie ai numerosi affinamenti che sono stati introdotti, e che ne migliorano sensibilmente le caratteristiche. I popolari microprocessori 8080 e MC6800 vengono costruiti con questa tecnologia.



| A | B | F |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

"0" = 0V  
 "1" =  $V_{DD}$

Fig. 3.-Porta NAND con CMOS.



## CMOS

La porta fondamentale NAND, a transistori complementari MOS è visibile in Fig. 3.

In Fig. 4 sono rappresentate le forme d'onda di uscita e della corrente in funzione dei segnali di ingresso.

Si osservi che circola corrente e, di conseguenza, si assorbe potenza dal-

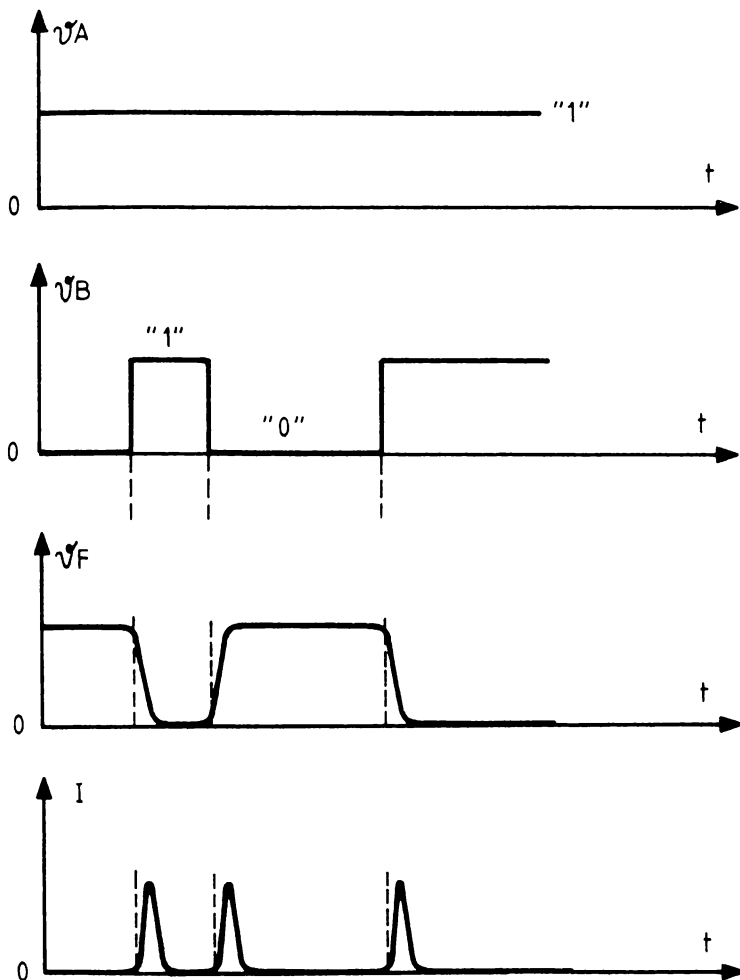


Fig. 4.-Diverse forme d'onda della porta NAND CMOS.

l'alimentazione, solo durante la transizione tra gli stati. In condizioni statiche il consumo, salvo le correnti di fuga, è nullo, in quanto i MOS tipo P, o gli N, sono interdetti.

Tra i vantaggi più rilevanti della famiglia CMOS si evidenziano:

- piccola dissipazione;
- ampio campo di alimentazione (tipicamente da 3 a 15 V non stabilizzati);
- ampi margini di rumore.

Gli svantaggi sono:

- senza tecniche raffinate occupano molto spazio sul chip;
- il processo di costruzione è relativamente complesso.

## TTL-SCHOTTKY

La TTL/S è una famiglia non saturata, e pertanto con elevata velocità di commutazione. Per non saturare i transistori viene integrato un diodo Schottky tra base e collettore (Fig. 5).

Poichè l'alluminio è un'impurezza accettrice (gruppo III), quando viene depositato su silicio di tipo N tende a formare una barriera metallo-semiconduttore, invece di un contatto ohmico. Pertanto il processo tecnologico di costruzione è relativamente semplice. La caduta di tensione con polarizzazione diretta è di circa 0,4 V e, poichè non si accumulano cariche nel diodo, questo risulta essere molto veloce.

Il circuito TTL/S ad alta velocità contiene, in tutti i transistori che si sa-

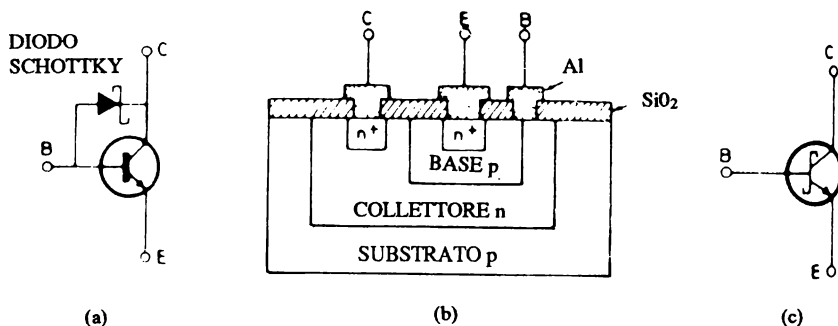


Fig. 5.-a) Transistore NPN con diodo Schottky di blocco. b) Sezione trasversale. c) Simbolo.

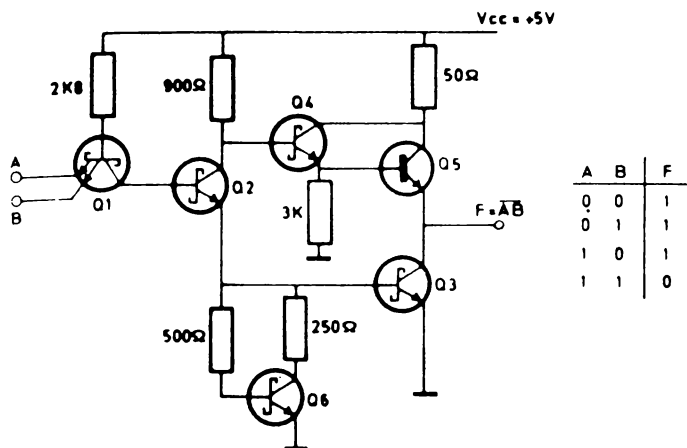


Fig. 6.-TTL-Schottky (NAND).

turano nel TTL standard, un diodo Schottky (Fig. 6).

Siccome Q5 non si può saturare, il diodo non è necessario.

Il principale inconveniente del TTL/S è l'elevata dissipazione; per questo motivo, e per competere con i CMOS, è stata sviluppata una versione leggermente diversa a bassa dissipazione (con perdita di velocità), detta Low Power Schottky TTL (LSTTL) (a basso consumo), molto utilizzata nei  $\mu P$  bipolari. Analogamente alle altre famiglie, è disponibile un'uscita addizionale ad alta impedenza, che caratterizza le versioni a tre stati (0, 1 e ad alta impedenza).

Tra i vantaggi vi sono:

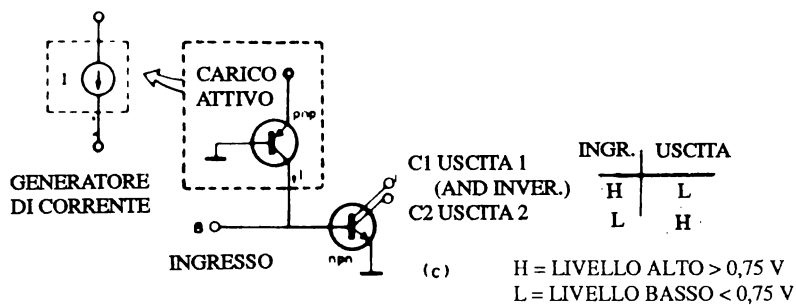
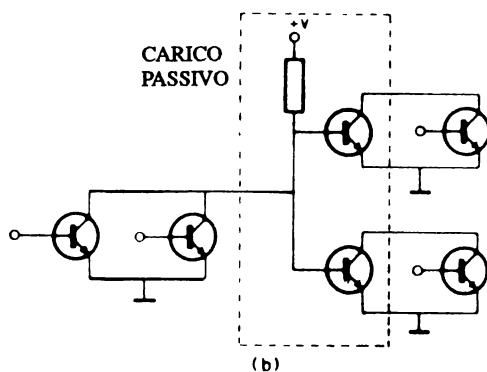
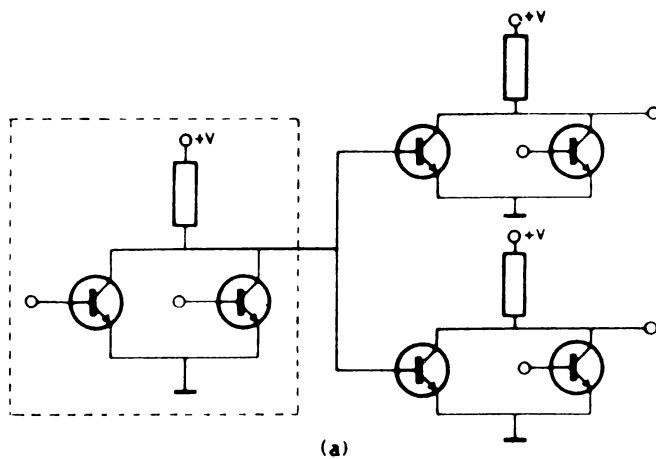
- elevata velocità;
- numerose periferiche in tecnologia TTL.

Inconvenienti:

- processo costruttivo complesso;
- necessità di sorgenti di alimentazione stabilizzate ( $5 \pm 0,5$  V);
- dissipazione elevata.

## ***I<sup>2</sup>L (Integrated Injection Logic)***

La famiglia  $I^2L$  (detta anche MTL-Merged Transistor Logic) (Fig. 7) derivata dalla DCTL, attualmente in disuso, presenta i vantaggi di una elevata



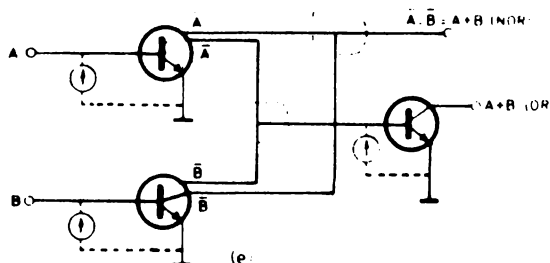
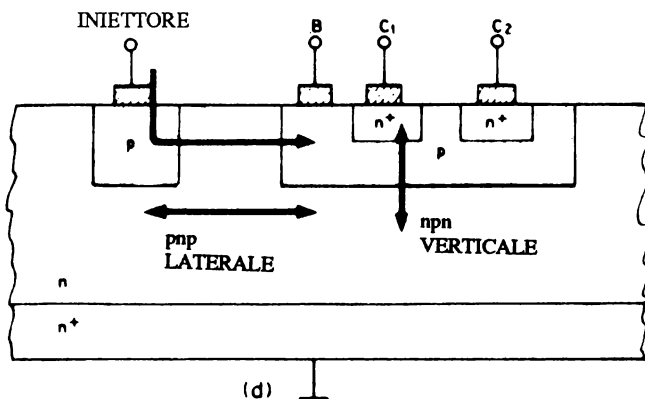


Fig. 7.-Logica ad iniezione integrata. a) Struttura DCTL. b) Transizione verso  $I^2L$ . c) Porta fondamentale  $I^2L$  (invertitore) in logica positiva. d) Sezione trasversale tipica. e) Esempio per generare le funzioni OR e NOR.

densità di integrazione, semplice processo di costruzione, e basso consumo.

Le sorgenti di corrente, i transistori laterali PNP, funzionano come carico. I transistori verticali NPN si comportano come invertitori.

Le porte  $I^2L$  vennero presentate nel 1972. Sia in DCTL che in  $I^2L$ , i transistori funzionano come invertitori, e le funzioni logiche (AND cablato) si ottengono collegando i collettori.

Non necessitando di resistenze integrate, la  $I^2L$  richiede poco spazio, per cui le capacità parassite sono molto piccole, e, essendo il salto logico ridotto ( $V_{BEON} - V_{CESat} \approx 0,6$  V), tali capacità si caricano e scaricano rapidamente, il che spiega il basso valore del fattore di merito ritardo-dissipazione.

In Fig. 7d) si può osservare che il funzionamento tipico del transistore integrato NPN in  $I^2L$  è opposto a quello dei transistori abituali, cioè l'iniezio-

ne di minoritari nella base si dirige verso la superficie. Fondamentalmente:

- il transistor NPN è saturato quando la sua base è fluttuante, cioè in circuito aperto ( $V_{BE} \approx 0,6 \text{ V}$ );
- il transistor NPN è interdetto quando la sua  $V_{BE} = 0 \text{ V}$ . ( $I_B$  andrà a massa e  $I_C = 0$ ).

La tavola della verità è visibile in Fig. 7c). La cellula isolata si comporta come un invertitore, dato che le uscite sono complemento dell'ingresso.

In Fig. 7e) è visibile un esempio di come si possano generare le funzioni OR e NOR, in logica positiva.

La famiglia  $I^2L$  è compatibile TTL. La tensione di alimentazione deve essere superiore a 0,85 V. Tipicamente il campo va da 1 a 15 V.

Si pensa che presto si raggiungeranno ritardi di propagazione di porta dell'ordine di 1 nsec, il che costituirà una forte concorrenzialità con la tecnologia attualmente più veloce: la ECL, che è quella dal consumo più elevato. Tra le migliori che si stanno sperimentando per rendere più rapida la  $I^2L$  ci sono:

- impiantazione ionica;
- diodo Schottky;
- isolamento passivo.

### ***ECL (emitter-coupled-logic)***

E' la più rapida di tutte le famiglie attualmente esistenti, in quanto i suoi transistori non entrano mai in saturazione.

Anche se ne esistono altre leggermente diverse, la porta fondamentale ECL è quella visibile in Fig. 8.

Il funzionamento del circuito è molto semplice. Siccome:

$$V_R = \frac{0 + 1}{2}$$

se l'ingresso è 0, Q1 sarà interdetto e Q2 funzionerà nella regione attiva; se l'ingresso è 1, la situazione è opposta. Le tensioni sui collettori di Q1 e Q2 giungono all'uscita tramite gli inseguitori di emettitore.

Tra i vantaggi si possono citare:

- elevata velocità;
- basso rumore.

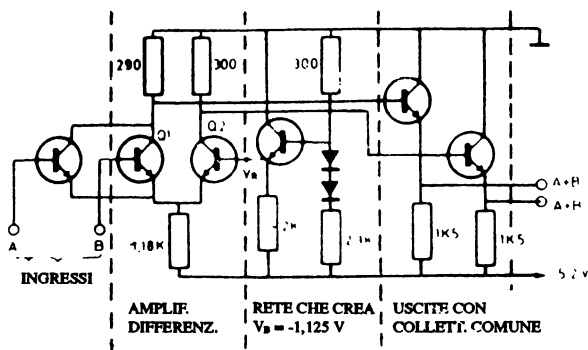


Fig. 8.-Porta fondamentale ECL. Le uscite sono funzioni OR e NOR degli ingressi.

Tra gli inconvenienti:

- elevata dissipazione;
- incompatibilità con le altre famiglie;
- processo di fabbricazione complesso;
- ridotta densità di integrazione.

Anche se sono state sinteticamente descritte le famiglie principali attualmente utilizzate nei  $\mu P$ , occorre non dimenticare che si stanno studiando nuove strutture di grande potenzialità per l'applicazione nei  $\mu P$ . A titolo di esempio, si citano le seguenti:

- $C^3L$  (Complementary Constant-Current Logic). Logica a corrente costante complementare;
- $T^3L$  (Incorpora uno stadio separatore aggiuntivo);
- Schottky  $I^2L$ ;
- 3D EFL (Triple-Diffused Emitter-Follower Logic). Logica ad inseguitore di emettitore a tripla diffusione.

### Miglioramenti tecnologici

Gli sforzi tecnologici sono incessantemente diretti a trarre il massimo vantaggio dalla tecnica LSI. I miglioramenti che vengono sperimentati sono

molto vari, ma hanno tutti i seguenti obiettivi:

- aumentare la densità di integrazione (strutture semplici e compatte);
- aumentare la velocità di commutazione (diminuzione delle capacità intrinseche e parassite, miglior allineamento);
- sorgenti di alimentazione semplici (unica alimentazione, bassa tensione non stabilizzata, tensione variabile, ecc.);
- massima immunità (al rumore, alle radiazioni, alle variazioni di temperatura...);
- minimo numero di terminali (interfacce semplici);
- maggior affidabilità;
- compatibilità con le altre famiglie logiche;
- maggior numero di circuiti senza difetti (maggior rendimento);
- prezzo ridotto.

La contropartita sta nella maggior complessità di costruzione e, in definitiva, nel costo.



## *MICROPROCESSORI A 8 BIT PIU' COMUNI (I)*



n questo capitolo saranno evidenziate le più importanti caratteristiche di alcuni microprocessori a 8 bit della INTEL, molto utilizzati.

### *Il microprocessore 8080*

Il microprocessore 8080 fu commercializzato dall'INTEL nel 1974; in realtà si tratta di una versione migliorata dell'8008 che è stato la prima CPU integrata a 8 bit. L'8080 è stato uno dei microprocessori di maggior attualità e, pertanto, dei più utilizzati.

### *Caratteristiche dell'8080*

Le sue caratteristiche più salienti sono:

- microprocessore a 8 bit;
- costruito con tecnologia MOS a canale N;
- capacità di indirizzamento di 64 Kbytes;
- frequenza tipica di clock: 500 KHz;
- 74 istruzioni di programmazione;
- possibili modi di indirizzamento: diretto esteso, immediato, implicito, diretto e indiretto tramite registro;
- alimentazione che necessita di tre diverse tensioni rispetto a



*Foto 1.-Chip 8084 prodotto da INTEL e NEC, con caratteristiche simili a quelle esposte nel testo.*

- massa: +5 V, -5 V, e +12 V;
- sistema di interrupt: possiede un terminale su cui si configurano tutte le richieste di interrupt, con 8 livelli di priorità.

### *Struttura interna dell' 8080*

Come indicato in Fig. 1, il  $\mu P$  8080 dispone di dieci registri interni per la sua programmazione.

- Un registro accumulatore a 8 bit, detto A, che è un registro di uso generale, attraverso cui entrano ed escono i dati esterni.
- Sei registri generali a 8 bit, che si possono considerare come sei registri da otto bit ciascuno, o come tre registri da sedici bit ciascuno. Il fatto di considerarli a otto o sedici bit viene determinato da ogni istruzione; esistono istruzioni che considerano i sei registri indipendentemente, e altre che li raggruppano a due a due per formare registri a sedici bit. Tali

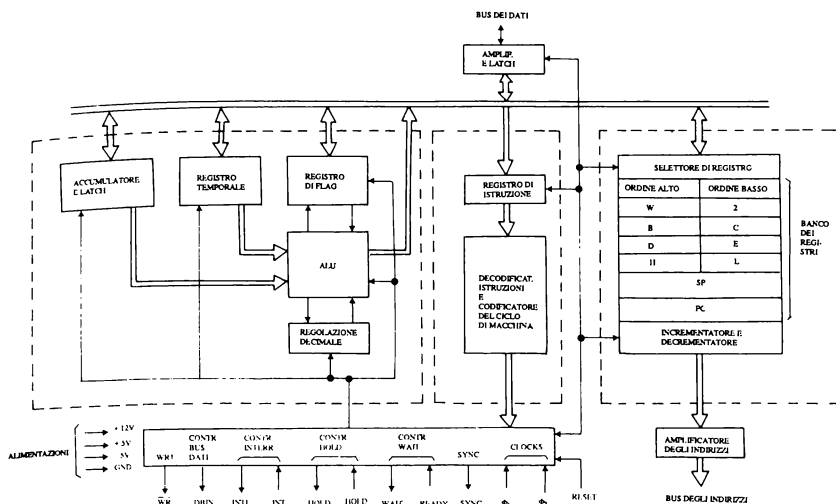


Fig. 1.-Diagramma a blocchi dell'8080.

registri si chiamano B, C, D, E, H, L. Le istruzioni che necessitano registri a sedici bit, li raggruppano a due a due nel modo seguente: B con C, D con E, e H con L.

- Un puntatore di stack a sedici bit per l'annotazione automatica dell'indirizzo di rientro dalle subroutine, detto SP.
- Un contatore di programma a sedici bit per indicare l'indirizzo dell'istruzione da eseguire, detto PC.
- Un registro di stato a otto bit che, come in tutti i microprocessori, rappresenta lo *stato*, e nel quale ognuno degli otto bit ha un proprio significato, il cui contenuto è relativo alle condizioni di funzionamento del  $\mu P$  in un dato istante. Il significato di ognuno di questi otto bit, rappresentati in Fig. 2, è il seguente:
  - Bit 0: carry, questo bit indica che il risultato dell'esecuzione dell'istruzione ha un riporto; quando esiste tale riporto il bit è zero.
  - Bit 1: non utilizzato, è sempre uno.
  - Bit 2: parità, questo bit indica se il dato elaborato è pari o dispari, quando il bit vale 1, il dato è dispari, se zero, pari.
  - Bit 3: non utilizzato, è sempre zero.
  - Bit 4: carry intermedio, questo bit indica il riporto tra mezzi bytes, tra il bit 3 e il 4 di un dato. Se un byte viene suddiviso

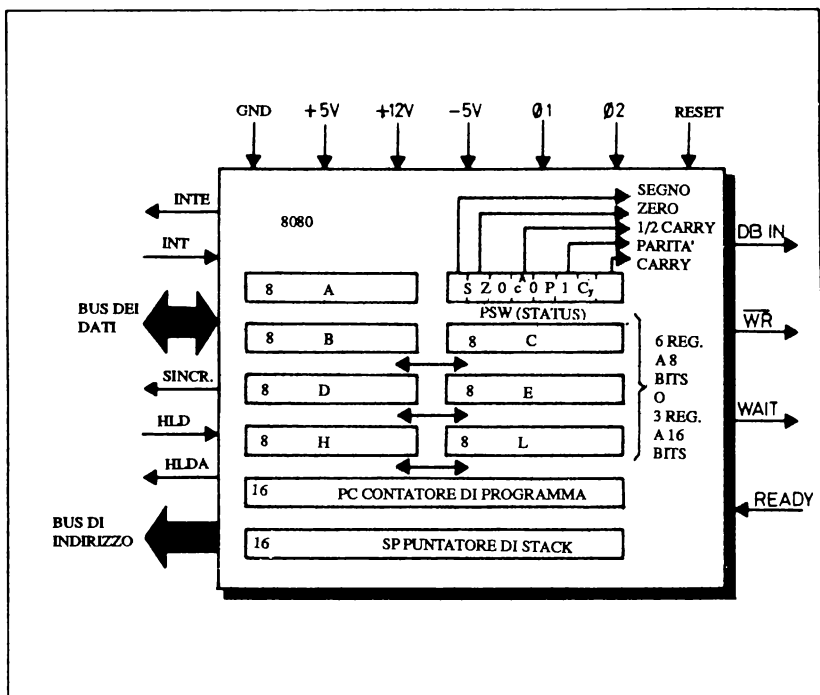


Fig. 2.-Registri interni del microprocessore 8080.

in due, si hanno due parti da 4 bit, ognuna di queste parti è in grado di rappresentare un esadecimale da 0 a F. Esistono casi in cui è necessario conoscere quando si passa da F a 0, incrementando un dato a 8 bit. Questo fatto è indicato dal bit di carry intermedio, che in tal caso assume valore 1.

- Bit 5: non utilizzato, è sempre zero.
- Bit 6: zero, questo bit vale zero se il risultato ottenuto in una operazione vale zero, altrimenti il bit vale 1.
- Bit 7: segno, indica il segno del risultato del dato elaborato; se il dato è negativo, il bit vale 1. (In esadecimale i numeri da 80 a FF sono negativi, cioè lo sono tutti quei dati in cui il bit 7 vale 1), mentre se il risultato è positivo (in esadecimale da 00 a 7F), il bit di segno è 0.

## Funzione dei terminali

In Fig. 3, dalla quale si può rilevare il numero del piedino che corrisponde a ciascuno dei 40 di cui dispone il microprocessore o chip, è indicata la distribuzione dei segnali sui terminali.

La funzione di ogni terminale è la seguente:

- A0-A15, da questi terminali esce verso l'esterno il bus degli indirizzi composto da 16 bit, per coprire il campo di 64 Kbytes di spazio indirizzabile.
- D0-D7, questi terminali costituiscono il bus dei dati attraverso cui circola il flusso di dati, da e verso l'esterno, quindi bidirezionale.

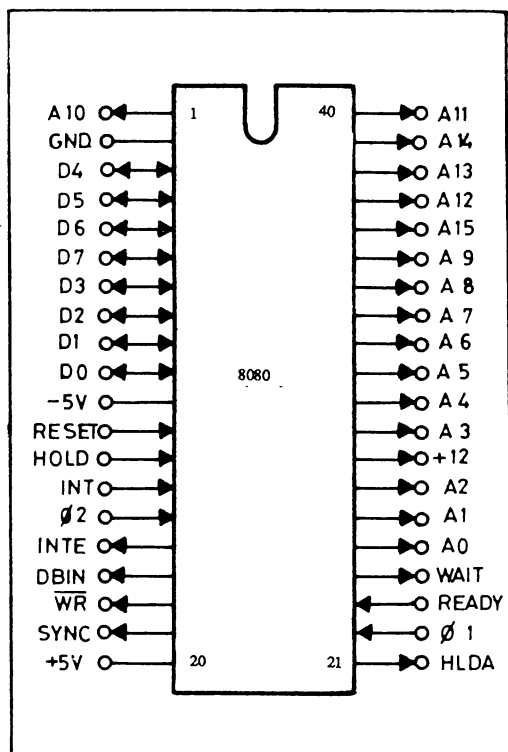


Fig. 3.-Distribuzione delle funzioni sui terminali del  $\mu P$  8080.

- RESET, segnale di azzeramento di tutti i registri interni del  $\mu P$ . Ciò avviene quando questo terminale è ad uno per almeno tre cicli di clock.
- HOLD, quando questo segnale di ingresso diventa 1, i bus degli indirizzi e dei dati passano in alta impedenza, vengono cioè *scollegati*, in modo che un altro  $\mu P$  o qualche periferica li usi per il tempo necessario.
- HLDA, tramite questo terminale il  $\mu P$  comunica all'esterno che è stato inserito un HOLD e, pertanto, i bus non sono disponibili, essendo ad alta impedenza.
- INT, segnale di richiesta di interrupt; tramite il corrispondente terminale, le periferiche possono sollecitare un interrupt dall'esterno; ciò avviene quando il terminale si porta a uno.
- INTE, segnale di accettazione dell'interrupt; tramite il terminale corrispondente a questo segnale il  $\mu P$  indica alla periferia richiedente l'interrupt, che lo stesso è stato accettato, portandosi a uno logico.
- DBIN, attraverso questo terminale il  $\mu P$  indica ai circuiti esterni che il bus dei dati è in ingresso, cioè in condizione di lettura dati, portandolo a uno.
- WR, tramite tale terminale il  $\mu P$  indica ai circuiti esterni che il bus dei dati è in scrittura, fatto che avviene quando il terminale diventa zero. I segnali DBIN e WR sono quelli che indicano alle periferiche e alla memoria se il  $\mu P$  è in lettura o scrittura.
- SYNC, terminale di sincronizzazione; il  $\mu P$  lo setta a uno quando inizia l'esecuzione di una nuova istruzione.
- ALIMENTAZIONE, il  $\mu P$  8080 necessita di tre tensioni di alimentazione +12 V, +5V, e -5 V, rispetto al punto di riferimento, cioè alla massa.
- SEGNALI DI CLOCK, questo  $\mu P$  necessita di due segnali di clock  $\Phi_1$  e  $\Phi_2$ , sfasati di  $180^\circ$ ; questi due terminali di clock non accettano livelli di tensione TTL, per cui occorre ricorrere a un circuito integrato come l'8224, progettato dal costruttore a tale scopo, per generare questi clock a partire da un oscillatore al quarzo.
- READY, segnale di pronto; settando a uno logico questo terminale, le periferiche o memorie lente, indicano al  $\mu P$  che sono pronte per mettersi in contatto con la CPU.

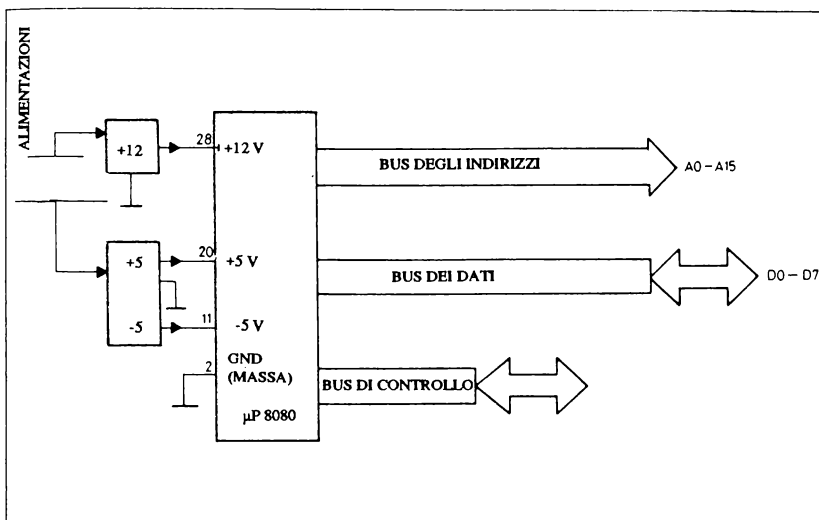
- WAIT, segnale di attesa; portando a uno logico questo terminale, il  $\mu P$  comunica all'esterno lo stato di attesa, che si verifica, ad esempio, quando si mette in contatto con una periferica lenta che ha fornito il segnale READY.

## *Sistema di collegamento dell' 8080*

Questo microprocessore non può essere direttamente collegato alla memoria e alle periferiche, ma necessita di due circuiti integrati esterni, progettati per interfacciarlo in alcuni dei suoi segnali e delle sue funzioni.

Questi circuiti sono: l'8224, utilizzato come generatore dei segnali di clock, reset e di altri sincronismi, e il circuito integrato 8228 utilizzato come amplificatore del bus dei dati, e come controllore di alcuni segnali, a cui viene fornita la forma adatta, per poter accedere alla memoria e alle periferiche.

Tutto il complesso, costituito dal  $\mu P$  e dai circuiti ausiliari è visibile in Fig. 6.



*Fig. 4.-Dettaglio dei bus e delle alimentazioni del  $\mu P$  8080.*

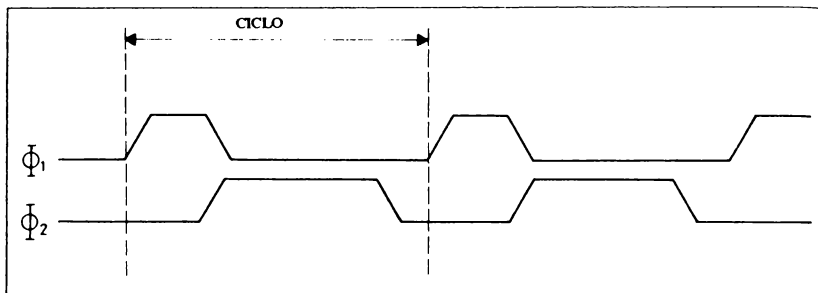


Fig. 5.-Dettaglio delle fasi del clock:  $\Phi_1$  e  $\Phi_2$ .

## 8224, generatore di clock

Questo circuito integrato ha il compito di generare le due fasi di clock  $\Phi_1$  e  $\Phi_2$ , a partire da un oscillatore interno, controllato esternamente da un cristallo di quarzo e da un circuito risonante LC, con il quale seleziona la frequenza fondamentale del quarzo, o un'armonica di tale fondamentale.

I valori assegnati a LC servono per aumentare il guadagno dell'armonica relativa alla frequenza adatta ai clock del microprocessore ( $\Phi_1$  e  $\Phi_2$ ); in tal modo il valore della frequenza del quarzo non è determinante.

Questo circuito integrato, inoltre, fornisce l'ampiezza minima del segna-

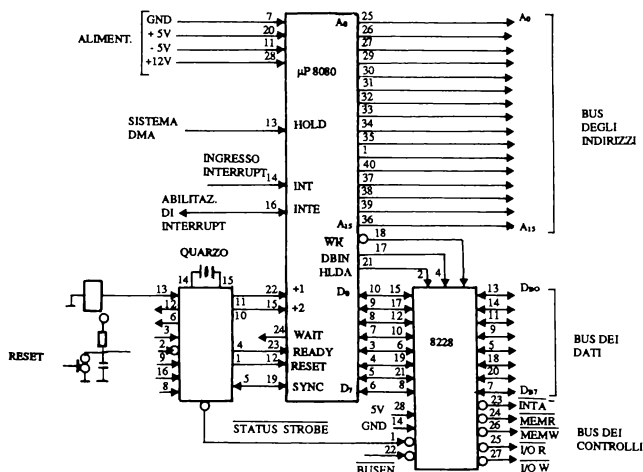


Fig. 6.-Schema dei collegamenti dell'8080 e dei suoi circuiti.



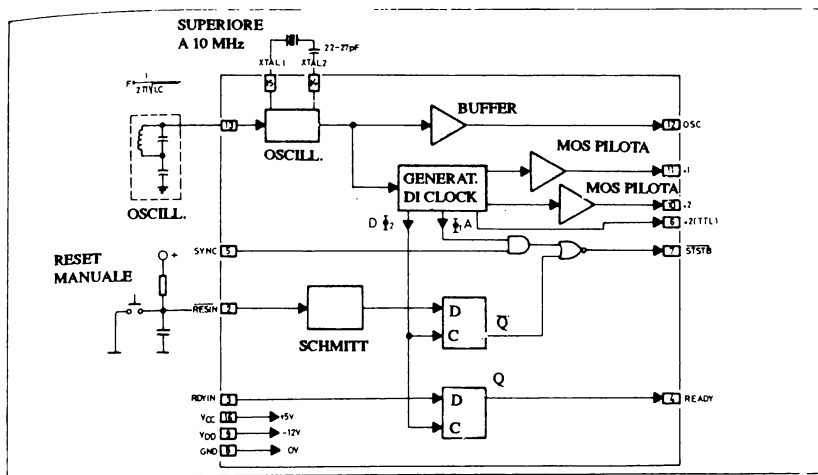


Fig. 7.-Schema interno a blocchi dell'8224.

le di reset del  $\mu P$ , a partire dal clock e dal reset manuale esterno.

Il circuito 8224 ha due altri compiti complementari, cioè fornire al  $\mu P$  il segnale di READY, in combinazione con il clock ( $\Phi_2$ ) ed il segnale di READY esterno.

Il secondo compito dell'8224 è di fornire il segnale STSTB (Status Strobe), che serve a dare passaggio allo *status* di funzionamento attraverso il bus dei dati verso il controller 8228, di cui si tratterà più avanti; il segnale STSTB è prodotto in combinazione con il segnale SYNC del microprocessore.

Si vedano le Figg. 6 e 7 per meglio comprendere quanto esposto.

## 8228, controller e amplificatore di bus

Questo circuito integrato ausiliare per il  $\mu P$  8080, gli fornisce un controllo completo sia del bus dei dati, sia dei segnali che consentono l'accesso alle memorie ed agli ingressi ed uscite speciali.

L'8228 è suddiviso, a livello di funzionamento, in due parti perfettamente differenziate, la prima controlla il bus dei dati, in entrambi i sensi, tramite i segnali HLDA e DBIN, provenienti dal  $\mu P$ , per determinare lo stato ad alta impedenza e di flusso dei dati, sia in scrittura (W), sia in lettura (R). Nello stesso istante in cui avviene l'attivazione di tali controlli, i bit subiscono un'amplificazione di corrente che permette un maggior FAN OUT del bus dei dati verso l'esterno del microprocessore.

La seconda parte di questo circuito integrato riceve il contenuto del bus dei dati elaborato dal  $\mu P$ , nel periodo di tempo in cui esiste il segnale SYNC nel primo ciclo di ogni istruzione; tale contenuto è lo stato di certi bit interni che indicano lo stato funzionale del  $\mu P$ .

Il significato di tali bit è interpretato, mediante una logica interna, dall'8228, che attiva in base ad essi i propri bit di uscita per controllo della memoria e delle periferiche.

I bit forniti dal  $\mu P$  hanno il seguente significato: D0=INTA è il segnale di riconoscimento dell'interrupt, D1=WO indica il modo di lettura o scrittura, D2=STK indica che il contenuto dello stack è nel bus degli indirizzi, D3=HLTA indica che il  $\mu P$  si è arrestato, D4=OUT indica che si sta eseguendo un'istruzione di uscita, D5=M1 indica che il  $\mu P$  è nel ciclo di ricerca, D6=INP indica che si sta eseguendo un'istruzione di ingresso, e D7=MEMR indica un'operazione di lettura della memoria.

## Uscite dell'8228

L'8228 esegue due compiti, il primo è di amplificare in entrambi i sensi il bus dei dati mediante i segnali di controllo WR, DBIN e HLDA; il secondo di generare i segnali di accesso alla memoria e alle periferiche prendendo l'in-

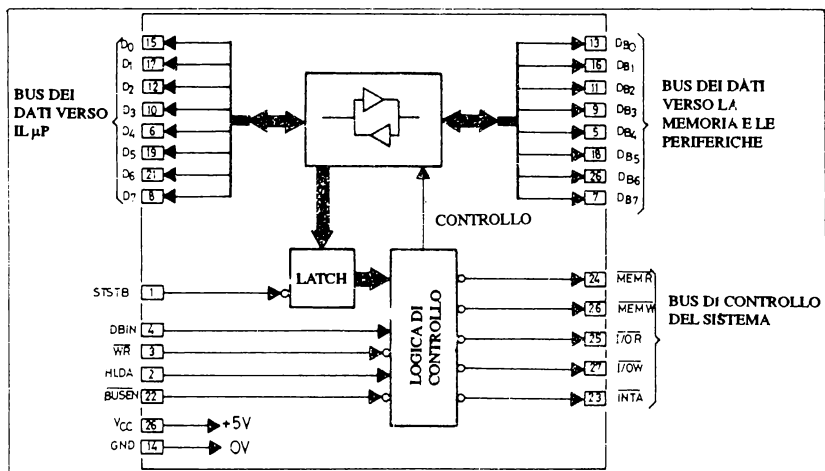


Fig. 8.-Schema interno a blocchi dell'8228.

formazione dal bus dei dati in un secondo ciclo tramite il segnale  $\overline{\text{STSTB}}$ , proveniente dal  $\mu\text{P}$ .

L'informazione fornita dal microprocessore attraverso il bus dei dati per questo secondo compito, attiva le uscite dell'8228, come è indicato nella tabella I. Queste uscite sono quelle che controllano e sincronizzano tutta la periferia della CPU.

- Il segnale  $\overline{\text{MEMR}}$  indica alla memoria l'operazione di lettura (R=READ), è un impulso basso o negativo, mediante il quale la memoria fornisce il dato contenuto nella cella indicata dall'indirizzo del corrispondente bus.
- Il segnale  $\overline{\text{MEMW}}$  indica alla memoria l'operazione di scrittura (W=WRITE), ed è anch'esso un impulso basso o negativo, mediante il quale viene scritto in memoria inserito il dato presente sul bus dei dati, nella cella corrispondente all'indirizzo indicato sul bus degli indirizzi.
- Il segnale  $\overline{\text{I/OR}}$  indica alle periferiche l'operazione di lettura; è un impulso negativo mediante il quale si trasferisce l'informazione da una periferica, ad esempio una porta (PORT), al microprocessore. Una porta è, in realtà, un insieme di LATCHES o bistabili progettati per essere collegati ad un  $\mu\text{P}$ .

|                |                        | <div> BUS DEI DATI<br/> NOME DEI BITS<br/> CICLO DI RICERCA<br/> MEM R<br/> MEM W<br/> STACK R<br/> STACK W<br/> R INGRESSO<br/> W USCITA<br/> INFA<br/> HALT<br/> HALT </div> |   |   |   |   |   |   |   |   |   |
|----------------|------------------------|--|---|---|---|---|---|---|---|---|---|
|                |                        | ①  | ② | ③ | ④ | ⑤ | ⑥ | ⑦ | ⑧ | ⑨ | ⑩ |
| D <sub>0</sub> | INTA                   | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| D <sub>1</sub> | $\overline{\text{WO}}$ | 1  | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| D <sub>2</sub> | STACK                  | 0  | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| D <sub>3</sub> | HLTA                   | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| D <sub>4</sub> | OUT                    | 0  | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| D <sub>5</sub> | M <sub>1</sub>         | 1  | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| D <sub>6</sub> | INP                    | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D <sub>7</sub> | MEMR                   | 1  | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |

Tabella I.-Interpretazione dei bits del bus dei dati attraverso l'8228 per produrre segnali di controllo della periferica.

- Il segnale  $\overline{I/O\overline{W}}$  indica alle periferiche l'operazione di scrittura, ed è un impulso negativo utilizzato per trasferire l'informazione dal  $\mu P$  alla periferica indirizzata.
- Il segnale INTA (Interrupt Acknowledge) è un'uscita che indica alla periferica, richiedente l'interrupt, che questo è stato accettato dal  $\mu P$ .
- Il segnale  $\overline{BUSEN}$  (Bus Enable Input) è un ingresso tramite il quale si portano ad alta impedenza le uscite suddette, per bloccare i controlli.

### *Circuito applicativo dell'8080*

Lo schema di un circuito applicativo è rappresentato nella Fig. 9, in cui sono indicati dettagliatamente i collegamenti dell'8080 ai suoi circuiti ausiliari, alla memoria, e alle periferiche.

Si osservi come l'8224 controlla il  $\mu P$  a partire dal clock generato da un quarzo, e dal reset ottenuto premendo il pulsante esterno. Si osservino anche i collegamenti dell'8228. Il bus dei dati si collega alla memoria, alle periferiche o all'esterno, mediante gli amplificatori interni; le uscite di lettura e scrittura, generate dall'8228, devono essere collegate alla ROM o EPROM, in cui si trova il firmware applicativo, alla RAM, alle porte e all'esterno.

Mentre i segnali di lettura e scrittura si collegano agli ultimi tre blocchi, quello di scrittura non viene inviato alla ROM, poichè quest'ultima può solo essere letta. I tre blocchi sono indirizzati da un decodificatore, i cui ingressi sono A13, A14 e A15, del bus degli indirizzi, in modo che l'indirizzo 0XXX attiva la ROM, 1XXX attiva la RAM, e 2XXX attiva la porta. I restanti indirizzi sono liberi.

Quando su un'uscita del decodificatore è presente, ad esempio, 2XXX significa che su tale uscita sono abilitati gli indirizzi compresi tra 2000 e 2FFF esadecimale. L'ingresso G del decodificatore è un ingresso di permesso, fornito dal microprocessore con il segnale HLDA, per ottenere il sincronismo tra periferiche e  $\mu P$ .

### *Tabella delle istruzioni dell'8080*

Il set di comandi per lo sviluppo del software per il microprocessore 8080 è costituito da 74 istruzioni. Queste possono, a loro volta, essere formate da uno, due e tre bytes. Le operazioni che sono in grado di effettuare, sono quel-

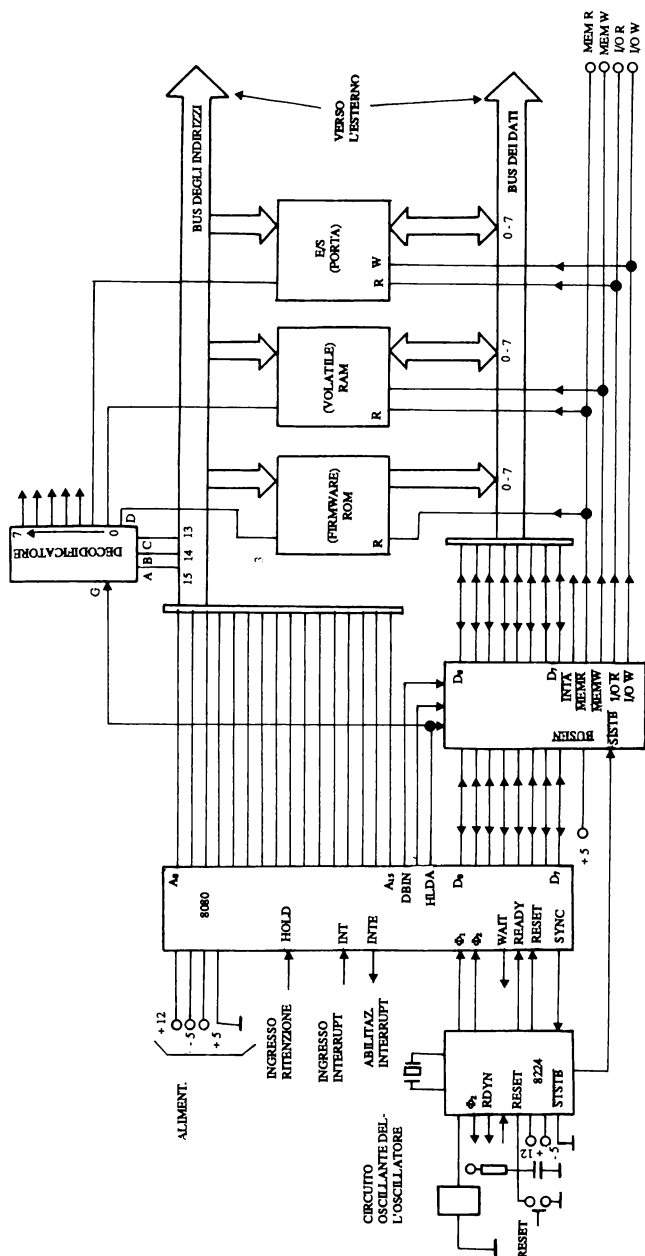


Fig. 9.-Schema completo di un circuito applicativo con il  $\mu P 8080$ .

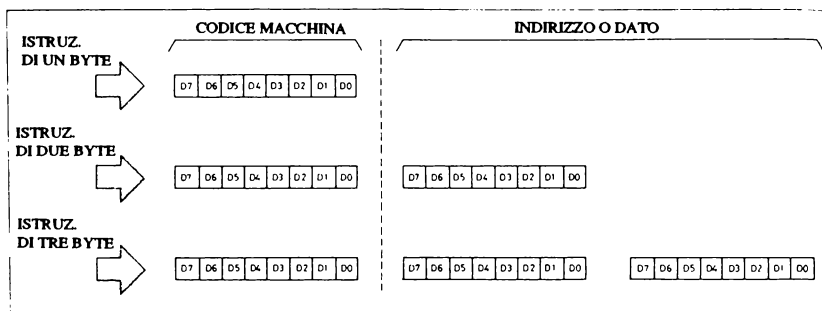


Fig. 10.-Formato delle istruzioni dell'8080. Il primo byte è occupato dal codice macchina o codice operativo, il secondo e terzo da un dato esadecimale o da un indirizzo.

le tipiche di quasi tutti i microprocessori.

Le loro denominazioni mnemoniche sono sigle delle funzioni in lingua inglese, di cui viene fornito un glossario.

### Nome delle istruzioni

L'insieme delle 74 istruzioni dell'8080, descritte nella tabella II, sono suddivise in cinque parti, raggruppate per funzioni simili.

- Nel gruppo di trasferimento vi sono:

- le istruzioni di caricamento LOAD, che caricano nell'accumulatore i registri o il puntatore dello STACK;
- le istruzioni di movimento MOVE, che scambiano tra loro il contenuto dei registri;
- le istruzioni di memorizzazione STORE, che trasferiscono il contenuto dell'accumulatore e dei registri nella memoria.

- Nel gruppo delle istruzioni aritmetiche si hanno:

- le istruzioni di controllo dell'interrupt, che lo permettono o no;
- le istruzioni di somma ADD, che eseguono la somma, con o senza riporto, tra i diversi registri e con la memoria; l'istruzione DAA opera in BCD;
- le istruzioni di decremento e incremento, che effettuano trasferimenti incrementando o decrementando il contenuto di una unità;

TABELLA II

| Mnemonico  | Descrizione                       | Operazione                  | Codice macchina             |   |   |   |   |   |   |   |   |    | Num. bytes<br>M | Num. cifre<br>T                            | Num. MC | Stato |   |    |  |  |
|------------|-----------------------------------|-----------------------------|-----------------------------|---|---|---|---|---|---|---|---|----|-----------------|--|---------|-------|---|----|--|--|
|            |                                   |                             | 7                           | 6 | 5 | 4 | 3 | 2 | 1 | 0 | S | Z  |                 |  |         | AC    | P | CY |  |  |
|            |                                   |                             | Istruzioni di trasferimento |   |   |   |   |   |   |   |   |    |                 |  |         |       |   |    |  |  |
| LDA        | Load Accumulator Direct           | (A) ← (byte 3) (byte 2)     | 0                           | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 3 | 4  | 13              | Queste istruzioni non influenzano lo stato |         |       |   |    |  |  |
| LDAX B     | Load Accumulator Indirect         | (A) ← (B)                   | 0                           | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 2  | 7               |  |         |       |   |    |  |  |
| LDAX D     | Load Accumulator Indirect         | (A) ← (D)                   | 0                           | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 2  | 7               |  |         |       |   |    |  |  |
| LHLD       | Load H and L Direct               | (L) ← (byte 3) (byte 2)     | 0                           | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 3 | 5  | 16              |  |         |       |   |    |  |  |
|            |                                   | (H) ← (byte 3) (byte 2+1))  |                             |   |   |   |   |   |   |   |   |    |                 |  |         |       |   |    |  |  |
| LXI B      | Load Immediate, Registers B and C | (B) ← (byte 3)              | 0                           | 0 | 0 | 0 | 0 | 0 | 1 | 3 | 3 | 10 |                 |  |         |       |   |    |  |  |
|            |                                   | (C) ← (byte 2)              |                             |   |   |   |   |   |   |   |   |    |                 |  |         |       |   |    |  |  |
| LXI D      | Load Immediate, Registers D and E | (D) ← (byte 3)              | 0                           | 0 | 0 | 1 | 0 | 0 | 1 | 3 | 3 | 10 |                 |  |         |       |   |    |  |  |
|            |                                   | (E) ← (byte 2)              |                             |   |   |   |   |   |   |   |   |    |                 |  |         |       |   |    |  |  |
| LXI H      | Load Immediate, Registers H and L | (H) ← (byte 3)              | 0                           | 0 | 1 | 0 | 0 | 0 | 1 | 3 | 3 | 10 |                 |  |         |       |   |    |  |  |
|            |                                   | (L) ← (byte 2)              |                             |   |   |   |   |   |   |   |   |    |                 |  |         |       |   |    |  |  |
| LXI SP     | Load Immediate, Stack Pointer     | (SPH) ← (byte 3)            | 0                           | 0 | 1 | 1 | 0 | 0 | 1 | 3 | 3 | 10 |                 |  |         |       |   |    |  |  |
|            |                                   | (SPL) ← (byte 2)            |                             |   |   |   |   |   |   |   |   |    |                 |  |         |       |   |    |  |  |
| MOV M, r   | Move to Memory                    | ((H) (L)) ← (r)             | 0                           | 1 | 1 | 1 | 0 | S | S | S | 1 | 2  | 7               |  |         |       |   |    |  |  |
| MOV r, M   | Move from Memory                  | ((H) (L)) ← (r)             | 0                           | 1 | D | D | D | 1 | 1 | 0 | 1 | 2  | 7               |  |         |       |   |    |  |  |
| MOV r1, r2 | Move Registers                    | (r1) ← (r2)                 | 0                           | 1 | D | D | D | S | S | S | 1 | 1  | 5               |  |         |       |   |    |  |  |
| MVI M      | Move to Memory Immediate          | ((H) (L)) ← (byte 2)        | 0                           | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 2 | 3  | 10              |  |         |       |   |    |  |  |
| MVI r      | Move immediate                    | (r) ← (byte 2)              | 0                           | 0 | D | D | D | 1 | 1 | 0 | 2 | 2  | 7               |  |         |       |   |    |  |  |
| SHLD       | Store H and L Direct              | ((byte 3) (byte 2)) ← (L)   | 0                           | 0 | 1 | 0 | 0 | 1 | 0 | 3 | 5 | 16 |                 |  |         |       |   |    |  |  |
|            |                                   | ((byte 3) (byte 2+1)) ← (H) |                             |   |   |   |   |   |   |   |   |    |                 |  |         |       |   |    |  |  |
| STA        | Store Accumulator Direct          | ((byte 3) (byte 2)) ← (A)   | 0                           | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 3 | 4  | 13              |  |         |       |   |    |  |  |
| STAX B     | Store Accumulator Indirect        | ((B)) ← (A)                 | 0                           | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 2  | 7               |  |         |       |   |    |  |  |
| STAX D     | Store Accumulator Indirect        | ((D)) ← (A)                 | 0                           | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 2  | 7               |  |         |       |   |    |  |  |
| XCHG       | Exchange H and L with D and E     | ((H) ← (D))<br>(L) ↔ (E)    | 1                           | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1  | 4               |  |         |       |   |    |  |  |

| Macro-<br>nimo     | Descrizione                | Operazione  | Codice macchina |   |   |   |   |   |   |   |   |   | Num.<br>bytes<br>M | Num.<br>cicE<br>T | Num.<br>MC | Stato |   |    |   |    |
|--------------------|----------------------------|---|-----------------|---|---|---|---|---|---|---|---|---|--------------------|-------------------|------------|-------|---|----|---|----|
|                    |                            |   |                 |   |   |   |   |   |   |   |   |   |                    |                   |            | S     | Z | AC | P | CY |
|                    |                            |   | 7               | 6 | 5 | 4 | 3 | 2 | 1 | 0 |   |   |                    |                   |            |       |   |    |   |    |
| Istruzioni logiche |                            |   |                 |   |   |   |   |   |   |   |   |   |                    |                   |            |       |   |    |   |    |
| ANA M              | AND Memory                 | $(A) \leftarrow (A) \wedge ((H) (L))$   | 1               | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1                  | 2                 | 7          | ↑     | ↑ | ↑  | ↑ | 0  |
| ANA r              | AND Register               | $(A) \leftarrow (A) \wedge (r)$   | 1               | 0 | 1 | 0 | 0 | 0 | S | S | S | S | 1                  | 2                 | 4          | ↑     | ↑ | ↑  | ↑ | 0  |
| ANI                | AND Immediate              | $(A) \leftarrow (A) \wedge (\text{byte } 2)$                                  | 1               | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 2                  | 7                 | ↑          | ↑     | ↑ | ↑  | ↑ | 0  |
| CMA                | Complement Accumulator     | $(A) \leftarrow (\bar{A})$  | 0               | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1                  | 4                 | 4          | ↑     | ↑ | ↑  | ↑ | 0  |
| CMC                | Complement Carry           | $(CY) \leftarrow (\bar{CY})$  | 0               | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1                  | 4                 | 4          | ↑     | ↑ | ↑  | ↑ | 0  |
| CMP M              | Compare Memory             | $(A) - ((H) (L))$   | 1               | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1                  | 2                 | 7          | ↑     | ↑ | ↑  | ↑ | 0  |
| CMP r              | Compare Register           | $(A) - (r)$   | 1               | 0 | 1 | 1 | 1 | 1 | S | S | S | S | 1                  | 4                 | 4          | ↑     | ↑ | ↑  | ↑ | 0  |
| CPI                | Compare Immediate          | $(A) - (\text{byte } 2)$  | 1               | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 2                  | 7                 | 7          | ↑     | ↑ | ↑  | ↑ | 0  |
| ORA M              | OR Memory                  | $(A) \leftarrow (A) \vee ((H) (L))$   | 1               | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 2                  | 7                 | 7          | ↑     | ↑ | ↑  | ↑ | 0  |
| ORA r              | OR Register                | $(A) \leftarrow (A) \vee (r)$   | 1               | 0 | 1 | 1 | 0 | S | S | S | S | 1 | 4                  | 4                 | 7          | ↑     | ↑ | ↑  | ↑ | 0  |
| ORI                | OR Immediate               | $(A) \leftarrow (A) \vee (\text{byte } 2)$                                    | 1               | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 2                  | 7                 | 7          | ↑     | ↑ | ↑  | ↑ | 0  |
| RAL                | Rotate Left through Carry  | $(A_n+1) \leftarrow A_n; (CY) \leftarrow (A_7)$<br>$(A_0) (CY)$               | 0               | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1                  | 4                 | 4          | ↑     | ↑ | ↑  | ↑ | 0  |
| RAR                | Rotate Right through Carry | $(A_n) \leftarrow (A_n+1); (CY) \leftarrow (A_0)$<br>$(A_7) \leftarrow (CY)$  | 0               | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1                  | 4                 | 4          | ↑     | ↑ | ↑  | ↑ | 0  |
| RLC                | Rotate Left                | $(A_n+1) \leftarrow (A_n); (A_0) \leftarrow (A_7)$<br>$(CY) \leftarrow (A_7)$ | 0               | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1                  | 4                 | 4          | ↑     | ↑ | ↑  | ↑ | 0  |
| RRC                | Rotate Right               | $(A_n) \leftarrow (A_n+1); (A_7) \leftarrow (A_0)$<br>$(CY) \leftarrow (A_0)$ | 0               | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1                  | 4                 | 4          | ↑     | ↑ | ↑  | ↑ | 0  |
| STC                | Set Carry                  | $(CY) \leftarrow 1$   | 0               | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1                  | 4                 | 4          | ↑     | ↑ | ↑  | ↑ | 0  |
| XRA M              | Exclusive OR Memory        | $(A) \leftarrow (A) \oplus ((H) (L))$   | 1               | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 2                  | 7                 | 7          | ↑     | ↑ | ↑  | ↑ | 0  |
| XRA r              | Exclusive OR Register      | $(A) \leftarrow (A) \oplus (r)$   | 1               | 0 | 1 | 0 | 1 | S | S | S | S | 1 | 4                  | 4                 | 7          | ↑     | ↑ | ↑  | ↑ | 0  |
| XRI                | Exclusive OR Immediate     | $(A) \leftarrow (A) \oplus (\text{byte } 2)$                                  | 1               | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 2                  | 7                 | 7          | ↑     | ↑ | ↑  | ↑ | 0  |



| Mnemonic | Descrizione                    | Operazioni  | Codice macchina        |   |   |   |   |   |   |   |   |   | Num. byte M | Num. cifre T | Num. MC | Stato |   |    |   |   |
|----------|--------------------------------|---|------------------------|---|---|---|---|---|---|---|---|---|-------------|--------------|---------|-------|---|----|---|---|
|          |                                |   | 7                      | 6 | 5 | 4 | 3 | 2 | 1 | 0 | S | Z |             |              |         | AC    | P | CY |   |   |
|          |                                |   | Istruzioni aritmetiche |   |   |   |   |   |   |   |   |   |             |              |         |       |   |    |   |   |
| ACI      | Add Immediate whit Carry       | $(A) \leftarrow (A) + (\text{byte } 2) + (CY)$                  | 1                      | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 2           | 7            | ↑       | ↑     | ↑ | ↑  | ↑ |   |
| ADC      | Add Memory whit Carry          | $(A) \leftarrow (A) + (H) (L) + (CY)$                           | 1                      | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1           | 2            | 7       | ↑     | ↑ | ↑  | ↑ | ↑ |
| ADC r    | Add Register whit Carry        | $(A) \leftarrow (A) + (r) + (CY)$                               | 1                      | 0 | 0 | 0 | 1 | S | S | S | 0 | 1 | 1           | 4            | 4       | ↑     | ↑ | ↑  | ↑ | ↑ |
| ADD      | Add Memory                     | $(A) \leftarrow (A) + (H) (L)$                                  | 1                      | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 2           | 7            | ↑       | ↑     | ↑ | ↑  | ↑ | ↑ |
| ADD r    | Add Register                   | $(A) \leftarrow (A) + (r)$                                      | 1                      | 0 | 0 | 0 | 0 | S | S | S | 0 | 1 | 1           | 4            | 4       | ↑     | ↑ | ↑  | ↑ | ↑ |
| ADI      | Add Immediate                  | $(A) \leftarrow (A) + (\text{byte } 2)$                         | 1                      | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 2           | 7            | ↑       | ↑     | ↑ | ↑  | ↑ | ↑ |
| DAA      | Decimal Adjust Accumulator     | 8bit number in Accumulator is converted to two 4 bit BCD digits | 0                      | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1           | 4            | 4       | ↑     | ↑ | ↑  | ↑ | ↑ |
| DAD      | Add B and C to H and L         | $(H) (L) \leftarrow (H) (L) + (B) (C)$                          | 0                      | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1           | 3            | 10      | ↑     | ↑ | ↑  | ↑ | ↑ |
| DAD D    | Add D and E to H and L         | $(H) (L) \leftarrow (H) (L) + (D) (E)$                          | 0                      | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1           | 3            | 10      |       |   |    |   |   |
| DAD H    | Add H and L to H and L         | $(H) (L) \leftarrow (H) (L) + (H) (L)$                          | 0                      | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1           | 3            | 10      |       |   |    |   |   |
| DAD SP   | Add Stack Pointer to H and L   | $(H) (L) \leftarrow (H) (L) + (SP)$                             | 0                      | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1           | 3            | 10      | ↑     | ↑ | ↑  | ↑ | ↑ |
| DCR      | Decrement Memory               | $(H) (L) \leftarrow ((H) (L)) - 1$                              | 0                      | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1           | 3            | 10      | ↑     | ↑ | ↑  | ↑ | ↑ |
| DCR r    | Decrement Register             | $(r) \leftarrow (r) - 1$  | 0                      | 0 | D | D | 1 | 0 | 1 | 0 | 1 | 1 | 1           | 1            | 5       |       |   |    |   |   |
| DCX      | Decrement Registers B and C    | $(B) (C) \leftarrow (B) (C) - 1$                                | 0                      | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1           | 1            | 5       |       |   |    |   |   |
| DCX D    | Decrement Registers D and E    | $(D) (E) \leftarrow (D) (E) - 1$                                | 0                      | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1           | 1            | 5       |       |   |    |   |   |
| DCX H    | Decrement Registers H and L    | $(H) (L) \leftarrow (H) (L) - 1$                                | 0                      | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1           | 1            | 5       |       |   |    |   |   |
| DCX SP   | Decrement Stack Pointer        | $(SP) \leftarrow (SP) - 1$                                      | 0                      | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1           | 1            | 5       |       |   |    |   |   |
| INR      | Increment Memory               | $(H) (L) \leftarrow ((H) (L)) + 1$                              | 0                      | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1           | 3            | 10      | ↑     | ↑ | ↑  | ↑ | ↑ |
| INR r    | Increment Register             | $(r) \leftarrow (r) + 1$  | 0                      | 0 | D | D | 1 | 0 | 1 | 0 | 0 | 1 | 1           | 1            | 5       |       |   |    |   |   |
| INX      | Increment Registers B and C    | $(B) (C) \leftarrow (B) (C) + 1$                                | 0                      | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1           | 1            | 5       |       |   |    |   |   |
| INX D    | Increment Registers D and E    | $(D) (E) \leftarrow (D) (E) + 1$                                | 0                      | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1           | 1            | 5       |       |   |    |   |   |
| INX H    | Increment Registers H and L    | $(H) (L) \leftarrow (H) (L) + 1$                                | 0                      | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1           | 1            | 5       |       |   |    |   |   |
| INX SP   | Increment Stack Pointer        | $(SP) \leftarrow (SP) + 1$                                      | 0                      | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1           | 1            | 5       |       |   |    |   |   |
| SBB      | Subtract Memory with Borrow    | $(A) \leftarrow (A) - ((H) (L)) - (CY)$                         | 1                      | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1           | 2            | 7       | ↑     | ↑ | ↑  | ↑ | ↑ |
| SBB r    | Subtract Register whit Borrow  | $(A) \leftarrow (A) - (r) - (CY)$                               | 1                      | 0 | 0 | 1 | 1 | S | S | S | 0 | 1 | 1           | 4            | 4       | ↑     | ↑ | ↑  | ↑ | ↑ |
| SBI      | Subtract Immediate with Borrow | $(A) \leftarrow (A) - (\text{byte } 2) - (CY)$                  | 1                      | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 2           | 7            | 7       | ↑     | ↑ | ↑  | ↑ | ↑ |
| SQB      | Subtract Memory                | $(A) \leftarrow (A) - (H) (L)$                                  | 1                      | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1           | 2            | 7       | ↑     | ↑ | ↑  | ↑ | ↑ |
| SUB      | Subtract Register              | $(A) \leftarrow (A) - (r)$                                      | 1                      | 0 | 0 | 1 | 0 | S | S | S | 0 | 1 | 1           | 4            | 4       | ↑     | ↑ | ↑  | ↑ | ↑ |
| SUI      | Subtract Immediate             | $(A) \leftarrow (A) - (\text{byte } 2)$                         | 1                      | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 2           | 7            | 7       | ↑     | ↑ | ↑  | ↑ | ↑ |

| Mnemonic            | Descrizione        | Operazione   | Codice macchina |   |   |   |   |   |   |   |   |   | Num. bytes<br>M | Num. cicli<br>T | Num. MC | Stato |   |    |  |
|---------------------|--------------------|--|-----------------|---|---|---|---|---|---|---|---|---|-----------------|-----------------|---------|-------|---|----|--|
|                     |                    |  | 7               | 6 | 5 | 4 | 3 | 2 | 1 | 0 | S | Z |                 |                 |         | AC    | P   | CY |  |
|                     |                    |  |                 |   |   |   |   |   |   |   |   |   |                 |                 |         |       |   |    |  |
| Istruzioni di salto |                    |  |                 |   |   |   |   |   |   |   |   |   |                 |                 |         |       |   |    |  |
| CALL                | Call Unconditional | $((SP)-1) \leftarrow (PCH)$<br>$((SP)-2) \leftarrow (PCL)$<br>$(SP) \leftarrow (SP)-2$<br>$(PC) \leftarrow (\text{byte } 3) (\text{byte } 2)$<br>If $CY=1$ ,<br>$((SP)-1) \leftarrow (PCH)$<br>$((SP)-2) \leftarrow (PCL)$<br>$(SP) \leftarrow (SP)-2$<br>$(PC) \leftarrow (\text{byte } 3) (\text{byte } 2)$<br>If $S=1$ ,<br>$((SP)-1) \leftarrow (PCH)$<br>$((SP)-2) \leftarrow (PCL)$<br>$(SP) \leftarrow (SP)-2$<br>$(PC) \leftarrow (\text{byte } 3) (\text{byte } 2)$<br>If $CY=0$ ,<br>$((SP)-1) \leftarrow (PCH)$<br>$((SP)-2) \leftarrow (PCL)$<br>$(SP) \leftarrow (SP)-2$<br>$(PC) \leftarrow (\text{byte } 3) (\text{byte } 2)$<br>If $Z=0$ ,<br>$((SP)-1) \leftarrow (PCH)$<br>$((SP)-2) \leftarrow (PCL)$<br>$(SP) \leftarrow (SP)-2$<br>$(PC) \leftarrow (\text{byte } 3) (\text{byte } 2)$<br>If $S=0$ ,<br>$((SP)-1) \leftarrow (PCH)$<br>$((SP)-2) \leftarrow (PCL)$<br>$(SP) \leftarrow (SP)-2$<br>$(PC) \leftarrow (\text{byte } 3) (\text{byte } 2)$ | 1               | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1               | 3               | 5       | 17    | Queste<br>istruzioni non<br>influenzano<br>lo stato |    |  |
| CC                  | Call on Carry      |  | 1               | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 3               | 3/5             | 11/17   |       |   |    |  |
| CM                  | Call on Minus      |  | 1               | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 3 | 3/5             | 3/5             | 11/17   |       |   |    |  |
| CNC                 | Call on No Carry   |  | 1               | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 3 | 3/5             | 3/5             | 11/17   |       |   |    |  |
| CNZ                 | Call on Not Zero   |  | 1               | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 3 | 3/5             | 3/5             | 11/17   |       |   |    |  |
| CP                  | Call on Positive   |  | 1               | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 3 | 3/5             | 3/5             | 11/17   |       |   |    |  |

| Mnemonic | Description         | Operazioni   | Codice macchina |   |   |   |   |   |   |   |   |     | Num. byte M | Num. cifre T                               | Num. MC | Stato |   |    |  |
|----------|---------------------|--|-----------------|---|---|---|---|---|---|---|---|-----|-------------|--|---------|-------|---|----|--|
|          |                     |  | 7               | 6 | 5 | 4 | 3 | 2 | 1 | 0 | S | Z   |             |  |         | AC    | P | CY |  |
|          |                     |  |                 |   |   |   |   |   |   |   |   |     |             |  |         |       |   |    |  |
| CPE      | Call on Parity Even | If P=1,<br>((SP)-1) ← (PCH)<br>((SP)-2) ← (PCL)<br>(SP) ← (SP)-2<br>(PC) ← (byte 3) (byte 2)<br>If P=0,<br>((SP)-1) ← (PCH)<br>((SP)-2) ← (PCL)<br>(SP) ← (SP)-2<br>(PC) ← (byte 3) (byte 2)<br>If Z=1,<br>((SP)-1) ← (PCH)<br>((SP)-2) ← (PCL)<br>(SP) ← (SP)-2<br>(PC) ← (byte 3) (byte 2)   | 1               | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 3 | 3/5 | 11/17       | Queste istruzioni non influenzano lo stato |         |       |   |    |  |
| CPO      | Call on Parity Odd  | ((SP)-1) ← (PCH)<br>((SP)-2) ← (PCL)<br>(SP) ← (SP)-2<br>(PC) ← (byte 3) (byte 2)<br>If P=0,<br>((SP)-1) ← (PCH)<br>((SP)-2) ← (PCL)<br>(SP) ← (SP)-2<br>(PC) ← (byte 3) (byte 2)<br>If Z=1,<br>((SP)-1) ← (PCH)<br>((SP)-2) ← (PCL)<br>(SP) ← (SP)-2<br>(PC) ← (byte 3) (byte 2)  | 1               | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 3 | 3/5 | 11/17       |  |         |       |   |    |  |
| CZ       | Call on Zero        | ((SP)-1) ← (PCH)<br>((SP)-2) ← (PCL)<br>(SP) ← (SP)-2<br>(PC) ← (byte 3) (byte 2)<br>If Z=1,<br>((SP)-1) ← (PCH)<br>((SP)-2) ← (PCL)<br>(SP) ← (SP)-2<br>(PC) ← (byte 3) (byte 2)  | 1               | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 3 | 3/5 | 11/17       |  |         |       |   |    |  |
| JC       | Jump on Carry       | If CY=1,<br>(PC) ← (byte 3) (byte 2)<br>If S=1,<br>(PC) ← (byte 3) (byte 2)<br>(PC) ← (byte 3) (byte 2)<br>If CY=0,<br>(PC) ← (byte 3) (byte 2)<br>If Z=0,<br>(PC) ← (byte 3) (byte 2)<br>If S=0,<br>(PC) ← (byte 3) (byte 2)<br>If P=1,<br>(PC) ← (byte 3) (byte 2)<br>If P=0,<br>(PC) ← (byte 3) (byte 2)<br>If Z=1,<br>(PC) ← (byte 3) (byte 2) | 1               | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 3 | 3   | 10          |  |         |       |   |    |  |
| JM       | Jump on Minus       | If S=1,<br>(PC) ← (byte 3) (byte 2)<br>(PC) ← (byte 3) (byte 2)<br>If CY=0,<br>(PC) ← (byte 3) (byte 2)<br>If Z=0,<br>(PC) ← (byte 3) (byte 2)<br>If S=0,<br>(PC) ← (byte 3) (byte 2)<br>If P=1,<br>(PC) ← (byte 3) (byte 2)<br>If P=0,<br>(PC) ← (byte 3) (byte 2)<br>If Z=1,<br>(PC) ← (byte 3) (byte 2)   | 1               | 1 | 1 | 1 | 0 | 1 | 0 | 3 | 3 | 10  |             |  |         |       |   |    |  |
| JMP      | Jump Unconditional  | (PC) ← (byte 3) (byte 2)<br>(PC) ← (byte 3) (byte 2)<br>If CY=0,<br>(PC) ← (byte 3) (byte 2)<br>If Z=0,<br>(PC) ← (byte 3) (byte 2)<br>If S=0,<br>(PC) ← (byte 3) (byte 2)<br>If P=1,<br>(PC) ← (byte 3) (byte 2)<br>If P=0,<br>(PC) ← (byte 3) (byte 2)<br>If Z=1,<br>(PC) ← (byte 3) (byte 2)  | 1               | 1 | 0 | 0 | 0 | 1 | 1 | 3 | 3 | 10  |             |  |         |       |   |    |  |
| JNC      | Jump on No Carry    | (PC) ← (byte 3) (byte 2)<br>If Z=0,<br>(PC) ← (byte 3) (byte 2)<br>If S=0,<br>(PC) ← (byte 3) (byte 2)<br>If P=1,<br>(PC) ← (byte 3) (byte 2)<br>If P=0,<br>(PC) ← (byte 3) (byte 2)<br>If Z=1,<br>(PC) ← (byte 3) (byte 2)  | 1               | 1 | 0 | 1 | 0 | 1 | 0 | 3 | 2 | 10  |             |  |         |       |   |    |  |
| JNZ      | Jump on Not Zero    | (PC) ← (byte 3) (byte 2)<br>If Z=0,<br>(PC) ← (byte 3) (byte 2)<br>If S=0,<br>(PC) ← (byte 3) (byte 2)<br>If P=1,<br>(PC) ← (byte 3) (byte 2)<br>If P=0,<br>(PC) ← (byte 3) (byte 2)<br>If Z=1,<br>(PC) ← (byte 3) (byte 2)  | 1               | 1 | 0 | 0 | 0 | 1 | 0 | 3 | 3 | 10  |             |  |         |       |   |    |  |
| JP       | Jump on Positive    | (PC) ← (byte 3) (byte 2)<br>If S=0,<br>(PC) ← (byte 3) (byte 2)<br>If P=1,<br>(PC) ← (byte 3) (byte 2)<br>If P=0,<br>(PC) ← (byte 3) (byte 2)<br>If Z=1,<br>(PC) ← (byte 3) (byte 2)   | 1               | 1 | 1 | 1 | 0 | 1 | 0 | 3 | 3 | 10  |             |  |         |       |   |    |  |
| JPE      | Jump on Parity Even | (PC) ← (byte 3) (byte 2)<br>If P=1,<br>(PC) ← (byte 3) (byte 2)<br>If P=0,<br>(PC) ← (byte 3) (byte 2)<br>If Z=1,<br>(PC) ← (byte 3) (byte 2)  | 1               | 1 | 1 | 0 | 1 | 0 | 1 | 3 | 3 | 10  |             |  |         |       |   |    |  |
| JPO      | Jump on Parity Odd  | (PC) ← (byte 3) (byte 2)<br>If P=0,<br>(PC) ← (byte 3) (byte 2)<br>If Z=1,<br>(PC) ← (byte 3) (byte 2)   | 1               | 1 | 1 | 0 | 0 | 1 | 0 | 3 | 3 | 10  |             |  |         |       |   |    |  |
| JZ       | Jump on Zero        | (PC) ← (byte 3) (byte 2)<br>If Z=1,<br>(PC) ← (byte 3) (byte 2)  | 1               | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 3 | 3   | 10          |  |         |       |   |    |  |

| Mnemonic | Description                | Operations  | Codier machines |   |   |   |   |   |   |   |   |     | Num. bytes<br>M | Num. clock<br>T                                     | Num. MC | State |   |    |  |
|----------|----------------------------|---|-----------------|---|---|---|---|---|---|---|---|-----|-----------------|---|---------|-------|---|----|--|
|          |                            |   | 7               | 6 | 5 | 4 | 3 | 2 | 1 | 0 | S | Z   |                 |   |         | AC    | P | CY |  |
|          |                            |   |                 |   |   |   |   |   |   |   |   |     |                 |   |         |       |   |    |  |
| PCHL     | H and L to Program Counter | (PCH) ← (H)<br>(PCL) ← (L)                                      | 1               | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1   | 5               | Queste<br>istruzioni non<br>influenzano<br>lo stato |         |       |   |    |  |
| RC       | Return on Carry            | If CY=1,<br>(PCL) ← ((SP))<br>(PCH) ← ((SP)+1)<br>(SP) ← (SP)+2 | 1               | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1/3 | 5/11            |   |         |       |   |    |  |
| RET      | Return                     | (PCL) ← ((SP))<br>(PCH) ← ((SP)+1)<br>(SP) ← (SP)+2             | 1               | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 3   | 10              |   |         |       |   |    |  |
| RM       | Return on Minus            | (PCH) ← ((SP)+1)<br>(SP) ← (SP)+2                               | 1               | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1/3 | 5/11            |   |         |       |   |    |  |
| RNC      | Return on No Carry         | If S=1,<br>(PCL) ← ((SP))<br>(PCH) ← ((SP)+1)<br>(SP) ← (SP)+2  | 1               | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1/3 | 5/11            |   |         |       |   |    |  |
| RNZ      | Return on Not Zero         | If CY=0,<br>(PCL) ← ((SP))<br>(PCH) ← ((SP)+1)<br>(SP) ← (SP)+2 | 1               | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1/3 | 5/11            |   |         |       |   |    |  |
| RP       | Return on Not Zero         | If Z=0,<br>(PCL) ← ((SP))<br>(PCH) ← ((SP)+1)<br>(SP) ← (SP)+2  | 1               | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1/3 | 5/11            |   |         |       |   |    |  |
| RPE      | Return on Parity Even      | If S=0,<br>(PCL) ← ((SP))<br>(PCH) ← ((SP)+1)<br>(SP) ← (SP)+2  | 1               | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1/3 | 5/11            |   |         |       |   |    |  |
| RPO      | Return on Parity Even      | If P=1,<br>(PCL) ← ((SP))<br>(PCH) ← ((SP)+1)<br>(SP) ← (SP)+2  | 1               | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1/3 | 5/11            |   |         |       |   |    |  |
|          | Return on Parity Odd       | If P=0,<br>(PCL) ← ((SP))<br>(PCH) ← ((SP)+1)<br>(SP) ← (SP)+2  | 1               | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1/3 | 5/11            |   |         |       |   |    |  |

| Mnemonic            | Descrizione    | Operazione   | Codice macchina |   |   |   |   |   |   |   |   |   | Num. bytes<br>M | Num. cicli<br>T | Num. MC | Stato                                      |   |    |  |  |
|---------------------|----------------|--|-----------------|---|---|---|---|---|---|---|---|---|-----------------|-----------------|---------|--|---|----|--|--|
|                     |                |  |                 |   |   |   |   |   |   |   |   |   |                 |                 |         |  |   |    |  |  |
|                     |                |  | 7               | 6 | 5 | 4 | 3 | 2 | 1 | 0 | S | Z |                 |                 |         | AC   | P | CY |  |  |
| Istruzioni di salto |                |  |                 |   |   |   |   |   |   |   |   |   |                 |                 |         |  |   |    |  |  |
| RST                 | Restart        | $((SP) - 1) \leftarrow (PCH)$<br>$((SP) - 2) \leftarrow (PCL)$<br>$(SP) \leftarrow (SP) - 2$<br>$(PC) \leftarrow 8 \cdot (NNIN)$<br>If $Z = 1$<br>$(PCL) \leftarrow ((SP))$<br>$(PCH) \leftarrow ((SP) + 1)$<br>$(SP) \leftarrow (SP) + 2$ | 1               | 1 | N | N | N | 1 | 1 | 1 | 1 | 1 | 1               | 3               | 11      | Queste istruzioni non influenzano lo stato |   |    |  |  |
| RZ                  | Return on Zero |  | 1               | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1/3             | 5/11            |         |  |   |    |  |  |

| Mnemonic                              | Descrizione        | Operazione  | Codice macchina |   |   |   |   |   |   |   | Num. bytes<br>M | Num. cicli<br>T | Num. MC | Stato |    |    |   |    |
|---------------------------------------|--------------------|---|-----------------|---|---|---|---|---|---|---|-----------------|-----------------|---------|-------|----|----|---|----|
|                                       |                    |   |                 |   |   |   |   |   |   |   |                 |                 |         |       |    |    |   |    |
|                                       |                    |   | 7               | 6 | 5 | 4 | 3 | 2 | 1 | 0 |                 |                 |         | S     | Z  | AC | P | CY |
| Istruzioni di controllo e dello stack |                    |   |                 |   |   |   |   |   |   |   |                 |                 |         |       |    |    |   |    |
| DI                                    | Disable Interrupts | The Interrupt system is disabled following the execution of the Di instruction. | 1               | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1               | 1               | 1       | 1     | 1  | 1  |   |    |
| EI                                    | Enable Interrupts  | The interrupt system is enabled following the execution of next instruction.    | 1               | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1               | 1               | 1       | 1     | 1  | 4  |   |    |
| HLT                                   | Halt               | Processor is stopped: registers and flags are unaffected.                       | 0               | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1               | 1               | 0       | 1     | 1  | 7  |   |    |
| IN                                    | Input              | (A) ← (data)  | 1               | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1               | 1               | 2       | 3     | 10 |    |   |    |

| Mnemonic  | Descrizione                            | Operazioni   | Codice macchina |   |   |   |   |   |   |   |   |   | Num. bytes<br>M | Num. ciclo<br>T | Num. MC | Stato |   |    |  |  |
|-----------|--|--|-----------------|---|---|---|---|---|---|---|---|---|-----------------|-----------------|---------|-------|---|----|--|--|
|           |  |  | 7               | 6 | 5 | 4 | 3 | 2 | 1 | 0 | S | Z |                 |                 |         | AC    | P | CY |  |  |
|           |  |  |                 |   |   |   |   |   |   |   |   |   |                 |                 |         |       |   |    |  |  |
| NOP       | No Operation                           | No operation is performed: registers and flags are unaffected.   | 0               | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1               | 1               | 4       |       |   |    |  |  |
| OUT POP B | Output Pop Registers B and C off Stack | (data) ← (A)<br>(C) ← (SP)<br>(B) ← (SP)+1<br>(SP) ← (SP)+2  | 1               | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 2 | 3               | 10              |         |       |   |    |  |  |
| POP D     | Pop Registers D and E off Stack        | (D) ← (SP)<br>(E) ← (SP)+1<br>(SP) ← (SP)+2  | 1               | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 3 | 10              |                 |         |       |   |    |  |  |
| POP H     | Pop Registers H and L off Stack        | (H) ← (SP)<br>(L) ← (SP)+1<br>(SP) ← (SP)+2  | 1               | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 3 | 10              |                 |         |       |   |    |  |  |
| POP PSW   | Pop Accumulator and Flags off Stack    | (CY) ← (SP) <sub>0</sub><br>(P) ← (SP) <sub>2</sub><br>(AC) ← (SP) <sub>4</sub><br>(Z) ← (SP) <sub>6</sub><br>(S) ← (SP) <sub>7</sub><br>(A) ← (SP)+1<br>(SP) ← (SP)+2 | 1               | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 3 | 10              |                 |         |       |   |    |  |  |
| PUSH B    | Push Registers B and C on Stack        | (SP)-1 ← (B)<br>(SP)-2 ← (C)<br>(SP) ← (SP)-2  | 1               | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 3 | 11              |                 |         |       |   |    |  |  |
| PUSH D    | Push Registers D and E on Stack        | (SP)-1 ← (D)<br>(SP)-2 ← (E)<br>(SP) ← (SP)-2  | 1               | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 3 | 11              |                 |         |       |   |    |  |  |
| PUSH H    | Push Registers H and L on Stack        | (SP)-1 ← (H)<br>(SP)-2 ← (L)<br>(SP) ← (SP)-2  | 1               | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 3 | 11              |                 |         |       |   |    |  |  |

| Mnemonico    | Descrizione   | Operazioni  | Codice macchina |   |   |   |   |   |   |   | Num. byte<br>M | Num. cicli<br>T | Num. MC | Stato |   |    |   |    |
|--------------|---|---|-----------------|---|---|---|---|---|---|---|----------------|-----------------|---------|-------|---|----|---|----|
|              |   |   | 7               | 6 | 5 | 4 | 3 | 2 | 1 | 0 |                |                 |         | S     | Z | AC | P | CY |
| PUSH PSW     | Push Accumulator and Flags on Stack                                 | $((SP)-1) \leftarrow (A)$<br>$((SP)-2)_0 \leftarrow (CY)$<br>$((SP)-2)_1 \leftarrow 1$<br>$((SP)-2)_2 \leftarrow (P)$<br>$((SP)-2)_3 \leftarrow 0$<br>$((SP)-2)_4 \leftarrow (AC)$<br>$((SP)-2)_5 \leftarrow 0$<br>$((SP)-2)_6 \leftarrow (Z)$<br>$((SP)-(SP)_7) \leftarrow (S)$<br>$(SP) \leftarrow (SP)-2$<br>$(SP) \leftarrow (H) (L)$<br>$(L) \leftrightarrow ((SP))$<br>$(H) \leftrightarrow ((SP)+1)$ | 1               | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1              | 3               | 11      | .     | . | .  | . |    |
| SPHL<br>XTHL | Move H and L to Stack Pointer<br>Exchange Top of Stack with H and L |   | 1               | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1              | 1               | 5       | 18    | . | .  | . | .  |

## NOTE:

- a)  $Z=1 \text{ se } (A)=(H)(L)$   
 $CY=1 \text{ se } (A)>(H)(L)$   
b)  $Z=1 \text{ se } (A)=0$   
 $CY=1 \text{ se } (A)>0$   
c)  $Z=1 \text{ se } (A)=(\text{byte } 2)$   
 $CY=1 \text{ se } (A)>(\text{byte } 2)$

## SIMBOLI E ABBREVIAZIONI

I seguenti simboli e abbreviazioni sono stati usati nella descrizione delle istruzioni del 8080: A=Registro A (Accumulatore), B=Registro B, C=Registro C, D=Registro D, E=Registro E, H=Registro H, L=Registro L, DDD:SSS=il bit padrone designa uno dei registri A, B, C, D, E, H, L. (DDD=Destinazione, SSS=Origine);

## DDD o SSS NOME DEL REG.

- 111 A  
000 B  
001 C  
010 D  
011 E  
100 H  
101 L

Byte 2: Il secondo byte dell'istruzione.

Byte 3: Il terzo byte dell'istruzione.

Riporte: 8-bit di indirizzo per un circuito di ingresso/uscita.

r, r1, r2: Uno dei registri A, B, C, D, E, H, L.

## SIMBOLI E SIGNIFICATO

PC = Registro contatore di programma a 16 bit (PCH e PCL), suddiviso in due gruppi da 8 bit. SP = Registro puntatore di Stack a 16 bit (SPH e SPL), suddiviso in due gruppi da 8 bit. ( ) = Il contenuto della memoria o del registro si racchiude tra parentesi.  $\leftarrow$  = E' sostituito da.  $\wedge$  = Logica AND.  $\vee$  = OR Esclusivo.  $\vee$  = OR inclusivo. + = Somma. - = Sottrazione, complemento a due. \* = Moltiplicazione.  $\leftrightarrow$  = Scambio. - = Complemento, esempio (A), n = numeri da 0 a 7, NNN = Rappresentazione in binario da 000 a 111, da 0 a 7, rispettivamente. = = Non influenzato. 0 = Reset. 1 = Set. x = Indifferente.  $\uparrow$  = Flags influenzati.

Tabella II.-Tabella delle istruzioni del  $\mu P$  8080.

- le istruzioni di differenza SUBTRACT, che effettuano una sottrazione tra registri, con o senza *borrow* (cioè prestito) e tra questi e la memoria.
- Nel gruppo delle istruzioni logiche esistono:
  - le istruzioni che eseguono AND, OR, e OR ESCLUSIVO, tra la memoria e l'accumulatore, o tra questo e i registri; l'istruzione di complemento esegue un'inversione del contenuto dell'accumulatore;
  - le istruzioni di confronto che effettuano un confronto logico bit a bit tra l'accumulatore e i registri; le istruzioni di rotazione ROTATE, che eseguono una rotazione, verso destra o sinistra, nei diversi registri, con o senza carry.
- Nel gruppo delle istruzioni di salto o ramificazione si trovano le istruzioni di salto JUMP con o senza condizione positiva, negativa, di zero, ecc., in base al contenuto dei bit dello status.
- Le istruzioni di ritorno RETURN, che servono per uscire da una subroutine, con o senza condizione di zero, negativa, positiva, ecc., secondo lo status.
- Tra le miscellanee vi sono:
  - le istruzioni HALT e NOP;
  - le istruzioni di uso dello STACK, che caricano il contenuto dei vari registri nello STACK.

### *Tabella delle istruzioni in funzione del codice macchina*

Nella tabella III sono contenute tutte le istruzioni già note, ma in base al loro codice macchina, per cui un'istruzione si cerca partendo dal suo codice esadecimale, in modo che il primo digit corrisponda alla colonna (I) e il secondo alla riga (J), nel punto di intersezione tra riga e colonna esiste l'istruzione cercata.

Questo metodo di ricerca di un'istruzione viene utilizzato per disassemblare pazientemente un programma di cui si conosce il codice macchina.

### *Il microprocessore 8085*

Questo microprocessore fu commercializzato dall'Intel dopo l'8080. Il  $\mu$ P 8085 ha quindi beneficiato di tutta l'evoluzione tecnica degli ultimi anni,



| $\begin{matrix} I \\ \backslash \\ J \end{matrix}$ | 0   | 1      | 2      | 3        | 4       | 5        | 6      | 7      | 8      | 9      | A      | B       | C      | D        | E       | F      |
|--|-----|--------|--------|----------|---------|----------|--------|--------|--------|--------|--------|---------|--------|----------|---------|--------|
| 0  | NOP | LXI    | B STAX | B INX    | B INR   | B DCR    | B MVI  | B RLC  |        | DAD    | B LDAX | B DCX   | B INR  | C DCR    | C MVI   | C RRC  |
| 1  |     | LXI    | D STAX | D INX    | D INR   | D DCR    | D MVI  | D RAL  |        | DAD    | D LDAX | D CDX   | D INR  | E DCR    | E MVI   | E RAR  |
| 2  |     | LXI    | H SHLD | H INX    | H INR   | H DCR    | H MVI  | H DAA  |        | DAD    | H LHLD | adr DCX | H INR  | L DCR    | L MVI   | L CMA  |
| 3  |     | LXI    | SP STA | adr INX  | SP INR  | M DCR    | M MVI  | M STC  |        | DAD    | SP LDA | adr DCX | SP INR | A DCR    | A MVI   | A CMC  |
| 4  | MOV | BB MOV | BC MOV | BD MOV   | BE MOV  | BH MOV   | BL MOV | M MOV  | BA MOV | CB MOV | CC MOV | CD MOV  | CE MOV | CH MOV   | CL MOV  | CM MOV |
| 5  | MOV | DB MOV | DC MOV | DD MOV   | DE MOV  | DH MOV   | DL MOV | DM MOV | DA MOV | EB MOV | EC MOV | ED MOV  | EE MOV | EH MOV   | EL MOV  | EM MOV |
| 6  | MOV | HB MOV | HC MOV | HD MOV   | HE MOV  | HH MOV   | HL MOV | HM MOV | HA MOV | LB MOV | LC MOV | LD MOV  | LE MOV | LH MOV   | LL MOV  | LM MOV |
| 7  | MOV | MB MOV | MC MOV | MD MOV   | ME MOV  | MH MOV   | ML MOV | HLT    | MA MOV | AB MOV | AC MOV | AD MOV  | AE MOV | AH MOV   | AL MOV  | AM MOV |
| 8  | ADD | B ADD  | C ADD  | D ADD    | E ADD   | H ADD    | L ADD  | M ADD  | A ADD  | B ADD  | C ADD  | D ADD   | E ADD  | H ADD    | L ADD   | M ADD  |
| 9  | SUB | B SUB  | C SUB  | D SUB    | F SUB   | H SUB    | L SUB  | M SUB  | A SUB  | B SUB  | C SUB  | D SUB   | E SUB  | H SUB    | L SUB   | M SUB  |
| A  | ANA | B ANA  | C ANA  | D ANA    | E ANA   | H ANA    | L ANA  | M ANA  | A ANA  | B ANA  | C ANA  | D ANA   | E ANA  | H ANA    | L ANA   | M ANA  |
| B  | ORA | B ORA  | C ORA  | D ORA    | E ORA   | H ORA    | L ORA  | M ORA  | A ORA  | B ORA  | C ORA  | D ORA   | E ORA  | H ORA    | L ORA   | M ORA  |
| C  | RNC | POP    | B INZ  | adr IMP  | adr CNZ | B PUSH   | ADI    | O RST  | A CMP  | B CMP  | C CMP  | D CMP   | E CMP  | H CMP    | L CMP   | M CMP  |
| D  | RNC | POP    | H INC  | adr OUT  | CNC     | adr PUSH | D SUI  | RST 2  | RZ     | RET    | JZ     | adr IN  | CZ     | adr CALL | adr ACI | RST 1  |
| E  | RPO | POP    | H PO   | adr NTHL | CPO     | adr PUSH | H ANI  | RST 4  | RPE    | SPHL   | JC     | adr JPE | CC     | adr XCHG | SBI     | RST 3  |
| F  | RP  | POP    | JP     | DI       | CP      | PUSH     | ORI    | RST    | RM     |        | JM     | adr EI  | CM     | adr      | CPI     | RST 4  |

Tabella III.-Tabella delle istruzioni dell'8080, in base al codice esadecimale.

Esempio: I=C, J=3, istruzione C3=JMP adr (istruzione di salto).

nel campo dell'integrazione. L'8085 è completamente compatibile, a livello logico, con l'8080, e molto migliorato per quanto riguarda gli interrupts. L'8085 contiene la circuiteria ausiliaria che era necessaria all'8080, come i circuiti integrati 8224 e 8228, per cui è un microprocessore molto utilizzato, e gode di grande popolarità nei circuiti tecnici professionali.

### *Caratteristiche dell'8085*

Il  $\mu$ P 8085 possiede le seguenti caratteristiche tecniche e funzionali:

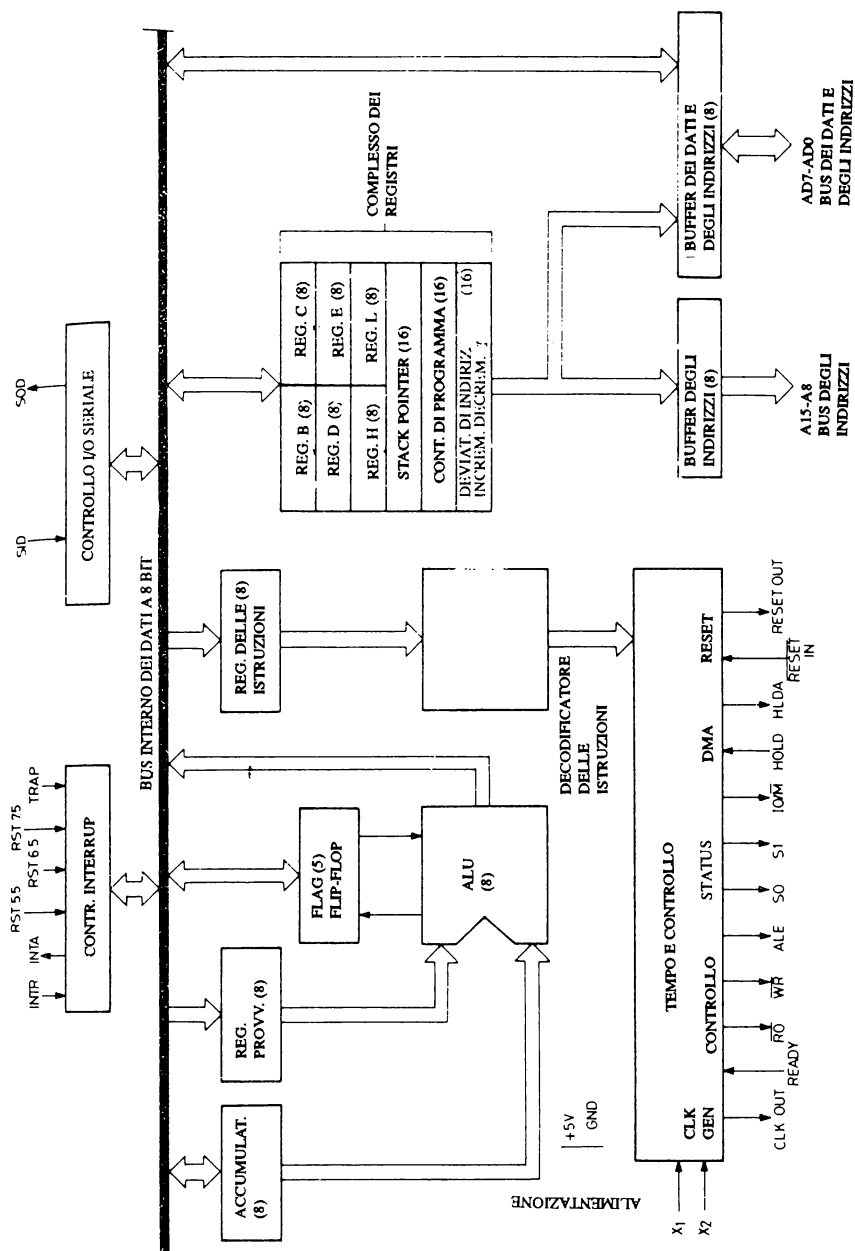
- è realizzato su un solo CHIP in tecnologia NMOS;
- ha una capacità di indirizzamento di 64 Kbytes;
- il periodo di clock è di 1,3  $\mu$ sec nell'8085 A, e di 0,8  $\mu$ sec nella versione A-2;
- prevede un'unica alimentazione a 5 V;
- dispone di 78 istruzioni (quattro in più rispetto all'8080);
- prevede gli indirizzamenti: diretto esteso, immediato, implicito, diretto e indiretto tramite registro;
- il sistema di interrupt dispone di un interrupt non mascherabile (TRAP), e altri livelli di interrupts vettorializzati e mascherabili.

### *Configurazione interna dell'8085*

La configurazione interna è quella classica di tutti i microprocessori, come mostrato in Fig. 11.

In questo microprocessore si distinguono le seguenti unità funzionali:

- il blocco dei registri;
- l'unità di controllo generale;
- l'unità di controllo degli interrupts;
- l'unità di controllo di input/output seriale;
- i registri sono i seguenti:
  - i registri di uso generale sono sei, ad essi ci si può riferire mediante le istruzioni del programma. Possono essere utilizzati come registri da otto bit, o come tre registri da sedici bit ciascuno in funzione del tipo di istruzione da eseguire (ogni istruzione utilizza uno o l'altro metodo). Tali registri si chiamano B, C, D, E, H, L. Per formare registri a sedici bit, si



raggruppano a due a due nel modo seguente: B con C, D con E, e H con L.

- il contatore di programma (PC) è un registro a sedici bit, contenente l'indirizzo in cui si trova la successiva istruzione da eseguire che si autoincrementa ad ogni ciclo di lavoro.
- il puntatore di stack (SP), registro a 16 bit che contiene l'indirizzo dell'ultima posizione occupata dello stack. Lo stack pointer è caricato con l'indirizzo dello stack, per il quale si può utilizzare qualsiasi posizione della memoria; il suo contenuto viene decrementato ogni volta che un indirizzo è memorizzato nello stack, e incrementato ogni volta che un indirizzo viene prelevato dallo stesso stack.
- registri temporanei a otto bit ad uso interno del  $\mu P$ , e quindi inaccessibili al programmatore.
- registro accumulatore a otto bit, detto A.
- registro di stato (status) a otto bit, dei quali solo cinque

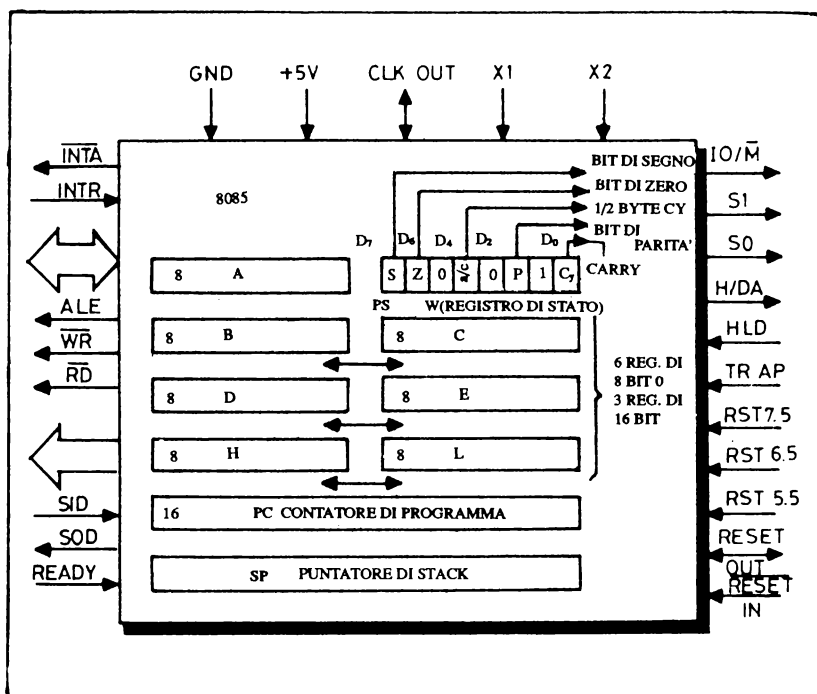


Fig. 12.-Registri interni del  $\mu P$  8085.

vengono utilizzati con il seguente significato:

D0=Cy, è l'indicatore di carry, utilizzato nelle operazioni aritmetiche; se esiste riporto vale uno, altrimenti zero. Solo se l'operazione è una sottrazione il carry funziona inversamente, e viene resettato a zero se esiste riporto.

D2=P, è l'indicatore di parità, settato a uno se la parità (numero di bit a uno) dell'accumulatore è pari, altrimenti vale zero.

D4=AC (Auxiliar Carry), questo indicatore è settato a uno se esiste un riporto sul bit quattro del dato elaborato, altrimenti a zero.

D6=Z (Zero), indicatore che vale uno se il risultato di un'operazione vale zero, se tale risultato è diverso da zero, viene resettato a zero.

D7=S, indicatore che è riferito al segno del dato nell'accumulatore; se vale uno, il dato è negativo, se zero, il dato è positivo.

## *Unità di controllo dell'8085*

Questo blocco interno del microprocessore è costituito dai seguenti elementi:

- il registro di istruzione da otto bit a cui viene inviato il primo ottetto o byte dell'istruzione, che rappresenta il codice dell'operazione;
- il decodificatore di istruzioni, che preleva il contenuto del registro di istruzione per determinare l'operazione da effettuare; l'uscita del decodificatore controlla tutti i registri, l'ALU, e i bus dei dati e degli indirizzi;
- il sequenziatore, che genera le microistruzioni di lettura o scrittura, in funzione dell'informazione fornita dal decodificatore;
- il generatore dei segnali interni di clock, incorporato nel  $\mu P$  8085, che richiede solo l'utilizzo di un cristallo di quarzo per stabilire la sequenza delle operazioni. La circuiteria interna del clock genera due segnali  $\Phi_1$  e  $\Phi_2$  che attivano il funzionamento del microprocessore, e non sono disponibili all'esterno.

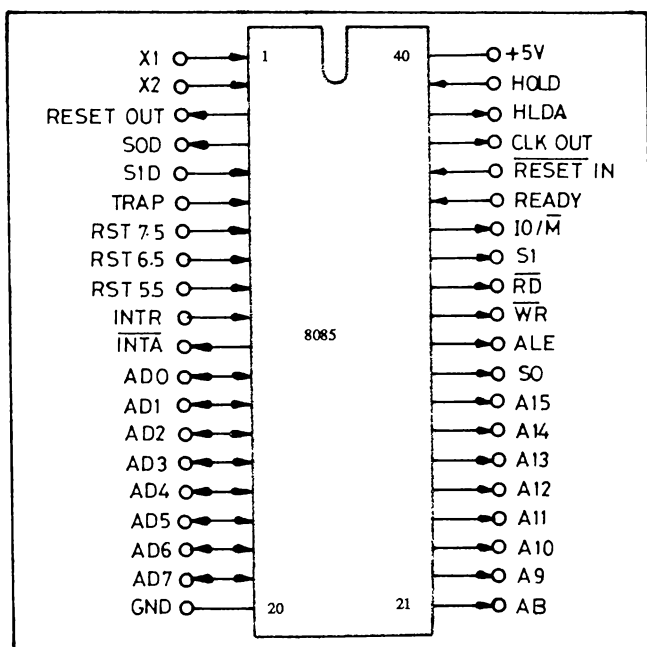


Fig. 13.-Distribuzione delle funzioni sui terminali del  $\mu P$  8085.

### Nomenclatura dei terminali dell'8085

Anche il microprocessore 8085 è contenuto nel classico contenitore a 40 piedini. In Fig. 13 si può rilevare la corrispondenza tra la funzione e il numero del terminale. Le funzioni sono le seguenti:

- A8-A15, è il bus degli indirizzi: dei 16 bit che compongono il bus per indirizzare il campo di 64 Kbytes, solo otto escono verso l'esterno, gli altri da A0 ad A7 escono al momento opportuno dal bus dei dati.
- AD0-AD7, questi terminali costituiscono il bus dei dati attraverso cui circola il flusso di dati, da e verso l'esterno, e la parte bassa dell'indirizzo.
- ALE (ADDRESS LATCH ENABLE) è il segnale che consente di caricare in LATCH la parte bassa del bus degli indirizzi per completare l'indirizzo da A0 ad A15. Questo segnale esce nel primo ciclo di ogni istruzione.

| <i>IO/M</i> | <i>S<sub>0</sub></i> | <i>S<sub>1</sub></i> | <i>Ciclo del microprocessore</i> |
|-------------|----------------------|----------------------|----------------------------------|
| 0           | 0                    | 1                    | MW (scrittura in memoria)        |
| 0           | 1                    | 0                    | MR (lettura in memoria)          |
| 1           | 0                    | 1                    | IOW (scrittura periferica IO)    |
| 1           | 1                    | 0                    | IOR (lettura periferica IO)      |
| 0           | 1                    | 1                    | (Ricerca del codice operativo)   |
| 1           | 1                    | 1                    | (Riconoscimento di interrupt)    |

*Tabella IV.- Relazione tra lo stato di alcuni segnali di controllo e il ciclo del microprocessore.*

- S<sub>0</sub>, S<sub>1</sub>, IO/M, questi segnali definiscono il ciclo di lavoro in cui i trova il  $\mu P$  in ogni istante, in base alla tabella IV.
- RD (READ CONTROL), quando questo segnale è zero, indica alla memoria e alla periferia che il  $\mu P$  esegue un'operazione di lettura.
- WR è simile al precedente, ma indica un'operazione di scrittura.
- READY, è un'ingresso mediante il quale le periferiche indicano al  $\mu P$  che sono pronte per mettersi in contatto con esso, settandolo a uno.
- HOLD è un ingresso attraverso il quale le periferiche o memorie lente occupano il microprocessore, per un certo tempo, mentre questo terminale è a uno logico.
- HLDA (HOLD ACKNOWLEDGE), riconoscimento di occupazione, è un'uscita tramite cui il  $\mu P$  comunica alle periferiche in risposta ad un segnale di HOLD, settandola a uno per tutto il tempo in cui permane il segnale HOLD.
- INTR (INTERRUPT REQUEST) è un ingresso tramite cui le periferiche possono sollecitare un interrupt portandolo a uno.
- INTA (INTERRUPT ACKNOWLEDGE), riconoscimento dell'interrupt, il  $\mu P$  porta a zero quest'uscita quando accetta un interrupt richiesto da una periferica.
- RST5.5, RST6.5, e RST7.5 (RESTART INTERRUPT) sono segnali di richiesta di interrupt dello stesso tipo di INTA, con la differenza che questi tre provocano una reinizializzazione interna del  $\mu P$ , posizionandolo a locazioni fisse di memoria.
- TRAP è un altro ingresso di richiesta di interrupt, ma con priorità su tutte le altre.
- RESET IN, ingresso di azzeramento per l'inizializzazione del  $\mu P$

dopo aver fornito la tensione, che lo posiziona all'indirizzo 0000.

- RESET OUT, questa uscita passa a uno durante l'inizializzazione del  $\mu P$  causata dal segnale precedente.
- CLK è un'uscita del clock del  $\mu P$ , e può servire come clock di sistema.
- X1, X2 sono due ingressi a cui si collega il cristallo di quarzo.
- SID (SERIAL INPUT DATA) è un ingresso seriale. Il contenuto di questo bit compare nel bit 7 dell'accumulatore, quando si esegue la corrispondente istruzione.
- SOD (SERIAL OUTPUT DATA) è un'uscita di dati seriali, il dato che esce è il contenuto del bit 7 dell'accumulatore.
- VCC e VSS sono i terminali di alimentazione, +5 V, e massa, rispettivamente.

### Ottenimento del bus degli indirizzi

Il  $\mu P$  dispone di 16 bit per il bus degli indirizzi, dei quali gli otto più significativi (A8-A15) escono direttamente dal  $\mu P$  verso l'esterno, gli altri otto meno significativi (A0-A7) escono con il bus dei dati durante un ciclo previ-

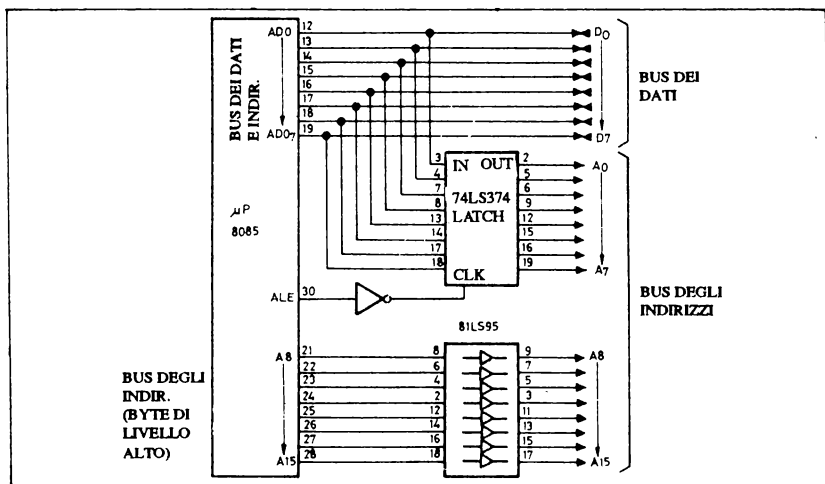


Fig. 14.-Dettaglio dell'ampliamento del bus degli indirizzi.



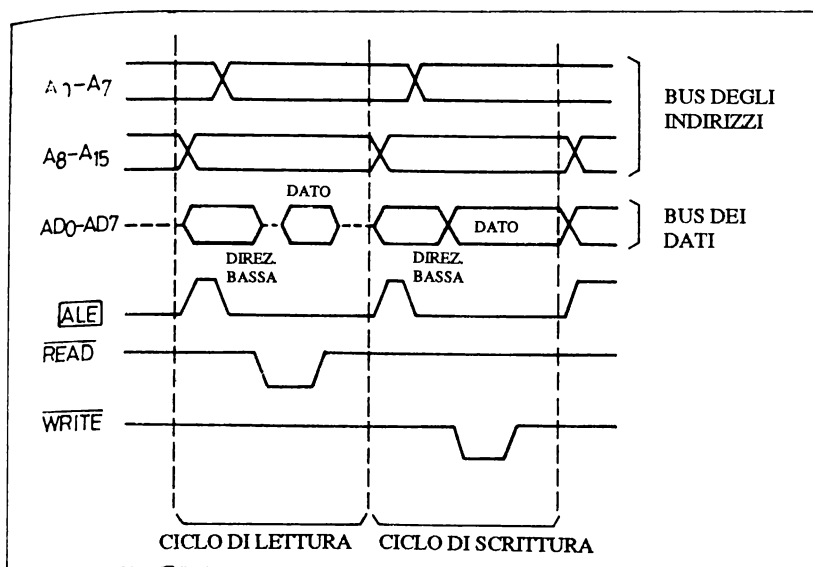


Fig. 15.-Diagramma dei tempi relativo al segnale ALE per ottenere il bus dei dati completo.

sto per l'esecuzione dell'istruzione; vale a dire che dal bus dei dati esce dapprima la parte bassa del bus degli indirizzi e poi il codice dell'istruzione. Questa tecnica permette di ridurre il numero dei terminali del  $\mu P$ , a soli quaranta.

Il segnale ALE (ADDRESS LATCH ENABLE) è quello che coordina la trasformazione del bus dei dati, in indirizzo o dato. Quando compare tale segnale nel primo ciclo di ogni istruzione che accede alla memoria, si caricano i latch con il contenuto del bus dei dati; presentandolo all'uscita, questo contenuto definisce gli otto bit meno significativi del bus degli indirizzi. Il segnale ALE attiva il clock dei latch per caricarli.

Nel diagramma dei tempi di Fig. 15, si possono rilevare i principali segnali del  $\mu P$  correlati con il segnale ALE.

### Uso degli ingressi ready e hold

Quando si porta a livello logico zero l'ingresso READY del  $\mu P$ , si genera lo stato di attesa (WAIT). Mediante questo stato il  $\mu P$  può accedere alle memorie o alle periferiche lente. Tramite questo ingresso, si può anche ottenere

l'esecuzione passo a passo di un programma. Se, tramite una circuiteria esterna ausiliaria, si resetta a zero questo terminale durante l'esecuzione di un'istruzione, i bus mantengono l'indirizzo e il dato, per cui l'utente può accertare se il programma viene correttamente eseguito. Da quanto detto, risulta che questo è un metodo lento, ma efficace, per il controllo di un programma.

Il segnale HOLD può essere utilizzato per implementare (realizzare) un sistema DMA (Direct Memory Access). Esistono alcune periferiche che, per funzionare più velocemente, necessitano di accedere direttamente alla memoria senza passare attraverso la CPU; per ottenere ciò è necessario che il  $\mu P$  si arresti e ponga i bus ad alta impedenza, per permettere in tal modo che la suddetta periferica possa accedere alla stessa memoria utilizzata dal  $\mu P$  per leggere o scrivere un blocco di dati. Con il  $\mu P$  8085 ciò si ottiene utilizzando l'ingresso HOLD; portando tale ingresso a uno, il  $\mu P$  completa l'esecuzione dell'istruzione in corso, e dopo aver settato a uno l'uscita HLDA, setta i bus ad alta impedenza. Quando la periferica termina di accedere alla memoria, resetta nuovamente a zero l'ingresso HOLD, per cui il microprocessore continuerà con l'istruzione successiva.

### *Esempio di collegamento del $\mu P$ 8085*

E' simile a quello del  $\mu P$  8080, ma più semplice e completo, poichè essendo più integrato, i collegamenti esterni sono semplificati. Nell'esempio proposto si osservano tre blocchi, corrispondenti a quelli a cui il  $\mu P$  dovrebbe essere collegato, ma il microprocessore si potrebbe collegare ad altri blocchi o periferiche in funzione dell'utilizzo a cui viene destinato. Bisogna però ricordare di posizionare la PROM, in cui risiedono le istruzioni iniziali, all'indirizzo 0000, in quanto il  $\mu P$ , dopo il reset, riparte da tale indirizzo.

Il blocco definito decodificatore di pagina è, in realtà, quello che si incarica di abilitare i blocchi corrispondenti, in funzione dei bit A13, A14 e A15 del bus diretto degli indirizzi, e il segnale di permesso IO/M che consente il passaggio sincronizzato al decodificatore.

Si osservi il segnale ALE che carica il latch con il contenuto del bus dei dati, per formare la parte bassa (L) del bus degli indirizzi. I segnali alla sinistra del  $\mu P$  sono segnali di interrupts, di cui si tratterà nel seguito, e gli altri sono segnali di controllo già visti.

### *Gli interrupts nell'8085*

Il microprocessore 8085 possiede un complesso e completo sistema di

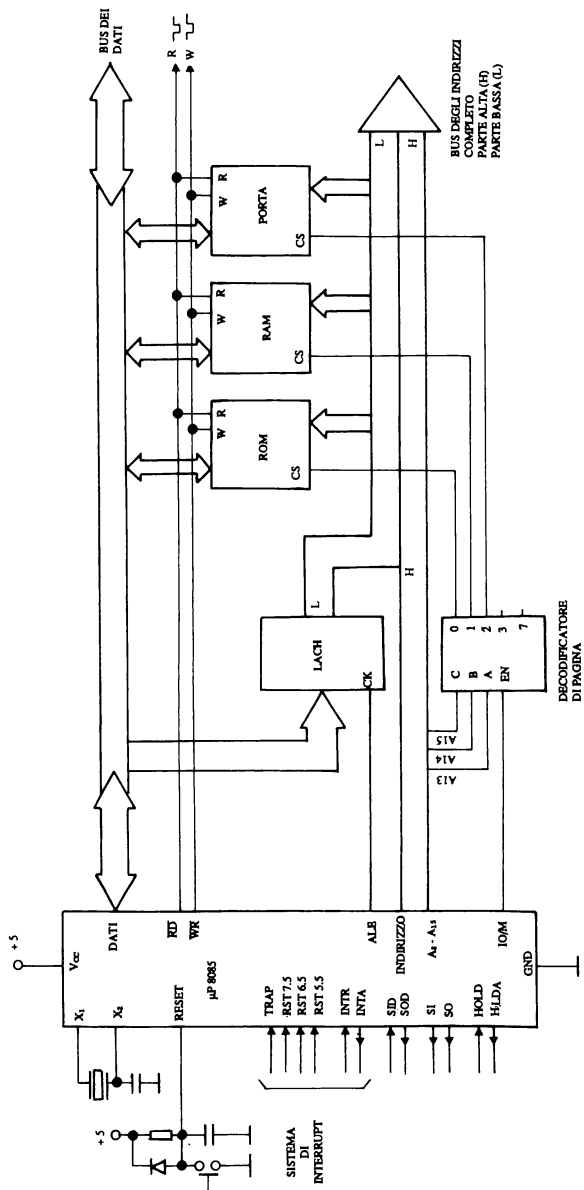


Fig. 16.-Schema dei collegamenti dell' 8085.

interrupts, ai quali sono dedicati i chiarimenti seguenti. Il  $\mu P$  8085 possiede cinque terminali destinati al trattamento degli interrupts.

Si ricorda che un interrupt è un artificio hardware/software, mediante il quale è possibile arrestare il programma in corso in modo che, quando avviene un determinato evento, il  $\mu P$  completi l'istruzione in corso ed effettui un salto ad una determinata subroutine, durante la quale si manterrà l'interrupt, dopo di che, il  $\mu P$  continuerà con l'istruzione successiva del programma principale.

L'8085 dispone di tre diversi modi per trattare gli interrupts, che gli arrivano attraverso i cinque terminali, i cui nomi sono:

- INTR (Interrupt Request): tramite questo ingresso si invia un interrupt che viene accettato o meno in base a quanto è stato indicato in precedenza mediante le istruzioni EI (interrupt permesso) o DI (interrupt non permesso); ciò significa che l'interrupt INTR è mascherabile con tali istruzioni, in modo tale che se nell'esecuzione di un programma è stata inclusa l'istruzione DI, anche se viene richiesto un interrupt tramite il terminale INTR, questo non sarà mai soddisfatto. Questo terminale INTR determina perciò solo uno dei tre modi di trattare un interrupt. Quando un interrupt è permesso e si è prodotto, la CPU cerca un'istruzione di reinizializzazione RST (a un solo byte), che viene fornita dall'esterno tramite la periferica che ha generato l'interrupt. Questo byte ha la forma di Fig. 17, in cui i bit indicati con XXX possono assumere i valori da 000 a 111; ciascuno di questi valori indica in quale degli otto possibili indirizzi, rappresentati anch'essi in Fig. 17, si trova la subroutine che elabora l'interrupt; al termine di tale subroutine deve esistere un'istruzione di ritorno per far sì che il programma, dopo averla eseguita, ritorni a quello principale.
- RST 5.5, RST 6.5, e RST 7.5: la funzione di questi tre terminali è di produrre un secondo modo di trattamento degli interrupts. I terminali RST 5.5 e RST 6.5 rilevano un interrupt solo se il segnale che viene loro applicato è ad uno logico o livello alto, per un certo tempo, come per il precedente ingresso INTR; l'ingresso corrispondente al terminale RST 7.5 è invece eccitato dal fronte di una variazione di segnale da zero a uno, fronte che cambia stato a un bistabile interno al  $\mu P$ . Terminato il trattamento dell'interrupt il  $\mu P$  resetta nuovamente tale bistabile.

Ciascuno di questi tre terminali, quando viene attivato, indica una richiesta di interrupt, e ad ognuno corrisponde un unico indirizzo a cui si trova la subroutine che tratta tali richieste, indirizzi visibili

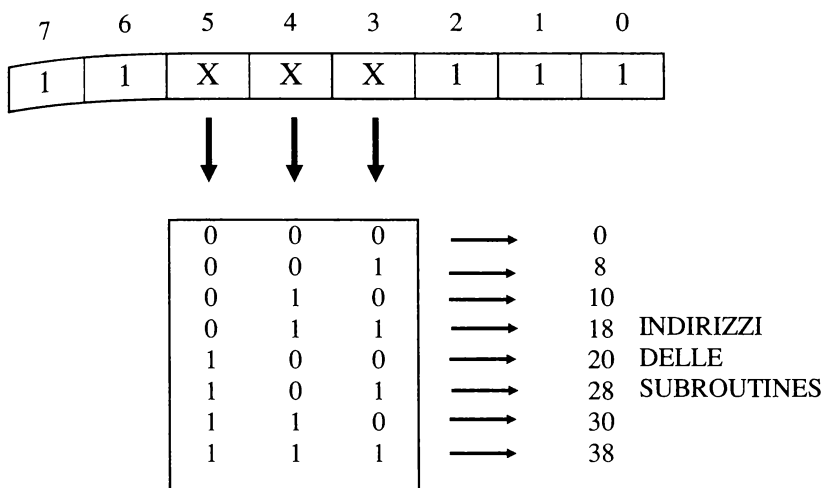


Fig. 17.-Significato dei bit del byte inviato dalla periferica che richiede un interrupt al terminale INTR.

nella tabella V.

Gli interrupts corrispondenti a RST 5.5, RST 6.5 e RST 7.5 possono essere permessi o vietati mediante le solite istruzioni EI e DI, ma sono anche mascherabili via software tramite l'istruzione SIM (Set Interrupt Mask) che permette o meno gli interrupts attraverso la scelta di alcuni bit indicatori di maschera, presenti nel registro accumulatore, che è stato caricato con il contenuto del secondo byte dell'istruzione che funziona da maschera. Dopo l'esecuzione di tale istruzione, saranno permessi solo gli interrupts il cui corrispondente bit sia stato settato a uno logico; ma la maschera potrà essere caricata solo se il bit tre dell'accumulatore è stato settato a uno dal  $\mu P$ . Il bit quattro resetta il bistabile dell'interrupt 7.5.

Mediante l'istruzione RIM (Read Interrupt Mask) è possibile leggere in ogni istante lo stato della maschera con l'informazione complementare dei bit 4, 5, e 6, che indicano gli interrupts corrispondenti che sono stati prodotti, ma che non sono stati accettati fino al momento dell'istruzione RIM.

## *Interrupts e controllo di I/O seriale*

L'interrupt TRAP è un interrupt non mascherabile, che viene attivato quando il terminale con tale denominazione (TRAP) viene settato a livello logico uno.

Quando il microprocessore lo esegue, questo indirizza una locazione fissa, la 24 esadecimale, a cui deve iniziare la subroutine che processa tale interrupt. Il segnale di interrupt proveniente dall'esterno, che viene fornito a tale terminale, agisce con il fronte di salita sui circuiti interni di interrupt via hardware, ma il livello logico uno deve rimanere finchè non avviene il completo riconoscimento da parte del microprocessore; questo interrupt è individuato dal fronte e dal livello.

Tale interrupt ha la priorità più elevata, per cui può essere utilizzato per trattare i fenomeni più importanti, come ad esempio gli errori, le cadute di alimentazione, ecc.

Nella tabella V sono elencati i vari interrupts, in base al livello di priorità.

Questi livelli sono quelli che determinano l'ordine con cui verranno eseguiti gli interrupts che siano richiesti simultaneamente.

Se durante un programma il microprocessore esegue uno dei cinque interrupt, a partire da tale istante bloccherà il permesso di eseguirne altri finchè non si rimuova tale blocco, via software, mediante l'istruzione EI (permesso di interrupt); tuttavia ciò non avviene per l'interrupt TRAP che, avendo priorità assoluta, può essere trattato in qualsiasi istante. L'esecuzione dell'istruzione RIM dopo che è avvenuto un interrupt TRAP, mostra il dato della maschera degli interrupts ricevuti precedentemente al TRAP, che non sono stati eseguiti.

| Livello di priorità | Nome dell'interrupt | Indirizzo della subroutine in esadecimale |
|---------------------|---------------------|---|
| Maggior priorità    | TRAP                | 24  |
| —                   | RST 5.5             | 2C  |
| —                   | RST 6.5             | 34  |
| —                   | RST 7.5             | 3C  |
| Minor priorità      | INTR                | —   |

*Tabella V.-Tabella degli interrupts del microprocessore 8085.*

## Controllo dell'input/output seriale

Le istruzioni RIM e SIM controllano anche il sistema di input/output seriale che si ottiene tramite il bit 7 dell'accumulatore.

Come è noto, quando il  $\mu P$  esegue un'istruzione RIM, il dato del terminale numero cinque del  $\mu P$ , detto SID, viene caricato nel bit 7 del registro accumulatore, mentre dal terminale numero quattro (SOD) uscirà il contenuto del bit 7 dell'accumulatore, quando viene eseguita l'istruzione SIM, ma solo se il bit 6 dello stesso è stato precedentemente settato a uno.

Il dato all'uscita del terminale SOD verrà mantenuto per un periodo indefinito, poichè tra l'accumulatore e l'uscita esiste un latch in grado di memorizzare tale bit.

In tutti i microprocessori, il sistema di input/output seriale ha una certa importanza, dato che esistono periferiche che comunicano serialmente con la CPU.

L'ingresso SID può essere considerato come un ingresso asincrono attraverso il quale si invia un bit di informazione proveniente dall'esterno, senza che sia necessario stabilire un indirizzo per lo stesso. Tale tipo di ingresso viene anche detto SENSE.

L'uscita SOD può essere considerata come un FLAG. Si ricorda che un FLAG è un segnale a un bit che, se previsto nel programma, può avere un'infinità di applicazioni, come attivare qualche circuito esterno in modo asincrono, o semplicemente attivare un diodo led per indicare se il programma ha percorso una certa subroutine, ecc.

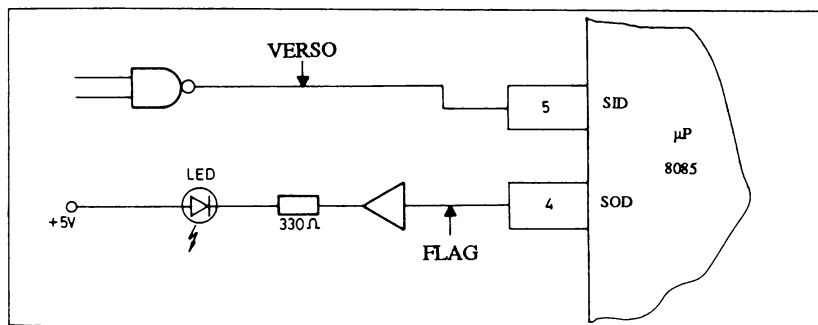


Fig. 18.-Ingresso SID considerato come SENSE e uscita SOD considerata come FLAG.

## *Simbologia delle istruzioni nell'8085*

Nelle tabelle delle istruzioni dell'8085 sono stati utilizzati le sigle e i simboli seguenti:

- r1, r2, e r3 rappresentano alcuni dei registri A, B, C, D, E, H, L e M; l'indirizzo di memoria è determinato mediante H (byte alto) e L (byte basso);
- DDD rappresenta il registro destinazione in cui viene caricato il dato, e SSS il registro sorgente dal quale il dato proviene;
- rp rappresenta una coppia di registri;
- i codici B(B-C): 0, D(D-E): 1, H(H-L): 2, e SP (Stack Pointer): 3, sono i valori I, in esadecimale, della tabella VII;
- n rappresenta un numero compreso tra 0 e 7, e indica una delle otto posizioni a cui salta il contatore di programma dopo un'istruzione RST;
- il codice NNN rappresenta un binario tra 000 e 111;
- CCC indica una condizione dipendente dallo stato dei flags;
- Adr indica un indirizzo.

## *Modi di indirizzamento dell'8085*

Il microprocessore 8085 dispone di vari sistemi di indirizzamento, che ampliano la funzione di ogni istruzione.

Le istruzioni possono essere a uno, due e tre bytes, in base al tipo specifico.

- **INDIRIZZAMENTO IMMEDIATO:** il byte che segue il codice operativo e il dato o operando, sono istruzioni a due bytes;
- **INDIRIZZAMENTO DIRETTO:** i due bytes che seguono il codice dell'operazione rappresentano l'indirizzo di memoria in cui si trova il dato;
- **INDIRIZZAMENTO TRAMITE REGISTRO:** uno dei registri B, C, E, H o L contiene il dato (in questo indirizzamento i registri sono considerati a otto bit);
- **INDIRIZZAMENTO TRAMITE COPPIA DI REGISTRI:** in questo caso il contenuto della coppia di registri indica l'indirizzo di memoria in cui si trova il dato; i registri sono accoppiati nel seguente modo: B con C, D con E, H con L.



Conoscere il banco dei registri dell'8085 è indispensabile per poterlo programmare, tenendo conto che un registro può essere utilizzato per una istruzione o semplicemente come luogo di immagazzinamento temporaneo di un dato o di un indirizzo.

**TABELLA VI**

| <b>Mnemonico</b>                    | <b>Descrizione</b>  | <b>1° byte dell'istruzione</b> |    |    |    |    |    |    |    |
|-------------------------------------|---|--------------------------------|----|----|----|----|----|----|----|
|                                     |   | D7                             | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| MOV r <sub>1</sub> , r <sub>2</sub> | r <sub>1</sub> ← r <sub>2</sub><br>L'informazione contenuta in r <sub>2</sub> passa in r <sub>1</sub>   | 0                              | 1  | D  | D  | D  | S  | S  | S  |
| MOV r <sub>1</sub><br>dato          | r <sub>1</sub> ← dato<br>Il dato passa in r <sub>1</sub>  | 0                              | 0  | D  | D  | D  | 1  | 1  | 0  |
| LXI rp<br>dato                      | rp ← dato<br>Il dato passa nella coppia di registri rp  | 0                              | 0  | R  | P  | 0  | 0  | 0  | 1  |
| LDA mem.                            | A ← mem<br>Il contenuto della locazione di memoria indirizzata si trasferisce nell'accumulatore   | 0                              | 0  | 1  | 1  | 1  | 0  | 1  | 0  |
| STA mem.                            | mem ← A<br>Il contenuto dell'accumulatore viene posto nella locazione di memoria indirizzata  | 0                              | 0  | 1  | 1  | 0  | 0  | 1  | 0  |
| LHLD mem.                           | HL ← mem<br>Il contenuto che esiste in una locazione di memoria viene posto nel registro L. Il contenuto dell'indirizzo successivo nel registro H | 0                              | 0  | 1  | 0  | 1  | 0  | 1  | 0  |

| Mnemonico | Descrizione   | 1 <sup>o</sup> byte dell'istruzione |    |    |    |    |    |    |    |
|-----------|---|-------------------------------------|----|----|----|----|----|----|----|
|           |   | D7                                  | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| SHLD mem. | mem $\leftarrow$ HL<br>Trasferisce il contenuto dei registri HL ad una locazione di memoria e alla successiva                                     | 0                                   | 0  | 1  | 0  | 0  | 0  | 1  | 0  |
| LDAX rp   | A $\leftarrow$ rp<br>Il contenuto della posizione di memoria indirizzata dalla coppia di registri rp passa nell'accumulatore                      | 0                                   | 0  | R  | P  | 1  | 0  | 1  | 0  |
| STAX rp   | rp $\leftarrow$ A<br>Il contenuto dell'accumulatore viene salvato nella locazione di memoria indirizzata da una coppia di registri                | 0                                   | 0  | R  | P  | 0  | 0  | 1  | 0  |
| XCHG      | H $\leftrightarrow$ D L $\leftrightarrow$ E<br>Il contenuto dei due registri H e L si scambia con il contenuto dei registri D e E rispettivamente | 1                                   | 1  | 1  | 0  | 1  | 0  | 1  | 1  |
| ADD, r    | A $\leftarrow$ A+r<br>Aggiunge il contenuto del registro r all'accumulatore   | 1                                   | 0  | 0  | 0  | 0  | S  | S  | S  |
| ADI, dato | A $\leftarrow$ A+dato<br>Il dato o 2 <sup>o</sup> byte viene sommato all'accumulatore   | 1                                   | 1  | 0  | 0  | 0  | 1  | 1  | 0  |
| ADC, r    | A $\leftarrow$ A+r+CY<br>Somma il contenuto del registro r e il riporto del CY all'accumulatore   | 1                                   | 0  | 0  | 0  | 1  | S  | S  | S  |
| ACI dato  | A $\leftarrow$ A+dato+CY<br>Il dato e il riporto vengono sommati all'accumulatore   | 1                                   | 1  | 0  | 0  | 1  | 1  | 1  | 0  |
| SUB r     | A $\leftarrow$ A-r<br>Sottrae il contenuto del registro r dall'accumulatore   | 1                                   | 0  | 0  | 1  | 0  | S  | S  | S  |
| SUI dato  | A $\leftarrow$ A-dato<br>Sottrae dato da accumul.re   | 1                                   | 1  | 0  | 1  | 0  | 1  | 1  | 0  |

| <i>Mnemonico</i> | <i>Descrizione</i>   | <i>1° byte dell'istruzione</i> |    |    |    |    |    |    |    |
|------------------|--|--------------------------------|----|----|----|----|----|----|----|
|                  |  | D7                             | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| SBBR             | $A \leftarrow A - r - CY$<br>Sottrae dall'accumulatore il contenuto del registro r e il riporto                                | 1                              | 0  | 0  | 1  | 1  | S  | S  | S  |
| SBI dato         | $A \leftarrow A - \text{dato} - CY$<br>Il dato e il riporto vengono sottratti dall'accumulatore                                | 1                              | 1  | 0  | 1  | 1  | 1  | 1  | 0  |
| INR r            | $r \leftarrow r + 1$<br>Incrementa contenuto di r  | 0                              | 0  | D  | D  | D  | 1  | 0  | 0  |
| DCR r            | $r \leftarrow r - 1$<br>Decrementa il contenuto del registro r   | 0                              | 0  | D  | D  | D  | 1  | 0  | 1  |
| INX rp           | $rp \leftarrow rp + 1$<br>Il contenuto della coppia di registri rp viene incrementato di un'unità                              | 0                              | 0  | R  | P  | 0  | 0  | 1  | 1  |
| DCX rp           | $rp \leftarrow rp - 1$<br>Il contenuto della coppia di registri rp viene decrementato di un'unità                              | 0                              | 0  | R  | P  | 1  | 0  | 1  | 1  |
| DAD rp           | $HL \leftarrow rp + HL$<br>Il contenuto della coppia di registri rp viene sommato al contenuto della coppia di registri H e L. | 0                              | 0  | R  | P  | 1  | 0  | 0  | 1  |
| DAA              | Regolazione decimale nell'accumulatore per ottenere 2 digit da 4 bits in BCD   | 0                              | 0  | 1  | 0  | 0  | 1  | 1  | 1  |
| ANA r            | $A \leftarrow A \wedge r$<br>Istruzione logica AND tra il registro r e l'accumulatore.   | 1                              | 0  | 1  | 0  | 0  | S  | S  | S  |
| ANI dato         | $A \leftarrow A \wedge \text{dato}$<br>AND di dato e accumulatore  | 1                              | 1  | 1  | 0  | 0  | 1  | 1  | 0  |
| XRA r            | $A \leftarrow A \vee r$<br>Esegue l'operazione logica di or esclusivo tra il registro r e l'accumulatore                       | 1                              | 0  | 1  | 0  | 1  | S  | S  | S  |

| <i>Mnemonico</i> | <i>Descrizione</i>  | <i>1<sup>o</sup> byte dell'istruzione</i> |    |    |    |    |    |    |    |
|------------------|---|---|----|----|----|----|----|----|----|
|                  |   | D7  | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| XRI dato         | $A \leftarrow A \nabla \text{dato}$<br>Operazione or esclusivo tra dato e accumulatore  | 1   | 1  | 1  | 0  | 1  | 1  | 1  | 0  |
| ORA r            | $A \leftarrow A \vee r$<br>Operazione logica or tra il contenuto del registro r e l'accumulatore  | 1   | 0  | 1  | 1  | 0  | S  | S  | S  |
| ORI dato         | $A \leftarrow A \vee \text{dato}$<br>OR logico tra il dato e l'accumulatore   | 1   | 0  | 1  | 1  | 1  | S  | S  | S  |
| CMP r            | A-r sottrazione<br>Confronta il contenuto del registro r e l'accumulatore. L'accumulatore non cambia  | 1   | 0  | 1  | 1  | 1  | S  | S  | S  |
| CPI dato         | A-dato<br>Confronta dato e accumulatore. L'accumulatore non cambia  | 1   | 1  | 1  | 1  | 1  | 1  | 1  | 0  |
| RLC              | Rotazione a sinistra dell'accumulatore $(A_{n+1}) \leftarrow (A_n)$ ; $(A_0) \leftarrow (A_7)$ ; $(CY) \leftarrow (A_7)$                    | 0   | 0  | 0  | 0  | 0  | 1  | 1  | 1  |
| RRC              | Rotazione a destra dell'accumulatore $(A_n) \leftarrow (A_{n+1})$ ; $(A_7) \leftarrow (A_0)$ ; $(CY) \leftarrow (A_0)$                      | 0   | 0  | 0  | 0  | 1  | 1  | 1  | 1  |
| RAL              | Rotazione dell'accumulatore a sinistra attraverso il carry $(A_{n+1}) \leftarrow (A_n)$ ; $(CY) \leftarrow (A_7)$ ; $(A_0) \leftarrow (CY)$ | 0   | 0  | 0  | 1  | 0  | 1  | 1  | 1  |
| RAR              | Rotazione dell'accumulatore a destra attraverso il carry $(A_n) \leftarrow (A_{n+1})$ ; $(CY) \leftarrow (A_0)$ ; $(A_0) \leftarrow (CY)$   | 0   | 0  | 0  | 1  | 1  | 1  | 1  | 1  |
| CMA              | $A \leftarrow A$<br>Complemento dell'accumulatore   | 0   | 0  | 1  | 0  | 1  | 1  | 1  | 1  |
| CMC              | $CY \leftarrow CY$<br>Complemento del carry   | 0   | 0  | 1  | 1  | 1  | 1  | 1  | 1  |

| <i>Mnemonico</i>         | <i>Descrizione</i>  | <i>1° byte dell'istruzione</i> |    |    |    |    |    |    |    |
|--------------------------|---|--------------------------------|----|----|----|----|----|----|----|
|                          |   | D7                             | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| STC                      | CY ← 1<br>Setta il Carry  | 0                              | 0  | 1  | 1  | 0  | 1  | 1  | 1  |
| JMP mem.                 | PC ← mem<br>Il contatore di programma è caricato con l'indirizzo di memoria del 2° e 3° byte  | 1                              | 1  | 0  | 0  | 0  | 0  | 1  | 1  |
| JXX mem.                 | PC ← mem; se CCC=1<br>Salto condizionale; dipende dai flags   | 1                              | 1  | C  | C  | C  | 0  | 1  | 0  |
| CALL mem.                | SP-1 ← PCH<br>SP-2 ← PCL<br>SP ← SP-2<br>PC ← mem (3° e 2° byte)<br>Salta incondizionatamente alla subroutine che occupa la locazione di memoria fornita dal 2° e 3° byte | 1                              | 1  | 0  | 0  | 1  | 1  | 0  | 1  |
| CXX mem.<br>Condizionato | Uguale a CALL se è verificata la condizione. Dipende dai flags  | 1                              | 1  | C  | C  | C  | 1  | 0  | 0  |
| RET                      | PCL ← SP<br>PCH ← SP+1<br>SP ← SP+2<br>Ritorno incondizionato da subroutine   | 1                              | 1  | 0  | 0  | 1  | 0  | 0  | 1  |
| RXX                      | Uguale a RET, se CCC=1.<br>Dipende dai flags  | 1                              | 1  | C  | C  | C  | 0  | 0  | 0  |
| RST n                    | SP-1 ← PCH<br>SP-2 ← PCL<br>SP ← SP-2<br>PC ← 8*NNN<br>Salto incondizionato a una subroutine con indirizzo n  | 1                              | 1  | N  | N  | N  | 1  | 1  | 1  |
| PCHL                     | PC ← HL<br>Il contenuto di HL, sostituisce il contenuto del contatore di programma  | 1                              | 1  | 1  | 0  | 1  | 0  | 0  | 1  |

| <i>Mnemonico</i> | <i>Descrizione</i>   | <i>1<sup>o</sup> byte dell'istruzione</i> |    |    |    |    |    |    |    |
|------------------|--|---|----|----|----|----|----|----|----|
|                  |  | D7  | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| PUSH rp          | $SP-1 \leftarrow rp$<br>$rp \leftarrow SP+1$<br>$SP \leftarrow SP-2$<br>Pone nello stack il contenuto della coppia di registri rp            | 1   | 1  | R  | P  | 0  | 1  | 0  | 1  |
| POP rp           | $rl \leftarrow SP$<br>$sh \leftarrow SP+1$<br>$SP \leftarrow SP+2$<br>Trasferisce il contenuto dello stack nella coppia di registri rp       | 1   | 1  | R  | P  | 0  | 0  | 0  | 1  |
| PUSH PSW         | Pone nello stack il contenuto del registro P   | 1   | 1  | 1  | 1  | 0  | 1  | 0  | 1  |
| POP PSW          | Trasferisce il contenuto dello stack nel registro P  | 1   | 1  | 1  | 1  | 0  | 0  | 0  | 1  |
| XTHL             | $L \leftrightarrow SP$<br>$H \leftrightarrow SP+1$<br>Il contenuto dei registri L e H viene scambiato con le prime due locazioni dello stack | 1   | 1  | 1  | 0  | 0  | 0  | 1  | 1  |
| SPHL             | $SP \leftarrow HL$<br>Il contenuto di H e L sostituisce il contenuto del puntatore di stack  | 1   | 1  | 1  | 1  | 1  | 0  | 0  | 1  |
| IN canale        | $A \leftarrow (\text{dati canale})$<br>L'ingresso <i>canale</i> invia i suoi 8 bit di informazione, che vengono depositati nell'accumulatore | 1   | 1  | 0  | 1  | 1  | 0  | 1  |    |
| OUT canale       | $(\text{dati canale}) \leftarrow A$<br>Il contenuto dell'accumulatore viene inviato al canale di uscita                                      | 1   | 1  | 0  | 1  | 0  | 0  | 1  | 1  |

Tabella VI.-Tabella delle istruzioni del  $\mu P$  8085.

## *Istruzioni specifiche dell'8085*

Praticamente tutto quanto detto relativamente al software dell'8080 è applicabile all'8085, in quanto l'8085 è un perfezionamento dell'8080.

Esistono due istruzioni specifiche del  $\mu P$  8085, cioè la RIM e la SIM con codice macchina o dell'operazione 20 e 30 esadecimale, rispettivamente. Queste istruzioni sono specifiche in quanto solo l'8085 possiede gli interrupts mascherabili, descritti precedentemente. L'istruzione RIM (Read Interrupt Mask), esegue la lettura della maschera degli interrupts, mentre la SIM (Set Interrupt Mask) scrive la maschera degli interrupts.

| $\begin{matrix} I \\ 0 \end{matrix}$ | $\begin{matrix} I \\ 1 \end{matrix}$ | $\begin{matrix} I \\ 2 \end{matrix}$ | $\begin{matrix} I \\ 3 \end{matrix}$ | $\begin{matrix} I \\ 4 \end{matrix}$ | $\begin{matrix} I \\ 5 \end{matrix}$ | $\begin{matrix} I \\ 6 \end{matrix}$ | $\begin{matrix} I \\ 7 \end{matrix}$ | $\begin{matrix} I \\ 8 \end{matrix}$ | $\begin{matrix} I \\ 9 \end{matrix}$ | $\begin{matrix} I \\ A \end{matrix}$ | $\begin{matrix} I \\ B \end{matrix}$ | $\begin{matrix} I \\ C \end{matrix}$ | $\begin{matrix} I \\ D \end{matrix}$ | $\begin{matrix} I \\ E \end{matrix}$ | $\begin{matrix} I \\ F \end{matrix}$ |
|--------------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|
| 0                                    | NOP                                  | LXI                                  | B STAX                               | B INX                                | B INR                                | B DCR                                | B DCR                                | B DCR                                | DAD                                  | B LDAX                               | B DCX                                | B INR                                | C DCR                                | C MVI                                | C RRC                                |
| 1                                    | LXI                                  | D STAX                               | D INX                                | D INR                                | D DCR                                | D DCR                                | D RAI                                | DAD                                  | DAD                                  | D LDAC                               | D DCX                                | D INR                                | E DCR                                | E MVI                                | E RAR                                |
| 2                                    | RIM                                  | LXI                                  | H SHLD                               | H INX                                | H DCR                                | H MVI                                | H DAA                                | DAD                                  | DAD                                  | H LHLD                               | H DCX                                | H INR                                | L DCR                                | L MVI                                | L CMA                                |
| 3                                    | SM                                   | LXI                                  | SP STA                               | INX                                  | SP INR                               | M DCR                                | M STC                                | DAD                                  | DAD                                  | SP LDA                               | SP DCX                               | SP INR                               | A DCR                                | A MVI                                | A CMC                                |
| 4                                    | MOV                                  | BB MOV                               | BC MOV                               | BD MOV                               | BE MOV                               | BL MOV                               | BM MOV                               | BA MOV                               | CB MOV                               | CC MOV                               | CD MOV                               | CE MOV                               | CH MOV                               | CL MOV                               | CM MOV                               |
| 5                                    | MOV                                  | DB MOV                               | DC MOV                               | DD MOV                               | DE MOV                               | DI MOV                               | DM MOV                               | DA MOV                               | EB MOV                               | EC MOV                               | ED MOV                               | EE MOV                               | EH MOV                               | FL MOV                               | FM MOV                               |
| 6                                    | MOV                                  | HB MOV                               | HC MOV                               | HD MOV                               | HE MOV                               | HL MOV                               | HM MOV                               | HA MOV                               | LB MOV                               | LC MOV                               | LD MOV                               | LE MOV                               | LH MOV                               | LL MOV                               | LM MOV                               |
| 7                                    | MOV                                  | MB MOV                               | MC MOV                               | MD MOV                               | ME MOV                               | ML MOV                               | MM MOV                               | MA MOV                               | AB MOV                               | AC MOV                               | AD MOV                               | AE MOV                               | AH MOV                               | AL MOV                               | AM MOV                               |
| 8                                    | ADD                                  | B ADD                                | C ADD                                | D ADD                                | E ADD                                | H ADD                                | M ADD                                | A ADD                                | B ADC                                | C ADC                                | D ADC                                | E ADC                                | H ADC                                | L ADC                                | M ADC                                |
| 9                                    | SUB                                  | B SUB                                | C SUB                                | D SUB                                | E SUB                                | H SUB                                | M SUB                                | A SUB                                | B SBB                                | C SBB                                | D SBB                                | E SBB                                | H SBB                                | L SBB                                | M SBB                                |
| A                                    | ANA                                  | B ANA                                | C ANA                                | D ANA                                | E ANA                                | H ANA                                | M ANA                                | A ANA                                | B XRA                                | C XRA                                | D XRA                                | E XRA                                | H XRA                                | L XRA                                | M XRA                                |
| B                                    | ORA                                  | B ORA                                | C ORA                                | D ORA                                | E ORA                                | H ORA                                | M ORA                                | A ORA                                | B CMP                                | C CMP                                | D CMP                                | E CMP                                | H CMP                                | L CMP                                | M CMP                                |
| C                                    | RNZ                                  | POP                                  | B INZ                                | IMP                                  | addr                                 | CNZ                                  | addr                                 | RZ                                   | RET                                  | JZ                                   | addr                                 | CZ                                   | addr                                 | CALL                                 | addr                                 |
| D                                    | RNC                                  | POP                                  | D INC                                | OUT                                  | addr                                 | CNC                                  | addr                                 | RC                                   | RET                                  | JC                                   | addr                                 | CC                                   | addr                                 | SBI                                  | RST                                  |
| E                                    | RPO                                  | POP                                  | H IPO                                | addr                                 | CPO                                  | PUSH                                 | H ANI                                | RPE                                  | SPHI                                 | PCHI                                 | addr                                 | XCHG                                 | CPE                                  | addr                                 | RST                                  |
| F                                    | RP                                   | POP                                  | PSW                                  | addr                                 | CP                                   | PUSH                                 | PSW                                  | RM                                   |                                      | JM                                   | addr                                 | CM                                   | addr                                 | CPI                                  | RST                                  |



| Tipo di istruzione | Numero dei cicli dell'istruzione | Cicli di clock | Tipo di istruzione | Numero dei cicli dell'istruzione | Cicli di clock |
|--------------------|----------------------------------|----------------|--------------------|----------------------------------|----------------|
| ALU R              | 1                                | 4              | LDAX               | 2                                | 7              |
| CMC                | 1                                | 4              | MVI                | 2                                | 7              |
| CMA                | 1                                | 4              | MOV M, R           | 2                                | 7              |
| DAA                | 1                                | 4              | MOV R, M           | 2                                | 7              |
| DCR R              | 1                                | 4              | STAX               | 2                                | 7              |
| DI                 | 1                                | 4              | CALL COND.         | 2/5                              | 9/18           |
| EI                 | 1                                | 4              | DAD                | 3                                | 10             |
| INR R              | 1                                | 4              | DCR                | 3                                | 10             |
| MOV R, R           | 1                                | 4              | IN                 | 3                                | 10             |
| NOP                | 1                                | 4              | INR M              | 3                                | 10             |
| ROTATE             | 1                                | 4              | IMP                | 3                                | 10             |
| RIM                | 1                                | 4              | LOARD PAIR         | 3                                | 10             |
| SIM                | 1                                | 4              | MVI N              | 3                                | 10             |
| STC                | 1                                | 4              | OUT                | 3                                | 10             |
| XCHG               | 1                                | 4              | POP                | 3                                | 10             |
| HLT                | 1                                | 5              | RET                | 3                                | 10             |
| DCX                | 1                                | 6              | PUSH               | 3                                | 12             |
| INX                | 1                                | 6              | RST                | 3                                | 12             |
| PCHL               | 1                                | 6              | LDA                | 2                                | 13             |
| RET COND.          | 1/3                              | 6/12           | STA                | 3                                | 13             |
| SPHL               | 1                                | 6              | LHLD               | 5                                | 16             |
| ALU I              | 2                                | 7              | SHLD               | 5                                | 16             |
| ALU M              | 2                                | 7              | XTHL               | 5                                | 16             |
| JNC                | 2/3                              | 7/10           | CALL               | 5                                | 18             |

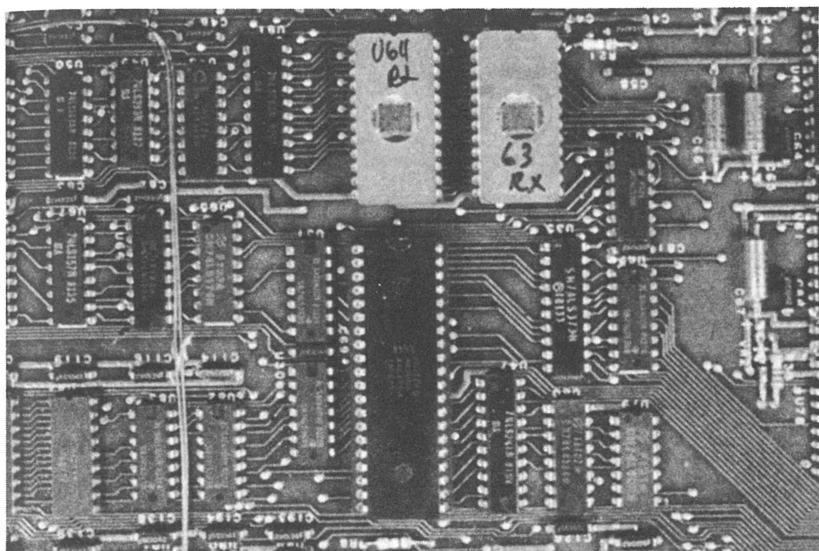
Tabella VIII.- Tabella delle istruzioni del  $\mu P$  8085 in funzione dei tempi di esecuzione.



*MICROPROCESSORI A 8 BIT PIU' COMUNI (II)*

Q

uesto capitolo intende fornire una panoramica delle più importanti caratteristiche del microprocessore a 8 bit della ZILOG, che è utilizzato in molti personal e home computer, tra i quali il popolare Spectrum.



*Foto 1.-Chip del microprocessore a 8 bit Z80, progettato dalla ZILOG, utilizzato come componente principale sulla piastra di un computer.*

## *Il microprocessore Z80*

E' stato posto in commercio dal 1976, e come il  $\mu$ P 8085, anche lo Z80 è un microprocessore simile all'8080, ma notevolmente migliorato. Anche lo Z80 è uno dei microprocessori più noti e più utilizzati, soprattutto negli home e nei personal computer, per cui verrà descritto in modo dettagliato.

Si fa rilevare che nel suo chip si trovano integrati circa ottomila transistori, quattromilacinquecento in più rispetto al predecessore 8080.

### *Caratteristiche dello Z80*

E' stato progettato dalla ZILOG, benchè attualmente esistano diverse case che lo costruiscono come: SGS, MOSTEK, NEC e SHARP.

- è costruito in tecnologia MOS a canale N, con gate al silicio;
- è un microprocessore con 8 bit per il bus dei dati e 16 bit per quello degli indirizzi, in grado quindi di indirizzare 64 Kbytes di memoria;
- è un microprocessore veloce, che ammette un clock di 2,5 MHz; nella versione Z80-A raggiunge i 4 MHz;
- possiede 158 istruzioni, che se si considerano i diversi modi di indirizzamento, possono arrivare fino a 696;
- i metodi di indirizzamento sono: implicito, immediato, relativo, diretto, e indicizzato;
- la sua alimentazione è unica, a +5 V;
- permette due tipi di interrupts INT (Maskable Interrupt) e NMI (Not Maskable Interrupt.).

Per quanto riguarda le particolarità, il  $\mu$ P Z80 dispone di:

- ingressi e uscite per indirizzare fino a 256 porte di accesso alle periferiche;
- un contatore a sette bit, ed i corrispondenti circuiti logici, per ottenere le funzioni di refresh per le memorie dinamiche collegate (DRAM);
- istruzioni adatte per la manipolazione, a livello di bit, dei registri e della memoria;
- istruzioni di copia e confronto a livello di blocchi.

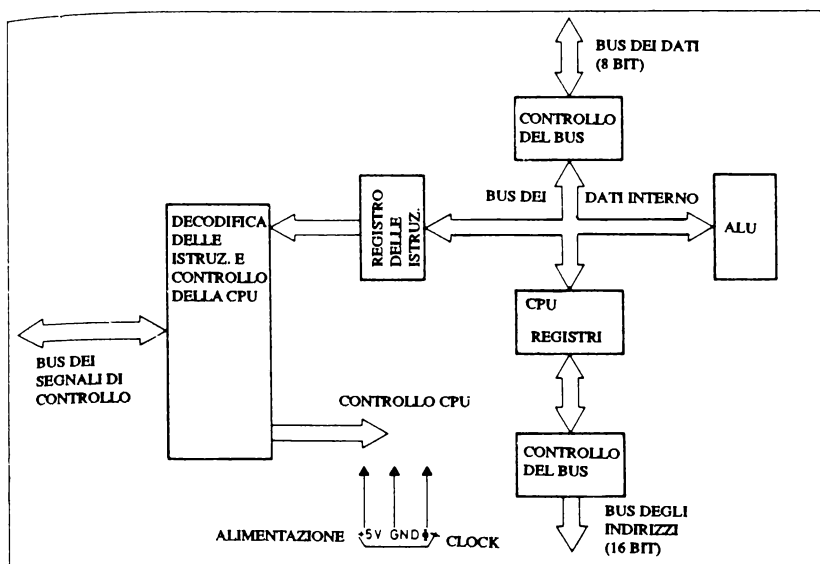


Fig. 1.-Blocchi interni del  $\mu P$  Z80.

## Terminali dello Z80

Il  $\mu P$  Z80 si trova incapsulato in un contenitore a 40 piedini, come rappresentato in Fig. 2, le cui funzioni sono:

- A0-A15 costituiscono il bus degli indirizzi a 16 bit per raggiungere i 64 Kbytes di memoria;
- D0-D7 costituiscono il bus dei dati a 8 bit, attraverso il quale i dati fluiscono in entrambi i sensi tra il  $\mu P$  e l'esterno;
- WR (Memory Write), quando tale terminale è a zero il  $\mu P$  indica all'esterno l'operazione di scrittura;
- RD (Memory Read), quando tale terminale è a zero, il  $\mu P$  indica all'esterno l'operazione di lettura;
- MREQ (Memory Request), quando il terminale è a livello basso, indica all'esterno che l'indirizzo del bus degli indirizzi è valido per leggere o scrivere;
- M1 (Machine Cycle One), tramite questo terminale il  $\mu P$  segnala all'esterno il primo ciclo di esecuzione di una istruzione (ciclo di ricerca);

- **HALT**, se il terminale è a zero il  $\mu P$  indica che in tale istante si arresta, in conseguenza dell'esecuzione dell'istruzione **HALT** proveniente da programma;
- **WAIT** è l'ingresso attraverso il quale le periferiche e memorie lente arrestano il  $\mu P$  per sincronizzarsi con esso, fatto che avviene quando viene portato a livello zero;
- **RESET**, portando a zero questo ingresso, si inizializza il funzionamento del  $\mu P$ ; ciò produce l'azzeramento del contatore di programma, in modo che quando è presente tale segnale il  $\mu P$  riparte dall'indirizzo 0000;
- **RFSH**, quando questa uscita si azzerà, indica che i sette bit meno significativi del bus degli indirizzi contengono l'indirizzo di refresh per le memorie dinamiche che si possono collegare al  $\mu P$ ;
- **IORQ** (Input/Output Request), se è basso indica che il  $\mu P$  non si rivolge alla memoria principale, ma ai circuiti di ingresso e uscita;
- **INT** (Interrupt Request) è un ingresso attraverso il quale le periferiche inviano una richiesta di interrupt al  $\mu P$ ; tale richiesta avviene quando l'ingresso viene posto a zero logico;
- **NMI** è un ingresso di interrupt non mascherabile. Questo ingresso si attiva con il fronte di discesa del segnale di interrupt inviato a tale terminale. L'interruzione richiesta tramite NMI ha una priorità assoluta sull'interruzione INT. Quando tale interrupt (NMI) viene attivato, il  $\mu P$  completa l'esecuzione dell'istruzione in corso e, indipendentemente dal contenuto del registro di stato, salta alla locazione 0066 esadecimale. Questo è l'indirizzo a cui deve iniziare la subroutine di trattamento dell'interrupt;
- **BUSRQ** (Bus Request) è un ingresso mediante il quale si portano ad alta impedenza le uscite del bus dei dati, degli indirizzi, e le uscite di controllo, che sono tutte del tipo three state. Questa possibilità è utile per mettere in parallelo vari microprocessori che accedono indistintamente, ma non simultaneamente, agli stessi indirizzi di memoria. Il  $\mu P$  porta le uscite ad alta impedenza, dopo aver completato l'istruzione in corso;
- **BUSAK** (Bus Acknowledge) è un'uscita per indicare all'esterno che il  $\mu P$  è scollegato per azione dell'ingresso BUSRQ. Ciò avviene se su tale uscita è presente uno zero logico;
- $\Phi$  è l'ingresso di clock che, per questo  $\mu P$ , ha una sola fase;
- **INGRESSI DI ALIMENTAZIONE**: l'alimentazione è unica a +5 V e deve essere collegata agli ingressi 11 e 29.

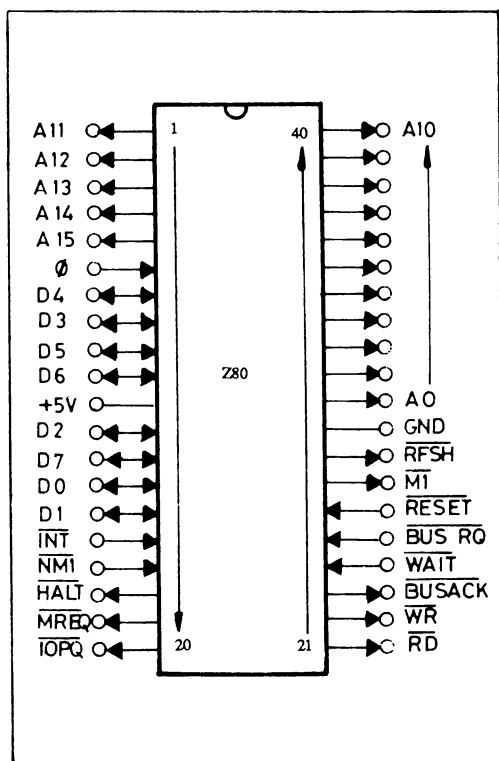


Fig. 2.-Terminali del  $\mu P$  Z80 e loro funzioni.

### *Dinamica dei segnali principali*

Nei diagrammi di Fig. 3 sono rappresentate le varie combinazioni dei segnali che escono dai relativi terminali nei momenti di lettura e scrittura nella memoria, e il tempo del ciclo di ricerca dell'istruzione. Tali tempi sono forniti, in quanto rappresentativi del  $\mu P$  Z80, poichè durante gli stessi agisce la maggioranza dei segnali descritti. Si fa notare che nel ciclo di lettura e scrittura, i segnali attivi sono RD e WR, e nel ciclo di ricerca M1.

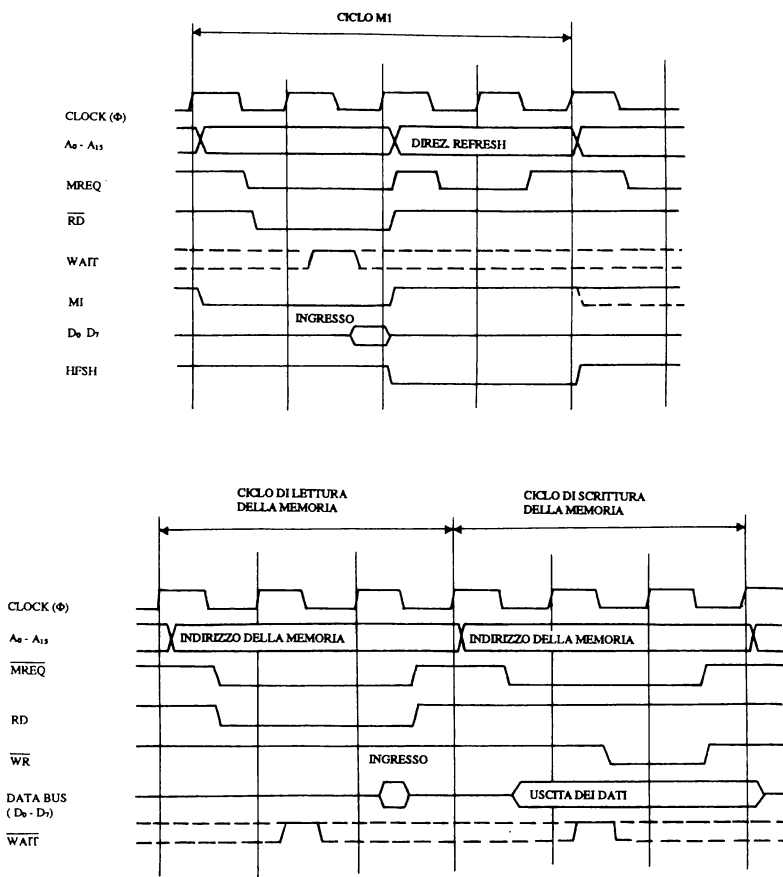


Fig. 3.-a) Diagramma dei tempi per il ciclo M1. b) Diagramma dei tempi di lettura e scrittura in memoria.

## I registri interni

Questo microprocessore contiene 22 registri disponibili per l'utente. Se si osserva la Fig. 4, in cui sono rappresentati, si può notare che quelli a sinistra si trovano duplicati a destra, cioè il  $\mu P$  dispone di due banchi completi di registri equivalenti. Il programmatore può lavorare con l'uno o l'altro gruppo di registri, passando dall'uno all'altro mediante le opportune istruzioni di scambio. I registri di uso generale, che sono tre, possono essere utilizzati come registri a 8 o 16 bit in base al tipo di istruzione. La funzione di ogni registro è la seguente:

- registri di uso generale: sono utilizzati in base alle esigenze del programmatore, e sono i registri accumulatori A e A'.



Da programma, si indica a quale dei due si vuole accedere. I registri B, C, D, E, H e L sono registri di uso generale, e si accede ad ognuno di essi in base all'indicazione dell'istruzione utilizzata. Si dispone anche di B', C', D', E', H' e L' nell'altro banco di registri. Come detto i registri possono essere accoppiati per lavorare con 16 bit. Quando si sceglie tale soluzione i registri si accoppiano come segue: B-C, D-E, H-L e B'-C', E'-D', H'-L'.

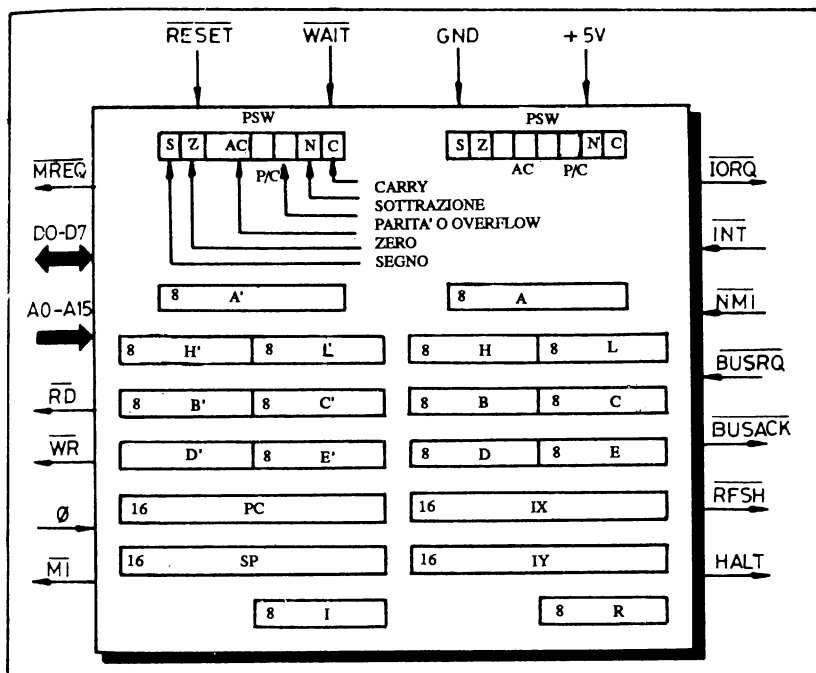


Fig. 4.-Registri interni del microprocessore Z80.

## *Registri di uso specifico nello Z80*

Il  $\mu$ P Z80 dispone di quattro registri di uso specifico a 16 bit, detti PC, SP, IX, IY, ed altri due registri specifici a otto bit detti I e R. Il loro funzionamento è il seguente:

- Pc (Program Counter), è il registro contatore di programma, a 16 bit, in cui è memorizzato l'indirizzo di memoria dell'istruzione in corso. Quando ne termina l'esecuzione, il PC si incrementa automaticamente di uno. Quando il programma presenta un'istruzione di salto, ne fornisce l'indirizzo al PC, per cui il salto viene eseguito e il programma continua dal nuovo indirizzo. E' il funzionamento tipico di tutti i registri PC;
- SP (Stack Pointer), è un registro a 16 bit il cui compito è quello di funzionare da puntatore, per indicare in quale indirizzo della RAM è ubicato lo stack delle subroutines; questo stack funziona come una memoria LIFO (Last Input First Output), il che significa che l'ultimo dato memorizzato è il primo ad essere recuperato. Il  $\mu$ P possiede due istruzioni dette PUSH e POP che, da programma, inseriscono e recuperano dati dallo stack.;
- IX e IY sono due registri indice indipendenti tra loro, ciascuno a 16 bit. Sono utilizzati dal  $\mu$ P come indice, per effettuare operazioni con indirizzamento indicizzato; tale tipo di indirizzamento semplifica la stesura di un programma, specialmente quando si accede a tabelle di dati.
- il registro I, detto Interrupt Page Address Register, è un registro a otto bit e può essere utilizzato quando è necessaria una chiamata indiretta a una locazione di memoria, in seguito ad un interrupt;
- il registro R, detto Memory Refresh Register, è ad 8 bit e può essere utilizzato per accedere a memorie dinamiche. Il contenuto di tale registro è automaticamente incrementato dopo ogni istruzione di ricerca. Questi otto bit possono uscire dal  $\mu$ P verso l'esterno, sulla parte bassa del bus degli indirizzi, per rinfrescare continuamente la memoria RAM dinamica a cui si vuole accedere.

## *Il registro di stato*

Il registro di stato di questo microprocessore è a 8 bit, dei quali solo sei

hanno un significato

- Bit 0=C (Carry), indica l'esistenza di un riporto nella parte più alta dell'accumulatore. E' utilizzato per le operazioni di somma e differenza, e in quelle di rotazione;
- Bit 6=Z (Zero), questo bit viene settato a uno se l'operazione eseguita ha risultato zero, oppure se è stato caricato uno zero nell'accumulatore, altrimenti rimane zero;
- Bit 7=S (Signe), tale bit indica il segno del dato contenuto nell'accumulatore, e verrà settato a uno quando il dato è negativo, a zero se il dato è positivo (un numero esadecimale è negativo se il bit 7 è uno);
- Bit 2=P/V (Parity/Overflow), questo bit può indicare due distinte funzioni: la parità di un risultato inviato all'accumulatore, quando è stata eseguita un'istruzione logica (Parità pari=1, Impari=0), e l'overflow o traboccamento quando è stata eseguita un'istruzione aritmetica;
- Bit 4=(H) è il bit che indica il carry di mezzo byte quando si opera in BCD, cioè il carry dei quattro bit meno significativi all'interno del byte;
- Bit 1=(N) indica il tipo di istruzione, somma o differenza, che è stata eseguita, operando in BCD.

## *Hardware di base per lo Z80*

Il sistema minimo di componenti che devono essere associati al micro-processore, perchè possa funzionare in modo completo, è visibile in Fig. 5. Il  $\mu P$  necessita, come è noto, di un clock generato esternamente, che gli viene fornito tramite il terminale detto  $\Phi$ .

I bus dei dati e degli indirizzi sono collegati in parallelo a tutto il sistema. Questo primo blocco deve essere formato da memorie ROM, in cui risiede il firmware specifico dell'applicazione del sistema, ed il cui indirizzo deve iniziare alla locazione 0000. Il secondo blocco è costituito dalla memoria RAM, necessaria per i diversi compiti di lettura e scrittura dei dati richiesti dal programma; nell'esempio è di soli 256 bytes. Il terzo blocco è costituito da un insieme di registri disposti in modo da costituire una porta per l'accesso alle periferiche.

La parte più interessante e specifica di questo circuito è il bus di controllo, integrato tramite i terminali detti MREQ, RD, WR, IORQ, e M1, le cui fun-

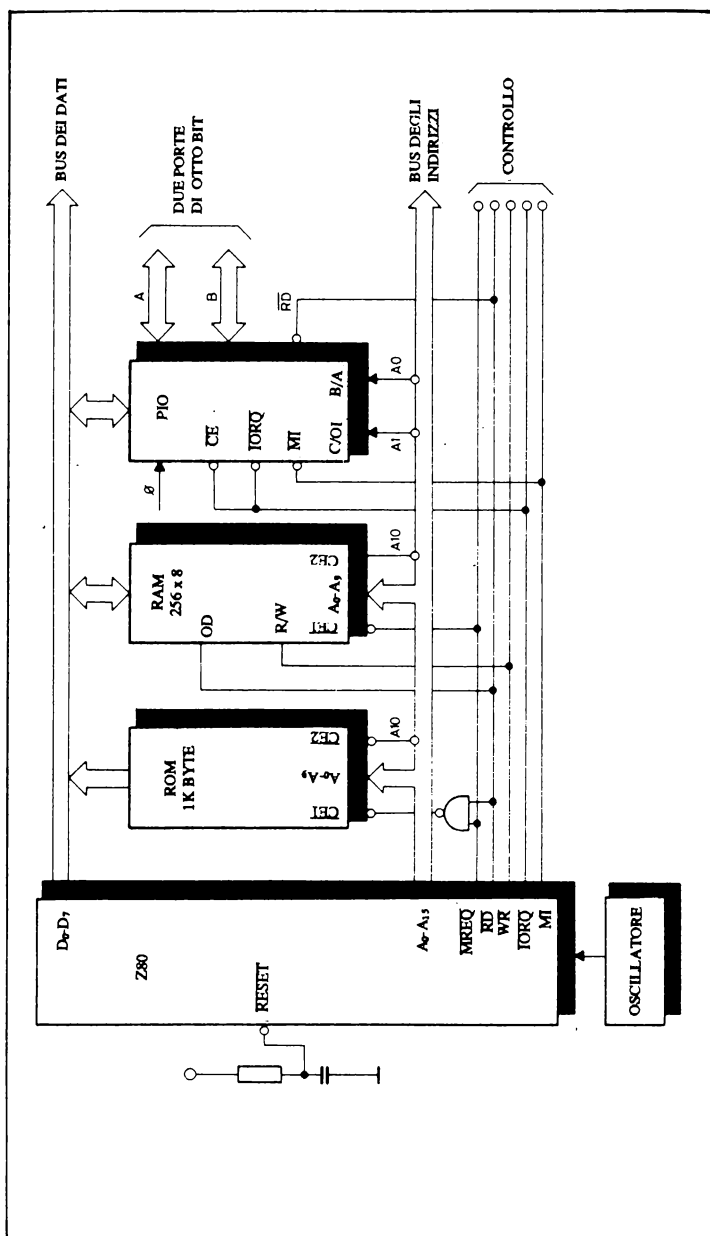


Fig. 5.-Semplice schema di collegamento del  $\mu P$  Z80.

zioni sono note.

MREQ è un segnale di permesso per accedere alla memoria, e andrà perciò collegato a quei blocchi che funzionano come memoria, cioè le ROM e la RAM, mentre il segnale IORQ che permette l'accesso alle periferiche, andrà collegato a quei circuiti che a queste accedono, come le PIO (Periferal Input/Output).

Questi due segnali, così disposti, distinguono un blocco dall'altro: i segnali RD e WR sono quelli che indicano se l'operazione è di scrittura o di lettura, per tanto accederanno alla memoria (alla ROM accede solo il segnale RD, in quanto può solo essere letta).

Infine il segnale M1, che è un segnale di sincronismo, dovrà essere collegato a quei circuiti per cui è necessaria una esatta sincronia con l'esecuzione dell'istruzione (si ricordi che M1 compare durante la prima parte dell'esecuzione dell'istruzione); come si vede nello schema la PIO necessita del segnale M1.

## *Gli interrupts nel $\mu$ P Z80*

Il microprocessore Z80 ha due ingressi di interrupt: l'ingresso di interrupt non mascherabile NMI e l'ingresso di interrupt mascherabile INT.

Quando viene generato l'interrupt NMI, non può essere disattivato da programma, e verrà accettato ogni volta che una periferica lo richiede. Questo interrupt sarà generalmente riservato alle funzioni più importanti, che devono necessariamente essere soddisfatte nel momento in cui si producono, come ad esempio la mancanza di alimentazione, ecc.

L'interrupt mascherabile INT può essere o meno permesso, in base alla necessità, dal programma. Questo fatto consente al programmatore di non preoccuparsi delle interruzioni nelle zone del programma che non devono essere interrotte.

Per quanto sopra, qualsiasi periferica collegata al terminale INT potrà richiedere un'interruzione in qualsiasi momento, ma questa non sarà soddisfatta se il programma ha preventivamente bloccato la possibilità di accesso.

La richiesta di interrupt potrà essere invece soddisfatta quando il programma la permetterà nuovamente.

All'interno del  $\mu$ P esiste un bistabile detto IFF, che può essere variato dal programma utilizzando l'istruzione EI (Enable Interrupt), per settarlo a uno, permettendo gli interrupts, e l'istruzione DI (Disable Interrupt), per resettarlo a zero, bloccando così l'ingresso di interrupt.

## *Uso degli interrupts*

Quanto detto relativamente al bistabile IFF è valido solo concettualmente, poichè in realtà non è costituito da un solo bistabile ma da due, detti: IFF<sub>1</sub>, che è quello che permette o meno gli interrupts, e IFF<sub>2</sub>, che può essere considerato come una locazione di memorizzazione temporanea dello stato di IFF<sub>1</sub>.

Quando avviene il reset generale del  $\mu P$ , all'accensione, questi due bistabili sono entrambi a zero e quindi gli interrupts saranno bloccati. Quando si esegue un'istruzione EI entrambi i bistabili vengono settati a uno, rendendo possibili gli interrupts. Se avviene un'interruzione, questa viene soddisfatta, ed entrambi i bistabili vengono automaticamente resettati a zero. In questo caso, ambedue funzionano all'unisono e necessitano di essere settati a uno da programma, dopo ogni trattamento di eventuali interrupts.

Lo scopo di IFF<sub>2</sub> è di salvaguardare il contenuto di IFF<sub>1</sub>, prima che sia cancellato. Tale funzionamento dei bistabili permette di conoscere in ogni momento quali interrupts sono stati eseguiti, e se ne rimane qualcuno in attesa di essere soddisfatto.

Il bistabile IFF può essere reinizializzato da programma, o tramite un'istruzione di ritorno da una subroutine di interrupt non mascherabile. Si rammenta che il trattamento di un interrupt si effettua generalmente mediante un sottoprogramma.

Gli interrupts, in generale, sono utili in qualsiasi punto del programma, poichè il programmatore può dimenticarsi di tenerli presenti. Quando avviene un interrupt, l'unica cosa che dovrà essere considerata è di creare una subroutine alla fine del programma principale, nella quale questa venga adeguatamente trattata; occorrerà però bloccare gli interrupts successivi, quando se ne sta già trattando qualcuno.

Programmare tutto il sistema di un microelaboratore e il suo sistema di interrupt, è soltanto questione di pratica e di perfetta conoscenza dello stesso. Si ricorda che si stanno trattando solo concetti generali e funzioni particolari dei sistemi; per approfondire questi argomenti occorre necessariamente ricorrere ai volumi informativi forniti dai costruttori.

## *Tabella delle istruzioni dello Z80*

Nella tabella delle istruzioni di questo microprocessore, i codici mnemonici o nomi inglesi delle istruzioni sono rappresentativi e hanno un riferimento diretto con ciascuna istruzione. I microprocessori possiedono pressochè le stesse istruzioni, ma alcune differiscono tra loro in base al criterio con cui il microprocessore è stato progettato. Nei microprocessori più evoluti si nota un

| Codice dell'Operazione | Mnemonico | Codice dell'Operazione | Mnemonico | Codice dell'Operazione | Mnemonico | Codice dell'Operazione | Mnemonico | Codice dell'Operazione | Mnemonico |
|------------------------|-----------|------------------------|-----------|------------------------|-----------|------------------------|-----------|------------------------|-----------|
| 00                     | NOP       | 56                     | LD        | D7                     | RST       | D7                     | RST       | ED 67                  | RST       |
| 01 yyyy                | LD        | 57 iss                 | LD        | D8                     | RET       | D8                     | RET       | ED 6F                  | RST       |
| 02                     | LD        | 5E                     | LD        | D9                     | EXX       | D9                     | EXX       | ED A0                  | RST       |
| 03                     | INC       | 6 0ass                 | LD        | DA ppqq                | IN        | DA ppqq                | IN        | ED A1                  | RST       |
| 04                     | INC       | 66                     | LD        | DB yy                  | CALL      | DB yy                  | CALL      | ED A2                  | RST       |
| 05                     | DEC       | 6 1ass                 | LD        | DC ppqq                | C, addr   | DC ppqq                | C, addr   | ED A3                  | RST       |
| 06 yy                  | LD        | 6E                     | LD        | DD 00xx 9              | ADD       | DD 00xx 9              | ADD       | ED A8                  | RST       |
| 07                     | RUCA      | 7 0ass                 | LD        | DD 21 yyyy             | LD        | DD 21 yyyy             | LD        | ED A9                  | RST       |
| 08                     | EX        | 75                     | HALT      | DD 22 ppqq             | LD        | DD 22 ppqq             | LD        | ED AA                  | RST       |
| 09                     | ADD       | 7 1ass                 | LD        | DD 2A ppqq             | INC       | DD 2A ppqq             | INC       | ED AB                  | RST       |
| 0A                     | LD        | 7E                     | LD        | DD 2B                  | LD        | DD 2B                  | LD        | ED B0                  | RST       |
| 0B                     | DEC       | 8 0trr                 | ADD       | DD 34 disp             | DEC       | DD 34 disp             | DEC       | ED B1                  | RST       |
| 0C                     | INC       | 86                     | ADD       | DD 35 disp             | INC       | DD 35 disp             | INC       | ED B2                  | RST       |
| 0D                     | DEC       | 8 1trr                 | ADC       | DD 36 disp yy          | DEC       | DD 36 disp yy          | DEC       | ED B3                  | RST       |
| 0E yy                  | LD        | 8E                     | ADC       | DD 01 dddd 10 disp     | LD        | DD 01 dddd 10 disp     | LD        | ED B8                  | RST       |
| 0F                     | PRCA      | 9 0trr                 | SUB       | DD 7 0ass disp         | LD        | DD 7 0ass disp         | LD        | ED BA                  | RST       |
| 10 disp-2              | DJNZ      | 96                     | SUB       | DD 86 disp             | LD        | DD 86 disp             | LD        | ED BB                  | RST       |
| 11 yyyy                | LD        | 9 1trr                 | SBC       | DD 8E disp             | LD        | DD 8E disp             | LD        | EE yy                  | RST       |
| 12                     | LD        | 9 E                    | SBC       | DD 9E disp             | LD        | DD 9E disp             | LD        | F0                     | RST       |
| 13                     | INC       | A 0trr                 | AND       | DD AE disp             | LD        | DD AE disp             | LD        | F1                     | RST       |
| 14                     | INC       | A5                     | AND       | DD BE disp             | LD        | DD BE disp             | LD        | F2 ppqq                | RST       |
| 15                     | DEC       | A 1trr                 | XOR       | DD CB disp 06          | LD        | DD CB disp 06          | LD        | F3                     | RST       |
| 16 yy                  | LD        | AE                     | XOR       | DD CB disp 0E          | LD        | DD CB disp 0E          | LD        | F4 ppqq                | RST       |
| 17                     | RLA       | B 0trr                 | OR        | DD CB disp 16          | LD        | DD CB disp 16          | LD        | F5                     | RST       |
| 18 disp-2              | JR        | B6                     | OR        | DD CB disp 1E          | LD        | DD CB disp 1E          | LD        | F6 yy                  | RST       |
| 19                     | ADD       | B 1trr                 | CP        | DD CB disp 2E          | LD        | DD CB disp 2E          | LD        | F7                     | RST       |
| 1A                     | LD        | BE                     | CP        |                        | LD        |                        | LD        | F8                     | RST       |
| 1B                     | DEC       | C0                     | RET       |                        | LD        |                        | LD        | F9                     | RST       |
| 1C                     | INC       | C1                     | POP       |                        | LD        |                        | LD        | FA ppqq                | RST       |
| 1D                     | DEC       | E                      | JP        |                        | LD        |                        | LD        |                        |           |
| 1E yy                  | LD        | E, data                | JP        |                        | LD        |                        | LD        |                        |           |

| Codice del-<br>Operazione | Mnemonico | Codice del-<br>Operazione | Mnemonico | Codice del-<br>Operazione | Mnemonico | Codice del-<br>Operazione | Mnemonico | Codice del-<br>Operazione | Mnemonico       |
|---------------------------|-----------|---------------------------|-----------|---------------------------|-----------|---------------------------|-----------|---------------------------|-----------------|
| 1F                        | RRA       | C4 ppoq                   | CALL      | NZ, addr                  | CALL      | DD CB disp 3E             | SRL       | FB                        | EI              |
| 20 disp-2                 | JR        | C5                        | PUSH      | BC                        | PUSH      | DD CB disp0bbb10          | BIT       | FC ppoq                   | CALL            |
| 21 yyy                    | LD        | C6 yy                     | ADD       | A, data                   | ADD       | DD CB disp10bbb10         | RES       | FD 00xx9                  | 1Y, rr          |
| 22 ppoq                   | LD        | C7                        | RST       | 00H                       | RST       | DD CB disp11bbb10         | SET       | FD 21 yyy                 | 1Y, data 16     |
| 23                        | INC       | C8                        | RET       | Z                         | RET       | DD EI                     | POP       | FD 22 ppoq                | (addr), 1Y      |
| 24                        | INC       | C9                        | RET       |                           | RET       | DD E1                     | EX        | FD 23                     | 1Y              |
| 25                        | DEC       | CA ppoq                   | JP        | z, addr                   | JP        | DD E3                     | PUSH      | FD 2A ppoq                | 1Y, (addr)      |
| 26 yy                     | LD        | CB 00rr                   | RLC       | reg                       | RLC       | DD E3                     | JP        | FD 2B                     | 1Y              |
| 27                        | DAA       | CB 06                     | RRC       | (HL)                      | RRC       | DD F9                     | LD        | FD 2A ppoq                | (IX)            |
| 28 disp-2                 | JR        | CB 01rr                   | RRC       | reg                       | RRC       | DD yy                     | SBC       | FD 34 disp                | (IX+disp)       |
| 29                        | ADD       | CB 0E                     | RRC       | (HL)                      | RRC       | DD yy                     | RST       | FD 35 disp                | (IX+disp)       |
| 2A ppoq                   | LD        | CB 10rr                   | RL        | reg                       | RL        | DF                        | POP       | FD 36 disp yy             | (IX+disp), data |
| 2B                        | DEC       | CB 16                     | RL        | (HL)                      | RL        | E0                        | RET       | FD 01ddd110 disp          | reg (1Y+disp)   |
| 2C                        | INC       | CB 11rr                   | RR        | reg                       | RR        | E1                        | POP       | FD 7 0sss disp            | (IX+disp) reg   |
| 2D                        | DEC       | CB 1E                     | RR        | (HL)                      | RR        | E2 ppoq                   | JP        | FD 86 disp                | A, (1Y+disp)    |
| 2E                        | LD        | CB 10rr                   | SRA       | reg                       | SRA       | E3                        | EX        | FD 8E disp                | A, (1Y+disp)    |
| 2F                        | CPL       | CB 26                     | SRA       | (HL)                      | SRA       | E4 ppoq                   | CALL      | FD 96 disp                | SUB             |
| 30 disp-2                 | JR        | CB 21rr                   | SRA       | reg                       | SRA       | E5                        | PUSH      | FD 9E disp                | SBC             |
| 31 yyy                    | LD        | CB 2E                     | SRA       | (HL)                      | SRA       | E6 yy                     | AND       | FD A6 disp                | AND             |
| 32 ppoq                   | LD        | CB 31rr                   | SRA       | reg                       | SRA       | E7                        | RST       | FD AE disp                | (IX+disp)       |
| 33                        | INC       | CB 3E                     | SRL       | (HL)                      | SRL       | E8                        | RET       | FD B6 disp                | (IX+disp)       |
| 34                        | INC       | CB 01bbrr                 | BIT       | b, reg                    | BIT       | E9                        | JP        | FD BE disp                | (IX+disp)       |
| 35                        | DEC       | CB 01bb10                 | BIT       | b, (HL)                   | BIT       | EA ppoq                   | JP        | FD CB disp 06             | (IX+disp)       |
| 36 yy                     | LD        | CB 10bbrr                 | RES       | b, reg                    | RES       | EB                        | EX        | FD CB disp 0E             | (IX+disp)       |
| 37                        | SCF       | CB 10bb10                 | RES       | b, (HL)                   | RES       | EC ppoq                   | CALL      | FD CB disp 1E             | (IX+disp)       |
| 38                        | JR        | CB 11bbrr                 | SET       | b, reg                    | SET       | ED 01ddd000               | IN        | FD CB disp 16             | (IX+disp)       |
| 39                        | ADD       | CB 11bb10                 | SET       | b, (HL)                   | SET       | ED 01sss001               | OUT       | FD CB disp 26             | (IX+disp)       |
| 3A ppoq                   | LD        | CC ppoq                   | CALL      | Z, addr                   | CALL      | ED 01xc2                  | SBC       | FD CB disp 2E             | (IX+disp)       |
| 3B                        | DEC       | CD ppoq                   | CALL      | addr                      | CALL      | ED 01xc3 ppoq             | LD        | FD CB disp 3E             | (IX+disp)       |
| 3C                        | INC       | CE yy                     | ADC       | A, data                   | ADC       | ED 44                     | NEG       | FD CB disp01bbb110        | b, (1Y+disp)    |
| 3D                        | DEC       | CF                        | RST       | 08H                       | RST       | ED 45                     | RETIN     | FD CB disp0bbb110         | b, (1Y+disp)    |
|                           |           |                           |           |                           |           | ED 010ml10                | IM        | FD CB disp11bbb110        | b, (1Y+disp)    |





| Mnemonico    | Simbolo della operazione | Registro di stato |   |   |     |   |   |   |   | N° dei bytes             | N° dei cicli | N° dei periodi di clock | Commenti |     |       |
|--------------|--------------------------|-------------------|---|---|-----|---|---|---|---|--------------------------|--------------|-------------------------|----------|-----|-------|
|              |                          | 7                 | 6 | 5 | 4   | 3 | 2 | 1 | 0 |                          |              |                         |          |     |       |
|              |                          | S                 | Z | H | P/V | N | C |   |   |                          |              |                         |          |     |       |
|              |                          | Codice macchina   |   |   |     |   |   |   |   |                          |              |                         |          | HEX |       |
|              |                          |                   |   |   |     |   |   |   |   |                          |              |                         |          |     |       |
| LD r, (Y+d)  | $r \leftarrow (Y+d)$     | •                 | • | X | •   | X | • | • | • | 11 111 101<br>01 r 110   | FD           | 3                       | 5        | 19  | 111 A |
| LD (HL), r   | $(HL) \leftarrow r$      | •                 | • | • | •   | X | • | • | • | 01 110 r                 |              | 1                       | 2        | 7   |       |
| LD (IX+d), r | $(IX+d) \leftarrow r$    | •                 | • | X | •   | X | • | • | • | 11 011 101<br>01 110 r   | DD           | 3                       | 5        | 19  |       |
| LD (Y+d), r  | $(Y+d) \leftarrow r$     | •                 | • | • | •   | X | • | • | • | 11 111 101<br>01 110 r   | FD           | 3                       | 5        | 19  |       |
| LD (HL), n   | $(HL) \leftarrow n$      | •                 | • | • | •   | X | • | • | • | 00 110 110               | 36           | 2                       | 3        | 10  |       |
| LD (IX+d), n | $(IX+d) \leftarrow n$    | •                 | • | • | •   | X | • | • | • | 11 011 101<br>00 110 110 | DD 36        | 4                       | 5        | 19  |       |
| LD (Y+d), n  | $(Y+d) \leftarrow n$     | •                 | • | • | •   | X | • | • | • | 11 111 101<br>00 110 110 | FD 36        | 4                       | 5        | 19  |       |
| LD A, (BC)   | $A \leftarrow (BC)$      | •                 | • | X | •   | • | • | • | • | 00 001 010               | 0A           | 1                       | 2        | 7   |       |
| LD A, (DE)   | $A \leftarrow (DE)$      | •                 | • | X | •   | • | • | • | • | 00 011 010               | 1A           | 1                       | 2        | 7   |       |
| LD A, (nn)   | $A \leftarrow (nn)$      | •                 | • | X | •   | • | • | • | • | 00 111 010               | 3A           | 3                       | 4        | 13  |       |
| LD (BC), A   | $(BC) \leftarrow A$      | •                 | • | • | •   | X | • | • | • | 00 000 010               | 02           | 1                       | 2        | 7   |       |
| LD (DE), A   | $(DE) \leftarrow A$      | •                 | • | • | •   | X | • | • | • | 00 010 010               | 12           | 1                       | 2        | 7   |       |
| LD (nn), A   | $(nn) \leftarrow A$      | •                 | • | • | •   | X | • | • | • | 00 110 010               | 32           | 3                       | 4        | 13  |       |

| Mnemonico   | Simbolo della operazione                         | Registro di stato |   |   |   |   |     |   |   | Codice macchina |     |     | N° dei bytes | N° dei cicli | N° dei periodi di clock | Commenti                                    |
|-------------|--|-------------------|---|---|---|---|-----|---|---|-----------------|-----|-----|--------------|--------------|-------------------------|---|
|             |  |                   |   |   |   |   |     |   |   |                 |     |     |              |              |                         |   |
|             |  | 7                 | 6 | 5 | 4 | 3 | 2   | 1 | 0 |                 |     |     |              |              |                         |   |
|             |  | S                 | Z |   | H |   | P/V | N | C | 76              | 543 | 210 | HEX          |              |                         |   |
| LD A, I     | $A \leftarrow I$                                 | ↑                 | ↑ | X | 0 | X | IFF | 0 | ● | 11              | 101 | 101 | ED           | 2            | 9                       | dd PARI<br>00 BC<br>01 DE<br>10 HL<br>11 SP |
| LD A, R     | $A \leftarrow R$                                 | ↑                 | ↑ | X | 0 | X | IFF | 0 | ● | 01              | 010 | 111 | 57           | 2            | 9                       |   |
| LD I, A     | $I \leftarrow A$                                 | ●                 | ● | X | ● | ● | ●   | ● | ● | 11              | 101 | 101 | ED           | 2            | 9                       |   |
| LD R, A     | $R \leftarrow A$                                 | ●                 | ● | X | ● | ● | ●   | ● | ● | 01              | 000 | 111 | 47           | 2            | 9                       |   |
| LD dd, nn   | $dd \leftarrow nn$                               | ●                 | ● | X | ● | ● | ●   | ● | ● | 00              | dd0 | 001 | 4F           | 3            | 10                      |   |
| LD IX, nn   | $IX \leftarrow nn$                               | ●                 | ● | X | ● | ● | ●   | ● | ● | 11              | 011 | 101 | DD           | 4            | 14                      |   |
| LD IY, nn   | $IY \leftarrow nn$                               | ●                 | ● | X | ● | ● | ●   | ● | ● | 00              | 100 | 001 | 21           | 4            | 14                      |   |
| LD HL, (nn) | $H \leftarrow (nn+1)$<br>$L \leftarrow (nn)$     | ●                 | ● | X | ● | ● | ●   | ● | ● | 11              | 111 | 101 | FD           | 3            | 16                      |   |
| LD dd, (nn) | $ddH \leftarrow (nn+1)$<br>$ddL \leftarrow (nn)$ | ●                 | ● | X | ● | ● | ●   | ● | ● | 00              | 101 | 101 | ED           | 4            | 20                      |   |

| Mnemonico   | Simbolo della operazione   | Registro di stato |   |   |   |   |   |   |   | Codice macchina |   |   |     | N° dei bytes | N° dei cicli | N° dei periodi di clock | Commenti |     |    |    |     |     |     |
|-------------|----------------------------|-------------------|---|---|---|---|---|---|---|-----------------|---|---|-----|--------------|--------------|-------------------------|----------|-----|----|----|-----|-----|-----|
|             |                            |                   |   |   |   |   |   |   |   |                 |   |   |     |              |              |                         |          |     |    |    |     |     |     |
|             |                            |                   |   |   |   |   |   |   |   |                 |   |   |     |              |              |                         |          |     |    |    |     |     |     |
|             |                            | 7                 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | S               | Z | H | P/V |              |              |                         |          | N   | C  | 76 | 543 | 210 | HEX |
| LD IX, (nn) | IXH ← (nn+1)<br>IXL ← (nn) | •                 | • | X | • | X | • | • | • | •               | • | • | •   | •            | 11           | 011                     | 101      | DD  | 4  | 6  | 20  |     |     |
|             |                            |                   |   |   |   |   |   |   |   |                 |   |   |     |              | 00           | 101                     | 010      | 2A  |    |    |     |     |     |
| LD IY, (nn) | IYH ← (nn+1)<br>IYL ← (nn) | •                 | • | X | • | X | • | • | • | •               | • | • | •   | •            | •            | 11                      | 111      | 101 | FD | 4  | 6   | 20  |     |
|             |                            |                   |   |   |   |   |   |   |   |                 |   |   |     |              | 00           | 101                     | 010      | 2A  |    |    |     |     |     |
| LD (nn), HL | (nn+1) ← H<br>(nn) ← L     | •                 | • | X | • | X | • | • | • | •               | • | • | •   | •            | •            | 00                      | 100      | 010 | 22 | 3  | 5   | 16  |     |
|             |                            |                   |   |   |   |   |   |   |   |                 |   |   |     |              | •            | •                       | •        | •   |    |    |     |     |     |
| LD (nn), dd | (nn+1) ← ddH<br>(nn) ← ddL | •                 | • | X | • | X | • | • | • | •               | • | • | •   | •            | •            | 11                      | 101      | 101 | ED | 4  | 6   | 20  |     |
|             |                            |                   |   |   |   |   |   |   |   |                 |   |   |     |              | 01           | dd0                     | 011      |     |    |    |     |     |     |
| LD (nn), IX | (nn+1) ← IXH<br>(nn) ← IXL | •                 | • | X | • | X | • | • | • | •               | • | • | •   | •            | •            | 11                      | 011      | 101 | DD | 4  | 6   | 20  |     |
|             |                            |                   |   |   |   |   |   |   |   |                 |   |   |     |              | 00           | 100                     | 010      | 22  |    |    |     |     |     |
| LD (nn), IY | (nn+1) ← IYH<br>(nn) ← IYL | •                 | • | X | • | X | • | • | • | •               | • | • | •   | •            | •            | 11                      | 111      | 101 | FD | 4  | 6   | 20  |     |
|             |                            |                   |   |   |   |   |   |   |   |                 |   |   |     |              | 00           | 100                     | 010      | 22  |    |    |     |     |     |
| LD SP, HL   | SP ← HL                    | •                 | • | X | • | X | • | • | • | •               | • | • | •   | •            | •            | 11                      | 111      | 001 | F9 | 1  | 1   | 6   |     |
| LD SP, IX   | SP ← IX                    | •                 | • | X | • | X | • | • | • | •               | • | • | •   | •            | •            | 11                      | 011      | 101 | DD | 2  | 2   | 10  |     |
|             |                            |                   |   |   |   |   |   |   |   |                 |   |   |     |              | 11           | 111                     | 001      | F9  |    |    |     |     |     |
| LD SP, IY   | SP ← IY                    | •                 | • | X | • | X | • | • | • | •               | • | • | •   | •            | •            | 11                      | 111      | 101 | FD | 2  | 2   | 10  |     |
|             |                            |                   |   |   |   |   |   |   |   |                 |   |   |     |              | 11           | 111                     | 001      | F9  |    |    |     |     |     |
|             |                            |                   |   |   |   |   |   |   |   |                 |   |   |     |              |              |                         | qq PARI  |     |    |    |     |     |     |

qq PARI

| Mnemonico | Simbolo della operazione     | Registro di stato |   |   |     |   |   |    |     | Codice macchina |     |     |     | N° dei bytes | N° dei cicli | N° dei periodi di clock | Commenti                         |
|-----------|------------------------------|-------------------|---|---|-----|---|---|----|-----|-----------------|-----|-----|-----|--------------|--------------|-------------------------|----------------------------------|
|           |                              | 7                 | 6 | 5 | 4   | 3 | 2 | 1  | 0   |                 |     |     |     |              |              |                         |                                  |
|           |                              | S                 | Z | H | P/V | N | C | 76 | 543 | 210             | HEX |     |     |              |              |                         |                                  |
| PUSH qq   | (SP-2) ← qqL<br>(SP-1) ← qqH | ●                 | ● | X | ●   | X | ● | ●  | ●   | ●               | 11  | qq0 | 101 | 1            | 3            | 11                      | 00 BC<br>01 DE<br>10 HL<br>11 AF |
| PUSH IX   | (SP-2) ← IXL<br>(SP-1) ← IXH | ●                 | ● | X | ●   | X | ● | ●  | ●   | ●               | 11  | 011 | 101 | DD           | 2            | 4                       | 15                               |
| PUSH IY   | (SP-2) ← IYL<br>(SP-1) ← IYH | ●                 | ● | X | ●   | X | ● | ●  | ●   | ●               | 11  | 100 | 101 | E5           | 2            | 4                       | 15                               |
| POP qq    | (SP-1) ← qqL<br>(SP-2) ← qqH | ●                 | ● | X | ●   | X | ● | ●  | ●   | ●               | 11  | 100 | 101 | E5           | 1            | 3                       | 10                               |
| POP IX    | (SP-1) ← IXL<br>(SP-2) ← IXH | ●                 | ● | X | ●   | X | ● | ●  | ●   | ●               | 11  | 011 | 101 | DD           | 2            | 4                       | 14                               |
| POP IY    | (SP-1) ← IYL<br>(SP-2) ← IYH | ●                 | ● | X | ●   | X | ● | ●  | ●   | ●               | 11  | 100 | 001 | E1           | 2            | 4                       | 14                               |

1, 2 = Rappresentano qualsiasi registro A, B, C, D, E, H, L.  
● = Bit di stato non influenzato, 0 = Bit azzerato a 1, X = Bit indifferente (0 o 1), 1 = Bit influenzato secondo il risultato dell'operazione.  
dd = Rappresenta qualsiasi coppia di registri BC, DE, HL, SP.  
qq = Rappresenta le coppie di registri AF, BC, DE, HL.  
BCI, ad esempio, rappresenta un registro a 8 bit della coppia BC, in questo caso, il C, poiché L indica l'ottavo meno significativo.

*r,d* = Rappresentano qualsiasi registro A, B, C, D, E, H, L.  
 \* = Bit di stato non influenzato, 0 = Bit azzerato, 1 = Bit settato a 1, X = Bit influenzato secondo il risultato dell'operazione.  
 dd = Rappresenta qualsiasi coppia di registri BC, DE, HL, SP.  
 qq = Rappresenta le coppie di registri AF, BC, DE, HL.  
 BCL, ad esempio, rappresenta un registro a 8 bit della coppia BC, in questo caso, il C, poiché L indica l'ottavo meno significativo.

**Gruppo di istruzioni di scambio, trasferimento e scelta**

| Mnemonic                                   | Simbolo della operazione   | Registro di stato |   |   |   |   |   |   |   | Codice macchina |   |   |    |     | N° dei bytes | N° dei cicli | N° dei periodi di clock | Commenti |     |     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |     |
|--|--|-------------------|---|---|---|---|---|---|---|-----------------|---|---|----|-----|--------------|--------------|-------------------------|----------|-----|-----|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|-----|
|  |  |                   |   |   |   |   |   |   |   |                 |   |   |    |     |              |              |                         |          |     |     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |     |
|  |  | 7                 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | P/V             | N | C | 76 | 543 |              |              |                         |          | 210 | HEX |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |     |
| EX DE, HL<br>EX AF, AF <sup>+</sup><br>EXX | DE ↔ HL<br>AF ↔ AF <sup>+</sup><br>(BC ↔ BC <sup>+</sup> )<br>DE ↔ DE <sup>+</sup><br>HL ↔ HL <sup>+</sup> | •                 | • | • | • | X | • | X | • | •               | • | • | •  | •   | •            | •            | •                       | •        | •   | •   | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | • | •</ |

**Gruppo di istruzioni di scambio, trasferimento e scelta**

| Mnemonico | Simbolo della operazione    | Registro di stato |   |   |     |   |   |    |     | N° dei bytes | N° dei cicli | N° dei periodi di clock | Commenti |
|-----------|-----------------------------|-------------------|---|---|-----|---|---|----|-----|--------------|--------------|-------------------------|----------|
|           |                             | Codice macchina   |   |   |     |   |   |    |     |              |              |                         |          |
|           |                             | 7                 | 6 | 5 | 4   | 3 | 2 | 1  | 0   |              |              |                         |          |
|           |                             | S                 | Z | H | P/V | N | C | 76 | 543 | 210          | HEX          |                         |          |
| CPI       | BC ← BC-1                   |                   |   |   |     |   |   |    |     |              |              |                         |          |
|           | Repeat until BC=0           |                   |   |   |     |   |   |    |     |              |              |                         |          |
|           | A ← (HL)                    |                   |   |   |     |   |   |    |     |              |              |                         |          |
|           | HL ← HL+1                   |                   |   |   |     |   |   |    |     |              |              |                         |          |
| CPIR      | BC ← BC-1                   |                   |   |   |     |   |   |    |     |              |              |                         |          |
|           | A ← (HL)                    |                   |   |   |     |   |   |    |     |              |              |                         |          |
|           | HL ← HL+1                   |                   |   |   |     |   |   |    |     |              |              |                         |          |
|           | BC ← BC-1                   |                   |   |   |     |   |   |    |     |              |              |                         |          |
| CPD       | Repeat until A=(HL) or BC=0 |                   |   |   |     |   |   |    |     |              |              |                         |          |
|           | A ← (HL)                    |                   |   |   |     |   |   |    |     |              |              |                         |          |
|           | HL ← HL-1                   |                   |   |   |     |   |   |    |     |              |              |                         |          |
|           | BC ← BC-1                   |                   |   |   |     |   |   |    |     |              |              |                         |          |
| CPDR      | Repeat until A=(HL) or BC=0 |                   |   |   |     |   |   |    |     |              |              |                         |          |
|           | A ← (HL)                    |                   |   |   |     |   |   |    |     |              |              |                         |          |
|           | HL ← HL-1                   |                   |   |   |     |   |   |    |     |              |              |                         |          |
|           | BC ← BC-1                   |                   |   |   |     |   |   |    |     |              |              |                         |          |

**Note:**  
 - Il bit P/V è posto a zero se il risultato di BC-1=0, altrimenti P/V è 1. Il bit Z è 1 se A=(HL), altrimenti Z=0.  
 - Bit di stato non influenzato, 0 = Bit azzerato, 1 = Bit settato a 1, X = Bit indifferente (0 o 1), ↑ = Bit influenzato secondo il risultato dell'operazione.

## Gruppo di istruzioni aritmetiche e logiche a 8 bit

| Mnemonico  | Simbolo della operazione  | Registro di stato  |  |  |  |  |  |  |  | Codice macchina                        |  |  | N° dei bytes                           | N° dei cicli                           | N° dei periodi di clock                | Commenti                         |                          |    |   |
|--|---|--|--|--|--|--|--|--|--|--|--|--|--|--|--|----------------------------------|--------------------------|----|---|
|  |   |  |  |  |  |  |  |  |  | HEX                                    |  |  |  |  |  |                                  |                          |    |   |
|  |   | 7  | 6                                      | 5                                      | 4                                      | 3                                      | 2                                      | 1                                      | 0                                      | S                                      | Z                                      | H                                      |  |  |  |                                  | P/V                      | N  | C                                       |
| ADD A, r<br>ADD A, n   | $A \leftarrow A + r$<br>$A \leftarrow A + n$  | $\uparrow$<br>$\uparrow$<br>$\uparrow$   | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | 10<br>11<br>$\leftarrow n$       | 000<br>000<br>r          | r  | Reg.<br>B<br>C<br>D<br>E<br>H<br>L<br>A |
| ADD A, (HL)<br>ADD A, (IX+d)   | $A \leftarrow A + (HL)$<br>$A \leftarrow A + (IX+d)$  | $\uparrow$<br>$\uparrow$<br>$\uparrow$   | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$ | 10<br>11<br>10<br>$\leftarrow d$ | 000<br>011<br>000<br>110 | DD |   |
| ADD A, (IY+d)  | $A \leftarrow A + (IY+d)$   | $\uparrow$   | $\uparrow$                             | $\uparrow$                             | $\uparrow$                             | $\uparrow$                             | $\uparrow$                             | $\uparrow$                             | $\uparrow$                             | $\uparrow$                             | $\uparrow$                             | $\uparrow$                             | $\uparrow$                             | $\uparrow$                             | $\uparrow$                             | 11<br>10<br>$\leftarrow d$       | 111<br>000<br>110        | FD |   |
| ADC A, S<br>SUB S<br>SBC A, S<br>AND S<br>OR S<br>XOR S<br>CP S<br>INC r<br>INC (HL)<br>INC (IX+d) | $A \leftarrow A + S + CY$<br>$A \leftarrow A - S$<br>$A \leftarrow A - S - CY$<br>$A \leftarrow A \wedge S$<br>$A \leftarrow A \vee S$<br>$A \leftarrow A (+)S$<br>$A \leftarrow S$<br>$r \leftarrow r + 1$<br>$(HL) \leftarrow (HL) + 1$<br>$(IX+d) \leftarrow (IX+d) + 1$ | $\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$<br>$\uparrow$ < |  |  |  |  |  |  |  |  |  |  |  |  |  |                                  |                          |    |   |



| Gruppo di istruzioni di controllo della CPU |  |                   |   |   |     |   |   |    |     |              |              |                         |          |   |   |  |
|---|--|-------------------|---|---|-----|---|---|----|-----|--------------|--------------|-------------------------|----------|---|---|--|
| Mnemonic                                    | Simbolo della operazione   | Registro di stato |   |   |     |   |   |    |     | N° dei bytes | N° dei clock | N° dei periodi di clock | Commenti |   |   |  |
|   |  | 7                 | 6 | 5 | 4   | 3 | 2 | 1  | 0   |              |              |                         |          |   |   |  |
|   |  | S                 | Z | H | P/V | N | C | 76 | 543 | 210          | HEX          |                         |          |   |   |  |
| DAA   | Converts acc. content into packed BCD following add or subtract with packed BCD operands<br>$A \leftarrow \bar{A}$<br>$A \leftarrow \bar{A} + 1$ | ↑                 | ↑ | X | ↑   | X | P | ●  | ↑   | 00           | 100          | 111                     | 27       | 1 | 4 | Regolazione decimale dell' accumulatore. |
| CPL   | $A \leftarrow \bar{A}$   | ●                 | X | 1 | X   | ● | ● | 1  | ●   | 00           | 101          | 111                     | 2F       | 1 | 4 |  |
| NEG   | $A \leftarrow \bar{A} + 1$   | ↑                 | ↑ | X | ↑   | X | V | 1  | ↑   | 11           | 101          | 101                     | ED       | 2 | 8 |  |
| CCF   | $CY \leftarrow \bar{CY}$   | ●                 | X | X | X   | ● | ● | 0  | ↑   | 00           | 111          | 111                     | 3F       | 1 | 4 |  |
| SCF   | $CY \leftarrow 1$  | ●                 | X | 0 | X   | ● | ● | 0  | 1   | 00           | 110          | 111                     | 37       | 1 | 4 |  |
| NOP   | No operation   | ●                 | X | ● | X   | ● | ● | ●  | ●   | 00           | 000          | 000                     | 00       | 1 | 4 | Complementa l'accumulatore.              |
| HALT  | CPU halted   | ●                 | X | ● | X   | ● | ● | ●  | ●   | 01           | 110          | 110                     | 76       | 1 | 4 | Pone a 1 il Bit carry.                   |
| D1*   | $IFF \leftarrow 0$   | ●                 | X | ● | X   | ● | ● | ●  | ●   | 11           | 110          | 011                     | F3       | 1 | 4 |  |
| E1*   | $IFF \leftarrow 1$   | ●                 | X | ● | X   | ● | ● | ●  | ●   | 11           | 111          | 011                     | FB       | 1 | 4 |  |
| IM 0  | Set interrupt mode 0   | ●                 | X | ● | X   | ● | ● | ●  | ●   | 11           | 101          | 101                     | ED       | 2 | 8 |  |
| IM 1  | Sep interrupt mode 1   | ●                 | X | ● | X   | ● | ● | ●  | ●   | 01           | 000          | 110                     | 46       | 2 | 8 |  |
| IM 2  | Sep interrupt mode 2   | ●                 | X | ● | X   | ● | ● | ●  | ●   | 11           | 101          | 101                     | ED       | 2 | 8 |  |
|   |  | ●                 | X | ● | X   | ● | ● | ●  | ●   | 01           | 010          | 110                     | 56       | 2 | 8 |  |
|   |  | ●                 | X | ● | X   | ● | ● | ●  | ●   | 11           | 101          | 101                     | ED       | 2 | 8 |  |
|   |  |                   |   |   |     |   |   |    |     | 01           | 011          | 110                     | 5E       |   |   |  |

Note:  
Il simbolo V nel bit P/V indica che tale bit contiene l'overflow del risultato dell'operazione. Analogamente il simbolo P indica la parità. V = 1 significa overflow, V = 0 significa non overflow. P = 1 significa parità pari. P = 0 significa disparità.  
IFF indica il permesso di interrupt.  
CY indica il bit di carry.  
● = Bit di stato non influenzato, 0 = Bit azzerato, 1 = Bit azzerato, 1 = Bit indifferente (0 o 1), ↑ = Bit influenzato secondo il risultato dell'operazione.

Notes:

Il simbolo V nel bit P/V indica che tale bit contiene l'overflow del risultato dell'operazione. Analogamente il simbolo P indica la parità. V = 1 significa overflow, V = 0 significa non overflow. P = 1 significa parità pari. P = 0 significa disparità.

IFF indica il permesso di interrupt.

CY indica il bit di carry.

● = Bit di stato non influenzato, 0 = Bit azzerato, 1 = Bit settato a 1, X = Bit indifferente (0 o 1), ↑ = Bit influenzato secondo il risultato dell'operazione.

# Gruppo di istruzioni aritmetiche a 16 bit

| Mnemonico | Simbolo della operazione | Registro di stato |   |   |   |   |   |   |   | Codice macchina |   |   |     | N° dei bytes | N° dei cicli | N° dei periodi di clock | Commenti |   |   |    |     |     |     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|-----------|--------------------------|-------------------|---|---|---|---|---|---|---|-----------------|---|---|-----|--------------|--------------|-------------------------|----------|---|---|----|-----|-----|-----|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|           |                          | 7                 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | S               | Z | H | P/V |              |              |                         |          | N | C | 76 | 543 | 210 | HEX |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|           |                          | •                 | • | X | X | X | • | • | 0 |                 |   |   |     |              |              |                         |          |   |   |    |     |     |     | ↑ | • | • | 0 | ↑ | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ | • | • | 0 | ↑ |

Note:

- ss: indica qualsiasi coppia di registri BC, DE, HL, SP. - pp: indica qualsiasi coppia di registri BC, DE, IX, SP. - rr: indica qualsiasi coppia di registri BC, DE, IX, SP. - • = Bit di stato non influenzato, 0 = Bit azzerato, 1 = Bit azzerato. - S = Bit Indifferente (0 o 1), Z = Bit influenzato secondo il risultato dell'operazione.

**Gruppo di Istruzioni di trattamento delle subroutines**

| Mnemonico   | Simbolo della operazione   | Registro di stato |   |   |   |   |   |   |   | Codice macchina |   |    |     | N° dei bytes | N° dei cicli | N° dei periodi di clock | Commenti |    |                    |
|-------------|--|-------------------|---|---|---|---|---|---|---|-----------------|---|----|-----|--------------|--------------|-------------------------|----------|----|--------------------|
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    |                    |
|             |  | 7                 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | P/V             | N | C  | 76  | 543          | 210          | HEX                     |          |    |                    |
| CALL nn     | (SP-1) ← PCH<br>(SP-2) ← PCL<br>PC ← nn<br>Se cc è falso continuo,<br>se no esegue CALL nn | ●                 | ● | X | ● | X | ● | ● | ● | ●               | ● | 11 | 001 | 101          | CD           | 3                       | 5        | 17 | Se cc è falso.     |
| CALL cc, nn | PC ← nn<br>Se cc è falso continuo,<br>se no esegue CALL nn                                 | ●                 | ● | X | ● | X | ● | ● | ● | ●               | ● | 11 | cc  | 100          |              | 3                       | 3        | 10 | Se cc è vero       |
| RET         | PCL ← (SP)<br>PCH ← (SP + 1)<br>Se cc è falso continuo,<br>se no esegue RET                | ●                 | ● | X | ● | X | ● | ● | ● | ●               | ● | 11 | 001 | 001          | C9           | 3                       | 5        | 17 |                    |
| RET cc      |  | ●                 | ● | X | ● | X | ● | ● | ● | ●               | ● | 11 | cc  | 000          |              | 1                       | 1        | 5  | Se cc è falso.     |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              | 1                       | 3        | 11 | Se cc è vero       |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    | cc                 |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    | Condizione         |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    | NZ non zero        |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    | 001 Z zero         |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    | 010 NC non carry   |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    | 011 C carry        |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    | 100 PO imparità    |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    | 101 PE parità pari |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    | 110 P segno pos.   |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    | 111 M segno neg.   |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    | t                  |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    | P                  |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    | 000 00H            |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    | 001 08H            |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    | 010 10H            |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    | 011 18H            |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    | 100 20H            |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    | 101 28H            |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    | 110 30H            |
|             |  |                   |   |   |   |   |   |   |   |                 |   |    |     |              |              |                         |          |    | 111 38H            |

| cc  | Condizione     |
|-----|----------------|
| 000 | NZ non zero    |
| 001 | Z zero         |
| 010 | NC non carry   |
| 011 | C carry        |
| 100 | PO imparità    |
| 101 | PE parità pari |
| 110 | P segno pos.   |
| 111 | M segno neg.   |

Note:  
 - RETN carica IPF<sub>2</sub> in IPF<sub>1</sub>. - e = Bit di stato non influenzato, 0 = Bit azzerato, 1 = Bit settato a 1, X = Bit indifferente (0 o 1), ↓ = Bit influenzato secondo il risultato dell'operazione.

**Gruppo di Istruzioni di salto**

| Mnemonico | Simbolo della operazione                               | Registro di stato |   |   |   |   |   |   |   | Codice macchina  |   |   | N° dei bytes | N° dei cicli | N° dei periodi di clock | Commenti |     |     |    |    |     |     |                          |
|-----------|--|-------------------|---|---|---|---|---|---|---|------------------|---|---|--------------|--------------|-------------------------|----------|-----|-----|----|----|-----|-----|--------------------------|
|           |  |                   |   |   |   |   |   |   |   | Condice macchina |   |   |              |              |                         |          |     |     |    |    |     |     |                          |
|           |  | 7                 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | S                | Z | H |              |              |                         |          | P/V | N   | C  | 76 | 543 | 210 | HEX                      |
| JP nn     | PC ← nn  | ●                 | ● | X | ● | X | ● | ● | ● | ●                | ● | ● | ●            | ●            | ●                       | 11       | 000 | 011 | C3 | 3  | 3   | 10  |                          |
| JP cc, nn | Si cc es<br>Se cc è vero<br>PC ← nn,<br>se no continua | ●                 | ● | X | ● | X | ● | ● | ● | ●                | ● | ● | ●            | ●            | ●                       | 11       | cc  | 010 |    | 3  | 3   | 10  |                          |
| JR e      | PC ← PC+e  | ●                 | ● | X | ● | X | ● | ● | ● | ●                | ● | ● | ●            | ●            | ●                       | 00       | 011 | 000 | 18 | 2  | 3   | 12  |                          |
| JR C, e   | Se C=0, continua<br>Se C=1                             | ●                 | ● | X | ● | X | ● | ● | ● | ●                | ● | ● | ●            | ●            | ●                       | 00       | 111 | 000 | 38 | 2  | 2   | 7   |                          |
| JR NC, e  | PC ← PC+e<br>Se C=1, continua<br>Se C=0                | ●                 | ● | X | ● | X | ● | ● | ● | ●                | ● | ● | ●            | ●            | ●                       | 00       | 110 | 000 | 30 | 2  | 2   | 7   | Se la condizione è vera. |
| JR Z, e   | PC ← PC+e<br>Se Z=0, continua<br>Se Z=1                | ●                 | ● | X | ● | X | ● | ● | ● | ●                | ● | ● | ●            | ●            | ●                       | 00       | 101 | 000 | 28 | 2  | 2   | 7   | Se la condizione è vera. |
| JR NZ, e  | PC ← PC+e<br>Se Z=1, continua                          | ●                 | ● | X | ● | X | ● | ● | ● | ●                | ● | ● | ●            | ●            | ●                       | 00       | 100 | 000 | 20 | 2  | 2   | 7   | Se la condizione è vera. |

**CC Condizione**  
 000 NZ non zero  
 001 Z zero  
 010 NC non carry  
 011 C carry  
 100 PO impurità  
 101 PE parità pari  
 110 P segno pos.  
 111 M segno neg.

Se la condizione è vera.  
 Se la condizione è vera.  
 Se la condizione è vera.  
 Se la condizione è vera.  
 Se la condizione è vera.  
 Se la condizione è vera.  
 Se la condizione è vera.



**Gruppo di istruzioni di ingresso/uscita**

| Mnemonico  | Simbolo della operazione   | Registro di stato |     |   |   |   |   |   |   | Codice macchina |             | N° dei bytes | N° dei cicli                              | N° dei periodi di clock | Commenti   |
|------------|--|-------------------|-----|---|---|---|---|---|---|-----------------|-------------|--------------|---|-------------------------|--|
|            |  | 7                 | 6   | 5 | 4 | 3 | 2 | 1 | 0 | 76              | 543 210     |              |   |                         |  |
| IN A, (n)  | $A \leftarrow (n)$   | •                 | •   | X | • | X | • | • | • | C               | 11 011 011  | 2            | 3   | 11                      | n to $A_0 \sim A_7$<br>$A_{CC}$ to $A_8 \sim A_{15}$ |
| IN r, (C)  | $r \leftarrow (C)$<br>Se $r=110$ ,<br>solo C sarà influenzato                                    | ↑                 | ↑   | X | ↑ | X | P | 0 | • | 01              | 101 101 000 | 2            | 3   | 12                      | C to $A_0 \sim A_7$<br>B to $A_8 \sim A_{15}$        |
| INI        | $(HL) \leftarrow (C)$<br>$B \leftarrow B-1$<br>$HL \leftarrow HL+1$                              | X                 | ① ↓ | X | X | X | X | 1 | • | 11 101 101      | 10 100 010  | 2            | 4   | 16                      | C to $A_0 \sim A_7$<br>B to $A_8 \sim A_{15}$        |
| INIR       | $(HL) \leftarrow (C)$<br>$B \leftarrow B-1$<br>$HL \leftarrow HL+1$<br>ripetere fino a che $B=0$ | X                 | 1   | X | X | X | X | 1 | • | 11 101 101      | 10 110 010  | 2            | 5<br>(If $B \neq 0$ )<br>4                | 21<br>16                | C to $A_0 \sim A_7$<br>B to $A_8 \sim A_{15}$        |
| IND        | $(HL) \leftarrow (C)$<br>$B \leftarrow B-1$  | X                 | ① ↓ | X | X | X | X | 1 | • | 11 101 101      | 10 101 010  | 2            | 4   | 16                      | C to $A_0 \sim A_7$<br>B to $A_8 \sim A_{15}$        |
| INDR       | $(HL) \leftarrow (C)$<br>$B \leftarrow B-1$<br>$HL \leftarrow H-1$<br>ripetere fino a che $B=0$  | X                 | 1   | X | X | X | X | 1 | • | 11 101 101      | 10 111 010  | 2            | 5<br>(If $B \neq 0$ )<br>4<br>(If $B=0$ ) | 21<br>16                | C to $A_0 \sim A_7$<br>B to $A_8 \sim A_{15}$        |
| OUT (n), A | $(n) \leftarrow A$   | •                 | •   | X | • | X | • | • | • | 11              | 010 011     | 2            | 3   | 11                      | n to $A_0 \sim A_7$<br>$A_{CC}$ to $A_8 \sim A_{15}$ |

## Gruppo di istruzioni di ingresso/uscita

| Memorizzo  | Simbolo della operazione                                      | Registro di stato |     |   |   |   |   |   |   | Codice macchina |   |   |     | N° dei bytes | N° dei clicks | N° dei periodi di clock | Commenti |   |   |          |     |               |     |   |
|------------|---|-------------------|-----|---|---|---|---|---|---|-----------------|---|---|-----|--------------|---------------|-------------------------|----------|---|---|----------|-----|---------------|-----|---|
|            |   | 7                 | 6   | 5 | 4 | 3 | 2 | 1 | 0 | Codice macchina |   |   |     |              |               |                         |          |   |   |          |     |               |     |   |
|            |   |                   |     |   |   |   |   |   |   | S               | Z | H | P/V |              |               |                         |          | N | C | 76       | 543 | 210           | HEX |   |
| OUT (C), r | (C) ← r   | •                 | •   | X | • | X | • | • | • | •               | • | • | •   | •            | •             | •                       | •        | • | • | ED       | 2   | 3             | 12  | C to A <sub>0</sub> ~A <sub>7</sub><br>B to A <sub>8</sub> ~A <sub>15</sub> |
| OUTI       | (C) ← (HL)<br>B ← B-1<br>HL ← HL+1                            | X                 | ① ↓ | X | X | X | X | X | 1 | •               | • | • | •   | •            | •             | •                       | •        | • | • | ED<br>A3 | 2   | 4             | 16  | C to A <sub>0</sub> ~A <sub>7</sub><br>B to A <sub>8</sub> ~A <sub>15</sub> |
| OTIR       | (C) ← (HL)<br>B ← B-1<br>HL ← HL+1<br>ripetere fino a che B=0 | X                 | 1   | X | X | X | X | X | 1 | •               | • | • | •   | •            | •             | •                       | •        | • | • | ED<br>B3 | 2   | 5<br>(If B≠0) | 21  | C to A <sub>0</sub> ~A <sub>7</sub><br>B to A <sub>8</sub> ~A <sub>15</sub> |
| OUTD       | (C) ← (HL)<br>B ← B-1<br>HL ← HL-1                            | X                 | ① ↓ | X | X | X | X | X | 1 | •               | • | • | •   | •            | •             | •                       | •        | • | • | ED<br>AB | 2   | 4             | 16  | C to A <sub>0</sub> ~A <sub>7</sub><br>B to A <sub>8</sub> ~A <sub>15</sub> |
| OTDR       | (C) ← (HL)<br>B ← B-1<br>HL ← HL-1<br>ripetere fino a che B=0 | X                 | 1   | X | X | X | X | X | 1 | •               | • | • | •   | •            | •             | •                       | •        | • | • | ED<br>BB | 2   | 5<br>(If B≠0) | 21  | C to A <sub>0</sub> ~A <sub>7</sub><br>B to A <sub>8</sub> ~A <sub>15</sub> |
|            |   |                   |     |   |   |   |   |   |   |                 |   |   |     |              |               |                         |          |   |   |          | 2   | 4<br>(If B=0) | 16  |   |

**Note:**

1 Significa che il bit Z è settato a 1 se il risultato di B-1 è zero, altrimenti verrà resettato a 0.

- = Significa bit di stato non influenzato, 0 = Bit resettato, 1 = Bit settato a 1, X = Bit indifferente (0 o 1),  $\uparrow$  = Il bit è influenzato secondo il risultato dell'operazione.

Gruppo di istruzioni di spostamento e rotazione

| Mnemonico  | Simbolo della operazione  | Registro di dato |   |   |   |     |   |   |   | Codice macchina      |            | N° dei bytes | N° dei cicli | N° dei periodi di clock | Commenti  |
|------------|---|------------------|---|---|---|-----|---|---|---|----------------------|------------|--------------|--------------|-------------------------|---|
|            |   | 7                | 6 | 5 | 4 | 3   | 2 | 1 | 0 | 76                   | 543 210    |              |              |                         |   |
|            |   | S                | Z | H |   | P/V | N | C |   | 00                   | 000 111    | 07           | 1            | 4                       | Rotazione circolare a sinistra dell'accumulatore. |
| RLCA       | $\boxed{CY} \leftarrow \boxed{7} \leftarrow 0$<br>A                 | •                | • | X | 0 | X   | • | 0 | • | 00                   | 000 111    | 07           | 1            | 4                       | Rotazione circolare a sinistra dell'accumulatore. |
| RLA        | $\boxed{CY} \leftarrow \boxed{7} \leftarrow 0$<br>A                 | •                | • | X | 0 | X   | • | 0 | • | 00                   | 010 111    | 17           | 1            | 4                       | Rotazione a sinistra dell'accumulatore.           |
| RRCA       | $\boxed{7} \rightarrow 0 \leftarrow \boxed{CY}$<br>A                | •                | • | X | 0 | X   | • | 0 | • | 00                   | 001 111    | 0F           | 1            | 4                       | Rotazione a destra dell'accumulatore.             |
| RRA        | $\boxed{7} \rightarrow 0 \leftarrow \boxed{CY}$<br>A                | •                | • | X | 0 | X   | • | 0 | • | 00                   | 011 111    | 1F           | 1            | 4                       | Rotazione a destra dell'accumulatore.             |
| RLCr       | A   | ↑                | ↑ | X | 0 | X   | P | 0 | • | 11                   | 001 011    | CB           | 2            | 8                       | Rotazione circolare a sinistra.                   |
| RLC (HL)   |   | ↑                | ↑ | X | 0 | X   | P | 0 | • | 00 $\boxed{000}$ r   | 11 001 011 | CB           | 2            | 15                      |   |
|            |   |                  |   |   |   |     |   |   |   | 00 $\boxed{000}$ 110 | 11 001 011 | CB           | 2            | 4                       | Reg. r  |
| RLC (IX+d) | $\boxed{CY} \leftarrow \boxed{7} \leftarrow 0$<br>r(HL)(IX+d)(IY+d) | ↑                | ↑ | X | 0 | X   | P | 0 | • | 11                   | 011 101    | DD           | 4            | 23                      | B   |
|            |   |                  |   |   |   |     |   |   |   | 11 001 011           | 11 001 011 | CB           | 4            | 6                       | C   |
| RLC (IY+d) | $\boxed{CY} \leftarrow \boxed{7} \leftarrow 0$<br>r(HL)(IX+d)(IY+d) | ↑                | ↑ | X | 0 | X   | P | 0 | • | 00 000 110           | 00 000 110 | FD           | 4            | 23                      | D   |
|            |   |                  |   |   |   |     |   |   |   | 11 111 101           | 11 001 011 | CB           | 4            | 6                       | E   |
|            |   |                  |   |   |   |     |   |   |   | 00 000 110           | 00 000 110 | FD           | 4            | 23                      | H   |
|            |   |                  |   |   |   |     |   |   |   | 11 111 101           | 11 001 011 | CB           | 4            | 23                      | L   |
|            |   |                  |   |   |   |     |   |   |   | 00 000 110           | 00 000 110 | FD           | 4            | 23                      | A   |



# Gruppo di Istruzioni di spostamento e rotazione

| Mnemonico | Simbolo della operazione   | Registro di stato     |   |   |     |   |   |                       | N° dei bytes             | N° dei periodi di clock | Commenti |
|-----------|--|-----------------------|---|---|-----|---|---|-----------------------|--------------------------|-------------------------|----------|
|           |  | 7                     | 6 | 5 | 4   | 3 | 2 | 1                     |                          |                         |          |
|           |  | S                     | Z | H | P/V | N | C | 0                     |                          |                         |          |
| RL s      | $\boxed{CY} \leftarrow \boxed{7} \leftarrow 0$<br>$S \leftarrow r(HL), (IX+d), (IY+d)$               | $\uparrow \downarrow$ | X | 0 | X   | P | 0 | $\uparrow \downarrow$ |                          |                         |          |
| RRC s     | $\boxed{7} \rightarrow 0 \leftarrow \boxed{CY}$<br>$S \leftarrow r(HL), (IX+d), (IY+d)$              | $\uparrow \downarrow$ | X | 0 | X   | P | 0 | $\uparrow \downarrow$ |                          |                         |          |
| RR s      | $\boxed{7} \rightarrow 0 \leftarrow \boxed{CY}$<br>$S \leftarrow r(HL), (IX+d), (IY+d)$              | $\uparrow \downarrow$ | X | 0 | X   | P | 0 | $\uparrow \downarrow$ |                          |                         |          |
| SLA s     | $\boxed{CY} \leftarrow \boxed{7} \leftarrow 0$<br>$S \leftarrow r(HL), (IX+d), (IY+d)$               | $\uparrow \downarrow$ | X | 0 | X   | P | 0 | $\uparrow \downarrow$ |                          |                         |          |
| SRA s     | $\boxed{7} \rightarrow 0 \leftarrow \boxed{CY}$<br>$S \leftarrow r(HL), (IX+d), (IY+d)$              | $\uparrow \downarrow$ | X | 0 | X   | P | 0 | $\uparrow \downarrow$ |                          |                         |          |
| SRL s     | $0 \leftarrow \boxed{7} \rightarrow 0 \leftarrow \boxed{CY}$<br>$S \leftarrow r(HL), (IX+d), (IY+d)$ | $\uparrow \downarrow$ | X | 0 | X   | P | 0 | $\uparrow \downarrow$ |                          |                         |          |
| RLD       | A $\boxed{7-4} \boxed{3-0} \leftarrow \boxed{7-4} \boxed{3-0}$ (HL)                                  | $\uparrow \downarrow$ | X | 0 | X   | P | 0 | ●                     | 11 101 101<br>01 101 111 | 2 5                     | 18       |
| RRD       | A $\boxed{7-4} \boxed{3-0} \leftarrow \boxed{7-4} \boxed{3-0}$ (HL)                                  | $\uparrow \downarrow$ | X | 0 | X   | P | 0 | ●                     | 11 101 101<br>01 100 111 | 2 5                     | 18       |

● = Significa bit di stato non influenzato, 0 = Bit resettato, 1 = Bit settato a 1, X = Bit indifferente (0 o 1),  $\uparrow \downarrow$  = Il bit è influenzato secondo il risultato dell'operazione.

Gruppo di istruzioni di azzeramento, settaggio a 1 e test

| Mnemonico   | Simbolo della operazione           | Registro di stato |   |   |   |     |   |   |    | Codice macchina |     |     |     | N° dei bytes | N° dei cicli | N° dei periodi di clock | Commenti  |
|---|------------------------------------|-------------------|---|---|---|-----|---|---|----|-----------------|-----|-----|-----|--------------|--------------|-------------------------|---|
|   |                                    | 7                 | 6 | 5 | 4 | 3   | 2 | 1 | 0  |                 |     |     |     |              |              |                         |   |
|   |                                    | S                 | Z | H |   | P/V | N | C | 76 | 543             | 210 | HEX |     |              |              |                         |   |
| BIT b, r<br><br>BIT b, (HL)<br><br>BIT b, (IX+d) <sub>b</sub> | $Z \leftarrow \neg b$              | X                 | ↑ | X | 1 | X   | X | 0 | ●  | 11              | 001 | 011 | CB  | 2            | 2            | 8                       | r<br>Reg.<br>000 B<br>001 C<br>010 D<br>011 E<br>100 H<br>101 L<br>111 A    |
|   | $Z \leftarrow (\overline{HL})_b$   | X                 | ↑ | X | 1 | X   | X | 0 | ●  | 11              | 001 | 011 | CB  | 2            | 3            | 12                      |   |
|   | $Z \leftarrow (\overline{IX+d})_b$ | ↑                 | X | 1 | X | X   | 0 | ● | ●  | 11              | 001 | 011 | DD  | 4            | 5            | 20                      |   |
| BIT b, (IY+d) <sub>b</sub>                                    |                                    |                   |   |   |   |     |   |   |    | 11              | 001 | 011 | CB  |              |              |                         |   |
|   |                                    |                   |   |   |   |     |   |   |    | 01              | b   | 110 | ← d |              |              |                         | b<br>Bit testato  |
|   | $Z \leftarrow (\overline{IY+d})_b$ | X                 | ↑ | X | 1 | X   | X | 0 | ●  | 11              | 111 | 101 | FD  | 4            | 5            | 20                      | 000 0 (Test)<br>001 1<br>010 2<br>011 3<br>100 4<br>101 5<br>110 6<br>111 7 |
| SET b, r<br><br>SET b, (HL)<br><br>SET b, (IX+d)              | $r \leftarrow 1$                   | ●                 | ● | X | ● | X   | ● | ● | ●  | 11              | 001 | 011 | CB  | 2            | 2            | 8                       |   |
|   | $(HL)_b \leftarrow 1$              | ●                 | ● | X | ● | X   | ● | ● | ●  | 11              | 001 | 011 | CB  | 2            | 4            | 15                      |   |
|   | $(IX+d)_b \leftarrow 1$            | ●                 | ● | X | ● | X   | ● | ● | ●  | 11              | 011 | 101 | DD  | 4            | 6            | 23                      |   |
| SET b, (IY+d)   | $(IY+d)_b \leftarrow 1$            | ●                 | ● | X | ● | X   | ● | ● | ●  | 11              | 001 | 011 | CB  |              |              |                         |   |
|   |                                    |                   |   |   |   |     |   |   |    | 11              | 111 | 101 | FD  | 4            | 6            | 23                      |   |

| Gruppo di istruzioni di azzeramento, settaggio a 1 e test |  |                   |   |   |   |   |     |   |   |                 |              |                         |          |  |     |
|---|--|-------------------|---|---|---|---|-----|---|---|-----------------|--------------|-------------------------|----------|--|-----|
| Mnemonico   | Simbolo della operazione                                   | Registro di stato |   |   |   |   |     |   |   | Codice macchina | N° dei bytes | N° dei periodi di clock | Commenti |  |     |
|   |  | 7                 | 6 | 5 | 4 | 3 | 2   | 1 | 0 |                 |              |                         |          |  |     |
|   |  | S                 | Z |   | H |   | P/V | N | C | 76              | 543          |                         |          | 210  | HEX |
| RES b, s  | S <sub>0</sub> ← 0<br>S ≡ r. (HL)<br>(IX + d),<br>(IY + d) |                   |   |   |   |   |     |   |   | 11              | 001          | 011                     | CB       | Il nuovo codice macchina sostituisce 11 di SET b,s con 10 e lo stato è influenzato nello stesso modo della istruzione SET. |     |
|   |  |                   |   |   |   |   |     |   |   |                 |              |                         |          |  |     |
|   |  |                   |   |   |   |   |     |   |   |                 |              |                         |          |  |     |
|   |  |                   |   |   |   |   |     |   |   |                 |              |                         |          |  |     |

La notazione S<sub>0</sub> indica bit b (0-7) alla posizione S.  
• = Significa bit di stato non influenzato, 0 = Bit resetato, 1 = Bit settato a 1, X = Bit indifferente (0 o 1), ↑ = Il bit è influenzato secondo il risultato dell'operazione.

La notazione Sb indica bit b (0-7) alla posizione S.  
 • = Significa bit di stato non influenzato, 0 = Bit resettato, 1 = Bit settato a 1, X = Bit indifferente (0 o 1),  $\uparrow$  = Il bit è influenzato secondo " risultato dell'operazione.

**Riassunto delle operazioni delle istruzioni dello Z-80 e loro relazione con il registro di stato.**  
*(I commenti sono in inglese poiché sono correlazionati alle istruzioni mnemoniche)*

| Istruzione  | Registro di stato |             |             |             |             |             |             | Commento   |
|---|-------------------|-------------|-------------|-------------|-------------|-------------|-------------|--|
|   | D7                | Z           | H           | P/V         | N           | D0          |             |  |
|   | S                 |             |             |             |             | C           |             |  |
| ADD A, s; ADC A, s<br>SUB, s; SBCA, s; CP, s; NEG | ↑<br>↑<br>↑       | ↑<br>↑<br>↑ | ↑<br>↑<br>↑ | X<br>X<br>X | V<br>V<br>V | 0<br>1<br>0 | ↑<br>↑<br>0 | 8 bit add or add with carry<br>8-bit subtract, subtract with carry, compare and negate accumulator |
| AND s   | ↑                 | ↑           | 1           | X           | P           | 0           | 0           | } Logical operations   |
| OR s; XOR s                                       | ↑                 | ↑           | 0           | X           | P           | 0           | 0           |  |
| INC s   | ↑                 | ↑           | ↑           | X           | V           | 0           | •           | 8-bit increment  |
| DEC s   | ↑                 | ↑           | ↑           | X           | V           | 1           | •           | 8-bit decrement  |
| ADD DD, SS  | •                 | •           | X           | X           | •           | 0           | ↑           | 16-bit add   |
| ADC HL, SS  | •                 | •           | X           | X           | •           | 0           | ↑           | 16-bit add with carry  |
| SBC HL, SS  | •                 | •           | X           | X           | V           | 1           | ↑           | 16-bit subtract with carry   |
| RLA; RLCA; RRA; RRCA                              | •                 | •           | 0           | X           | •           | 0           | ↑           | Rotate accumulator   |
| RL s; RLC s; RR s; RRC s; SLA s; SRA s; SRL s     | •                 | •           | 0           | X           | P           | 0           | ↑           | Rotate and shift locations   |
| RLD; RRD  | ↑                 | ↑           | 0           | X           | P           | 0           | •           | Rotate digit left and right  |
| DAA   | ↑                 | ↑           | ↑           | X           | P           | •           | ↑           | Decimal adjust accumulator   |
| CPL   | •                 | •           | 1           | X           | •           | 1           | •           | Complement accumulator   |
| SCF   | •                 | •           | 0           | X           | •           | 0           | ↓           | Set carry  |
| CCF   | •                 | •           | 0           | X           | •           | 0           | ↑           | Complement carry   |
| IN r, (C)   | ↑                 | ↑           | 0           | X           | P           | 0           | •           | Input register indirect  |
| INI; IND, OUTI, OUTD                              | X                 | ↑           | 0           | X           | X           | 1           | •           | Block input and output   |
| INIR; INDR; OTIR; OTDR                            | X                 | 1           | X           | X           | X           | 1           | •           | Z=0 if B≠0 otherwise Z=1   |
| LDI; LDD  | X                 | X           | 0           | X           | ↑           | 0           | •           | Block transfer instructions  |
| LDIR; LDDR  | X                 | X           | 0           | X           | 0           | 0           | •           | P/V=1 if BC≠0, otherwise P/V=0   |
| CPI; CPIR; CPD; CPDR                              | X                 | ↑           | X           | X           | ↑           | 1           | •           | Block search instructions  |
|   |                   |             |             |             |             |             |             | Z=1 if A=(HL), otherwise Z=0<br>P/V=1 if BC≠0, otherwise P/V=0                                     |
| LDA, I, LD A, R                                   | ↑                 | ↑           | 0           | X           | IFF         | 0           | •           | The content of the interrupt enable flip-flop (IFF) is copied into the P/V flag                    |
| BIT b, s  | X                 | ↑           | 1           | X           | X           | 0           | •           | The state of bit b of location s is copied into the Z flag   |

**Tabella 1.-Tabella delle istruzioni del  $\mu P$  Z80.**

certo incremento del numero delle istruzioni e della potenza delle stesse: cioè più grande è il numero delle istruzioni più specifica sarà l'operazione di ciascuna. Nelle tabelle sono esposte dettagliatamente le istruzioni dello Z80 in base al codice mnemonico. LD significa load o caricamento e consiste nel trasferire un dato da un registro all'altro o da una locazione di memoria a un registro. PUSH è utilizzata per trasferire il dato da un registro allo stack. POP è l'operazione inversa a PUSH. EX permette di scambiare i dati senza alterarli da un registro a un altro. CP è l'operazione di confronto. ADD è l'operazione di somma. SUB è l'operazione di sottrazione. AND è l'operazione logica dello stesso nome. OR è anch'essa l'operazione logica OR. XOR è l'operazione logica OR esclusivo. INC significa incrementare, DEC decrementare. DAA è un'operazione in BCD. CPL è un'operazione di complemento o inversione. NEG è un'operazione di complemento a due. CCF è un'operazione di complemento del bit di carry. SCF setta a uno il bit di carry. NOP è un'istruzione di non operare. HALT arresta il  $\mu$ P. DI, EI, IM0, IM1, IM2, sono istruzioni di trattamento degli interrupts. RL e RR sono operazioni di rotazione verso sinistra e verso destra rispettivamente. BIT è un'operazione di test. SET è un'operazione di settaggio a uno. RES è un'operazione di settaggio a zero. JP è un'operazione di salto. JR è un'operazione di salto relativo. DJNZ decrementa e salta se non è zero. CALL richiamo di subroutine. RET ritorno da subroutine. IN ingresso da una periferica. OUT uscita verso una periferica.

### *Tipi di indirizzamento*

Il microprocessore dispone dei seguenti modi di indirizzamento:

- indirizzamento immediato: è utilizzato con istruzioni a due bytes, il primo è quello dell'istruzione, il secondo è l'operando o dato da trattare;
- indirizzamento immediato esteso: è utilizzato con istruzioni a tre bytes; l'operando occupa gli ultimi due;
- indirizzamento relativo: si utilizza per operare in prossimità dell'istruzione in corso ed è quindi in relazione con il contatore di programma; occupa due bytes;
- indirizzamento esteso: è utilizzato con istruzioni in cui si indica l'effettivo indirizzo; occupa tre bytes;
- indirizzamento indicizzato: l'indirizzo effettivo si ottiene sommando il contenuto dei registri indice con quello del terzo byte; necessita di tre bytes;
- indirizzamento tramite registri: è utilizzato per operazioni di

trattamento dei registri;

- indirizzamento indiretto: l'indirizzo effettivo si trova alla locazione di memoria indicata dall'istruzione.

### *Riassunto del significato delle notazioni utilizzate nella tabella*

Le notazioni utilizzate nella tabella delle istruzioni del microprocessore Z80 sono le seguenti:

- C = bit di carry;
- Z = bit di risultato zero nell'accumulatore;
- S = bit del segno del risultato nell'accumulatore;
- P/V = bit di parità e overflow;
- H = bit di carry di mezzo byte;
- N = bit che indica il tipo dell'operazione precedente (somma o differenza);
- = il corrispondente bit non cambia nell'operazione;
- 0 = il bit corrispondente è stato resettato durante l'operazione eseguita;
- 1 = il corrispondente bit è stato settato durante l'operazione effettuata;
- X = il valore del corrispondente bit è indifferente;
- V = il bit corrispondente viene influenzato dal carry durante l'operazione effettuata;
- P = il bit corrispondente viene influenzato dalla parità del risultato;
- r = si riferisce a uno dei registri A, B, C, D, E, H, L;
- s = si riferisce a qualsiasi locazione di memoria a cui si accede con otto bit;
- ss = si riferisce a qualsiasi locazione di memoria a cui si accede con sedici bit;
- R = si riferisce al contatore di refresh;
- n = con questa lettera si indica un qualsiasi valore binario a otto bit, compreso tra 0 e 255 decimale;
- nn = con queste lettere si indica un qualsiasi valore binario a sedici bit, compreso tra 0 e 65535 decimale.

## *MICROPROCESSORI A 8 BIT PIU' COMUNI (III)*



uesto capitolo è dedicato a due microprocessori attualmente molto utilizzati come il 6800 della Motorola e il 6502 della Rockwell, che hanno caratteristiche molto simili.

Benchè prodotti da costruttori diversi, vengono considerati assieme per la loro similitudine di funzionamento e di programmazione.

### *Il microprocessore 6800*

Il microprocessore 6800 venne messo in commercio dalla Motorola ed è attualmente uno dei più utilizzati, con altri della stessa serie che lo seguirono, come il 6802.

Le caratteristiche di questo  $\mu$ P sono: il 6800 è costruito in tecnologia NMOS, ha una capacità di indirizzamento di memoria di 64 Kbytes e frequenza che può variare dal MHz del 6800-A fino ai 2 MHz del 6800-B, prevede 72 istruzioni, e l'alimentazione è unica a +5 V, i modi di indirizzamento possono essere: diretto, relativo, immediato, indicizzato, esteso, o implicito. Relativamente agli interrupts, ne possiede uno logico (SWI) e due hardware (IRQ e NMI). Il suo bus dei dati è a otto bit.

### *Costituzione interna del 6800*

In Fig. 1 si vedono i blocchi interni di questo microprocessore, dei quali si deve evidenziare come specifico, il blocco dei registri, mentre il resto è simile a quanto previsto generalmente in tutti i microprocessori.

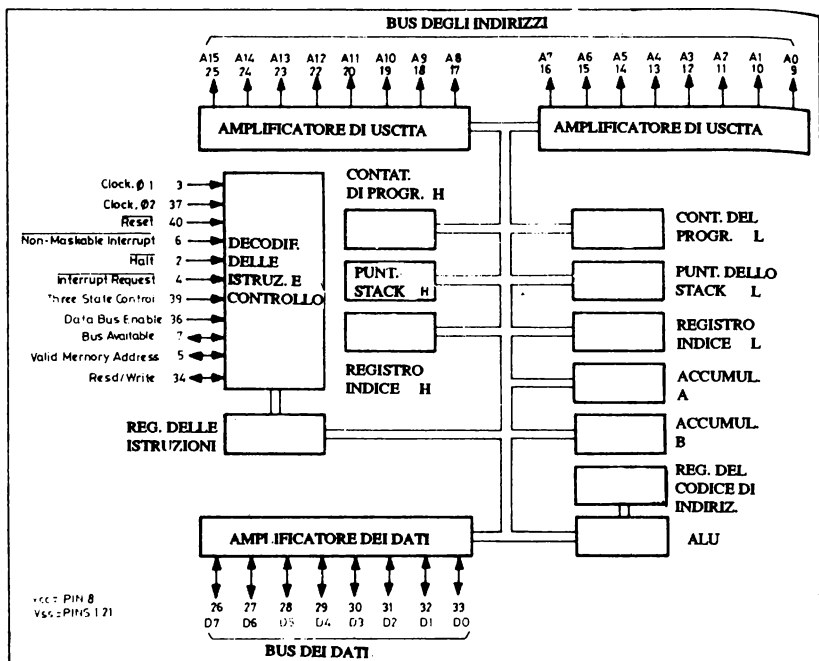


Fig. 1.-Diagramma interno a blocchi del microprocessore 6800.

Il 6800 dispone di sei registri interni ai quali farà riferimento il software.

L'uso di ogni registro è il seguente:

- **accumulatori:** il 6800 ha due registri accumulatori ai quali nelle istruzioni ci si riferisce come A e B. Ognuno di questi è da 8 bit, e attraverso di loro passano gli operandi diretti al  $\mu P$  ed i risultati forniti dall'ALU.
- **registro indice:** è un registro a due bytes, utilizzato come memoria temporanea per i dati o come registro indice per l'immagazzinamento dei dati in memoria con indirizzamento indicizzato.
- **contatore di programma:** è un registro a due bytes (16 bit) in cui viene memorizzato l'indirizzo di memoria dell'istruzione in corso.
- **puntatore di stack:** è un registro a due bytes in cui si memorizza



l'indirizzo della posizione di memoria nella quale è ubicato lo stack, in cui si memorizzano gli indirizzi di ritorno dalle subroutines o altri indirizzi o dati necessari. Questo stack può essere costituito da una memoria RAM, meglio se non volatile, che si può ottenere con una batteria tampone che rende l'alimentazione indipendente da quella del sistema.

- registro di stato: è un registro a otto bit, ognuno dei quali ha una propria funzione relativa allo stato funzionale del microprocessore. Il loro significato è:
  - i bit 6 e 7 non vengono utilizzati e sono settati permanentemente a uno;
  - il bit 5, detto H, serve per il carry di mezzo byte, il suo uso è indispensabile per le operazioni in BCD;
  - il bit 4, detto I, è quello che permette gli interrupts mascherati, il permesso è concesso se tale bit è settato a uno;
  - il bit 3, detto N, è quello che indica il segno del dato elaborato. Quando N=1, il dato è negativo (bit 7 del dato = 1);
  - il bit 2, detto Z, indica quando il dato elaborato vale zero (quando ciò accade Z=1);
  - il bit 1, detto V, indica, quando è settato a uno, che è avvenuto un overflow su qualche dato;
  - il bit 0, detto C, è il bit di carry che si porta a uno quando l'operazione aritmetica eseguita ha prodotto un riporto.

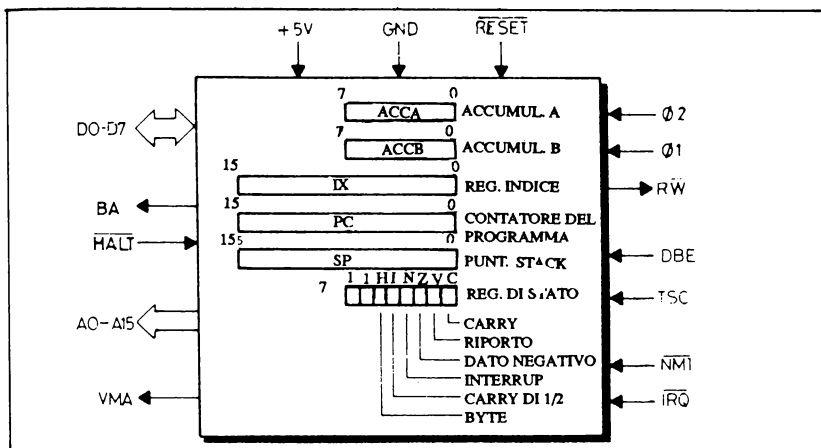


Fig. 2.-Dettaglio del blocco dei registri interni del µP 6800.

## *Descrizione dei terminali del 6800*

Il microprocessore 6800 si trova in un contenitore a 40 piedini, le cui funzioni sono:

- A0-A15: è il bus degli indirizzi formato da 16 bit per poter indirizzare 64 Kbytes di memoria. Queste uscite possono essere collegate a circuiti TTL (Fan out=1).
- D0-D7: sono i terminali corrispondenti al bus dei dati, gli otto three state interni possiedono un fan out=1, per cui possono essere collegati a circuiti TTL con fan in=1.

Si ricorda che il fan out è il numero di ingressi TTL con carico tipico di 1,6 mA (fan in) che si possono collegare in parallelo a uno di tali terminali.

- HALT: è un ingresso, resettando il quale si può arrestare il  $\mu P$ . Ciò accade solo dopo che è terminata l'istruzione in corso.
- TSC: è un ingresso mediante il quale si può portare ad alta impedenza il bus degli indirizzi ed il segnale di lettura- scrittura (R/W). Ciò accade quando TSC (Three State Control) = 1.
- R/W (Read/Write): è un'uscita, mediante la quale il  $\mu P$  indica alla memoria e alle periferiche se l'operazione è di lettura (livello logico uno) o scrittura (livello logico zero).
- VMA (Valid Memory Address): questa uscita indica in quale istante l'indirizzo è valido. E' effettivamente un impulso positivo di sincronismo che serve per indicare il momento in cui si può prendere l'indirizzo dal bus, dopo che è trascorso il tempo critico dei fronti.
- DBE (Data Bus Enable): è un ingresso attraverso il quale si controlla il bus dei dati, portandolo ad alta impedenza o consentendo il flusso dei dati.
- BA (Bus Available): è un'uscita che, quando viene settata a uno, indica all'esterno che in quel momento il  $\mu P$  si è messo in attesa liberando i bus.

Questa uscita si attiva quando il  $\mu P$  ha ricevuto un HALT, o quando si trova in stato di attesa in seguito all'esecuzione dell'istruzione WAIT.

- IRQ (Interrupt Request): mediante questo ingresso le periferiche possono richiedere un interrupt al microprocessore, il quale completa l'istruzione che sta eseguendo in tale istante, e salta ad

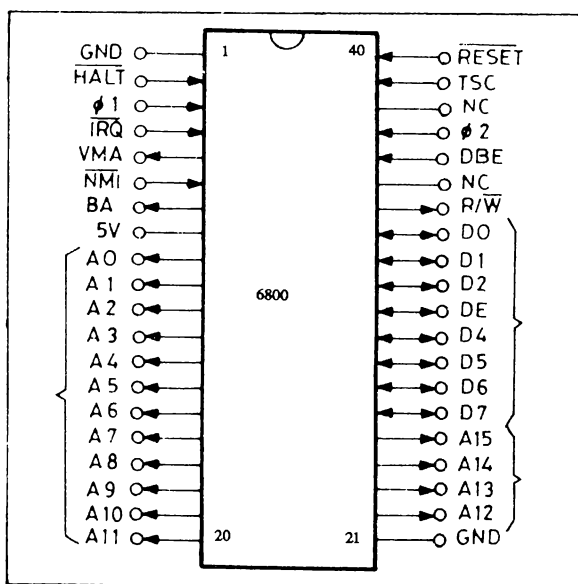


Fig. 3.-Terminali del microprocessore 6800 e loro funzioni.

un programma interno di trattamento degli interrupts.

L'ingresso HALT deve essere settato a uno perchè gli interrupts siano permessi. L'ingresso IRQ necessita di una resistenza esterna da 3 K $\Omega$  collegata a +5 V perchè tutti gli ingressi di richiesta di interrupts possano essere collegati a questo terminale, formando un OR cablato.

- RESET: è l'ingresso di azzeramento, utilizzato per azzerare e inizializzare il microprocessore dopo averlo alimentato, o quando si vuole reinizializzarlo.

Questo segnale avvia una routine interna al microprocessore che obbliga ad iniziare l'esecuzione del programma principale.

In tale istante sul bus degli indirizzi comparirà l'indirizzo FFFE, locazione di memoria che, con la successiva FFFF, contiene l'indirizzo da cui inizia il programma principale, poichè il contenuto di queste locazioni sarà fornito al contatore di programma (PC) come primo indirizzo di esecuzione. La routine interna di inizializzazione resetta anche il bit di interrupt dello status per bloccare tutte le interruzioni durante l'inizializzazione.

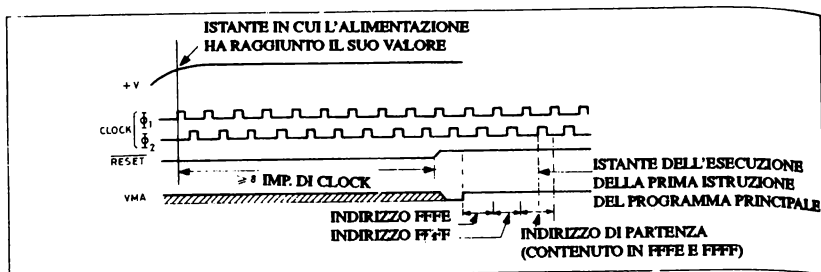


Fig. 4.-Diagramma dei tempi dei segnali che intervengono nell'azzeramento e nell'inizializzazione del microprocessore. L'indirizzo iniziale del programma principale è contenuto alle locazioni FFFE e FFFF.

In Fig. 4 si vede il diagramma dei tempi dei segnali che intervengono in questa fase.

- NMI (Not Maskable Interrupt): è un ingresso attraverso il quale si richiede un interrupt non mascherabile. Quando questo ingresso viene resettato a zero, il  $\mu P$  completa l'istruzione in corso e soddisfa la richiesta della periferica.

### Hardware di base per il 6800

In Fig. 5 è indicato lo schema a blocchi di un semplice sistema da aggiungere al microprocessore 6800 perchè possa eseguire un programma.

Fondamentalmente comprende il microprocessore pilotato dalle due fasi di clock fornite dal circuito integrato 6871, un blocco con i circuiti di inizializzazione per ottenere l'inizio dell'esecuzione del programma dopo aver fornito l'alimentazione, il blocco di memoria in cui una ROM o una EPROM contiene il programma, e una RAM per l'elaborazione dei dati e delle variabili. L'ultimo blocco è costituito da una PIA, o circuito che agisce da interfaccia tra il microprocessore e le periferiche.

Tutti questi blocchi sono collegati al  $\mu P$  mediante i bus dei dati, degli indirizzi e di controllo.

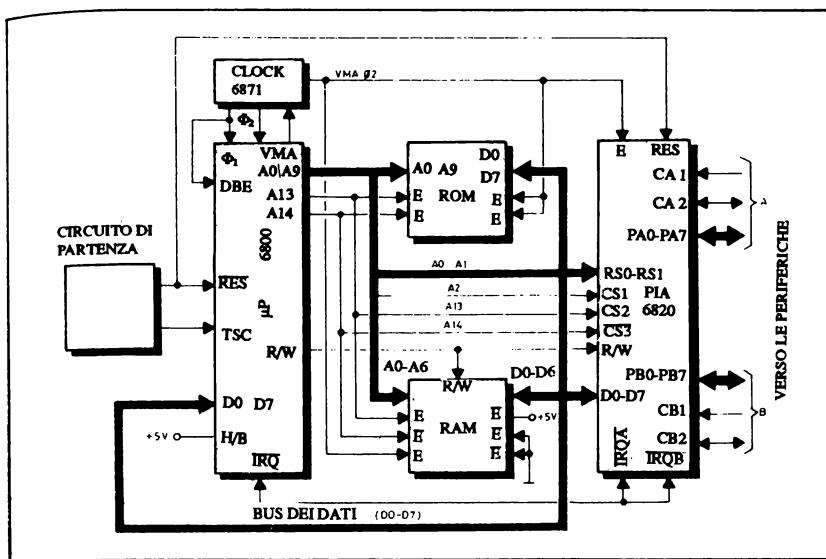


Fig. 5.-Diagramma a blocchi di un sistema di principio per il funzionamento del microprocessore 6800.

## La PIA MC-6820

Il circuito integrato PIA MC-6820 fornisce al  $\mu P$  6800 un metodo flessibile per il suo collegamento alle periferiche. Benchè sia internamente un circuito relativamente complesso, permette una certa varietà di collegamenti con la CPU mediante un minimo di circuiteria logica aggiuntiva, ed una semplificazione della programmazione.

Attualmente quasi tutti i microprocessori dispongono di circuiti associati per facilitare i collegamenti della CPU con l'esterno. Questi circuiti sono integrati di tipo LSI in contenitori a quaranta piedini, ai quali deve essere collegato il  $\mu P$  da un lato, e le periferiche dall'altro. Sono interfacce complete denominate PIA (Peripheral Interface Adapter). La Motorola costruttrice del  $\mu P$  6800 e della PIA MC-6820 è stata la prima ad introdurre questo tipo di circuito nei suoi sistemi; in seguito diversi fabbricanti hanno seguito tale esempio. La PIA MC-6820 è costituita da due blocchi di accesso al bus dei dati, il blocco A e il blocco B, entrambi a otto bit. Ognuno di questi bit viene collegato internamente al corrispondente bit del bus dei dati del  $\mu P$ , e può essere trasformato in ingresso o uscita, in base al contenuto precedentemente caricato

nel registro interno DDRA.

Quando un bit di tale registro è settato a uno, il corrispondente bit del blocco di accesso A diventa di uscita, al contrario, quando tale bit è resettato, l'omologo del blocco di accesso A diventa di ingresso. Ciò permette un controllo indipendente di ognuno degli otto bit del blocco di accesso A, e B (DDRA controlla il blocco A e DDRB controlla il blocco B), il flusso di dati tra il bus dei dati ed i blocchi passa attraverso i registri ORA e ORB, mentre DDRA e DDRB indicano il senso di trasferimento.

Infine i registri CRA e CRB sono registri di controllo, monitorizzano gli interrupts tra CPU e periferiche, e indirizzano i registri dei dati e il loro trasferimento.

Il bus di controllo della PIA è costituito da CS0, CS1 e CS2, che servono per indirizzarla. RS0 e RS1 servono per selezionare il blocco A o B, R/W per interpretare l'operazione di lettura o scrittura del  $\mu P$ , Reset per azzerare il contenuto di tutti i registri interni, mentre Enable è collegato al clock del sistema.

Per comprendere meglio quanto esposto, si faccia riferimento alla Fig. 6.

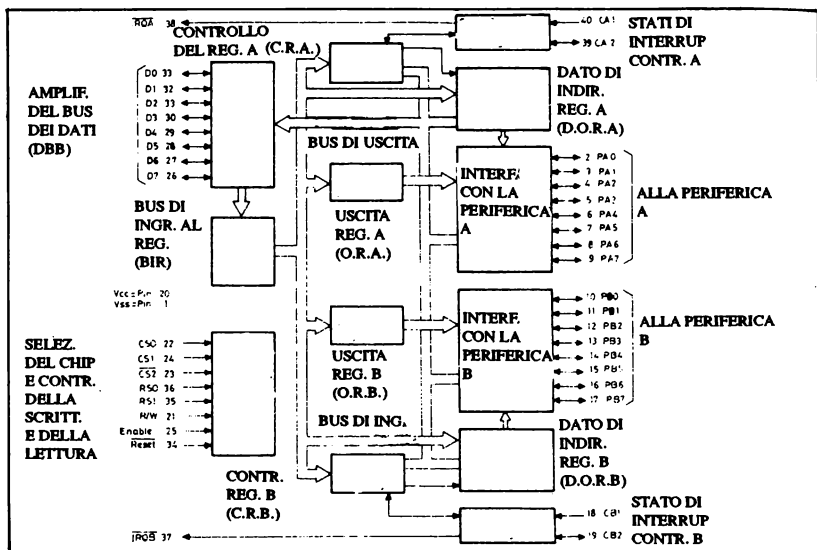


Fig. 6.-Blocchi interni della PIA MC-6820, collegabile al microprocessore 6800.

## Modi di indirizzamento del 6800

Il microprocessore 6800 dispone dei seguenti modi di indirizzamento:

- *immediato*: in questo tipo di indirizzamento il dato o operando dell'istruzione è fornito dopo il byte del codice macchina;
- *diretto*: questo indirizzamento si utilizza quando il dato si trova in una locazione di memoria indicata, a partire dall'inizio della pagina, dal secondo byte, perciò questo indirizzamento può solo arrivare a 256 posizioni a partire dall'inizio di una pagina. Viene anche detto paginato;
- *indicizzato*: il contenuto del secondo byte viene sommato a quello del registro indice per ottenere l'indirizzo finale, ma il contenuto di tale registro non si altera poichè il risultato viene salvato in un registro ausiliario a 16 bit;
- *esteso*: il secondo e il terzo byte determinano l'indirizzo assoluto su cui opererà la corrispondente istruzione;
- *implicito o inerente*: il dato o operando è implicito nell'istruzione;
- *relativo*: il contenuto del secondo byte dell'istruzione viene sommato al contenuto del registro contatore di programma per ottenere l'indirizzo finale. Si usa nelle istruzioni di salto.

## Interrupts del microprocessore 6800

Gli interrupts nel  $\mu P$  6800 sono trattati mediante i segnali introdotti via hardware sui terminali NMI e IRQ nel modo seguente: quando viene attivato il segnale NMI: resettandolo a basso livello, inizia una sequenza di *interrupt non mascherabile*, il che significa che l'interrupt viene sempre trattato, indipendentemente dal valore del bit 1 del registro di stato.

Questo interrupt provoca il caricamento del PC con il contenuto del vettore memoria, costituito dagli indirizzi FFFC e FFFD, in cui è memorizzato l'indirizzo iniziale della routine di trattamento di tale interrupt.

Quando il segnale IRQ viene attivato dall'esterno, sempre mediante un livello basso di tensione, viene eseguita una sequenza di *interrupt mascherabile*, cioè l'interrupt verrà eseguito solo se il bit 1 del registro di stato è a livello zero.

Il vettore memoria per il trattamento di tale interrupt è localizzato alle posizioni FFF8 e FFF9.

# ISTRUZIONI DEL REGISTRO INDICE E DELLO STACK

| Operazioni                         | Mnemonico | Modo di indirizzamento |         |             |        |           | Registro di stato                            |  |  |  |  |  |  |  |  |  |
|------------------------------------|-----------|------------------------|---------|-------------|--------|-----------|--|--|--|--|--|--|--|--|--|--|
|                                    |           | Immediato              | Diretto | Indirizzato | Esteso | Implicito | Commenti                                     |  |  |  |  |  |  |  |  |  |
|                                    |           | OP ~ #                 | OP ~ #  | OP ~ #      | OP ~ # | OP ~ #    |  |  |  |  |  |  |  |  |  |  |
| Compare Index Reg                  | CPX       | 8C 3 3                 | 9C 4 2  | AC 6 2      | 8C 5 3 |           | $X_i \leftarrow M[X_i - (M+1)]$              |  |  |  |  |  |  |  |  |  |
| Decrement Index Reg                | DEX       |                        |         |             |        | 09 4 1    | $X \leftarrow X - 1$                         |  |  |  |  |  |  |  |  |  |
| Decrement Stack Pntr               | DES       |                        |         |             |        | 34 4 1    | $SP \leftarrow SP - 1$                       |  |  |  |  |  |  |  |  |  |
| Increment Index Reg                | INX       |                        |         |             |        | 03 4 1    | $X \leftarrow X + 1$                         |  |  |  |  |  |  |  |  |  |
| Increment Stack Pntr               | INS       |                        |         |             |        | 31 4 1    | $SP \leftarrow SP + 1$                       |  |  |  |  |  |  |  |  |  |
| Load Index Reg                     | LDX       | CE 3 3                 | DE 4 2  | EE 6 2      | FE 5 3 |           | $M \rightarrow X_i, (M+1) \rightarrow X_i$   |  |  |  |  |  |  |  |  |  |
| Load Stack Pntr                    | LDS       | 8E 3 3                 | 9E 4 2  | AE 6 2      | BE 5 3 |           | $M \rightarrow SP_i, (M+1) \rightarrow SP_i$ |  |  |  |  |  |  |  |  |  |
| Store Index Reg                    | STX       | OF 5 2                 | EF 7 2  | FF 6 3      |        |           | $X_i \rightarrow M[X_i - (M+1)]$             |  |  |  |  |  |  |  |  |  |
| Store Stack Pntr                   | STS       | 9F 5 2                 | AF 7 2  | BF 6 3      |        |           | $SP_i \rightarrow M[SP_i - (M+1)]$           |  |  |  |  |  |  |  |  |  |
| Index Reg $\rightarrow$ Stack Pntr | TXS       |                        |         |             |        | 35 4 1    | $X \leftarrow SP$                            |  |  |  |  |  |  |  |  |  |
| Stack Pntr $\rightarrow$ Index Reg | TSX       |                        |         |             |        | 30 4 1    | $SP \leftarrow X$                            |  |  |  |  |  |  |  |  |  |

## SIGNIFICATO DEI SIMBOLI UTILIZZATI

OP = Codice di operazione. # = Numero di cicli. # = Numero di bytes. + = Somma. - = Differenza. \* = Funzione logica AND.  
M-SP = Posizione di memoria mediante il puntatore di stack. + = Funzione logica OR. + = Complemento di M.  $\rightarrow$  = Trasferimento logico.  
0 = Bit = 0. 00 = Byte = 0.

## SIMBOLI DEL REGISTRO DI STATO

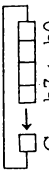
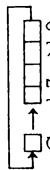

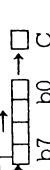
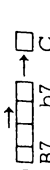
H = Carry di mezzo byte. I = Interrupt mascherato. N = segno negativo. Z = dato uguale a zero. V = Overflow. C= Carry. R = Reset. S = Set.  
 $\uparrow$  = Test e set se la condizione è verificata. \* = Bit non influenzato.



# TABELLE DELLE ISTRUZIONI DEL $\mu P$ 6800

Istruzioni dell'accumulatore e della memoria

| Operazioni               | Modo di indirizzamento |           |        |         |        |             |     |        |     |                                    | Commenti | Registro di dato |   |   |   |   |   |    |
|--------------------------|------------------------|-----------|--------|---------|--------|-------------|-----|--------|-----|------------------------------------|----------|------------------|---|---|---|---|---|----|
|                          | Memoristico            | Immediato |        | Diretto |        | Indirizzato |     | Esteso |     | Implicito                          |          | 8                | 4 | 3 | 2 | 1 | 0 |    |
|                          |                        | OP        | ~ #    | OP      | ~ #    | OP          | ~ # | OP     | ~ # |                                    |          |                  |   |   |   |   |   | OP |
| Add                      | ADDA                   | 9B        | 2 2 9B | 3 2 AB  | 5 2 BB | 4 3         |     |        |     |                                    | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
| Add Acmltr               | ADDB                   | DB        | 2 2 DB | 3 2 EB  | 5 2 FB | 4 3         |     |        |     |                                    | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
| Add with Carry           | ABA                    |           |        |         |        |             |     |        |     | 1B 2 1                             | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
| And                      | ADCA                   | 89        | 2 2 99 | 3 2 A9  | 5 2 B9 | 4 3         |     |        |     |                                    | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
|                          | ADCB                   | C9        | 2 2 D9 | 3 2 E9  | 5 2 F9 | 4 3         |     |        |     |                                    | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
|                          | AND4                   | B4        | 2 2 94 | 3 2 A4  | 5 2 B4 | 4 3         |     |        |     |                                    | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
|                          | ANDB                   | C4        | 2 2 D4 | 3 2 E4  | 5 2 F4 | 4 3         |     |        |     |                                    | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
| Bit Test                 | BITA                   | 85        | 2 2 95 | 3 2 A5  | 5 2 B5 | 4 3         |     |        |     |                                    | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
|                          | BITB                   | C5        | 2 2 D5 | 3 2 E5  | 5 2 F5 | 4 3         |     |        |     |                                    | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
| Clear                    | CLR                    |           |        | 6F      | 7 2 7F | 6 3         |     |        |     |                                    | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
|                          | CLRA                   |           |        |         |        |             |     |        |     | 4F 2 1 00                          | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
|                          | CLRB                   |           |        |         |        |             |     |        |     | 5F 2 1 00                          | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
| Compare                  | CMPA                   | 81        | 2 2 91 | 3 2 A1  | 5 2 B1 | 4 3         |     |        |     |                                    | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
| Compare Acmltr           | CMPB                   | C1        | 2 2 D1 | 3 2 E1  | 5 2 F1 | 4 3         |     |        |     |                                    | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
| Complement, 1's          | CBA                    |           |        |         |        |             |     |        |     | 51 2 1 A-B                         | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
|                          | COM                    |           |        |         |        |             |     |        |     |                                    | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
|                          | COMA                   |           |        | 63      | 7 2 73 | 6 3         |     |        |     |                                    | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
|                          | COMB                   |           |        |         |        |             |     |        |     | 43 2 1 A→A                         | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
| Complement, 2's (Negate) | NEG                    |           |        |         |        |             |     |        |     | 53 2 1 8→B                         | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
|                          | NEGA                   |           |        | 80      | 7 2 70 | 6 3         |     |        |     |                                    | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
|                          | NEGB                   |           |        |         |        |             |     |        |     | 40 2 1 00-A→A                      | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
| Decimal Adjust, A        | DAA                    |           |        |         |        |             |     |        |     | 50 2 1 00-B→B                      | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
|                          |                        |           |        |         |        |             |     |        |     | 19 2 1                             | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
| Decrement                | DEC                    |           |        | 6A      | 7 2 7A | 6 3         |     |        |     | CONVERTIRE I DATI<br>BINARI IN BCD | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
|                          | DECA                   |           |        |         |        |             |     |        |     | M-1→M                              | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
|                          | DECB                   |           |        |         |        |             |     |        |     | 4A 2 1 A-1→A                       | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |
|                          |                        |           |        |         |        |             |     |        |     | 5A 2 1 B-1→B                       | ↑        | ●                | ↑ | ↑ | ↑ | ↑ |   |    |

| Operazioni                | Modo di indirizzamento       |                                      |                                      |                                      |                                      |                                      | Commenti  | Registro di stato |                  |                  |                  |                  |                  |   |
|---------------------------|------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|---|-------------------|------------------|------------------|------------------|------------------|------------------|---|
|                           | Mnemonic                     | Immediato                            | Diretto                              | Indirizzato                          | Esteso                               | Implicito                            |   | 8                 | 4                | 3                | 2                | 1                | 0                |   |
|                           |                              | OP ~ #                               | OP ~ #                               | OP ~ #                               | OP ~ #                               | OP ~ #                               |   | OP ~ #            | H                | I                | N                | Z                | V                | C |
| Exclusive OR<br>Increment | EORA<br>EORB<br>INC          | 88 2 2<br>C8 2 2                     | 98 3 2<br>D8 3 2                     | A8 5 2<br>E8 5 2<br>6C 7 2           | 88 4 3<br>F8 4 3<br>7C 6 3           |                                      | A + M → A<br>8 + M → B<br>M + 1 → M   | •<br>•<br>•       | •<br>•<br>•      | ↑<br>↑<br>↑      | ↑<br>↑<br>↑      | ↑<br>↑<br>↑      | 0<br>V<br>C      |   |
|                           | INCA<br>INCB                 |                                      |                                      |                                      |                                      | 4C 2 1<br>5C 2 1                     | A → A<br>1 B + 1 → B  | •<br>•            | •<br>•           | ↑<br>↑           | ↑<br>↑           | ↑<br>↑           | •<br>•           |   |
|                           | LDAA<br>LDAB<br>ORAA<br>ORAB | 86 2 2<br>C6 2 2<br>8A 2 2<br>CA 2 2 | 96 3 2<br>06 3 2<br>9A 3 2<br>DA 3 2 | A6 5 2<br>E6 5 2<br>AA 5 2<br>EA 5 2 | B6 4 3<br>F6 4 3<br>BA 4 3<br>FA 4 3 |                                      | M → A<br>M → B<br>A + M → A<br>B + M → B  | •<br>•<br>•<br>•  | •<br>•<br>•<br>• | ↑<br>↑<br>↑<br>↑ | ↑<br>↑<br>↑<br>↑ | ↑<br>↑<br>↑<br>↑ | •<br>•<br>•<br>• |   |
| Push Data<br>Pull Oata    | PSHA<br>PSHB<br>PULA<br>PULB |                                      |                                      |                                      |                                      | 36 4 1<br>37 4 1<br>32 4 1<br>33 4 1 | A → M <sub>SP</sub><br>B → M <sub>SP</sub><br>SP + 1 → SP<br>SP + 1 → SP          | •<br>•<br>•<br>•  | •<br>•<br>•<br>• | •<br>•<br>•<br>• | •<br>•<br>•<br>• | •<br>•<br>•<br>• | •<br>•<br>•<br>• |   |
|                           | ROL<br>ROLA<br>ROLB          |                                      |                                      | 69 7 2                               | 79 6 3                               | M<br>49 2 1 A<br>59 2 1 B            |  | •<br>•<br>•       | •<br>•<br>•      | •<br>•<br>•      | •<br>•<br>•      | •<br>•<br>•      | •<br>•<br>•      |   |
|                           | ROR<br>RORA<br>RORB          |                                      |                                      | 66 7 2                               | 76 6 3                               | M<br>46 2 1 A<br>56 2 1 B            |  | •<br>•<br>•       | •<br>•<br>•      | •<br>•<br>•      | •<br>•<br>•      | •<br>•<br>•      | •<br>•<br>•      |   |
| Shift Left, Arithmetic    | ASL<br>ASLA<br>ASLB          |                                      |                                      | 68 7 2                               | 78 6 3                               | M<br>48 2 1 A<br>58 2 1 B            |  | •<br>•<br>•       | •<br>•<br>•      | •<br>•<br>•      | •<br>•<br>•      | •<br>•<br>•      | •<br>•<br>•      |   |
|                           | ASR<br>ASRA<br>ASRB          |                                      |                                      | 67 7 2                               | 77 6 3                               | M<br>47 2 1 A<br>57 2 1 B            |  | •<br>•<br>•       | •<br>•<br>•      | •<br>•<br>•      | •<br>•<br>•      | •<br>•<br>•      | •<br>•<br>•      |   |
|                           | LSR<br>LSRA<br>LSRB          |                                      |                                      | 64 7 2                               | 74 6 3                               | M<br>44 2 1 A<br>54 2 1 B            |  | •<br>•<br>•       | •<br>•<br>•      | •<br>•<br>•      | •<br>•<br>•      | •<br>•<br>•      | •<br>•<br>•      |   |

| Operazioni:         | Modo di indirizzamento |           |     |         |     |             |     |        |     |           | Registro di stato |          |      |   |   |   |   |   |
|---------------------|------------------------|-----------|-----|---------|-----|-------------|-----|--------|-----|-----------|-------------------|----------|------|---|---|---|---|---|
|                     | Mnemonic               | Immediato |     | Diretto |     | Indirizzato |     | Esteso |     | Implicito |                   | Commenti | 8    | 4 | 3 | 2 | 1 | 0 |
|                     |                        | Op        | ~ # | Op      | ~ # | Op          | ~ # | Op     | ~ # | Op        | ~ #               |          |      |   |   |   |   |   |
|                     |                        |           |     |         |     |             |     |        |     |           |                   |          |      |   |   |   |   |   |
| Store Acnfltr       | STAA                   |           | 97  | 4 2     | A7  | 6 2         | B7  | 5 3    |     |           | A → M             | ●        | ●    | ↑ | ↑ | ↑ | ↑ | R |
| Subtract            | STAB                   |           | D7  | 4 2     | E7  | 6 2         | F7  | 5 3    |     |           | B → M             | ●        | ●    | ↑ | ↑ | ↑ | ↑ | R |
|                     | SUBA                   | 80        | 2 2 | 90      | 3 2 | A0          | 5 2 | B0     | 4 3 |           | A-M → A           | ●        | ●    | ↑ | ↑ | ↑ | ↑ | R |
| Subtract Acnfltrs   | SUBB                   | C0        | 2 2 | D0      | 3 2 | E0          | 5 2 | F0     | 4 3 |           | B-M → B           | ●        | ●    | ↑ | ↑ | ↑ | ↑ | R |
|                     | SBA                    |           |     |         |     |             |     |        | 10  | 2 1       | A-B → A           | ●        | ●    | ↑ | ↑ | ↑ | ↑ | R |
| Subtr. with Carry   | SBCA                   | 82        | 2 2 | 92      | 3 2 | A2          | 5 2 | B2     | 4 3 |           | A-M-C → A         | ●        | ●    | ↑ | ↑ | ↑ | ↑ | R |
|                     | SBCB                   | C2        | 2 2 | D2      | 3 2 | E2          | 5 2 | F2     | 4 3 |           | B-M-C → B         | ●        | ●    | ↑ | ↑ | ↑ | ↑ | R |
| Transfer Acnfltrs   | TAB                    |           |     |         |     |             |     |        | 16  | 2 1       | A → B             | ●        | ●    | ↑ | ↑ | ↑ | ↑ | R |
|                     | TBA                    |           |     |         |     |             |     |        | 17  | 2 1       | B → A             | ●        | ●    | ↑ | ↑ | ↑ | ↑ | R |
| Test. Zero or Minus | TST                    |           |     |         |     |             |     |        |     |           | M-00              | ●        | ●    | ↑ | ↑ | ↑ | ↑ | R |
|                     | TSTA                   |           |     |         |     |             |     |        | 4D  | 2 1       | A-00              | ●        | ●    | ↑ | ↑ | ↑ | ↑ | R |
|                     | TSTB                   |           |     |         |     |             | 6D  | 7 2    | 7D  | 6 3       | 5D                | 2 1      | B-00 | ● | ● | ↑ | ↑ | R |

## Istruzioni di salto

| Operazioni               | Mnemonic | Modo di indirizzamento |   |             |   |        |    |           |   |  |   | Commenti | Registro di stato |   |   |   |  |  |  |
|--------------------------|----------|------------------------|---|-------------|---|--------|----|-----------|---|--|---|----------|-------------------|---|---|---|--|--|--|
|                          |          | Immediato              |   | Indirizzato |   | Esteso |    | Implicito |   | 8  | 4 |          | 3                 | 2 | 1 | 0 |  |  |  |
|                          |          | OP                     | # | OP          | # | OP     | #  | OP        | # |  |   |          |                   |   |   |   |  |  |  |
|                          |          |                        |   |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Branch Always            | BRA      | 20                     | 4 |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Branch If Carry Clear    | BCC      | 24                     | 4 |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Branch If Carry Set      | BCS      | 25                     | 4 |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Branch If =Zero          | BEQ      | 27                     | 4 |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Branch If ≥Zero          | BGE      | 2C                     | 4 |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Branch If >Zero          | BGT      | 2E                     | 4 |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Branch If Higher         | BHI      | 22                     | 4 |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Branch If ≤Zero          | BLE      | 2F                     | 4 |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Branch If Lower Or Same  | BLS      | 23                     | 4 |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Branch If < Zero         | BLT      | 2D                     | 4 |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Branch If Minus          | BMI      | 2B                     | 4 |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Branch If Not Equal Zero | BNE      | 26                     | 4 |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Branch If Overflow Clear | BVC      | 28                     | 4 |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Branch If Overflow Set   | BVS      | 29                     | 4 |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Branch If Plus           | BPL      | 2A                     | 4 |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Branch To Subroutine     | BSR      | 8D                     | 8 |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Jump                     | JMP      |                        |   | 6E          | 4 | 2      | 7E | 3         | 3 | }<br>2 1<br>3B 10 1<br>39 5 1<br>3F 12 1<br>3E 9 1 |   |          |                   |   |   |   |  |  |  |
| Jump To Subroutine       | JSR      |                        |   | AD          | 8 | 2      | BD | 9         | 3 |  |   |          |                   |   |   |   |  |  |  |
| No Operation             | NOP      |                        |   |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Return From Interrupt    | RTI      |                        |   |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Return From Subroutine   | RTS      |                        |   |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Software Interrupt       | SWI      |                        |   |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |
| Wait for interrupt       | WAI      |                        |   |             |   |        |    |           |   |  |   |          |                   |   |   |   |  |  |  |

| Significato dei simboli utilizzati   |  |
|--|--|
| <p>           OP = Codice di operazione<br/>           n = Numero di cicli<br/>           # = Numero di bytes.<br/>           + = Somma<br/>           - = Differenza<br/>           • = Funzione logica AND<br/>           M, SP = Posizione di memoria mediante il puntatore di stack         </p> | <p>           + = Funzione logica OR<br/>           ⊕ = Funzione logica XOR<br/>           M = Complemento di M<br/>           → = Trasferimento verso.<br/>           0 = Bit = 0<br/>           00 = Byte = 0         </p> |
| <p>NOTA: Le istruzioni con indirizzamento dell'accumulatore sono incluse nella colonna di indirizzamento implicito.</p>  |  |

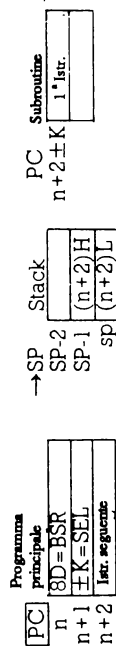
| Istruzioni di manipolazione del registro di stato  |  |  |                                      |                                      |  |                                      |                                      |                                      |                                      |                                      |                                      |   |
|--|--|--|--------------------------------------|--------------------------------------|--|--------------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|---|
| Operazioni   | Mnemonic   | Indirizzamento                               |                                      |                                      | Commenti   | Registro di stato                    |                                      |                                      |                                      |                                      |                                      |   |
|  |  | OP   | Implicito                            |                                      |  | S                                    | 4                                    | 3                                    | 2                                    | 1                                    | 0                                    |   |
|  |  |  | ~                                    | #                                    |  |                                      |                                      |                                      |                                      |                                      |                                      |   |
|  |  |  |                                      |                                      |  |                                      |                                      |                                      |                                      |                                      |                                      | H |
| Clear Carry<br>Clear Interrupt Mask<br>Clear Overflow<br>Set Carry<br>Set Interrupt Mask<br>Set Overflow<br>Acmltr A → CCR<br>CCR → Acmltr A | CLC<br>CLI<br>CLV<br>SEC<br>SEI<br>SEV<br>TAP<br>TPA | 0C<br>0E<br>0A<br>0D<br>0F<br>0B<br>06<br>07 | 2<br>2<br>2<br>2<br>2<br>2<br>2<br>2 | 1<br>1<br>1<br>1<br>1<br>1<br>1<br>1 | 0 → C<br>0 → I<br>0 → V<br>1 → C<br>1 → I<br>1 → V<br>A → CCR<br>CCR → A | •<br>•<br>•<br>•<br>•<br>•<br>•<br>• | •<br>R<br>•<br>•<br>S<br>•<br>•<br>• | •<br>•<br>•<br>•<br>•<br>•<br>•<br>• | •<br>•<br>•<br>•<br>•<br>•<br>•<br>• | •<br>•<br>•<br>•<br>•<br>•<br>•<br>• | •<br>•<br>•<br>•<br>•<br>•<br>•<br>• |   |

| Note al registro di stato |  |
|---------------------------|--|
| 1                         | (Bit V) test: risultato = 100000007  |
| 2                         | (Bit C) test: risultato = 000000007  |
| 3                         | (Bit C) test: il valore decimale del carattere BCD è maggiore di 9?  |
| 4                         | (Bit V) test: l'operando = 10000000 maggiore all'esecuzione?   |
| 5                         | (Bit V) test: l'operando = 01111111 maggiore all'esecuzione?   |
| 6                         | (Bit V) test: lo pone uguale al risultato di NOC dopo lo spazzamento   |
| 7                         | (Bit N) test: il bit di segno = 1?   |
| 8                         | (Bit N) test: complemento a due della sottrazione?   |
| 9                         | (Bit N) test: risultato minore di zero? (Bit 15 = 1)   |
| 10                        | (Tutti) carica il registro di stato dallo stack  |
| 11                        | (Bit I) selezionato quando avviene un interrupt. Se è stato preventivamente effettuato un interrupt non mascherato richiede lo stato di attesa             |
| 12                        | (Tutti) Selezionati secondo il contenuto dell' accumulatore A. (Il bit selezionato è settato a uno se la condizione è vera e resettato in caso contrario.) |

| Simboli del registro di stato |   |
|-------------------------------|---|
| II                            | = Carry di mezzo byte                       |
| I                             | = Interrupt mascherato                      |
| N                             | = Segno negativo                            |
| Z                             | = Dato uguale a zero                        |
| V                             | = Overflow                                  |
| C                             | = Carry                                     |
| R                             | = Reset (settaggio a zero)                  |
| S                             | = Set (settaggio a uno)                     |
| •                             | = Bit non influenzato                       |
| ↑                             | = Test e settaggio se la condizione è vera. |



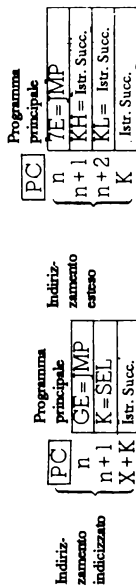
### BSR: ISTRUZIONI DI CHIAMATA A SUBROUTINE



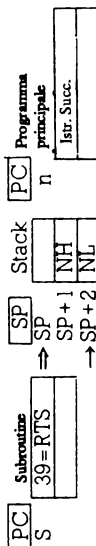
K = Dato a 8 bit  $n+2$  formato da  $(n+2)$

6 (ae + 2) L

### JMP: ISTRUZIONI DI SALTO



### RST: ISTRUZIONI DI RIENTRO DA SUBROUTINE



### RTI: ISTRUZIONI DI RITORNO CON INTERRUPT

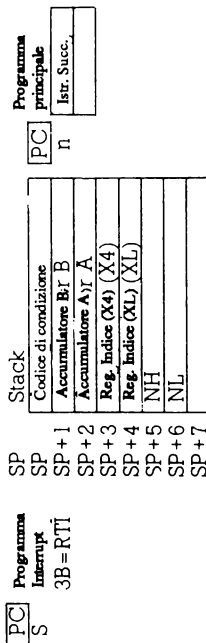




Tabella del tipo di indirizzamento di ogni istruzione e suo tempo di esecuzione in microcicli

|     | Doppio<br>operando | Accumul. X | Immediato | Diretto | Ritardo | Ritardo | Indicizzato | Implicito | Relativo |
|-----|--------------------|------------|-----------|---------|---------|---------|-------------|-----------|----------|
| ABA |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | ●        |
| ADC | X                  | ●          | 2         | 3       | 4       | ●       | 5           | ●         | 2        |
| ADD | X                  | ●          | 2         | 3       | 4       | ●       | 5           | ●         | ●        |
| AND | X                  | ●          | 2         | 3       | 4       | ●       | 5           | ●         | ●        |
| ASL |                    | 2          | ●         | ●       | 6       | ●       | 7           | ●         | ●        |
| ASR |                    | 2          | ●         | ●       | 6       | ●       | 7           | ●         | ●        |
| BCC |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| BCS |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| BEA |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| BGE |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| BGT |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| BHI |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| BIT | X                  | ●          | 2         | 3       | 4       | ●       | 5           | ●         | ●        |
| BLE |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| BLS |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| HLT |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| HLT |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| BMI |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| BNE |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| BPL |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| BRA |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| BSR |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 8        |
| BVC |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| BVS |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| CBA |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 2        |
| CLC |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | ●        |
| CLI |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 2        |
| CLR |                    | 2          | ●         | ●       | 6       | ●       | 7           | ●         | 2        |
| CLV |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | ●        |
| CMV | X                  | ●          | 2         | 3       | 4       | ●       | 5           | ●         | 2        |
| COM |                    | 2          | ●         | ●       | 6       | ●       | 7           | ●         | ●        |
| CPX |                    | ●          | 3         | 4       | 5       | ●       | 6           | ●         | ●        |
| DAA |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 2        |
| DEC |                    | 2          | ●         | ●       | 6       | ●       | 7           | ●         | ●        |
| DES |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| DEX |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| EOR | X                  | ●          | 2         | 3       | 4       | ●       | 5           | ●         | ●        |
| INC |                    | 2          | ●         | ●       | 6       | ●       | 7           | ●         | ●        |
| INS |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| INX |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| JMP |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| JSR |                    | ●          | ●         | ●       | 3       | ●       | 4           | ●         | ●        |
| LDA | X                  | ●          | 2         | 3       | 4       | ●       | 5           | ●         | 2        |
| LDS |                    | ●          | 3         | 4       | 5       | ●       | 6           | ●         | 2        |
| LDX |                    | ●          | 3         | 4       | 5       | ●       | 6           | ●         | 2        |
| LSR |                    | 2          | ●         | ●       | 6       | ●       | 7           | ●         | ●        |
| NEG |                    | 2          | ●         | ●       | 6       | ●       | 7           | ●         | ●        |
| NOP |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 2        |
| ORA | X                  | ●          | 2         | 3       | 4       | ●       | 5           | ●         | ●        |
| PSH |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| PUL |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| ROL |                    | 2          | ●         | ●       | 6       | ●       | 7           | ●         | ●        |
| ROR |                    | ●          | ●         | ●       | 6       | ●       | 7           | ●         | ●        |
| RTI |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 10       |
| RTS |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 5        |
| SBA |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 2        |
| SBC | X                  | ●          | 2         | 3       | 4       | ●       | 5           | ●         | ●        |
| SEC |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 2        |
| SEI |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 2        |
| SEV |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 2        |
| STA | X                  | ●          | ●         | 4       | 5       | ●       | 6           | ●         | ●        |
| STS |                    | ●          | ●         | 5       | 6       | ●       | 7           | ●         | ●        |
| STX |                    | ●          | ●         | 5       | 6       | ●       | 7           | ●         | ●        |
| SUB | X                  | ●          | 2         | 3       | 4       | ●       | 5           | ●         | 12       |
| SWI |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 2        |
| TAB |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 2        |
| TAP |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 2        |
| TBA |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 2        |
| TPA |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 2        |
| TST |                    | 2          | ●         | ●       | 6       | ●       | 7           | ●         | ●        |
| TSX |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| TSX |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 4        |
| WAI |                    | ●          | ●         | ●       | ●       | ●       | ●           | ●         | 9        |

*Tabella del significato delle istruzioni*

|                                      |  |
|--------------------------------------|--|
| ABA Somma di accumulatori            | NOP Nessuna operazione   |
| ADD Somma con carry                  | ORA Funzione or con l'accumulatore                             |
| ADC Somma                            | PSH Inserisce dato in stack                                    |
| AND Funzione logica And              | PUL Estrae dato da stack                                       |
| ASL Spostamento a sinistra           | ROL Rotazione verso sinistra                                   |
| BRA Salto sempre                     | ROR Rotazione verso destra                                     |
| BSR Salto a subroutine               | RTI Ritorno da sub. di interrupt                               |
| BVC Salto se non c'è overflow        | RTS Ritorno da subroutine                                      |
| BVS Salto se c'è overflow            | SBA Sottrazione con l'accumulatore                             |
| CBA Confronto tra accumulatori       | SBC Sottrazione con carry                                      |
| CLC Azzerà carry                     | SEC Selezione di carry   |
| CLI Cancella interrupt mascherato    | BIT Test del bit   |
| CLR Azzerà                           | BLE Salto se minore o uguale                                   |
| CLV Azzerà overflow                  | BLS Salto se inferiore a dato                                  |
| CMP Confronto                        | BLT Salto se minore di zero                                    |
| COM Complemento                      | BMI Salto se minore  |
| CPX Confronto con il registro indice | BNE Salto se diverso da zero                                   |
| DAA Regolazione decimale             | BPL Salto se maggiore  |
| DEC Decremento                       | SEI Selezione interrupt mascherato                             |
| DES Incrementa il puntatore di stack | SEV Selezione dell'overflow                                    |
| DEX Decrementa il registro indice    | STA Memorizzazione nel-<br>l'accumulatore                      |
| EOR Funzione logica or esclusivo     | STS Memorizzazione del registro di<br>stack                    |
| INC Incrementa                       | STX Memorizzazione registro indice                             |
| INS Incrementa il puntatore di stack | SUB Sottrazione  |
| ASR Spostamento verso destra         | SWI Interrupt via software                                     |
| BCC Salto se non c'è carry           | TAB Trasferimento tra accumulatori                             |
| BCS Salto e c'è carry                | TAP Trasferimento degli<br>accumulatori allo status            |
| BEQ Salto se diverso da zero         | TBA Trasferimento tra accumulatori                             |
| BGE Salto se maggiore o uguale zero  | TPA Trasferimento dallo status<br>agli accumulatori            |
| BGT Salto se è maggiore di zero      | TST Test   |
| BHI Salto se superiore               | TSX Trasferimento del puntatore di<br>stack al registro indice |
| INX Incrementa il registro indice    | TXS Trasferimento del registro<br>indice al puntatore di stack |
| JMP Salto                            | WAI Attesa di interrupt  |
| JSR Salto a subroutine               |  |
| LDA Carica l'accumulatore            |  |
| LDS Carica il puntatore di stack     |  |
| LDX Carica il registro indice        |  |
| LSR Rotazione a destra               |  |
| NEG Negazione                        |  |

## *Repertorio delle istruzioni del 6800 suddivise per operazioni*

### *1. Operazioni sui registri a 8 bit*

- |    |                                     |                            |
|----|-------------------------------------|----------------------------|
| A. | Operazioni aritmetiche a 2 operandi | ABC ADC ADD SBA<br>SBC SUB |
| B. | Operazioni aritmetiche a 1 operando | CLR DAA DEC INC NEG        |
| C. | Confronti e tests (Prove)           | CBA CMP TST                |
| D. | Scorrimento e rotazione             | ASL ASR LSR ROL ROR        |
| E. | Funzioni logiche                    | AND BIT COM EOR ORA        |
| F. | Caricamento e memorizzazione        | LDA STA PSH PUL            |
| G. | Trasferimenti                       | TAB TBA                    |

### *2. Controllo dei salti*

- |    |                          |   |
|----|--------------------------|---|
| A. | Salti condizionati       | BCC BCS BEQ BGE BGT<br>BHI BLE BLS BLT BMI<br>BNE BPL BVC BVS |
| B. | Salti incondizionati     | BRA NOP JMP   |
| C. | Controlli di subroutines | BSR JSR RTS   |
| D. | Controlli di interrupts  | RTI SWI WAI   |

### *3. Controlli del registro indice e del puntatore di stack*

- |    |                    |                     |
|----|--------------------|---------------------|
| A. | Registro indice    | DEX INX LDX STX CPX |
| B. | Puntatore di stack | DES INS LDS STS     |
| C. | Trasferimenti      | TSX TXS             |

### *4. Controllo del registro di stato*

- |    |                       |                            |
|----|-----------------------|----------------------------|
| A. | Controllo di bit      | CLC CLI CLV SEC SEI<br>SEV |
| B. | Trasferimento di byte | TAP TPA                    |

### *5. Specifiche complessive*

END EQU FCB FCC FDB  
MON NAM OPT ORG  
PAGE RMB SPC

## Il microprocessore 6802

Questo microprocessore venne commercializzato dalla Motorola come variante del 6800 dopo i vari progressi nel campo dell'integrazione. In effetti il 6802 comprende nello stesso CHIP i circuiti completi del 6800, il circuito di clock e 128 bytes di memoria RAM.

In tal modo il 6802 è completamente compatibile, per quanto riguarda il software con il 6800.

Il microprocessore 6802 si presenta incapsulato nel classico contenitore a quaranta piedini. La distribuzione dei segnali sugli stessi è la stessa del 6800, tranne per quattro terminali che cambiano denominazione e che sono:

- XTAL 1 e XTAL 2, che nel 6802 costituiscono i collegamenti esterni per il cristallo di quarzo che regola la frequenza del generatore di clock interno su quella propria del quarzo (nel 6800 il generatore del clock è completamente esterno); nel 6800 XTAL 1 è occupato da TSC (controllo del three state) e XTAL 2 da un terminale non collegato.
- MR, che è un terminale per collegare il  $\mu P$  a memorie e periferiche lente; nel 6800 questo terminale è occupato dall'ingresso  $\Phi 1$  del clock esterno.
- ME, terminale tramite cui si accede alla memoria RAM interna;

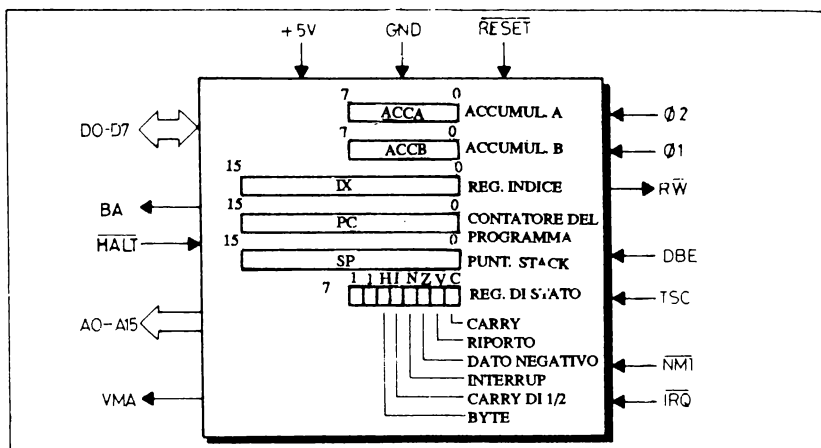


Fig. 7.-Dettaglio del blocco dei registri interni e della Ram a 128 bytes, integrati nel  $\mu P$  6802.

è l'ingresso che permette di leggere o scrivere in questa RAM. Nel 6800 questo terminale è occupato da DBE che controlla il bus dei dati.

- VCC-S, terminale che permette di preservare la memoria interna da black-out dell'alimentazione, collegandogli una tensione esterna a 5 V, fornita da pile o batterie che lo alimentino in qualsiasi caso.

L'inserimento di memoria RAM o PROM nello stesso CHIP del microprocessore, è una tendenza degli ultimi progetti, in quanto in tal modo, il CHIP del microprocessore si trasforma in una completa CPU per applicazioni semplici.

Per quanto riguarda la struttura interna del  $\mu P$  6802, questo è costituito da un blocco da 128 bytes o ottetti indirizzabili in esadecimale da 0000 a 007F, con la particolarità che i primi 32 sono non volatili e possono essere salvaguardati con il metodo descritto in precedenza.

Disporre di una zona di RAM non volatile è molto utile per il programmatore, poichè esiste la sicurezza che in essa rimarranno i dati più importanti ottenuti nel corso del programma e che si perderebbero dopo aver tolto l'alimentazione.

Gli altri registri sono identici a quelli del 6800.

## *Collegamenti alle periferiche nel 6802*

Il microprocessore 6802 dispone di una PIA progettata su misura, con codice 6846, alla quale si possono collegare le periferiche secondo il tipico modo di accoppiamento mediante PIA. Il circuito 6846, oltre a disporre della possibilità di otto uscite in parallelo, possiede anche terminali per il bus di controllo e interrupts, ed un contatore-temporizzatore interno a cui si accede tramite programma e che serve per ottenere ritardi, conteggi per display, ecc.

## *Caratteristiche del $\mu P$ 6502*

Il microprocessore che costituisce la CPU è il 6502 della Rockwell. Le sue caratteristiche sono:

- alimentazione a 5 V;
- costruito in tecnologia MOS, canale N;
- bus dei dati a otto bit;

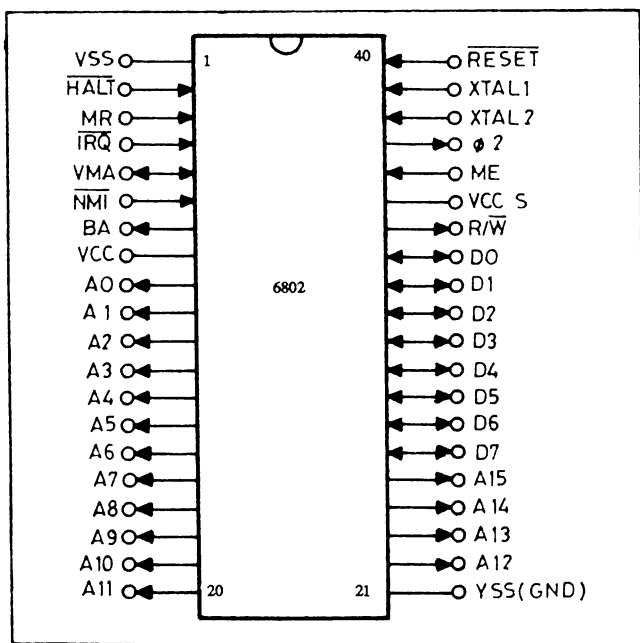


Fig. 8.-Terminali del microprocessore 6802 e loro funzioni.

- 56 istruzioni fondamentali;
- aritmetica in binario e decimale;
- modi di indirizzamento: diretto, relativo, immediato, indicizzato, implicito e indiretto;
- interrupts: logici (BRK), fisici (IRA), interrupt prioritario non mascherato (NMI);
- capacità di memoria: fino a 64 Kbytes;
- clock: da 1 MHz a 2 MHz.

In Fig. 9 si vede il contenitore a 40 terminali con il relativo pin-out che è il seguente:

- DB0-DB7: bus dei dati;
- AB0-AB15: bus degli indirizzi;
- R/W: segnale di lettura (R read) e scrittura (W write), per indicare alla memoria o alle periferiche quale sarà l'operazione successiva;

- RESET: azzeramento di tutti i registri interni per l'inizializzazione;
- RDY: segnale di sincronismo per le memorie lente;
- IRQ: richiesta di interrupt mascherabile;
- NMI: richiesta di interrupt non mascherabile;
- SYNC: segnale indicatore del ciclo di ricerca dell'istruzione;
- SO: flag di overflow;

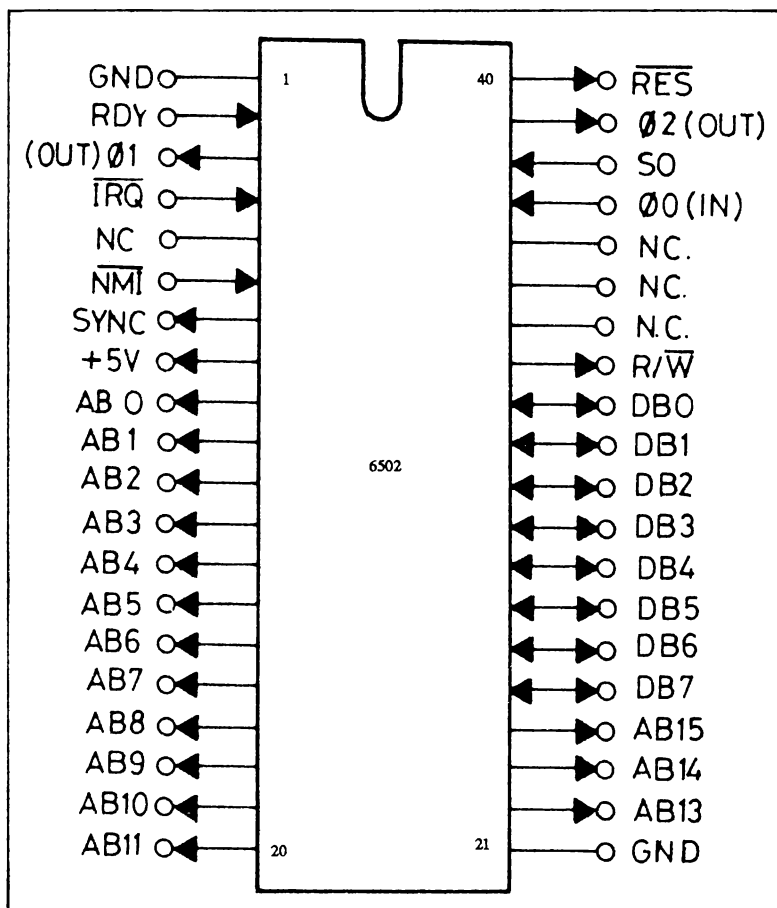


Fig. 9.-Aspetto esterno e distribuzione delle funzioni per ogni PIN del  $\mu P$  6502

- GND: massa (-);
- +5: alimentazione a 5 V;
- $\Phi_0$ : segnale di clock della CPU;
- $\Phi_1, \Phi_2$ : segnale di clock del sistema;
- NC: terminale non collegato.

## Configurazione interna del $\mu P$

La sua configurazione interna è quella caratteristica di tutti i  $\mu P$ , ma la sua particolarità è di disporre di sei registri interni:

- 1 accumulatore a otto bit per uso generale, detto A;
- 2 puntatori a otto bit ciascuno, utilizzati nei registri indice X e Y;
- 1 contatore di programma a 16 bit (gli otto più alti PCH e gli otto più bassi PCL);
- 1 puntatore di stack per il rientro da subroutine a otto bit detto S;
- 1 registro di stato, i cui bit hanno il significato seguente:

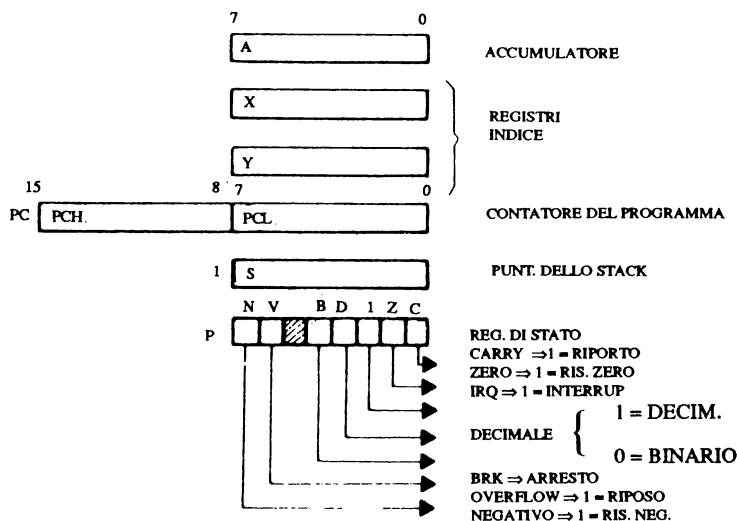


Fig. 10.-Registri interni del  $\mu P$  6502.



- 0-C bit di carry (Carry);
- 1-Z bit di zero (Zero);
- 2-I abilitazione interrupt;
- 3-D bit di operazione binaria o decimale;
- 4-B break per l'arresto completo;
- 5 non usato
- 6-V bit di overflow;
- 7-N bit di segno.

E' utile esaminare più dettagliatamente i bit del registro di stato:

- 0-C=bit di carry: indica se l'operazione precedente ha provocato o no un riporto, se il riporto si è verificato viene settato a 1;
- 1-Z=bit di zero: indica se l'operazione precedente ha avuto risultato zero, se il bit è uno il risultato zero si è verificato, altrimenti il bit è a zero;
- 2-I=abilitazione interrupt: se il bit è 1 l'interrupt è permesso, se zero non permesso;
- 3-D=bit di operazione binaria o decimale: se vale uno il modo è decimale, se zero, binario;
- 4-B=break di arresto: se 0 non esiste arresto, se uno il  $\mu P$  si blocca completamente;
- 5=non utilizzato;

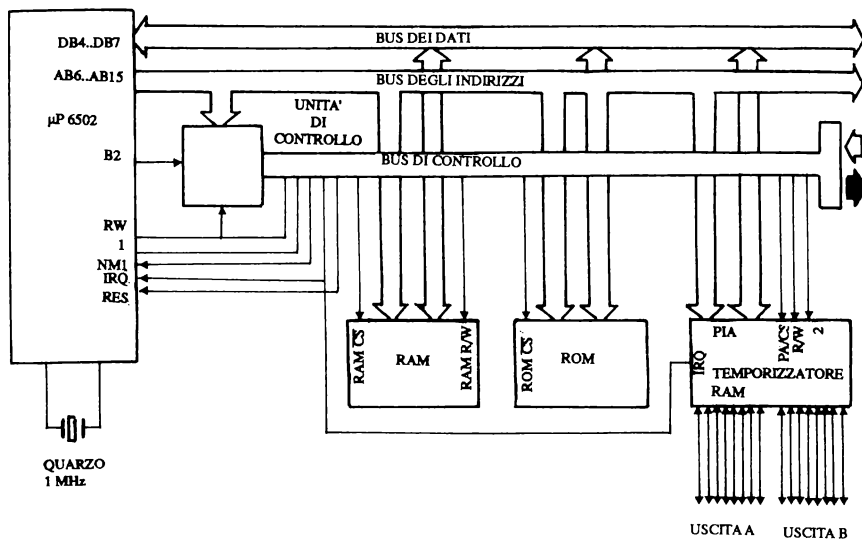


Fig. 11.-Blocchi funzionali del Junior Computer con dettaglio dei suoi bus.

- 6-V=bit di overflow: se viene settato a uno si è verificato un overflow;
- 7-N=bit di segno: se è uno indica che l'operazione precedente ha avuto risultato negativo.

## *Istruzioni del $\mu P$ 6502*

Il  $\mu P$  6502 dispone di 56 istruzioni di programmazione, che sono riferite ai registri interni ed al modo di indirizzamento.

Le descrizioni mnemoniche sono in inglese, poichè il campo dell'informatica si è sviluppato nei paesi con tale lingua. Tali istruzioni sono:

- di caricamento (LOAD): LDA, LDX, LDY carica il contenuto della memoria nel registro specificato (A, X, Y);
- di memorizzazione (STORE): STA, STX, STY memorizza il contenuto del registro specificato nella memoria;
- di trasferimento (TRANSFER): TAX, TAY, ecc. trasferisce il dato tra i registri specificati;
- di somma (ADD): ADC somma la memoria con l'accumulatore;
- di differenza (SUBTRACT): SBC sottrae l'accumulatore e la memoria;
- logiche: AND, ORA, EOR esegue una operazione logica tra la memoria e l'accumulatore;
- di salto condizionato (BRANCH): BBC, BCS, ecc. salta da una locazione di memoria ad un'altra, se si verifica una condizione;
- di azzeramento e settaggio a uno dello status (CLEAR e SET): CLC, CLD, ecc., e SEC, SED, ecc. azzerava e setta a uno, rispettivamente, il registro di stato indicato dall'istruzione;
- di confronto (COMPARE): CMP, CPX, ecc. confronta il registro indicato con la memoria;
- di incremento e decremento (INCREMENT, DECREMENT): INC, INX, ecc., e DEC, DEX, ecc. incrementa o decrementa di un'unità il registro indicato dall'istruzione;
- di salto (JUMP): JMP, JSR salta alla locazione di memoria indicata;
- di rotazione (ROTATE): ROL, ROR, ecc. ruota verso sinistra o destra il registro indicato dall'istruzione;

|     |  |     |  |
|-----|--|-----|--|
| ADC | Add Memory to Accumulator with Carry       | LDA | Load Accumulator with Memory                 |
| AND | «AND» Memory with Accumulator              | LDX | Load Index X with Memory                     |
| ASL | Shift left One Bit (Memory or Accumulator) | LDY | Load Index Y with Memory                     |
| BCC | Branch on Carry Clear                      | LSR | Shift One Bit Right (Memory or Accumulator)  |
| BCS | Branch on Carry Set                        | NOP | No Operation                                 |
| BEQ | Branch on Result Zero                      | ORA | «OR» Memory with Accumulator                 |
| BIT | Test Bits in Memory with Accumulator       | PHA | Push Accumulator on Stack                    |
| BMI | Branch on Result Minus                     | PHP | Push Processor Status on Stack               |
| BNE | Branch on Result not Zero                  | PLA | Pull Accumulator from Stack                  |
| BPL | Branch on Result Plus                      | PLP | Pull Processor Status from Stack             |
| BRK | Force Break                                | ROL | Rotate One Bit Left (Memory or Accumulator)  |
| BVC | Branch on Overflow Clear                   | ROR | Rotate One Bit Right (Memory or Accumulator) |
| BVS | Branch on Overflow Set                     | RTI | Return from Interrupt                        |
| CLC | Clear Carry Flag                           | RTS | Return from Subroutine                       |
| CLD | Clear Decimal Mode                         | SBC | Subtract Memory from Accumulator with Borrow |
| CLI | Clear Interrupt Disable Bit                | SEC | Set Carry Flag                               |
| CLV | Clear Overflow Flag                        | SED | Set Decimal Mode                             |
| CMP | Compare Memory and Accumulator             | SEI | Set Interrupt Disable Status                 |
| CPX | Compare Memory and Index X                 | STA | Store Accumulator in Memory                  |
| CPY | Compare Memory and Index Y                 | STX | Store Index X in Memory                      |
| DEC | Decrement Memory by One                    | STY | Store Index Y in Memory                      |
| DEX | Decrement Index X by One                   | TAX | Transfer Accumulator to Index X              |
| DEY | Decrement Index Y by One                   | TAY | Transfer Accumulator to Index Y              |
| EOR | «Exclusive» Memory with Accumulator        | TSX | Transfer Stack Pointer to Index X            |
| INC | Increment Memory by One                    | TXA | Transfer Index X to Accumulator              |
| INC | Increment Index X by One                   | TXS | Transfer Index X to Stack Register           |
| INY | Increment Index Y by One                   | TYA | Transfer Index Y to Accumulator              |
| JMP | Jump to New Location                       |     |  |
| JSR | Jump to New Location Saving Return Address |     |  |

| $\begin{array}{c c} I & I \end{array}$ | 0       | 1         | 2       | 3 | 4 | 5       | 6       | 7 | 8   | 9       | A   | B   | C        | D        | E        | F |
|--|---------|-----------|---------|---|---|---------|---------|---|-----|---------|-----|-----|----------|----------|----------|---|
| 0                                      | BRK     | ORA ind X |         |   |   | ORA PO  | ASL PO  |   | PHP | ORA imm | ASL | A   |          | ORA abs  | ASL abs  |   |
| 1                                      | BPL     | ORA ind Y |         |   |   | ORA POX | ASL POX |   | CLC | ORA abs | Y   |     |          | ORA absX | ASL absX |   |
| 2                                      | JSR     | AND ind X |         |   |   | AND PO  | ROL PO  |   | PLP | AND imm | ROL | A   |          | AND abs  | ROL abs  |   |
| 3                                      | BMI     | AND ind Y |         |   |   | AND POX | ROL POX |   | SEC | AND abs | Y   |     | BIT abs  | AND absX | ROL absX |   |
| 4                                      | RTI     | EOR ind X |         |   |   | EOR PO  | LSR PO  |   | PHA | EOR imm | LSR | A   | JMP abs  | EOR abs  | LSR abs  |   |
| 5                                      | BVC     | EOR ind Y |         |   |   | EOR POX | LSR POX |   | CLJ | EOR abs | Y   |     |          | EOR absX | LSR absX |   |
| 6                                      | RTS     | ADC ind X |         |   |   | ADC PO  | ROR PO  |   | PLA | ADC imm | ROR | A   | JMP ind  | ADC abs  | ROR abs  |   |
| 7                                      | BVS     | ADC ind Y |         |   |   | ADC POX | ROR POX |   | SEI | ADC abs | Y   |     |          | ADC absX | ROR absX |   |
| 8                                      | BCC     | STA ind X |         |   |   | STA PO  | STX PO  |   | DEY |         | TXA |     | STY abs  | STA abs  | STX abs  |   |
| 9                                      | LDY imm | STA ind Y |         |   |   | STA POX | STX POX |   | TYA | STA abs | Y   | TXS | LDY abs  | STA absX | LDX abs  |   |
| A                                      | BCS     | LDA ind X | LDX imm |   |   | LDA PO  | LDX PO  |   | TAY | LDA imm | Y   | TAX | LDY abs  | LDA abs  | LDX abs  |   |
| B                                      | CPY imm | LDA ind Y |         |   |   | LDA POX | LDX POX |   | CLV | LDA abs | Y   | TSX | LDY absX | LDA absX | LDX absX |   |
| C                                      | BCS     | CMP ind X |         |   |   | CMP PO  | DEC PO  |   | INY | CMP imm | Y   | DEX | CPY abs  | CMP abs  | DEC abs  |   |
| D                                      | CPY imm | CMP ind Y |         |   |   | CMP POX | DEC POX |   | CLD | CMP abs | Y   |     | CPX abs  | CMP absX | DEC absX |   |
| E                                      | BCQ     | SBC ind X |         |   |   | SBC PO  | INC PO  |   | INX | SBC imm | Y   |     | CPX abs  | SBC absX | INC abs  |   |
| F                                      | BEQ     | SBC ind Y |         |   |   | SBC POX | INC POX |   | SED | SBC abs | Y   |     |          | SBC absX | INC absX |   |

Tabella III. -Mappa delle istruzioni del  $\mu P$  6502. Esempio di utilizzo: 94=STY, pagina 0 indicizzato tramite X.

TABELLA IV

| Abbreviazioni<br>memoriche +<br>spiegazione  | Modo di<br>indirizzamento (5)                                       | Codice<br>esadecimale                                       | Numero di<br>impulsi<br>di clock     | Numero<br>di bytes                   | Flag<br>influenzati |
|--|---|---|--------------------------------------|--------------------------------------|---------------------|
| ADC<br>Add memory to accumulator<br>with carry<br>$A + M + C \rightarrow A(1)$<br>Somma M e A con carry  | IMM<br>ABS<br>Z<br>(IND, X)<br>(IND), Y<br>ZX<br>ABS, X<br>ABS, Y   | 69<br>6D<br>65<br>61<br>71<br>75<br>7D<br>79                | 2<br>4<br>3<br>6<br>5<br>4<br>4<br>4 | 2<br>3<br>2<br>2<br>2<br>2<br>3<br>3 | NV ..... ZC         |
| AND<br>«AND» memory with<br>accumulator<br>$A \wedge M \rightarrow A(1)$<br>Funzione AND tra M e A       | IMM<br>ABS<br>Z<br>(IND, X)<br>(IND), Y<br>Z, X<br>ABS, X<br>ABS, Y | 29<br>2D<br>25<br>21<br>31<br>35<br>3D<br>39                | 2<br>4<br>3<br>6<br>5<br>4<br>4<br>4 | 2<br>3<br>2<br>2<br>2<br>2<br>3<br>3 | N ..... Z...        |
| ASL<br>Shift left one bit<br>Spostare un bit a sinistra<br>$C \leftarrow \boxed{Z} \leftarrow \emptyset$ | ABS<br>Z<br>A<br>Z, X<br>ABS, X                                     | $\emptyset E$<br>$\emptyset 6$<br>$\emptyset A$<br>16<br>1E | 6<br>5<br>2<br>6<br>7                | 3<br>2<br>1<br>2<br>3                | N ..... ZC          |
| BCC<br>Branch on carry clear (2)<br>Salto se C=0   | REL   | $9\emptyset$  | 2                                    | 2                                    |                     |
| BCS<br>Branch on carry set (2)<br>Salto se C=1   | REL   | B $\emptyset$   | 2                                    | 2                                    |                     |

| <i>Abbreviazione<br/>macromarca+<br/>spiegazione</i>                               | <i>Modo di<br/>indirizzamento (3)</i> | <i>Codice<br/>esadecimale</i> | <i>Numero di<br/>input<br/>di clock</i> | <i>Numero<br/>di bytes</i> | <i>Flag<br/>influenzati</i> |
|--|---------------------------------------|-------------------------------|---|----------------------------|-----------------------------|
| BEQ<br>Branch on result zero (2)<br>Salto se Z=1                                   | REL                                   | FØ                            | 2                                       | 2                          |                             |
| BIT<br>Test bits in memory<br>Esigine AM<br>$M_7 \rightarrow N; M_6 \rightarrow V$ | ABS<br>Z                              | 2C<br>24                      | 4<br>3                                  | 3<br>2                     | $M_7 M_6 \dots Z$           |
| BMI<br>Branch on result minus (2)<br>Salto se N=1                                  | REL                                   | 3Ø                            | 2                                       | 2                          |                             |
| BNE<br>Branch on result not zero (2)<br>Salto se Z=0                               | REL                                   | DØ                            | 2                                       | 2                          |                             |
| BPL<br>Branch on result plus (2)<br>Salto se N=0                                   | REL                                   | 1Ø                            | 2                                       | 2                          |                             |
| BRK<br>Force break<br>Interruzione obbligatoria                                    | IMP                                   | ØØ                            | 7                                       | 1                          | $\dots 1 1 \dots$<br>B 1    |
| BVC<br>Branch on overflow clear (2)<br>Salto se V=0                                | REL                                   | 5Ø                            | 2                                       | 2                          |                             |

| <i>Abbreviazione<br/>macroistruzioni+<br/>spiegazione</i>         | <i>Modo di<br/>indirizzamento (5)</i>                                     | <i>Codice<br/>esadecimale</i>                 | <i>Numero di<br/>impulsi<br/>di clock</i> | <i>Numero<br/>di bytes</i>      | <i>Flag<br/>influenzati</i> |
|---|---|---|---|---------------------------------|-----------------------------|
| BVS<br>Branch on overflow set (2)<br>Salvo se V=1                 | REL   | 70  | 2   | 2                               |                             |
| CLC<br>Clear carry flag<br>Disattivare C; 0-C                     | IMP   | 18  | 2   | 1                               | 0.....<br>C                 |
| CLD<br>Clear decimal mode Desactivar<br>Disattivare D; 0-D        | IMP   | D8  | 2   | 1                               | 0.....<br>D ,               |
| CLI<br>Clear interrupt flag<br>Disattivare I; 0-I                 | IMP   | 58  | 2   | 1                               | 0.....<br>I                 |
| CLV<br>Clear overflow flag<br>Disattivare V; 0-V                  | IMP   | B8  | 2   | 1                               | 0.....<br>V                 |
| CMP<br>Compare memory and<br>accumulator A-M<br>Confrontare M e A | IMM<br>ABS<br>Z<br>(IND, X)<br>(IND), Y<br>Z, X<br>ABS, X<br>DD<br>ABS, Y | C9<br>CD,<br>C5<br>C1<br>D1<br>D5<br>DD<br>D9 | 2<br>4<br>3<br>6<br>5<br>4<br>4<br>4      | 2<br>3<br>2<br>2<br>2<br>3<br>3 | N.....ZC                    |

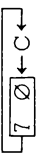

| <i>Abbreviazioni<br/>mnemoniche +<br/>spiegazione</i>                                       | <i>Modo di<br/>indirizzamento (5)</i>                               | <i>Codice<br/>esadecimale</i>                | <i>Numero di<br/>impulsi<br/>di clock</i> | <i>Numero<br/>di bytes</i>           | <i>Flag<br/>influenzati</i> |
|---|---|--|---|--------------------------------------|-----------------------------|
| CPX<br>Compare memory and index X.<br>Controllare M e Y                                     | IMM<br>ABS<br>Z   | E0<br>EC<br>E4                               | 2<br>4<br>3                               | 2<br>3<br>2                          | N.....ZC                    |
| CPY<br>Compare memory and index Y<br>Controllare M e Y; M: Y                                | IMM<br>ABS<br>Z   | C0<br>CC<br>C4                               | 2<br>4<br>3                               | 2<br>3<br>2                          | N.....ZC                    |
| DEC<br>Decrement memory<br>Togliere 1 da M<br>M-1 → M                                       | ABS<br>Z<br>Z, X<br>ABS, X  | CE<br>C6<br>D6<br>DE                         | 6<br>5<br>6<br>7                          | 3<br>2<br>2<br>3                     | N.....Z...                  |
| DEX<br>Decrement index X by one<br>Togliere 1 da X  | IMP   | CA   | 2   | 1                                    | N.....Z...                  |
| DEY<br>Decrement index<br>Y by one Y-1 → Y<br>Togliere 1 da Y                               | IMP   | 88   | 2   | 1                                    | N.....Z...                  |
| EOR<br>«Exclusive or» memory with<br>accumulator<br>A V M → A (1)<br>Funzione XOR tra M e A | IMM<br>ABS<br>Z<br>(IND, X)<br>(IND), Y<br>Z, X<br>ABS, X<br>ABS, Y | 49<br>4D<br>45<br>41<br>51<br>55<br>5D<br>59 | 2<br>4<br>3<br>6<br>5<br>4<br>4<br>4      | 2<br>3<br>2<br>2<br>2<br>2<br>3<br>3 | N.....Z...                  |



| <i>Abbreviazione<br/>mnemonica+<br/>spiegazione</i>  | <i>Modo di<br/>indirizzamento (S)</i> | <i>Codice<br/>esadecimale</i> | <i>Numero di<br/>impulsi<br/>di clock</i> | <i>Numero<br/>di bytes</i> | <i>Flag<br/>influenzati</i> |
|--|---------------------------------------|-------------------------------|---|----------------------------|-----------------------------|
| INC<br>Increment memory by one<br>$X+1 \rightarrow M$<br><b>Sommare 1 a M</b>  | ABS<br>Z<br>Z, X<br>ABS, X            | EE<br>E6<br>F6<br>FE          | 6<br>5<br>6<br>7                          | 3<br>2<br>2<br>3           | N.....Z...                  |
| INX<br>Increment index X by one<br>$X+1 \rightarrow X$<br><b>Sommare 1 a X</b>   | IMP                                   | E8                            | 2   | 1                          | N.....Z...                  |
| INY<br>Increment index<br>Y by one $Y+1 \rightarrow Y$<br><b>Sommare 1 a Y</b>   | IMP                                   | C8                            | 2   | 1                          | N.....Z...                  |
| JMP<br>Jump to new loc.<br><b>Salto</b><br>(PC+1) $\rightarrow$ PCL<br>(PC+2) $\rightarrow$ PCH  | ABS<br>IND                            | 4C<br>6C                      | 3<br>5                                    | 3<br>3                     |                             |
| JSR<br>Jump to new location saving<br>return address<br><b>Salto con ritorno</b><br>(PC+1) $\rightarrow$ PCL<br>(PC+2) $\rightarrow$ PCH | ABS                                   | 20                            | 6   | 3                          |                             |

| <i>Abbreviazione<br/>mнемоніca+<br/>spiegazione</i>                     | <i>Modo di<br/>indirizzamento (S)</i>                               | <i>Codice<br/>esadecimale</i>                | <i>Numero di<br/>impulsi<br/>di clock</i> | <i>Numero<br/>di bytes</i>           | <i>Flag<br/>influenzati</i> |
|---|---|--|---|--------------------------------------|-----------------------------|
| LDA<br>Load accumulator with memory<br>M → A (1)<br>Caricare A con M    | IMM<br>ABS<br>Z<br>(IND, X)<br>(IND), Y<br>Z, X<br>ABS, X<br>ABS, Y | A9<br>AD<br>A5<br>A1<br>B1<br>B5<br>BD<br>B9 | 2<br>4<br>4<br>3<br>6<br>5<br>4<br>4<br>4 | 2<br>3<br>2<br>2<br>2<br>2<br>3<br>3 | N.....Z...                  |
| LDX<br>Load index X with memory<br>M → X (1)<br>Caricare X con M        | IMM<br>ABS<br>Z<br>Z, Y<br>ABS, Y                                   | A2<br>AE<br>A6<br>B6<br>BE                   | 2<br>4<br>4<br>3<br>4<br>4<br>4           | 2<br>3<br>2<br>2<br>3                | N.....Z...                  |
| LDY<br>Load Index Y with memory<br>M → Y (1)<br>Caricare Y con M        | IMM<br>ABS<br>Z<br>Z, X<br>ABS, X                                   | A0<br>AC<br>A4<br>B4<br>BC                   | 2<br>4<br>4<br>3<br>4<br>4<br>4           | 2<br>3<br>2<br>2<br>3                | N.....Z...                  |
| LSR<br>Shift right one bit<br>Spostare un bit a sinistra<br>0 → 7 0 → C | ABS<br>Z<br>A<br>Z, X<br>ABS, X                                     | 4E<br>46<br>4A<br>56<br>5E                   | 6<br>5<br>2<br>6<br>7                     | 3<br>2<br>1<br>2<br>3                | 0.....ZC<br>N               |
| NOP<br>No operation   | IMP   | EA   | 2   | 1                                    |                             |

| <i>Abbreviazione<br/>mnemonica +<br/>spiegazione</i>                         | <i>Modo di<br/>indirizzamento (S)</i>                                     | <i>Codice<br/>esadecimale</i>                | <i>Numero di<br/>impulsi<br/>di clock</i> | <i>Numero<br/>di bytes</i>      | <i>Flag<br/>influenzati</i> |
|--|---|--|---|---------------------------------|-----------------------------|
| ORA<br>«OR» memory with accumulator<br>AVM → A<br>Funzione OR tra M e A      | IMM<br>ABS<br>Z<br>(IND, X)<br>(IND), Y<br>Z, X<br>ABS, X<br>ID<br>ABS, Y | Ø9<br>ØD<br>Ø5<br>Ø1<br>11<br>15<br>1D<br>19 | 2<br>4<br>3<br>6<br>5<br>4<br>4<br>4      | 2<br>3<br>2<br>2<br>2<br>3<br>3 | N ..... Z...                |
| PHA<br>Push accumulator on stack AI<br>Inviare A allo stack                  | IMP   | 48   | 3   | 1                               |                             |
| PHP<br>Push processor status on stack;<br>PI<br>Inviare P allo stack         | IMP   | Ø8   | 3   | 1                               |                             |
| PLA<br>Pull accumulator from stack AI<br>Ripercorrere A dallo stack          | IMP   | 68   | 4   | 1                               | N ..... Z...                |
| PLP<br>Pull processor status from<br>stack: PI<br>Ripercorrere P dallo stack | IMP   | 28   | 4   | 1                               | (reset)                     |

| <i>Abbreviazioni<br/>marmoscello+<br/>spiegazione</i>  | <i>Modo di<br/>indirizzamento (S)</i>                               | <i>Codice<br/>assemblatore</i>               | <i>Numero di<br/>impulsi<br/>di clock</i> | <i>Numero<br/>di bytes</i>           | <i>Flag<br/>influenzati</i> |
|--|---|--|---|--------------------------------------|-----------------------------|
| ROL<br>Rotate one bit left<br>Ruotare un bit a sinistra<br> | ABS<br>Z<br>Z, X<br>ABS, X<br>A                                     | 2E<br>26<br>36<br>3E<br>2A                   | 6<br>5<br>6<br>7<br>2                     | 3<br>2<br>2<br>3<br>1                | N.....ZC                    |
| ROR<br>Rotate one bit right<br>Ruotare un bit a destra<br>  | ABS<br>Z<br>A<br>Z, X<br>ABX, X                                     | 6E<br>66<br>6A<br>76<br>7E                   | 6<br>5<br>2<br>6<br>7                     | 3<br>2<br>1<br>2<br>3                | N.....ZC                    |
| RTI<br>Return from interrupt PCi, PI<br>Ritorno da interrupt rup.  | IMP   | 40   | 6   | 1                                    | (reset)                     |
| RTS<br>Return from subroutine PCi;<br>PC+1-PC<br>Ritorno da subroutine   | IMP   | 60   | 6   | 1                                    |                             |
| SBC<br>Subtract memory from<br>accumulator with borrow (3)<br>A-M- $\bar{C}$ $\rightarrow$ A<br>Sottrarre M e carry negato da A                | IMM<br>ABS<br>Z<br>(IND, X)<br>(IND), Y<br>Z, X<br>ABS, X<br>ABS, Y | E9<br>ED<br>E5<br>E1<br>F1<br>F5<br>FD<br>F9 | 2<br>4<br>3<br>6<br>5<br>4<br>4<br>4      | 2<br>3<br>2<br>2<br>2<br>2<br>3<br>3 | N.....ZC                    |

| <i>Abbreviazione<br/>memorizza+<br/>spiegazione</i>                           | <i>Modo di<br/>indirizzamento (S)</i>                        | <i>Codice<br/>esadecimale</i>          | <i>Numero di<br/>impulsi<br/>di clock</i> | <i>Numero<br/>di bytes</i>      | <i>Flag<br/>influenzati</i> |
|---|--|--|---|---------------------------------|-----------------------------|
| SEC<br>Set carry flag $1 \rightarrow C$<br>Scrivere C                         | IMP  | 38                                     | 2   | 1                               | .....1                      |
| SED<br>Scrivere D var D   | IMP  | F8                                     | 2   | 1                               | .....1.....<br>D            |
| SEI<br>Set interrupt; $1 \rightarrow I$<br>Scrivere I                         | IMP  | 78                                     | 2   | 1                               | .....1.....<br>I            |
| STA<br>Store accumulator in memory<br>$A \rightarrow M$<br>Memorizzare A in M | ABS<br>Z<br>(IND, X)<br>(IND), Y<br>Z, X<br>ABS, X<br>ABS, Y | 8D<br>85<br>81<br>91<br>95<br>9D<br>99 | 4<br>3<br>6<br>6<br>4<br>5<br>5           | 3<br>2<br>2<br>2<br>2<br>3<br>3 |                             |
| STX<br>Store index X in M<br>Memorizzare X in M                               | ABS<br>Z<br>Z, Y   | 8E<br>86<br>96                         | 4<br>3<br>4                               | 3<br>2<br>2                     |                             |
| STY<br>Store index Y in M<br>Memorizzare Y in M                               | ABS<br>Z<br>Z, X   | 8C<br>84<br>94                         | 4<br>3<br>4                               | 3<br>2<br>2                     |                             |

| <i>Abbreviazione<br/>mnemonica+<br/>spiegazione</i>                        | <i>Modo di<br/>indirizzamento (S)</i> | <i>Codice<br/>esadecimale</i> | <i>Numero di<br/>impulsi<br/>di clock</i> | <i>Numero<br/>di bytes</i> | <i>Flag<br/>influenzati</i> |
|--|---------------------------------------|-------------------------------|---|----------------------------|-----------------------------|
| TAX<br>(A → X)<br>Transfer accumulator to index X<br>Trasferire A a X      | IMP                                   | AA                            | 2   | 1                          | N.....Z...                  |
| TAY<br>(A → Y)<br>Transfer accumulator to index Y<br>Trasferire A a Y      | IMP                                   | A8                            | 2   | 1                          | N.....Z...                  |
| TSX<br>(S → X)<br>Transfer stack pointer to index X<br>Trasferire S a X    | IMP                                   | BA                            | 2   | 1                          | N.....Z...                  |
| TXA<br>(X → A)<br>Transfer index X to accumulator<br>Trasferire X a A      | IMP                                   | 8A                            | 2   | 1                          | N.....Z...                  |
| TXS<br>(X → S)<br>Transfer index X to stack<br>pointer<br>Trasferire X a S | IMP                                   | 9A                            | 2   | 1                          | N.....Z...                  |
| TYA<br>(Y → A)<br>Transfer index Y to accumulator<br>Trasferire Y a A      | IMP                                   | 98                            | 2   | 1                          | N.....Z...                  |

*Tabella IV.-Elenco delle istruzioni del  $\mu P$  6502.*

- di copia (PUSH, PULL): PHA, PHP, ecc. trasferisce il dato tra il registro indicato e lo stack;
- di ritorno (RETURN): RTI, RTS ritorna da subroutine;
- di non operare: NOP questa istruzione non esegue alcunchè, e agisce da riempitivo.

### *Note relative alla tabella delle istruzioni*

(1) Sommare 1 a N se si supera il limite di pagina. (2) Sommare 1 a N se il salto si esegue alla stessa pagina, sommare 2 a N se il salto si esegue ad altra pagina. (3) Senza carry. (4) A=accumulatore, M=memoria, C=flag di carry, Z=flag di segno, V=flag di overflow, N=flag di segno negativo, D=flag di numero decimale, I=flag di interrupt, X=indice X, Y=indice Y.

(5)

|          |  |
|----------|--|
| IMM:     | indirizzamento immediato;                                      |
| ABS:     | indirizzamento assoluto;                                       |
| Z:       | indirizzamento in pagina zero;                                 |
| A:       | indirizzamento dell'accumulatore;                              |
| IMP:     | indirizzamento implicito;                                      |
| (IND, X) | indirizzamento preindicizzato;                                 |
| (IND, Y) | indirizzamento preindicizzato;                                 |
| Z, X     | indirizzamento indicizzato in pagina zero (utilizza registro); |
| Z, Y     | indirizzamento indicizzato in pagina zero (utilizza registro); |
| ABS, X   | indirizzamento assoluto indicizzato (utilizza registro);       |
| ABS, Y   | indirizzamento assoluto indicizzato (utilizza registro);       |
| REL      | indirizzamento relativo;                                       |
| IND      | indirizzamento diretto.  |

### *Modi di indirizzamento del 6502*

Le varie istruzioni di questo µP dispongono dei seguenti modi di indirizzamento:

- Indirizzamento dell'accumulatore:  
Sono istruzioni a un solo byte che si riferiscono ad una operazione con il registro accumulatore (A).
- Indirizzamento immediato:  
Sono istruzioni a due bytes, di cui il primo è l'istruzione e il secondo l'operando; questo indirizzamento non necessita di

indirizzo di memoria.

- Indirizzamento assoluto:

Le istruzioni che utilizzano questo indirizzamento sono a tre bytes: il primo è l'istruzione, il secondo e terzo costituiscono l'indirizzo di memoria in cui si vuole operare. Con questo tipo di indirizzamento è possibile coprire i 64 Kbytes di cui dispone il  $\mu P$  6502.

- Indirizzamento in pagina zero:

Utilizzano questo tipo di indirizzamento le istruzioni che operano solo in pagina zero.

- Indirizzamento indicizzato in pagina zero:

Le istruzioni che utilizzano questo indirizzamento, lo fanno lavorando in pagina zero con l'indicizzazione del registro X e Y.

- Indirizzamento indicizzato assoluto:

Questo tipo di indirizzamento è come quello assoluto, ma indicizzato con i registri X e Y.

L'indirizzo effettivo è il risultato della somma del contenuto del registro X o Y, e dell'indirizzo assoluto indicato dal secondo e terzo byte.

- Indirizzamento implicito:

Nel modo implicito, l'indirizzo dell'operazione è contenuto nell'operando.

- Indirizzamento relativo:

Questo tipo di indirizzamento è utilizzato solo con le istruzioni di salto. Sono istruzioni a due bytes, in cui il secondo indica se il salto è in avanti o all'indietro, ed il numero di locazioni da saltare.

I limiti del salto sono di 127 posizioni in avanti e 128 all'indietro rispetto alla posizione in cui si trova l'istruzione che si sta eseguendo.

- Indirizzamento indicizzato indiretto:

L'indirizzo finale dell'operazione si trova nella locazione di memoria indicata mediante l'indirizzamento indicizzato; il registro indice può essere X o Y.

- Indirizzamento assoluto indiretto:

L'indirizzo effettivo si trova mediante un indirizzamento assoluto.



**L**'espansione dei campi di applicazione e delle possibilità di impiego dei microprocessori a 8 bit, è stata così ampia che, a partire dal 1980, questi microprocessori erano utilizzati al limite delle loro possibilità. Inoltre erano prevedibili applicazioni che superavano tali possibilità.

Per soddisfare queste necessità di mercato nacquero i microprocessori a 16 bit. Quelli a 32 bit furono realizzati subito dopo.

Quanto accadde ai microprocessori a 4 bit con l'avvento di quelli a 8 bit, sta ora accadendo a questi ultimi con l'avvento di quelli a 16.

D'altra parte, chi è in grado di sviluppare sistemi con i microprocessori a 8 bit, non avrà molti problemi con quelli a 16 bit, vista la loro somiglianza. Analogamente accadrà con quelli a 32 bit.

L'obiettivo di questo capitolo è di descrivere, in modo semplice, la filosofia dei microprocessori a 16 bit a un lettore che conosca quelli a 8 bit.

Perciò vengono forniti solo i concetti basilari, relativi all'hardware e al software, per permettere di *iniziare a lavorare* con i nuovi componenti. Infatti solo l'effettiva sperimentazione aiuta, quando si inizia a trattare in modo approfondito gli argomenti chiave.

Tra i microprocessori a 16 bit, sono stati scelti solo quelli per cui si possono prevedere più applicazioni:

- 1. Il classico 8086, che è il più venduto nella sua categoria, e che è supportato da una eccellente famiglia di circuiti ausiliari. L'inserimento dell'8088, versione dell'8086 con il bus dei dati a 8 bit, nel personal computer IBM, fa prevedere un grande sviluppo dell'hardware e del software di tale famiglia.
- 2. Il 68000, che possiede un'architettura protesa verso i 32 bit, e

un impianto software orientato ai sistemi operativi moderni ed ai linguaggi strutturali ad alto livello.

## *Generazione dei microprocessori a 16 bit*

La scoperta della tecnologia HMOS, da parte dell'INTEL CORPORATION, ha propiziato l'aumento della densità di integrazione, e il passaggio alla tecnologia VLSI, con cui sono stati ampiamente superati i 100.000 transistori in un chip, fa sperare di raggiungere tra breve il milione.

Storicamente INTEL fu la prima costruttrice di un microprocessore a 16 bit, all'inizio del 1978. Dopo l'8086 dell'INTEL, apparvero lo Z8000 della ZILOG, il 68000 della MOTOROLA, l'NS 1600 della NATIONAL, il 99000 della TEXAS, il  $\mu$ COM 70K, successore del 6502 della NEC, e altri ancora.

In questo momento l'applicazione più importante dei microprocessori a 16 bit è quella nei microelaboratori ad uso generale.

La nascita dei microprocessori a 16 e 32 bit è stata necessaria per adeguarli ai requisiti delle nuove applicazioni, ed al miglioramento e semplificazione del software.

L'hardware di questi microprocessori, seguendo in molti casi la linea degli 8 bit, si orienta verso la simmetria e la possibilità di ampliamenti e variazioni, come dimostra l'inserimento in molti modelli dell'Unità di Controllo in Multiprogrammazione.

Parallelamente, l'uso delle nuove tecnologie, come l'HMOS, nella fabbricazione dei circuiti integrati, permette di aumentare la densità di integrazione e migliorare il rapporto velocità/consumo.

Tuttavia, il miglioramento più importante dei moderni microprocessori è relativo al software. Il suo costo ed i problemi che presenta all'utente hanno spinto i progettisti ad una architettura speciale, che semplifichi e minimizzi tutti questi fattori.

Così, nella maggioranza dei microprocessori di progettazione recente, esistono caratteristiche sconosciute in quelli a 8 bit, come:

- 1. repertorio delle istruzioni orientato verso l'implementazione di linguaggi ad alto livello;
- 2. elevata varietà dei modi di indirizzamento;
- 3. grande diversità nei formati dei dati;
- 4. possibilità di lavorare in memoria virtuale, mediante la quale il microprocessore è in grado di operare in un campo di memoria più ampio di quello che esiste effettivamente nel sistema;
- 5. impiego del modo Supervisor, per la protezione dei programmi,

| MODELLO | FABBRICANTE | FORMATO DEI DATI | MULTIPLEXER DEI DATI | REGISTRI  | FREQUENZA DI CLOCK      | CAPACITA' MEMORIA | ISTR. PIU' LUNGA | CODA DI ATTESA DELLE ISTRUZIONI | SECONDA FONTE   |
|---------|-------------|------------------|----------------------|---|-------------------------|-------------------|------------------|---------------------------------|---|
| 8086    | INTEL       | 16/16            | SI                   | 14 A 16 BITS<br>DI USO NON GENERALE                                   | 4, 5 y 8 MHz            | 7 M BYTES         | 20 $\mu$ s       | 6 BYTES                         | AMD, FUJITSU<br>HARRIS, MATRA<br>MITSUBISHI,<br>NEC E SIEMENS |
| 8001    | ZILOG       | 16/16            | SI                   | 7 + 4 A<br>16 BITS<br>SPECIALIZZATI                                   | 4, 6 Y 10 MHz           | 8 M BYTES         | 140 $\mu$ s      | NO                              | AMD,<br>SOS E<br>SHARP  |
| 68000   | MOTOROLA    | 32/16            | NO                   | 18 A 32 BITS<br>+ 1 A 16 BITS<br>SPECIALIZZATI                        | 4, 6, 8, 10<br>y 12 MHz | 16 M BYTES        | 20 $\mu$ s       | NO                              | HITACHI,<br>MOSTEK,<br>ROCKWELL,<br>SIGNETICS E<br>THOMPSON   |
| 99000   | TEXAS       | 16/16            | NO                   | 8 A 16 BITS<br>SPECIALI   | 4 MHz                   | 64 K BYTES        | 31 $\mu$ s       | NO                              | AMI, ITT  |
| 16032   | NATIONAL    | 32/16            | SI                   | 6 A 24 BITS +<br>2 A 16 BITS<br>SPECIALI<br>E 8 A 32 BITS<br>GENERALI | 10 MHz                  | 16 M BYTES        | 8 $\mu$ s        | 8 BYTES                         | EUROTECHNIQUE<br>E<br>FAIRCHILD                               |

*Tabella I.-Riepilogo delle caratteristiche più importanti di alcuni microprocessori a 16 bits.*

e adeguamento ai moderni sistemi operativi;

- 6. sistema completo di interrupts;
- 7. nuove istruzioni, inesistenti in quelli a 8 bit, come moltiplicazione, divisione, trasferimento blocchi di dati, ecc.

Come già accadde con i 4 bit, i microprocessori a 8 bit verranno soppiantati dai 16 e 32 bit, il che non significa che saranno completamente accantonati.

D'altra parte il piccolo aumento di prezzo dei microprocessori a 16 e 32 bit, rispetto alle loro straordinarie possibilità, contribuirà a renderli presto di uso comune.

## *Il microprocessore 68000*

L'aspetto più rilevante del 68000 sta nel fatto che può essere considerato come un effettivo microprocessore a 32 bit, anche se viene compreso tra quelli a 16. La versione siglata 68020 è disponibile in commercio, come CPU a 32 bit, dal 1983.

Nel 68000 sono riunite una serie di caratteristiche che lo evidenziano tra i microprocessori, come l'assoluta flessibilità per la realizzazione di sistemi multiprocessore o multiarea, unita ad altre che gli permettono di adattarsi a prodotti standard, tipici dei microprocessori a 8 bit, memorie e moduli di ingresso e uscita, con formati e velocità di lavoro variabili.

Questa possibilità di utilizzare nella configurazione di sistemi basati sul 68000, elementi molto noti presenti in commercio, riduce significativamente i costi del progetto.

Il 68000 è simile ad un microprocessore a 32 bit suddivisi in due parole da 16 bit. Ciò è conseguenza del fatto che, non multiplexando i bus dei dati e degli indirizzi, sarebbe necessario un contenitore a 96 piedini per utilizzare tutti i segnali inerenti al formato a 32 bit.

Il costo e le restrizioni tecnologiche ne motivarono la produzione in contenitori a 64 terminali, ma il passaggio ai 32 bit è semplice ed è già stato effettuato.

Il 68000 ha migliorato molti aspetti del suo predecessore 6800:

- 1. è stato ampliato moltissimo il numero delle istruzioni, comprese quelle proprie dei 16 bit, come moltiplicazione e divisione. Altre, come PEA e LEA, consentono la programmazione strutturata;
- 2. il numero dei modi di indirizzamento è stato portato a 14, tra cui alcuni di indubbia operatività;

- 3. si è cercato di semplificare l'adattamento dei programmi del 6800 al 68000;
- 4. si è permessa l'elaborazione di dati con formato variabile da 1 a 32 bit;
- 5. sono stati incorporati segnali propri del 6800, come VPA, VMA, ed E, che lo rendono direttamente compatibile con i moduli di ingresso e uscita della famiglia del microprocessore a 8 bit.

I costruttori principali sono quelli sottoelencati:

- 1. MOTOROLA (MC6800);
- 2. ROCKWELL (R68000);
- 3. HITACHI AMERICA LTD. (HD68000);
- 4. SIGNETICS/PHILIPS (SP68000)
- 5. MOSTECK CORP. (MK68000);
- 6. EFCIS (THOMSON-CFS) (EF68000).

## *Architettura del 68000*

Due grandi blocchi di registri a 32 bit, uno per i dati e l'altro per gli indirizzi, costituiscono la base su cui poggia l'hardware e, di conseguenza, il software di questo microprocessore.

Esistono inoltre un'Unità Aritmetico-Logica, un registro di stato, e il registro delle istruzioni, assieme alla logica di controllo e temporizzazione, come si vede in Fig. 1.

La ALU è in grado di effettuare operazioni logiche e aritmetiche con dati a 32 bit, comprese la moltiplicazione e la divisione, con e senza segno.

I rimanenti componenti del 68000 intervengono in modo molto diverso nelle istruzioni, per cui queste devono essere ben conosciute dal progettista.

I registri che intervengono, in modo preponderante, nella programmazione del 68000, così come il complesso di segnali che lo mettono in comunicazione con l'esterno, sono graficamente rappresentati in Fig. 2.

## *Registro dei dati*

E' un banco di 8 registri a 32 bit ciascuno, destinati a contenere dati e operandi. Per dare un'idea della sua flessibilità, si può dire che equivale ad 8 accumulatori, di quelli visti nei microprocessori a 8 bit.

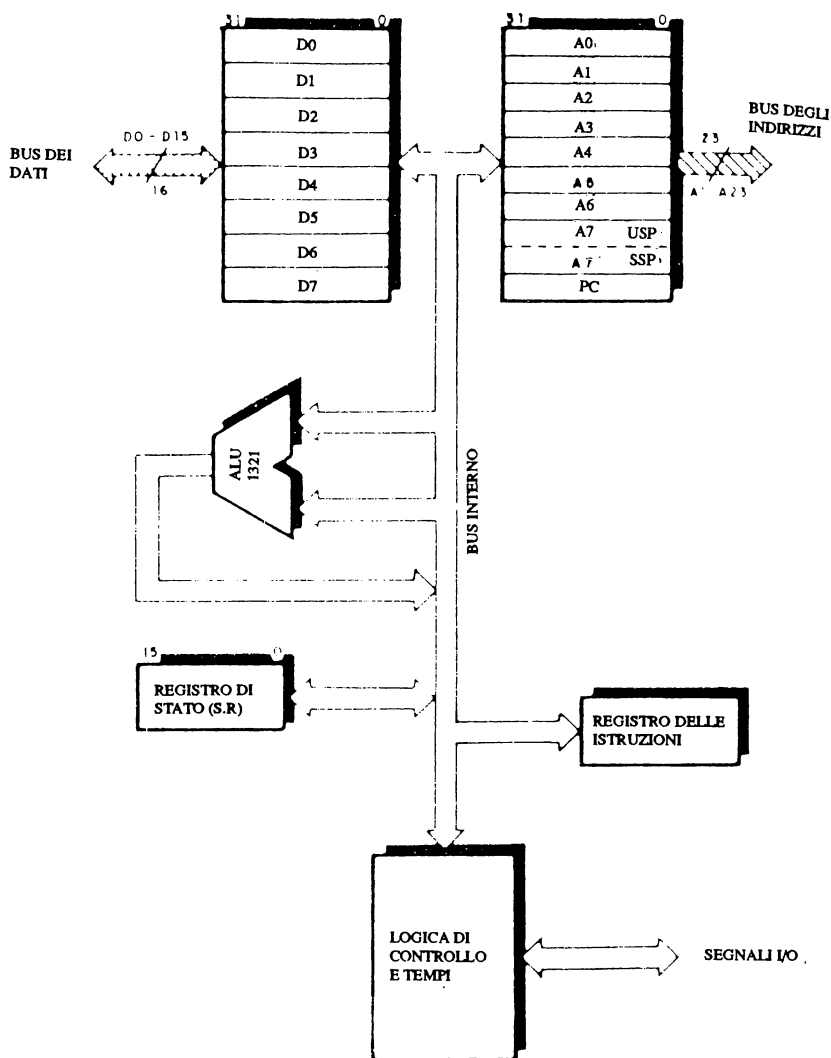


Fig. 1.-Architettura del 68000. La sua semplicità è unita ad una enorme flessibilità dei due banchi di registri, il che spiega la sua potenza operativa.

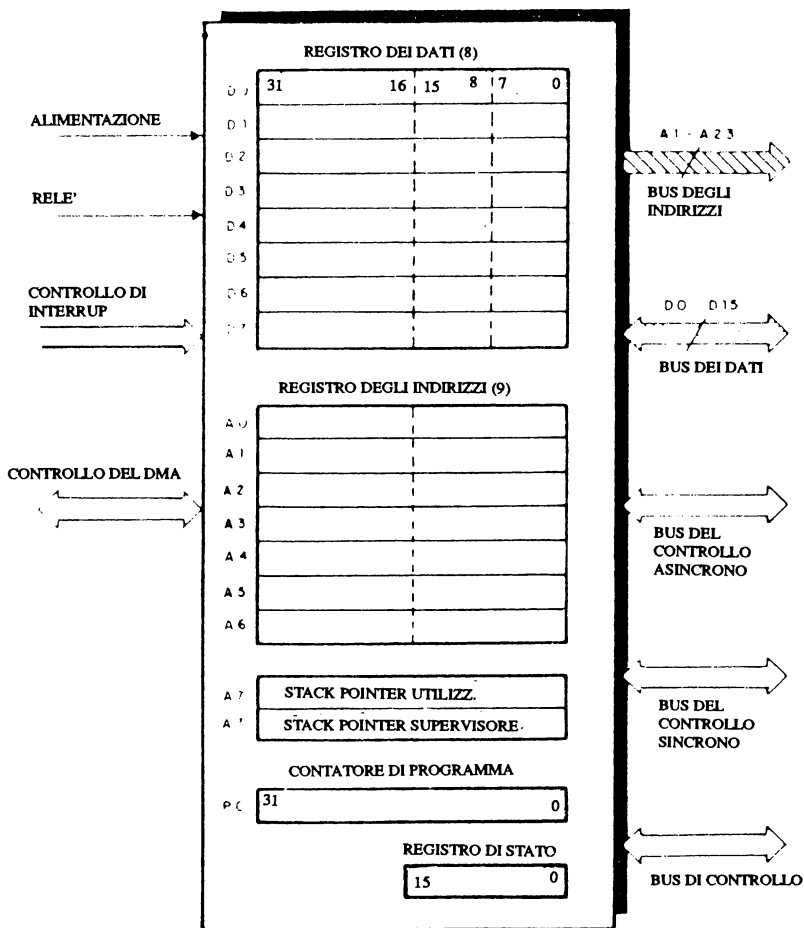


Fig. 2.-Organizzazione interna del 68000, dal punto di vista della programmazione dei banchi di registri.

L'informazione elaborata nei registri dei dati può essere a 8, 16, 32 bit.

Per determinare le dimensioni dell'informazione utilizzata in un registro di dati, si pone nel codice mnemonico della corrispondente istruzione, un punto seguito da B, W, o L, secondo che si tratti di 8, 16, o 32 bit, rispettivamente.

Le operazioni con i registri di dati influenzano solo i bit utilizzati nell'i-

struzione, gli altri non vengono variati, anche in istruzioni di spostamento.

## *Registri degli indirizzi*

E' costituito da 7 registri a 32 bit, che hanno il compito di memorizzare gli indirizzi di memoria. Pertanto il loro contenuto deve essere correlato con il modo di indirizzamento delle diverse lunghezze dei dati.

Si osservi che gli indirizzi di memoria che si riferiscono a parole lunghe, si succedono di 4 in 4. Non rispettare queste norme provoca un *indirizzo illegale*.

Le dimensioni che possono essere utilizzate da registri degli indirizzi, quando agiscono su un operando sorgente dell'istruzione, sono di 16 e 32 bit.

Tuttavia, quando gli operandi sono di destinazione, tutti i 32 bit del registro sono influenzati, indipendentemente dalle dimensioni dell'operando.

Cioè se si elabora un operando a 16 bit, gli altri 16 del registro degli indirizzi destinazione sono influenzati, con l'estensione a tali bit del segno (il bit più significativo dei 16 bit del risultato viene ripetuto nei 16 bit non utilizzati).

Un altro importante dettaglio relativo a tali registri, consiste nel fatto che le operazioni che si effettuano con essi non influiscono sui flags di condizione (CCR) del registro di stato.

## *I registri puntatori di stack*

I registri degli indirizzi A7 e A7' sono due puntatori di stack, che è una zona di memoria in cui si salva lo stato della CPU quando esiste un salto a subroutine o interviene un interrupt. Il registro A7 corrisponde al puntatore di stack dell'ambiente utente, l'A7' a quello del supervisore. In ogni istante solo uno dei due è in funzione.

Normalmente il puntatore di stack dell'ambiente utente funziona nelle chiamate a subroutine di tale ambiente. L'A7' funziona durante gli interrupts.

I due puntatori di stack indicano l'ultimo dato contenuto. Si caricano verso l'alto e si scaricano verso il basso.

## *Contatore di programma*

E' un registro a 32 bit, che viene utilizzato per contenere l'indirizzo della parola dell'istruzione successiva a cui si deve accedere.



Sono utilizzati solo 24 bit, anche se il bus degli indirizzi viene caricato con soli 23, poichè non accoglie il bit meno significativo A0.

Con 23 bit di possono indirizzare  $2^{23} = 8.388.608$  parole a 16 bit.

Il bit meno significato del PC si utilizza per determinare all'interno dell'indirizzo, della parola cui si accede, se si riferisce al byte pari (LDS) o dispari (UDS).

Tenendo conto della funzione di questo bit, si possono indirizzare 16 Mbytes.

Se A0=0 l'indirizzo è pari e può essere riferito a un byte pari o a una parola. In questo caso, si attiva il segnale UDS, che esce dalla CPU, abilitando gli 8 bit più significativi del bus dei dati (D8-D15).

Se A0=1, l'indirizzo è dispari e si riferisce a un byte dispari. Si attiva il segnale LDS, che abilita gli 8 bit meno significativi del bus dei dati (D0-D7).

## *Registro di stato*

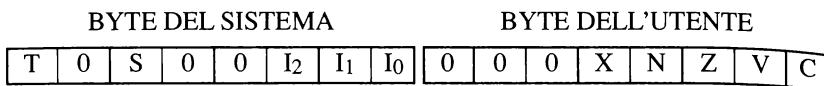
E' costituito da 16 bit, suddivisi in due bytes, uno destinato al sistema e l'altro all'utente.

La sua struttura è quella di Fig. 3.

I cinque bit rappresentativi nel byte dell'utente (gli altri tre sono sempre 0) sono i flags o codici di condizione dell'operazione effettuata nell'ultima istruzione eseguita dalla CPU.

Il significato di ognuno di essi è il seguente:

- C: carry, viene settato a 1 quando esiste riporto nei bit più significativi di una somma, o prestito per una differenza. Interviene in altre istruzioni, come quelle di spostamento, rotazione, ecc.;
- V: overflow, ha significato solo nelle operazioni in cui intervengono numeri con segno. Diventa 1 quando si sommano due numeri con lo stesso segno o se ne sottraggono due di segno opposto ed il risultato supera il campo dell'operando in complemento a 2;
- Z: zero, questo bit viene settato a 1 quando il risultato dell'operazione vale zero;
- N: segno, ha significato solo quando si lavora con numeri con segno. Diventa 1 quando, dopo un'operazione, il risultato è negativo, cioè il bit più significativo è 1.
- X: carry esteso, è un bit di carry (C), per operazioni in multipla precisione. Serve per concatenare operazioni in cui intervengono



*Fig. 3.-Configurazione e denominazione dei 16 bits del registro di stato (SR).*

dati di lunghezza maggiore. E' trasparente al flusso dei dati e, nelle istruzioni in cui agisce, ha lo stesso valore del registro C.

Anche il byte del registro di stato relativo al sistema possiede 5 bit significativi, con i seguenti significati:

- T: modo trace, si può settare a 1 solo tramite lo stato supervisore, il che può accadere dopo ogni istruzione, per eseguire un programma passo a passo, e permettere all'utente di conoscere lo stato del sistema. Per ogni eccezione esiste un byte che rappresenta un vettore il quale, moltiplicato per 4, fornisce l'indirizzo che contiene la locazione di inizio del programma di gestione dell'eccezione. Per implementare il trace, istruzione per istruzione, si accede a un vettore che serve una routine in grado di esaminare i registri della CPU, le locazioni di memoria, ecc;
- S: Supervisore, indica quando il 68000 funziona in modo supervisore (S=1). In questo modo, tutte le istruzioni sono eseguibili, e si può modificare la maschera degli interrupts (I<sub>0</sub>-I<sub>2</sub>). Si può anche accedere al registro di stato e ai puntatori di stack;

| LIVELLO | I <sub>0</sub> | I <sub>1</sub> | I <sub>2</sub> |   |
|---------|----------------|----------------|----------------|---|
| 0       | 0              | 0              | 0              | SENZA PRIORITA'                         |
| 1       | 1              | 0              | 0              | MINIMA PRIORITA'                        |
| 2       | 0              | 1              | 0              |   |
| 3       | 1              | 1              | 0              |   |
| 4       | 0              | 0              | 1              |   |
| 5       | 1              | 0              | 1              |   |
| 6       | 0              | 1              | 1              |   |
| 7       | 1              | 1              | 1              | MASSIMA PRIORITA'<br>(NON MASCHERABILE) |

*Tabella II.-Tabella dei livelli di priorità che riflette la maschera degli interrupts, costituita dai bits I<sub>0</sub>-I<sub>2</sub>. Tutte le richieste di interrupt con un livello uguale o inferiore a quello stabilito con la maschera, non sono accettati dalla CPU.*

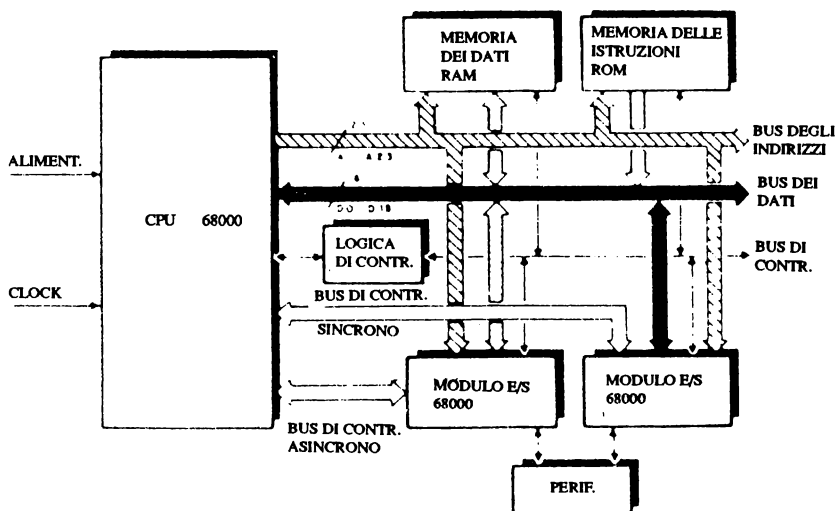
- (I<sub>0</sub>-I<sub>2</sub>): maschera degli interrupts, la loro combinazione logica riflette il livello di priorità a partire dal quale si accettano gli interrupts. Nella tabella II sono visibili i vari livelli di priorità.

### *Configurazione di un sistema basato sul 68000*

I 64 terminali del 68000 implementano tutte le funzioni necessarie per sviluppare un potente complesso di bus di dati, indirizzi, e controllo, che permettono di costruire un ottimo sistema microelaboratore, il cui schema di principio è visibile in Fig. 4.

A tale sistema si possono accoppiare tutti i tipi di memoria e moduli di ingresso e uscita standard, come quelli della famiglia del 6800.

L'esistenza di due bus di controllo, uno sincrono e l'altro asincrono, amplia notevolmente la possibilità di inserire una gran varietà di periferiche nel sistema.



*Fig. 4.-Configurazione fondamentale di un sistema microelaboratore, basato sul 68000.*

## *Schema generale di collegamento del 68000*

Il chip del 68000 è inserito in un contenitore di tipo DIL a 64 piedini.

Oltre al bus dei dati a 16 linee (D0-D15) e al bus degli indirizzi a 23 (A1-A23), esistono altri segnali che si possono raggruppare nel modo seguente:

- 1. bus di controllo sincrono per periferiche della famiglia 6800;
- 2. bus di controllo asincrono per periferiche del 68000;
- 3. controllo di DMA o di gestione del bus;
- 4. controllo della funzione o stato della CPU;
- 5. controllo del sistema.

La Fig. 5 indica la numerazione dei terminali assegnati ad ogni linea. I segnali rappresentati in modo sopralineato sono attivi quando il loro livello logico è basso. Per una miglior comprensione si descrive il compito delle linee del microprocessore raggruppate per funzioni:

- alimentazione: esistono due piedini, il 14 e il 49, mediante i quali si fornisce la tensione di alimentazione positiva il cui valore deve essere di 5 V, con una tolleranza del 5%. I terminali 16 e 53 si collegano al polo negativo o massa (GND).
- clock: l'ingresso corrispondente al piedino 15 del circuito integrato, fornisce un segnale di frequenza stabile, TTL compatibile, che serve per ottenere internamente gli impulsi del clock di sincronismo.
- bus dei dati: è costituito da 16 linee bidirezionali, denominate D0-D15, che sono distribuite, strategicamente, sui terminali della zona superiore del contenitore, allo scopo di semplificare il disegno delle piste che convogliano l'informazione sul circuito stampato. La limitazione del numero delle linee del bus dei dati implica che per eseguire il trasferimento con un numero di bit superiore alla parola, è necessario più di un ciclo.
- bus degli indirizzi e segnali UDS e LDS: è costituito da 23 linee unidirezionali, che escono dalla CPU, mediante i terminali detti A1-A23. I segnali UDS e LDS selezionano il byte pari o dispari di una parola, per cui, considerando anch'essi, si possono indirizzare fino a 16 Mbytes.
- bus di controllo asincrono: la funzione delle linee principali che costituiscono il bus di controllo asincrono del 68000, per il controllo di periferiche caratteristiche della famiglia 68000, è la seguente:

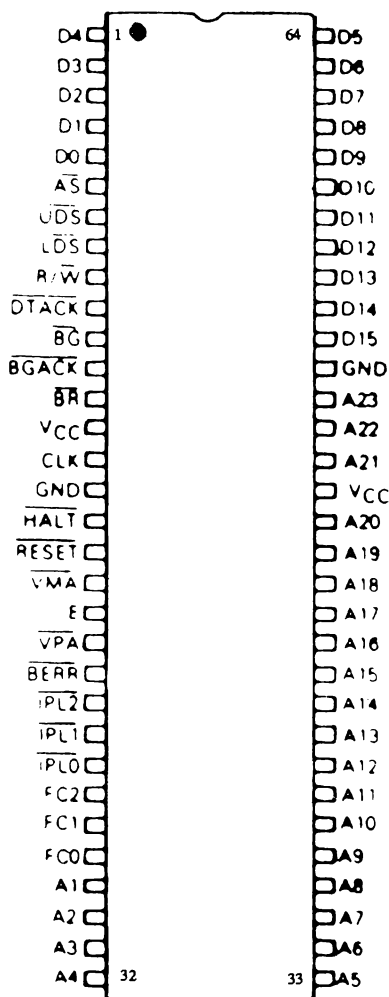


Fig. 5.-Distribuzione dei terminali del microprocessore 68000.

- $R/\overline{W}$  (Read/Write): mediante tale segnale il 68000 definisce se un ciclo in esecuzione è di lettura o scrittura;
- $\overline{AS}$  (Address Strobe): quando esiste un livello basso, che è quello attivo, su questa linea, la CPU comunica all'esterno la presenza di un indirizzo valido sul bus degli indirizzi, per le periferiche asincrone;

- $\overline{\text{DTACK}}$  (Data Transfer Acknowledge): questo segnale di ingresso, informa il 68000 della fine di un trasferimento di dati. Quando la CPU riconosce DTACK, durante un ciclo di lettura, raccoglie l'informazione dal bus dei dati e termina il ciclo. Il riconoscimento, in scrittura, determina la fine del ciclo.
- Bus di controllo sincrono:
  - E (Enable): su questa uscita, la CPU genera un segnale sincrono, con una frequenza di clock fissa. Si tratta di un segnale fondamentale per le periferiche sincrone del 6800;
  - $\overline{\text{VPA}}$  (Valid Peripheral Address): mediante tale segnale di ingresso, si comunica alla CPU che l'elemento indirizzato è una periferica sincrona della famiglia 6800;
  - $\overline{\text{VMA}}$  (Valid Memory Address): segnale di uscita che indica alle periferiche del 6800 che esiste un indirizzo valido sul bus degli indirizzi e che la CPU è sincronizzata per la sua abilitazione.  $\overline{\text{VMA}}$  è un segnale di risposta a  $\overline{\text{VPA}}$ .
- Linee di controllo dello stato del microprocessore: sono tre segnali che escono dalla CPU che forniscono informazioni sulle modalità di lavoro (utente o supervisore), e se la zona a cui si accede corrisponde a programmi o dati. In realtà si controllano quattro mappe di memoria da 16 Mbytes ciascuna. L'informazione fornita da questi terminali, detti FC0, FC1, e FC2 è valida, sempre che il segnale  $\overline{\text{AS}}$  si trovi attivato.
- Linee di controllo degli interrupts: questo gruppo di terminali, costituiti da quelli detti IPL0, IPL1, e IPL2, hanno il compito di ricevere il livello di priorità del dispositivo che richiede un interrupt alla CPU.
- Linee di controllo DMA: le linee che comprendo questa funzione, servono per cedere il controllo dei bus del sistema a elementi in comune con altri microprocessori, o per l'accesso diretto alla memoria:
  - $\overline{\text{BR}}$  (Bus request): è un ingresso al 68000 che gli comunica la richiesta da parte di un dispositivo esterno del controllo dei bus del sistema;
  - $\overline{\text{BG}}$  (Bus Grant): è un'uscita con cui la CPU informa di avere ricevuto una richiesta di controllo del bus;
  - $\overline{\text{BGACK}}$  (Bus Grant Acknowledge): la periferica, mediante questo segnale, conferma alla CPU che ha preso il controllo

**TABELLA III**

| <b>SIMBOLO</b>  | <b>DENOMINAZIONE</b>              | <b>COMPITO</b>                                   | <b>STATO ATTIVO</b> | <b>INGRESSO USCITA (I o O)</b> | <b>TRHEE STATE</b> |
|---|-----------------------------------|--|---------------------|--------------------------------|--------------------|
| A1-A23  | BUS DEGLI INDIRIZZI               | INDIRIZZAMENTO                                   | ALTO                | O                              | SI                 |
| D0-D15  | BUS DEI DATI                      | TRASFERIMENTO INFORMAZIONI                       | ALTO                | I/O                            | SI                 |
| $\overline{AS}$   | INDIRIZZO ATTIVATO                | SEGNALA PRESENZA DI INDIRIZZO                    | BASSO               | O                              | SI                 |
| $R/\overline{W}$  | LETTURA / SCRITTURA               | INDICA CICLO LETTURA O SCRITTURA                 | ALTO / BASSO        | O                              | SI                 |
| $\overline{UDS}$ e $\overline{LDS}$                         | DATO PIU' O MENO SIGNIFICATIVO    | SEGNALA BYTE CUI ACCEDERE                        | BASSO               | O                              | SI                 |
| $\overline{DTACK}$  | RICONOSCIMENTO TRASFERIMENTO DATI | INDICA TRASFERIMENTO DI DATI                     | BASSO               | I                              | NO                 |
| $\overline{BR}$   | RICHIESTA DI BUS                  | RICHIESTE I BUS                                  | BASSO               | I                              | NO                 |
| $\overline{BG}$   | CONCESSIONE DI BUS                | CONCEDE I BUS                                    | BASSO               | O                              | NO                 |
| $\overline{BGACK}$  | RICONOSCIMENTO CONCESSIONE DI BUS | SEGNALA IL POSSESSO DEI BUS                      | BASSO               | I                              | NO                 |
| $\overline{IPL0}$<br>$\overline{IPL1}$<br>$\overline{IPL2}$ | LIVELLO PRIORITARIO DI INTERRUPT  | INDICANO IL LIVELLO DI INTERRUPT DEL RICHIEDENTE | BASSO               | I                              | NO                 |
| $\overline{BERR}$   | ERRORE DI BUS                     | INDICA UN PROBLEMA NEL CICLO                     | BASSO               | I                              | NO                 |
| $\overline{RESET}$  | RESET                             | RICHIESTA DI RESET                               | BASSO               | I/O                            | NO                 |
| $\overline{HALT}$   | ALT                               | RICHIESTA DI ALT                                 | BASSO               | I/O                            | NO                 |
| E   | ABILITAZIONE                      | SEGNALE DI CLOCK (6800)                          | ALTO                | O                              | NO                 |
| VMA   | INDIRIZZO DI MEMORIA VALIDO       | VALIDAZIONE DELL'INDIRIZZO (6800)                | BASSO               | O                              | SI                 |
| VPA   | INDIRIZZO DI PERIFERICA VALIDO    | VALIDAZIONE DELLA PERIFERICA (6800)              | BASSO               | I                              | NO                 |

| SIMBOLO               | DENOMINAZIONE             | COMPITO                         | STATO ATTIVO | INGRESSO USCITA (I o O) | TRHEE STATE |
|-----------------------|---------------------------|---------------------------------|--------------|-------------------------|-------------|
| FC0<br>FC1<br>FC2     | USCITA CODICE DI FUNZIONE | INDICANO MODO E CICLO           | ALTO         | O                       | SI          |
| CLK                   | CLOCK                     | SEGNALE DI CLOCK DI RIFERIMENTO | ALTO         | I                       | NO          |
| V <sub>CC</sub> e GND | +5V $\pm$ 5% e MASSA      | ALIMENTAZIONE                   | —            | I                       | —           |

*Tabella III.-Tabella riassuntiva delle caratteristiche più rappresentative dei 64 terminali del 68000.*

dei bus. A partire da tale istante, disattiva  $\overline{BR}$  e controlla i bus, mentre mantiene attivo il segnale BGACK. Quando viene disattivato  $\overline{BR}$ , il 68000 disattiva  $\overline{BG}$ .

- Linee di controllo del sistema:
  - $\overline{RESET}$ : è una linea bidirezionale, cioè sia la CPU, sia un dispositivo esterno, è in grado di attivarlo e originare una reinizializzazione del sistema;
  - $\overline{HALT}$ : quando funziona come ingresso ed è attivato dall'esterno, permette di implementare un programma passo a passo, per debuggarlo;
  - $\overline{BERR}$  (Bus Error): si tratta di un segnale di ingresso che informa il microprocessore del verificarsi di un problema nel sistema, come un accesso illegale alla memoria, o l'errore nella risposta di qualche periferica.

## *Repertorio delle istruzioni*

Il 68000 è stato progettato con un repertorio di sole 56 istruzioni, che sono state ampliate nei modelli più recenti come il 68010. Questo ridotto numero di istruzioni, combinato con l'elevata varietà dei modi di indirizzamento e di dati, offre al programmatore un potente strumento per sviluppare il software dei sistemi basati su tale microprocessore. Le varie istruzioni, suddivise per campo funzionale, si possono raggruppare come segue:

- a) istruzioni aritmetiche: ADD, ADDA, ADDI, ADDQ, ADDX,



MULS, DIVU, DIVS, EXT, CLR, CMP, CMPA, CMPI, CMPM, TST, e TAS;

- b) istruzioni logiche: AND, ANDI, OR, ORI, EOR, EORI, e NOT;
- c) istruzioni di scorrimento e rotazione: ASI, ASR, LSL, LSR, ROL, ROR, ROXL, e ROXR;
- d) istruzioni per la manipolazione di bit: BTST, BSET, BCLR, e BCHG;
- e) istruzioni di trasferimento: MOVE, MOVEM, MOVEP, MOVEQ, EXG, SWAP, LEA, e PEA;
- f) istruzioni per il controllo sequenziale del programma: JMP, BRA, NOP, Bcc, DBcc, e Scc;
- g) istruzioni per il controllo delle subroutines: JSR, BSR, RTS, LINK, e UNLK;
- h) istruzioni per il controllo del sistema: RESET, STOP, RTE, MOVE-SR, ANDI-SR, ORI-SR, EORI-SR, TRAP, TRAPV, e CHK.

### *Modi di indirizzamento*

Il 68000 dispone di 14 diversi modi per l'indirizzamento degli operandi, e risulta, sotto questo aspetto, uno dei più potenti della sua categoria. Tale varietà consente, ai programmatori abili, possibilità straordinarie per ottimizzare le loro realizzazioni software. I modi di indirizzamento si possono suddividere in 6 categorie:

- 1. indirizzamento diretto tramite registro;
- 2. indirizzamento indiretto tramite registro;
- 3. indirizzamento assoluto;
- 4. indirizzamento relativo al contatore di programma;
- 5. indirizzamento immediato;
- 6. indirizzamento implicito.

### *La famiglia del microprocessore 68000*

Come conseguenza dello sviluppo da parte di vari costruttori di periferiche per il microprocessore 68000, nel corso degli ultimi anni, si è avuto un

## TABELLA IV

@+: modo di indirizzamento indiretto con postincremento

•: influenzato

—: non influenzato

| MNEMONICO | OPERAZIONE   | CODICI DI CONDIZIONE |   |   |   |   |
|-----------|--|----------------------|---|---|---|---|
|           |  | X                    | N | Z | V | C |
| ABCD      | (DESTINAZIONE) <sub>10</sub> +(SORGENTE) <sub>10</sub> → DESTINAZIONE                                  | •                    | U | • | U | • |
| ADD       | (DESTINAZIONE)+(SORGENTE) → DESTINAZIONE   | •                    | • | • | • | • |
| ADDA      | (DESTINAZIONE)+(SORGENTE) → DESTINAZIONE   | —                    | — | — | — | — |
| ADDI      | (DESTINAZIONE)+DATO IMMEDIATO → DESTINAZIONE   | •                    | • | • | • | • |
| ADDQ      | (DESTINAZIONE)+DATO IMMEDIATO → DESTINAZIONE   | •                    | • | • | • | • |
| ADDX      | (DESTINAZIONE)+(SORGENTE)+X → DESTINAZIONE   | •                    | • | • | • | • |
| AND       | (DESTINAZIONE) ∧ (SORGENTE) → DESTINAZIONE   | —                    | • | • | 0 | 0 |
| ANDI      | (DESTINAZIONE) ∧ DATO IMMEDIATO → DESTINAZIONE   | —                    | • | • | 0 | 0 |
| ASL, ASR  | (DESTINAZIONE) SPIAZZATO DA <CONT> → DESTINAZIONE  | •                    | • | • | • | • |
| Bcc       | SE cc ALLORA PC+d → PC   | —                    | — | — | — | — |
| BCHG      | ≈(<NUMERO BIT>) DI DESTINAZIONE → Z<br>≈(<NUMERO BIT>) DI DESTINAZIONE<br><NUMERO BIT> DI DESTINAZIONE | —                    | — | • | — | — |
| BCLR      | ≈(<NUMERO BIT>) DI DESTINAZIONE → Z<br>0 → (<NUMERO BIT>) DI DESTINAZIONE                              | —                    | — | • | — | — |
| BRA       | PC+d → PC  | —                    | — | — | — | — |
| BSET      | ≈(<NUMERO BIT>) DI DESTINAZIONE → Z<br>1 → (<NUMERO BIT>) DI DESTINAZIONE                              | —                    | — | • | — | — |
| BSR       | PC → SP @-; PC+d → PC  | —                    | — | — | — | — |
| BTST      | ≈(<NUMERO BIT>) DI DESTINAZIONE → Z  | —                    | — | • | — | — |
| CHK       | SE Dn < 0 O Dn > <ea> ALLORA TRAP  | —                    | • | U | U | U |
| CLR       | 0 → DESTINAZIONE   | —                    | 0 | 1 | 0 | 0 |
| CMP       | (DESTINAZIONE)-(SORGENTE)  | —                    | • | • | • | • |

| MNEMONICO  | OPERAZIONE  | CODICI DI CONDIZIONE |   |   |   |   |
|------------|---|----------------------|---|---|---|---|
|            |   | X                    | N | Z | V | C |
| CMPA       | (DESTINAZIONE)-(SORGENTE)   | —                    | • | • | • | • |
| CMPI       | (DESTINAZIONE)-DATO IMMEDIATO   | —                    | • | • | • | • |
| CMPM       | (DESTINAZIONE)-(SORGENTE)   | —                    | • | • | • | • |
| DBCC       | SE $\approx_{CC}$ ALLORA $D_n-1 \rightarrow D_n$ ;<br>SE $D_n \neq -1$ ALLORA $PC+d \rightarrow PC$ | —                    | — | — | — | — |
| DIVS       | (DESTINAZIONE)/(SORGENTE) $\rightarrow$ DESTINAZIONE  | —                    | • | • | • | 0 |
| DIVU       | (DESTINAZIONE)/(SORGENTE) $\rightarrow$ DESTINAZIONE  | —                    | • | • | • | 0 |
| EOR        | (DESTINAZIONE) $\oplus$ (SORGENTE) $\rightarrow$ DESTINAZIONE                                       | —                    | • | • | 0 | 0 |
| EORI       | (DESTINAZIONE) $\oplus$ DATO IMMEDIATO $\rightarrow$ DESTINAZIONE                                   | —                    | • | • | 0 | 0 |
| EXG        | $R_x \leftrightarrow R_y$   | —                    | — | — | — | — |
| EXT        | (DESTINAZIONE) SEGNO ESTESO $\rightarrow$ DESTINAZIONE  | —                    | • | • | 0 | 0 |
| JMP        | DESTINAZIONE $\rightarrow$ PC   | —                    | — | — | — | — |
| JSR        | $PC \rightarrow SP @-;$ DESTINAZIONE $\rightarrow$ SP   | —                    | — | — | — | — |
| LEA        | DESTINAZIONE $\rightarrow A_n$  | —                    | — | — | — | — |
| LINK       | $A_n \rightarrow SP @-;$ $SP \rightarrow A_n$ ; $SP+d \rightarrow SP$                               | —                    | — | — | — | — |
| LSL, LSR   | (DESTINAZIONE) SPIAZZATA DA <CONT> $\rightarrow$ DESTINAZIONE                                       | •                    | • | • | 0 | • |
| MOVE       | (SORGENTE) $\rightarrow$ DESTINAZIONE   | —                    | • | • | 0 | 0 |
| MOVE A CCR | (SORGENTE) $\rightarrow$ CCR  | •                    | • | • | • | • |
| MOVE A SR  | (SORGENTE) $\rightarrow$ SR   | •                    | • | • | • | • |
| MOVE ?     | SR $\rightarrow$ DESTINAZIONE   | —                    | — | — | — | — |
| MOVE USP   | USP $\rightarrow A_n$ ; $A_n \rightarrow$ USP   | —                    | — | — | — | — |
| MOVEA      | (SORGENTE) $\rightarrow$ DESTINAZIONE   | —                    | — | — | — | — |
| MOVEM      | REGISTRI $\rightarrow$ DESTINAZIONE<br>(SORGENTE) $\rightarrow$ DESTINAZIONE                        | —                    | — | — | — | — |
| MOVEP      | (SORGENTE) $\rightarrow$ DESTINAZIONE   | —                    | — | — | — | — |

| MNEMONICO     | OPERAZIONE   | CODICI DI CONDIZIONE |   |   |   |   |
|---------------|--|----------------------|---|---|---|---|
|               |  | X                    | N | Z | V | C |
| MOVEQ         | DATO IMMEDIATO → DESTINAZIONE  | —                    | • | • | 0 | 0 |
| MULS          | (DESTINAZIONE)•(SORGENTE) → DESTINAZIONE                                 | —                    | • | • | 0 | 0 |
| MULU          | (DESTINAZIONE)•(SORGENTE) → DESTINAZIONE                                 | —                    | • | • | 0 | 0 |
| MBCD          | 0-(DESTINAZIONE) <sub>10</sub> -X → DESTINAZIONE                         | •                    | U | • | U | • |
| NEG           | 0-(DESTINAZIONE) → DESTINAZIONE  | •                    | • | • | • | • |
| NEGX          | 0-(DESTINAZIONE)-X → DESTINAZIONE  | •                    | • | • | • | • |
| NOP           | —  | —                    | — | — | — | — |
| NOT           | ≈(DESTINAZIONE) → DESTINAZIONE   | —                    | • | • | 0 | 0 |
| OR            | (DESTINAZIONE) E (SORGENTE) → DESTINAZIONE                               | —                    | • | • | 0 | 0 |
| ORI           | (DESTINAZIONE) E DATO IMMEDIATO → DESTINAZIONE                           | —                    | • | • | 0 | 0 |
| PEA           | (DESTINAZIONE) → SP@-  | —                    | — | — | — | — |
| RESET         | —  | —                    | — | — | — | — |
| ROL,<br>ROR   | (DESTINAZIONE) RUOTATO DA <CONT> → DESTINAZIONE                          | —                    | • | • | 0 | • |
| ROXL,<br>ROXR | (DESTINAZIONE) RUOTATO DA <CONT> → DESTINAZIONE                          | •                    | • | • | 0 | • |
| RTE           | SP@+ → SR; SP@+ → PC   | •                    | • | • | • | • |
| RTR           | SP@+ → CC; SP@+ → PC   | •                    | • | • | • | • |
| RTS           | SP@+ → PC  | —                    | — | — | — | — |
| SBCD          | (DESTINAZIONE) <sub>10</sub> -(SORGENTE) <sub>10</sub> -X → DESTINAZIONE | •                    | U | • | U | • |
| Scc           | UN BYTE A 00 O FF IN ACCORDO CON LA CONDIZIONE                           | —                    | — | — | — | — |
| STOP          | DATO IMMEDIATO → SR; STOP  | •                    | • | • | • | • |
| SUB           | (DESTINAZIONE)-(SORGENTE) → DESTINAZIONE                                 | •                    | • | • | • | • |
| SUBA          | (DESTINAZIONE)-(SORGENTE) → DESTINAZIONE                                 | —                    | — | — | — | — |
| SUBI          | (DESTINAZIONE)-DATO IMMEDIATO → DESTINAZIONE                             | •                    | • | • | • | • |

| MNEMONICO | OPERAZIONE   | CODICI DI CONDIZIONE |   |   |   |   |
|-----------|--|----------------------|---|---|---|---|
|           |  | X                    | N | Z | V | C |
| SUBQ      | (DESTINAZIONE)-DATO IMMEDIATO → DESTINAZIONE         | •                    | • | • | • | • |
| SUBX      | (DESTINAZIONE)-(SORGENTE)-X → DESTINAZIONE           | •                    | • | • | • | • |
| SWAP      | REGISTRO [31:16] ↔ REGISTRO [15:0]                   | —                    | • | • | 0 | 0 |
| TAS       | (DESTINAZIONE) TESTATA → CC; 1 → [7] DI DESTINAZIONE | —                    | • | • | 0 | 0 |
| TRAP      | PC → SSP @; SR → SSP @-; (VETTORE) → PC              | —                    | — | — | — | — |
| TRAPV     | SE V ALLORA TRAP                                     | —                    | — | — | — | — |
| TST       | (DESTINAZIONE) TESTATA → CC                          | —                    | • | • | 0 | 0 |
| UNLK      | An → SP; SP @+ → An                                  | —                    | — | — | — | — |

*Tabella IV.-Operazioni relative ad ogni istruzione, e funzionamento dei flag di condizione.*

veloce potenziamento della sua famiglia. D'altra parte, la sua flessibilità aumenta notevolmente, per la compatibilità degli elementi della famiglia del 6800 (PIA, ACIA, CONTROLLER CRT, ecc.). Attualmente la famiglia del 68000 è costituita dai seguenti circuiti integrati:

- 1. 68000 (CPU a 16 bit);
- 2. 68008 (CPU a 8/16 bit);
- 3. 68010 (CPU con memoria virtuale);
- 4. 68020 (CPU a 32 bit);
- 5. 68200 (microelaboratore monochip);
- 6. 68881 (coprocessore matematico a virgola mobile);
- 7. 68120 e 68121 IPC (controller intelligente di periferiche);
- 8. 68230 MCU, PI/T (interfaccia parallela e timer);
- 9. 68430 DMAI (interfaccia DMA);
- 10. 68440 DDMA (doppio accesso diretto alla memoria);
- 11. 68450 DMAC (controller DMA);
- 12. 68451 MMU (unità di utilizzazione della memoria);
- 13. 68452 BAM (modulo di arbitraggio del bus);

- 14. 68454 IMDC;
- 15. 68459 DPLL;
- 16. 68562 DUSCC;
- 17. 68564 SIO (I/O seriale);
- 18. 68590 LANCE (controller per rete ETHERNET);
- 19. 68652 MPCC (controller di comunicazione con multiprotocollo);
- 20. 68561 MPCC II;
- 21. 68653 PGC;
- 22. 68661 EPCI;
- 23. 68681 DUART (UART a doppio canale);
- 24. 68901 MFP (periferica multifunzione).

Anche se qualcuno di questi componenti non è ancora in produzione, lo sarà in breve tempo.

## *Il microprocessore 8086*

I microprocessori a otto bit sono attualmente i più utilizzati, sia in campo professionale, sia in quello informatico e dell'automazione. I microprocessori a 16 bit sono stati finora utilizzati esclusivamente in campo professionale, per il fatto che, tra l'altro, la realizzazione di una CPU con questi microprocessori è costosa e complessa, principalmente perchè il microprocessore è costituito da vari chips. Con la comparsa dell'8086 nel 1978 si iniziò a prendere in considerazione i microprocessori a 16 bit, poichè tale  $\mu P$  presentava notevoli vantaggi sui predecessori a 16 bit.

Le caratteristiche principali dell'8086 sono:

- il suo bus dei dati è a 16 bit;
- la sua capacità di indirizzamento raggiunge 1 Mbyte (1.048.576 bytes);
- è fabbricato in tecnologia HMOS (MOS ad alta velocità), il che permette di operare ad una frequenza minima di 5 MHz, e consente tempi di esecuzione delle istruzioni più brevi di 500 nsec circa;
- è concepito per lavorare con linguaggi ad alto livello (Basic, Cobol, Fortran, o Pascal), e per l'utilizzo in microelaboratori;
- possiede un vasto repertorio di istruzioni che gli permettono di

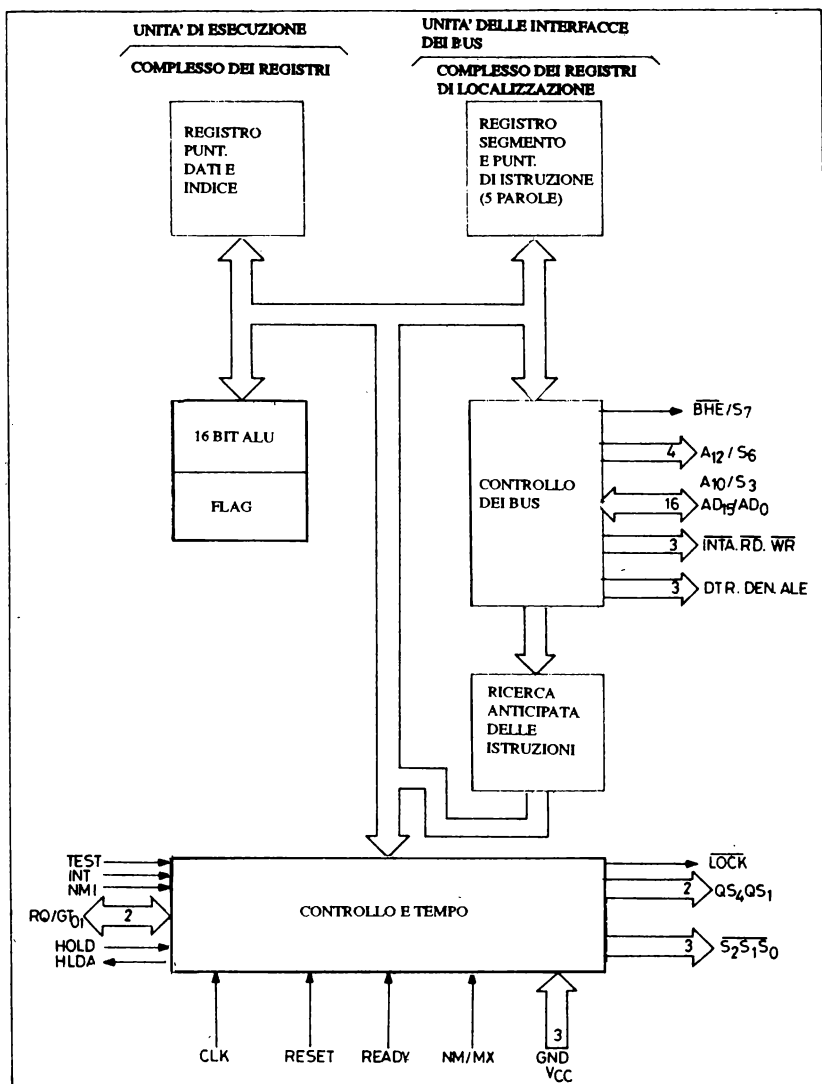


Fig. 6.-Blocchi interni del microprocessore 8086.

- operare sia ad 8 che a 16 bit;
- necessita di alimentazione unica a +5 V.

### *Descrizione dei terminali dell'8086*

L'8086 è inserito in un contenitore con il classico formato a 40 piedini, le cui funzioni sono:

- AD0-AD15: sono i terminali del bus degli indirizzi e dei dati, poichè i segnali sui due bus escono in tempi diversi. Esistono quattro tempi di funzionamento T1, T2, T3, e T4; durante T1 su tali terminali è presente l'indirizzo, da T2 a T4 i dati.
- A16-A19: sono quattro terminali su cui esce la parte alta dell'indirizzo per poter accedere al megabyte di memoria; ciò avviene durante T1. Durante T2, T3, e T4 questi terminali sono utilizzati per indicare l'uso dei registri interni (da S3 a S6).
- $\overline{\text{BHE}}$ : nel tempo T1 tale terminale, in combinazione con l'uscita AD0 esegue le seguenti funzioni: BHE=0, AD0=0 (bus dei dati a 16 bit), BHE=1, AD0=0 (il byte inferiore è riferito agli indirizzi pari), BHE=0, AD0=1 (il byte superiore del bus dei dati è riferito agli indirizzi dispari), BHE=1, AD0=1 non ha significato.
- RD: uscita per indicare all'esterno il ciclo di lettura (READ).
- READY: ingresso per adattare il  $\mu\text{P}$  alle memorie e periferiche lente (ciò è molto utile nell'8086, a causa della sua elevata velocità).
- INTR: ingresso per interrupt mascherabile.
- NMI: ingresso per interrupt non mascherabile.
- TEST: ingresso di prova associato all'istruzione con lo stesso nome (Test).
- RESET: ingresso di inizializzazione del ciclo di lavoro del  $\mu\text{P}$ .
- CLX: ingresso del clock esterno (circa 5 MHz).
- VCC e GND: terminali di alimentazione a +5 V e massa, rispettivamente.
- MN/MX: questo terminale fa variare alcuni segnali, in funzione del fatto che il sistema operi con un solo microprocessore (MN/MX=1) o con più microprocessori (MN/MX=0). Per MN/MX=1 i terminali dal 32 al 24 hanno i compiti ordinatamente sottoelencati (vedere Figg. 10 e 11).



- HOLD: ingresso per richieste di accesso al bus provenienti da qualche periferica.
- HOLDA: uscita di riconoscimento della richiesta di accesso, da parte dell'8086, e concessione della stessa.
- WR: uscita indicatrice del ciclo di scrittura (WRITE).
- $M/\overline{IO}$ : tramite questa uscita settata a uno, il  $\mu P$  attiva la memoria, per eseguire qualche istruzione ad essa riferita, o, se è a zero, attiva i circuiti di INPUT/OUTPUT (I/O) per l'esecuzione di qualche istruzione riferita a tali circuiti, a cui si possono collegare periferiche o blocchi di dati fino a 64 Kbytes, che rappresentano

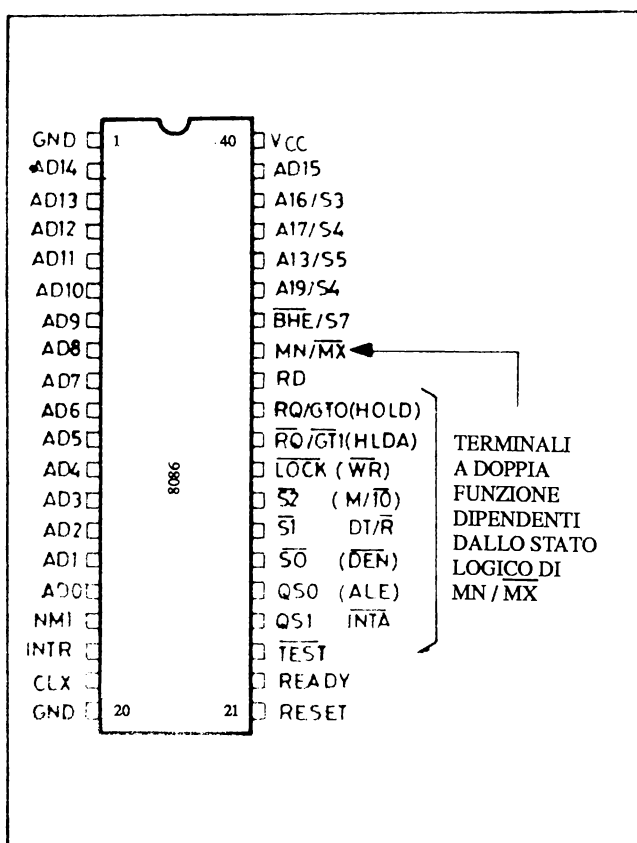


Fig. 7.-Descrizione dei terminali del microprocessore 8086.

l'estensione dell'I/O.

- DT/R: è un'uscita attraverso la quale il  $\mu P$  indica il ciclo di trasmissione o ricezione di dati da circuiti specializzati, che gli si aggiungono in un sistema monoprocessoire, ad esempio un 8287.
- DEN: uscita per abilitare i dati in un sistema monoprocessoire che utilizza il trasmettitore 8287.
- ALE: questa uscita abilita il latch degli indirizzi.
- INTA: tramite tale uscita il microprocessore indica ad una periferica che ha richiesto un interrupt, che a partire da tale istante lo stesso viene accettato.

Quando il terminale  $MN/\overline{MX}=0$ , i terminali numerati dal 32 al 34 hanno i compiti elencati nell'ordine seguente (si ricorda, per una migliore comprensione, di osservare le Figg. 10 e 11).

- $\overline{RQ}/\overline{GT0}$ : tramite questo ingresso, un altro  $\mu P$  associato può richiedere l'accesso al bus comune, dopo che il  $\mu P$  precedente ha terminato l'istruzione in corso.
- $\overline{RQ}/\overline{GT1}$ : per mezzo di tale uscita il microprocessore richiede l'accesso al bus condiviso con vari microprocessori in un sistema multiprocessore.
- LOCK: è un'uscita tramite cui il  $\mu P$  indica agli altri presenti nel sistema, che non possono accedere al bus comune mentre questo si trova in stato zero.
- S0-S2: sono tre terminali che decodificano alcuni degli otto possibili stati sottoelencati:
  - S2=0, S1=0, S0=0 (riconoscimento di interrupt);
  - S2=0, S1=0, S0=1 (lettura di circuiti I/O);
  - S2=0, S1=1, S0=0 (scrittura di circuiti I/O);
  - S2=0, S1=1, S0=1 (pari, halt);
  - S2=1, S1=0, S0=0 (codice di accesso a istruzione) (Fetch);
  - S2=1, S1=0, S0=1 (lettura memoria);
  - S2=1, S1=1, S0=0 (scrittura memoria);
  - S2=1, S1=1, S0=1 (non opera).
- QS0 e QS1: sono uscite che indicano lo stato del microprocessore nel trattamento delle istruzioni:
  - QS1=0, QS0=0 (non opera);
  - QS1=0, QS0=1 (primo byte del codice operativo dalla coda);

|   |   |  |  |  |   |
|---|---|--|--|--|---|
| 8086  | Funzioni<br>generales   | INTR<br>CLK<br>Vcc<br>GND  | Interrupt Request<br>Interrupt Request<br>System Clock<br>+5V<br>Ground                              | DEN<br>ALE<br>INTA   | Data Enable<br>Address Latch<br>Enable<br>Interrupt Acknow-<br>ledge  |
| AD15-ADO<br>—<br>A 19/S6<br>A 16/S3<br>BHE/S7                               | Address/Data Bus<br>Address Bus<br>Address/Status<br><br>Bus High Enable/<br>Status   | <b>Funzioni per un<br/>sistema semplice<br/>(Terminale<br/>MN/MX =1)</b>           |  | <b>Funzioni per un<br/>sistema complesso<br/>(Terminale<br/>MN/MX =0)</b>        |   |
| MN/ $\overline{MX}$<br><br>$\overline{RD}$<br>TEST<br>READY<br>RESET<br>NMI | Minimum/Maximum<br>Mode Control<br>Read Control<br>Wait on Test Control<br>Wait State Control<br>System Reset<br>Non-Maskable | HOLD<br>HLDA<br>$\overline{WR}$<br>M/ $\overline{IO}$<br>DT/ $\overline{R}$<br>DEN | Hold Request<br>Hold Acknowledge<br>Write Control<br>Memory I/O Control<br>Data Transmit/<br>Receive | $\overline{RQ}/\overline{GT}10$<br><br>$\overline{LOK}$<br>S2-S0<br><br>QS1, QS0 | Request/Grant Bus<br>Access Control<br>Bus Prior Lock Cont<br>Bus Cycle Status<br>Instruction Queue<br>Status |

*Tabella V.-Terminali del  $\mu P$  8086, con il significato dei nomi in inglese. Sono raggruppati in tre blocchi.*

- QS1=1, QS0=0 (coda vuota);
- QS1=1, QS0=1 (byte successivo dalla coda. Non era il primo).

## *Registri dell'8086*

L'8086 possiede un repertorio di registri, rappresentato in Fig. 8.

Il tipo di istruzioni di questo  $\mu P$  permette di operare con 8 o 16 bit, in base all'istruzione, per cui il primo blocco di registri costituisce degli accumulatori a 8 bit, detti AH, AL, BH, BL, CH, CL, DH, e DL (A, B, C, D indica l'ordine del registro, H la parte alta e L la parte bassa). Quando questi accumulatori sono trattati come 16 bit vengono definiti AX, BX, CX, DX.

Il blocco di registri successivo contiene il puntatore di stack e altri tre puntatori detti base, sorgente e destinazione, per il movimento dei dati interno, o rispetto alla memoria.

Il terzo blocco contiene il puntatore dell'istruzione che funziona da contatore di programma, e il registro dei flags, che è effettivamente il registro di stato, tipico di tutti i  $\mu P$ . Questi bit indicano il segno del dato elaborato, il carry, l'overflow, la parità, lo stato degli interrupts mascherabili mediante il bit 1 (num. 9), l'autoincremento o decremento di un registro, un bit ausiliario detto trap.

L'ultimo blocco comprende quattro registri a 16 bit, che definiscono in modo indipendente quattro aree di memoria da 64 Kbytes ciascuna.

L'8086 è, come detto, un microprocessore completo e complesso, per cui

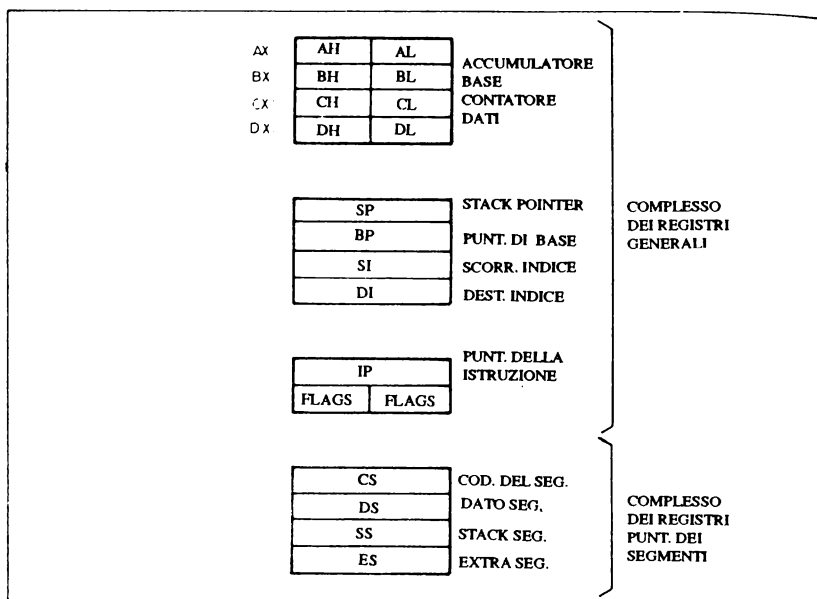


Fig. 8.-Mappa dei registri del microprocessore 8086.

si fornisce solo una panoramica dello stesso, per ampliare la comprensione dei microprocessori. Si rimanda perciò, per chi fosse interessato a questo microprocessore, alle informazioni fornite dai costruttori.

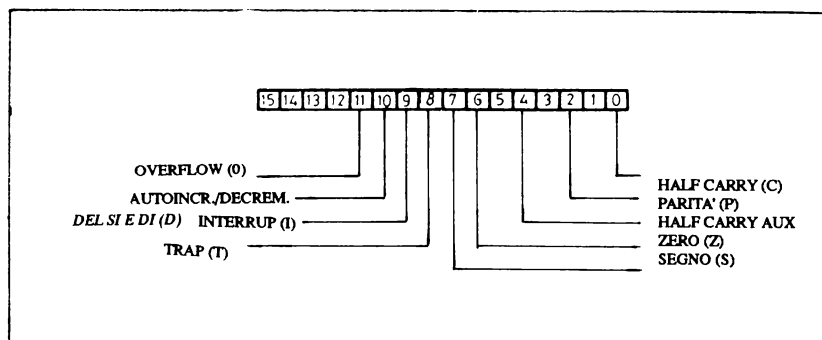


Fig. 9.-Registro di stato del  $\mu P$  8086.

## Sistema mono e multiprocessore nell' 8086

Il  $\mu P$  8086 è concepito per essere utilizzato in due modi di configurazione: secondo il sistema mono e multiprocessore.

Nel sistema monoprocessore il  $\mu P$  lavora indipendentemente con la sua circuiteria tipica; è il sistema di lavoro di qualsiasi altro microprocessore. La sua configurazione schematica è rappresentata in Fig. 10.

Tutti i microprocessori della serie 80 hanno molto in comune, ad esempio il circuito 8284 è il tipo di clock utilizzato per gli altri.

Come già detto nella descrizione, il suo bus a 16 bit è comune ai dati e agli indirizzi, in tempi diversi, per cui è necessario un latch che memorizzi l'indirizzo che esce dal bus nel primo ciclo, per poi operare sui dati che escono sul bus, diretti a tale indirizzo.

In realtà, tale sistema è seguito anche da altri microprocessori a 16 bit inseriti in contenitori a 40 terminali. Il circuito integrato 8286 è fondamentale un amplificatore bidirezionale del bus dei dati, che opera sul flusso in ingresso e in uscita.

Il sistema di collegamento multiprocessore, è concepito per operare in sistemi molto più complessi e per l'utilizzo con altri microprocessori 8086, in modo interlacciato.

In un sistema multiprocessore, in cui operano indipendentemente vari microprocessori che condividono a volte il bus, esiste sempre uno di essi il cui software è strutturato in modo che funzioni come principale, mentre gli altri gli sono subordinati. In tale sistema, inoltre, è possibile aggiungere ai circuiti

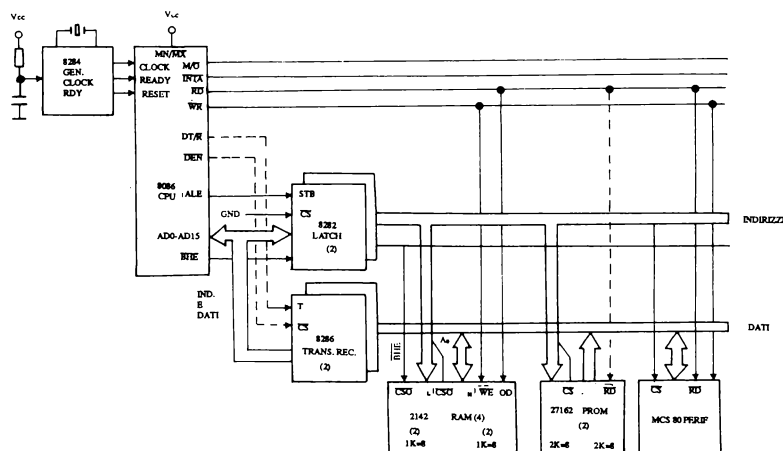


Fig. 10.-Schema a blocchi dei collegamenti dei circuiti di un 8086, secondo il sistema a monoprocessore.

generatori di clock, ai latch e amplificatori del bus, altri circuiti come l'8288 controllore dei bus e l'8285 controllore degli interrupts, che nel  $\mu P$  8086 sono simili a quelli del predecessore 8085.

Negli schemi delle varie figure sono rappresentati un sistema semplice ed uno più complesso, ed i vari blocchi con tutti i circuiti integrati LSI, da inserire opzionalmente in sistemi basati sul  $\mu P$  8086, per cui sono stati progettati.

La tendenza di questo microprocessore è quella valida attualmente per tutti i microprocessori a 16 bit, che apre nuove prospettive per il futuro.

Questa tendenza è orientata a fornire una sempre maggior potenza delle istruzioni, per ridurre al massimo il costo del software, e creare gli LSI necessari che, agendo in combinazione con il microprocessore, facilitino il suo collegamento e le sue possibilità di impiego.

## Tabella delle istruzioni dell' 8086

- Istruzioni di trasferimento:

MOV: trasferisce una parola (16 bit) o un byte (8 bit);

XCHG: scambia una parola o un byte;

LEA: carica un indirizzo in un registro;

LDS: carica un puntatore tramite DS;

LES: carica un puntatore mediante ES;

LAHF: carica i flags;

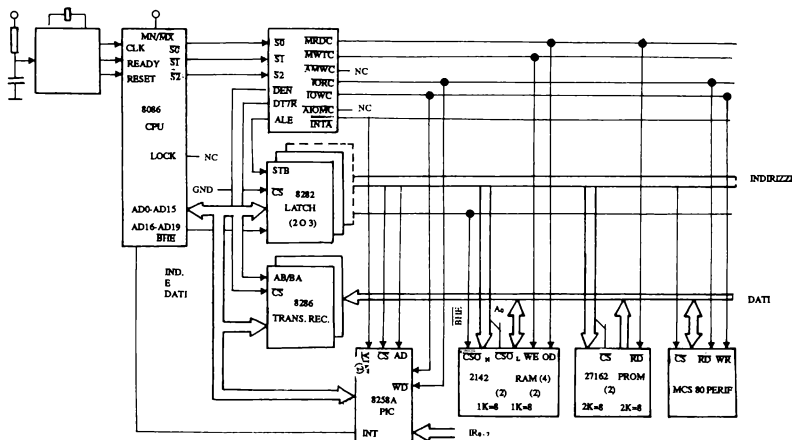


Fig. 11.-Schema a blocchi dei collegamenti dei circuiti di un 8086, in un sistema a multiprocessore.

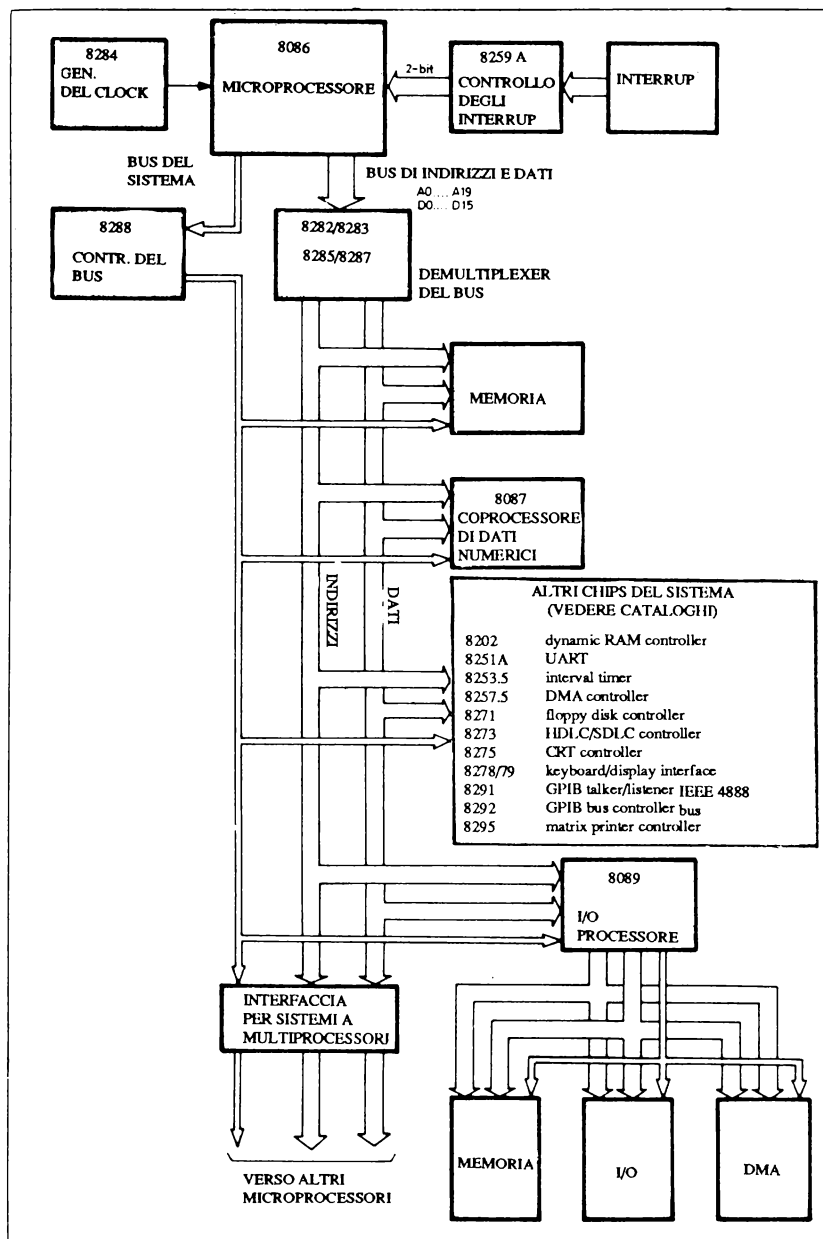


Fig. 12.-Schema dei diversi circuiti integrati in LSI disponibili per il  $\mu P$  8086.

SAHF: trasferisce i flags;  
PUSHF: mette i flags nello stack;  
POPF: estrae i flags dallo stack;  
XLAT: trasforma un carattere di un codice (EBCDIC) in un altro (ASCII).

*- Istruzioni aritmetiche:*

ADD: somma due parole o bytes con o senza segno;  
ADC: somma tenendo conto del carry;  
SUB: sottrae;  
SUBB: sottrae tenendo conto del prestito;  
MUL: moltiplicazione di parole o bytes, con o senza segno;  
IMUL: moltiplicazione di dati interi;  
DIV: divisione di dati a 32 e 16 bit per dati a 16 e 8 bit senza segno;  
IDIV: divisione intera con segno;  
CMP: confronto di parole o dati;  
CBN: estensione di segno passando da 8 a 16 bit;  
CWD: estensione di segno passando da 16 a 32 bit;  
DAA: regolazione decimale per sommare;  
DAS: regolazione decimale per sottrarre;  
DEG: negazione di un numero;  
AAM: regolazione decimale per moltiplicare;  
AAD: regolazione decimale per dividere;  
INC: incremento;  
DEC: decremento.

*- Istruzioni logiche:*

NOT: negazione;  
OR: operazione OR;  
AND: operazione AND;  
XOR: operazione OR esclusivo;  
TEST: operazione AND per verificare i flags.

*- Istruzioni di salto e interrupt:*

JMP: salto incondizionato;  
J: salto dipendente da 30 condizioni diverse;  
JCX: salto se CX=0;  
LOOP: regola di cinque modi alla fine di un loop;  
INT: interrupt via software;  
INTO interrupt condizionato (se O= $\Phi$  overflow);  
IRET: ritorno da interrupt;  
LODS: trasferisce un dato da AL e AX;  
STOS: trasferisce un dato dai registri accumulatori AL e AX;  
SCAS: trasferisce un dato con i registri AL e AX;



CMPS: confronta due caratteri;

MOVSB e MOVSM: trasferisce un byte o una parola.

- Istruzioni di controllo:

NOP: non opera;

WAIT: per il processore fino a che TEST sia 0;

HALT: per il processore;

LOCK: permette a LOCK di passare a 0 durante l'esecuzione dell'istruzione successiva;

CLD e CLI: settano a 0 i flags D e I, rispettivamente;

STD e STI: settano a 1 i flags D e I, rispettivamente;

CLC e STC: settano a 0 e 1 il flag di carry;

CMC: complementa il flag di carry;

IN: questa istruzione di ingresso trasferisce il dato da una periferica ai registri AL o AX;

OUT: questa istruzione di uscita trasferisce il dato dai registri AL o AX a una periferica;

ROL: rotazione a sinistra;

ROR: rotazione a destra;

RCL: rotazione a sinistra includendo il bit (o flag) di carry del registro di stato;

RCR: rotazione a destra con il bit (o flag) di carry del registro di stato;

CALL: chiamata a subroutine;

RET: ritorno da subroutine;

PUSH: carica un dato nello stack;

POP: estrae un dato dallo stack;

REP: ripetere;

REPE: ripetere se uguale;

REPNE: ripetere se diverso;

REPZ, REPNZ: ripetere se è zero o se non è zero, rispettivamente. Sono istruzioni che elaborano catene di caratteri, utilizzando come puntatori i registri SI e DI.

## *Significato delle abbreviazioni*

Nell'insieme delle istruzioni si utilizzano abbreviazioni e note il cui significato è il seguente:

AL=registro accumulatore a 8 bit;

AX=registro accumulatore a 16 bit;

CX=registro contatore;

DS=segmento dei dati;

ES=segmento extra;

ABOVE=al di sopra (riferito al valore assegnato);

BELOW=al di sotto (riferito al valore assegnato);

GREATER=valore più positivo;

LESS=valore meno positivo (più negativo), riferito al segno dei valori.

Se  $d=1$ , allora il valore è più negativo di quello del registro. Se  $d=0$ , il valore è più positivo di quello del registro. Se  $W=1$ , allora l'istruzione è di parola (tiene conto del segno). Se  $W=0$ , l'istruzione è di byte.

## Condizioni

- se  $Mod.=11$ , allora  $r/m$  è trattato come un registro;
- se  $Mod.=00$ , allora  $DISP=0$  (senza spostamento, eccetto se  $r/m=110$ , allora  $EA$  indica lo spostamento);
- se  $Mod.=01$ , allora ( $DISP$ ) lo spostamento basso è di 16 bit con segno (non c'è spostamento alto);
- se  $Mod.=10$ , allora  $DESPL\ ALTO=DESPL\ BASSO$ ;
- se  $r/m=000$ , allora  $EA=(BX)+(SI)+spostamento$ ;
- se  $r/m=001$ , allora  $EA=(BX)+(DI)+spostamento$ ;
- se  $r/m=010$ , allora  $EA=(BP)+(SI)+spostamento$ ;
- se  $r/m=011$ , allora  $EA=(BP)+(DI)+spostamento$ ;
- se  $r/m=100$ , allora  $EA=(SI)+spostamento$ ;
- se  $r/m=101$ , allora  $EA=(DI)+spostamento$ ;
- se  $r/m=110$ , allora  $EA=(BP)+spostamento$ ;
- se  $r/m=111$ , allora  $EA=(BX)+spostamento$ ;
- se  $W=01$ , allora prende il dato come 16 bit;
- se  $W=11$ , allora il dato di un byte viene esteso a 16 bit;
- se  $V=0$ , allora  $Count=1$ , se  $V=1$  allora  $Count$  in  $CL$ ;
- $X$ =qualsiasi valore;
- prefisso di segmento=001 reg 1101;
- Reg viene assegnato in base alla tabella seguente:

Le istruzioni che si riferiscono ai bit o flags dello status usano 16 bit.

Il simbolo dei flags è:

FLAGS=XXXX(OF): (DF): (IF): (TF): (SF): (ZF): X: (AF): X: (PF): X(CF).



## ISTRUZIONI LOGICHE

**NOT=Invert**

|                 |               |
|-----------------|---------------|
| 1 1 1 1 0 1 1 w | mod 0 1 0 r/m |
|-----------------|---------------|

**SHL/SAL=Shift logical/arithmetic**

left

|                 |               |
|-----------------|---------------|
| 1 1 0 1 0 0 v w | mod 1 0 0 r/m |
|-----------------|---------------|

**SHR=Shift logical right**

|                 |               |
|-----------------|---------------|
| 1 1 0 1 0 0 v w | mod 1 0 1 r/m |
|-----------------|---------------|

**SAR=Shift arithmetic right**

|                 |               |
|-----------------|---------------|
| 1 1 0 1 0 0 v w | mod 1 1 1 r/m |
|-----------------|---------------|

**ROL=Rotate left**

|                 |               |
|-----------------|---------------|
| 1 1 0 1 0 0 v w | mod 0 0 0 r/m |
|-----------------|---------------|

**ROR=Rotate right**

|                 |               |
|-----------------|---------------|
| 1 1 0 1 0 0 v w | mod 0 0 1 r/m |
|-----------------|---------------|

**RCL=Rotte through carry flag left**

|                 |               |
|-----------------|---------------|
| 1 1 0 1 0 0 v w | mod 0 1 0 r/m |
|-----------------|---------------|

**RCR=Rotate through carry right**

|                 |               |
|-----------------|---------------|
| 1 1 0 1 0 0 v w | mod 0 1 1 r/m |
|-----------------|---------------|

**AND=And:**

Reg/memory and register to either

|                 |             |
|-----------------|-------------|
| 0 0 1 0 0 0 d w | mod reg r/m |
|-----------------|-------------|

Immediate to register/memory

|                 |               |      |             |
|-----------------|---------------|------|-------------|
| 1 0 0 0 0 0 0 w | mod 1 0 0 r/m | data | data if w=1 |
|-----------------|---------------|------|-------------|

Immediate to accumulator

|                 |      |             |
|-----------------|------|-------------|
| 0 0 1 0 0 1 0 w | data | data if w=1 |
|-----------------|------|-------------|

## ISTRUZIONI ARITMETICHE

**ADD=Add:**

Reg/memory with register to either

|                 |             |
|-----------------|-------------|
| 0 0 0 0 0 0 d w | mod reg r/m |
|-----------------|-------------|

Immediate to register/memory

|                 |               |      |                |
|-----------------|---------------|------|----------------|
| 1 0 0 0 0 0 s w | mod 0 0 0 r/m | data | data if s w=01 |
|-----------------|---------------|------|----------------|

Immediate to accumulator

|                 |      |             |
|-----------------|------|-------------|
| 0 0 0 0 0 1 0 w | data | data if w=1 |
|-----------------|------|-------------|

**ADC=Add with carry:**

Reg/memory with register to either

|                 |             |
|-----------------|-------------|
| 0 0 0 1 0 0 d w | mod reg r/m |
|-----------------|-------------|

Immediate to register/memory

|                 |               |      |                |
|-----------------|---------------|------|----------------|
| 1 0 0 0 0 0 s w | mod 0 1 0 r/m | data | data if s w=01 |
|-----------------|---------------|------|----------------|

Immediate to accumulator

|                 |      |             |
|-----------------|------|-------------|
| 0 0 0 1 0 1 0 w | data | data if w=1 |
|-----------------|------|-------------|

**INC=Increment:**

Register/memory

|                 |               |
|-----------------|---------------|
| 1 1 1 1 1 1 i w | mod 0 0 0 r/m |
|-----------------|---------------|

Register

|               |
|---------------|
| 0 1 0 0 0 reg |
|---------------|

**AAA=ASCII adjust for add**

|                 |
|-----------------|
| 0 0 1 1 0 1 1 1 |
|-----------------|

**DAA=Decimal adjust for add**

|                 |
|-----------------|
| 0 0 1 0 0 1 1 1 |
|-----------------|

**SUB=Subtract:**

Reg/memory and register to either

|                 |             |
|-----------------|-------------|
| 0 0 1 0 1 0 d w | mod reg r/m |
|-----------------|-------------|

Immediate from register/memory

|                 |               |      |                |
|-----------------|---------------|------|----------------|
| 1 0 0 0 0 0 s w | mod 1 0 1 r/m | data | data if s w=01 |
|-----------------|---------------|------|----------------|

Immediate from accumulator

|                 |      |             |
|-----------------|------|-------------|
| 0 0 1 0 1 1 0 w | data | data if w=1 |
|-----------------|------|-------------|

**SBB=Subtract with borrow**

Reg/memory and register to either

|                 |             |
|-----------------|-------------|
| 0 0 0 1 1 0 d w | mod reg r/m |
|-----------------|-------------|

Immediate from register/memory

|                 |               |      |                |
|-----------------|---------------|------|----------------|
| 1 0 0 0 0 0 s w | mod 0 1 1 r/m | data | data if s w=01 |
|-----------------|---------------|------|----------------|

Immediate from accumulator

|                 |      |             |
|-----------------|------|-------------|
| 0 0 0 1 1 1 0 w | data | data if w=1 |
|-----------------|------|-------------|

**TEST=And function to flags, no result:**

Register/memory and register

|                 |             |
|-----------------|-------------|
| 1 0 0 0 0 1 0 w | mod reg r/m |
|-----------------|-------------|

Immediate data and register/memory

|                 |               |      |             |
|-----------------|---------------|------|-------------|
| 1 1 1 1 0 1 1 w | mod 0 0 0 r/m | data | data if w=1 |
|-----------------|---------------|------|-------------|

Immediate data and accumulator

|                 |      |             |
|-----------------|------|-------------|
| 1 0 1 0 1 0 0 w | data | data if w=1 |
|-----------------|------|-------------|

**OR=Or:**

Reg/memory and register to either

|                 |             |
|-----------------|-------------|
| 0 0 0 0 1 0 d w | mod reg r/m |
|-----------------|-------------|

Immediate to register/memory

|                 |               |      |             |
|-----------------|---------------|------|-------------|
| 1 0 0 0 0 0 0 w | mod 0 0 1 r/m | data | data if w=1 |
|-----------------|---------------|------|-------------|

Immediate to accumulator

|                 |      |             |
|-----------------|------|-------------|
| 0 0 0 0 1 1 0 w | data | data if w=1 |
|-----------------|------|-------------|

**XOR=Exclusive or:**

Reg/memory and register to either

Immediate to register/memory

Immediate to accumulator

|                 |             |             |             |
|-----------------|-------------|-------------|-------------|
| 0 0 1 1 0 0 d w | mod reg r/m |             |             |
| 1 0 0 0 0 0 w   | mod 110 r/m | data        | data if w=1 |
| 0 0 1 1 0 1 w   | data        | data if w=1 |             |

**ISTRUZIONI DI CONTROLLO****REP**=Repeat**MOVS**=Move byte/word**CMPS**=Compare byte/word**SCAS**=Scan byte/word**LODS**=Load byte/wd to AL/AX**STDS**=Stor byte/wd from AL/A

|                 |
|-----------------|
| 1 1 1 1 0 0 1 z |
| 1 0 1 0 0 1 0 w |
| 1 0 1 0 0 1 1 w |
| 1 0 1 0 1 1 1 w |
| 1 0 1 0 1 1 0 w |
| 1 0 1 0 1 0 1 w |

**ISTRUZIONI DI CONTROLLO DEL TRASFERIMENTO****CALL**=Call:

7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0

Direct within segment

Indirect within segment

Direct intersegment

Indirect intersegment

|                 |             |             |
|-----------------|-------------|-------------|
| 1 1 1 0 1 0 0 0 | disp-low    | disp-high   |
| 1 1 1 1 1 1 1 1 | mod 010 r/m |             |
| 1 0 0 1 1 0 1 0 | offset-low  | offset-high |
|                 | seg-low     | seg-high    |
| 1 1 1 1 1 1 1 1 | mod 011 r/m |             |

**JMP**=Unconditional jump:

Direct within segment

Direct within segment-short

Indirect within segment

Direct intersegment

Indirect intersegment

|                 |             |             |
|-----------------|-------------|-------------|
| 1 1 1 0 1 0 0 1 | disp-low    | disp-high   |
| 1 1 1 0 1 0 1 1 | disp        |             |
| 1 1 1 1 1 1 1 1 | mod 100 r/m |             |
| 1 1 1 0 1 0 1 0 | offset-low  | offset-high |
|                 | seg-low     | seg-high    |
| 1 1 1 1 1 1 1 1 | mod 101 r/m |             |

**RET**=Return from CALL:

Within segment

Within seg adding immed to SP

Intersegment

Intersegment adding immediate to SP

**JE/JZ**=jump on equal/zero**JL/JNGE**=jump on less/not greater on equal**JLE/JNG**=jump on less or equal/not greater**JB/JNAE**=jump on below/not above or equal**JBE/JNA**=jump on below or equal/not above**JP/JPE**=jump on parity/parity even**JO**=jump on overflow**JS**=jump on sign**JNE/JNZ**=jump on not equal/not zero**JNL/JGE**=jump on not less/greater or equal**JNLE/JG**=jump on not less or equal/greater**JNB/JAE**=jump on not below/above or equal**JNBE/JA**=jump on not below or equal/above**JNP/JPO**=jump on not par/par odd**JNO**=jump on not overflow**JNS**=jump on not sign**LOOP**=Loop. CX times**LOOPZ/LOOPE**=Loop with zero/equal

|                 |          |           |
|-----------------|----------|-----------|
| 1 1 0 0 0 0 1 1 |          |           |
| 1 1 0 0 0 0 1 0 | data-low | data-high |
| 1 1 0 0 1 0 1 1 |          |           |
| 1 1 0 0 1 0 1 0 | data-low | data-high |
| 0 1 1 1 0 1 0 0 | disp     |           |
| 0 1 1 1 1 1 0 0 | disp     |           |
| 0 1 1 1 1 1 1 0 | disp     |           |
| 0 1 1 1 0 0 1 0 | disp     |           |
| 0 1 1 1 0 1 1 0 | disp     |           |
| 0 1 1 1 1 0 1 0 | disp     |           |
| 0 1 1 1 0 0 0 0 | disp     |           |
| 0 1 1 1 1 0 0 0 | disp     |           |
| 0 1 1 1 0 1 0 1 | disp     |           |
| 0 1 1 1 1 1 0 1 | disp     |           |
| 0 1 1 1 1 1 1 1 | disp     |           |
| 0 1 1 1 0 0 1 1 | disp     |           |
| 0 1 1 1 0 1 1 1 | disp     |           |
| 0 1 1 1 1 0 1 1 | disp     |           |
| 0 1 1 1 0 0 0 1 | disp     |           |
| 0 1 1 1 1 0 0 1 | disp     |           |
| 1 1 1 0 0 0 1 0 | disp     |           |
| 1 1 1 0 0 0 0 1 | disp     |           |

|  |                 |               |
|--|-----------------|---------------|
| <b>LOOPNZ/LOOPNE</b> =Loop while not zero/equal    | 1 1 1 0 0 0 0 0 | disp          |
| <b>JCZX</b> =jump on CX zero                       | 1 1 1 0 0 0 1 1 | disp          |
| <b>INT</b> =Interrupt                              |                 |               |
| Type specified                                     | 1 1 0 0 1 1 0 1 | type          |
| Type 3   | 1 1 0 0 1 1 0 0 |               |
| <b>INTO</b> =Interrupt on overflow                 | 1 1 0 0 1 1 1 0 |               |
| <b>IRET</b> =Interrupt return                      | 1 1 0 0 1 1 1 1 |               |
| <b>ISTRUZIONI DI CONTROLLO DEL MICROPROCESSORE</b> |                 |               |
| <b>CLC</b> =Clear carry                            | 1 1 1 1 1 0 0 0 |               |
| <b>CMC</b> =Complement carry                       | 1 1 1 1 0 1 0 1 |               |
| <b>STC</b> =Set carry                              | 1 1 1 1 1 0 0 1 |               |
| <b>CLD</b> =Clear direction                        | 1 1 1 1 1 1 0 0 |               |
| <b>STD</b> =Set direction                          | 1 1 1 1 1 1 0 1 |               |
| <b>CLI</b> =Clear interrupt                        | 1 1 1 1 1 0 1 0 |               |
| <b>STI</b> =Set interrupt                          | 1 1 1 1 1 0 1 1 |               |
| <b>HLT</b> =Halt                                   | 1 1 1 1 0 1 0 0 |               |
| <b>WAIT</b> =Wait                                  | 1 0 0 1 1 0 1 1 |               |
| <b>ESC</b> =Escape (to external device)            | 1 1 0 1 1 x x x | mod x x x r/m |
| <b>LOCK</b> =Bus lock prefix                       | 1 1 1 1 0 0 0 0 |               |

Tabella VI.-Istruzioni del  $\mu P$  8086.

## Modi di indirizzamento

I modi fondamentali di indirizzamento dell'8086 sono:

- mediante registro base: l'indirizzo si ottiene mediante la somma del contenuto di un registro base e il dato dell'istruzione;
- immediato: indicato dalla stessa istruzione;
- mediante registro indiretto: l'indirizzo si trova in BX, BP, SI, DI;
- diretto: l'indirizzo è nella stessa istruzione;
- relativo: utilizzato per istruzioni che necessitano salti, e per l'accesso a subroutine;
- mediante registro indicizzato: l'indirizzo si ottiene sommando a un registro indice il dato dell'istruzione;
- mediante registro base indicizzato: l'indirizzo finale è la somma del contenuto del registro base, dell'indice, e del dato dell'istruzione.

## CONCLUSIONI

**I** microprocessori, che sono realmente unità centrali di elaborazione (CPU) realizzati con uno (o pochi) circuiti integrati, sono componenti la cui evoluzione è molto legata ai progressi della tecnologia di fabbricazione dei circuiti integrati e all'architettura degli elaboratori.

Il microprocessore è un componente elettronico, un circuito integrato, che esegue le funzioni di unità centrale di un elaboratore. Quando al  $\mu P$  si aggiungono circuiti di memoria, e di ingresso/uscita si ottiene un  $\mu C$ , ossia un elaboratore realizzato con un microprocessore.

Il microcomputer è il livello inferiore nella classificazione degli elaboratori per la sua dimensione (e prezzo): microcomputer, minicomputer, e grande elaboratore (o mainframe, come viene normalmente chiamato). L'obiettivo della ricerca nel campo degli elaboratori è di ottenere maggiori velocità di elaborazione e potenza di calcolo, utilizzando a questo scopo componenti elettronici con tempo di propagazione minore, minori dimensioni, e minore consumo, e organizzando la macchina con architetture più efficienti.

Nello stesso tempo i progettisti di minielaboratori, basandosi sull'architettura di quelli grandi, realizzano semplificazioni degli stessi e utilizzano altri componenti e altre tecnologie che, anche se diminuiscono le prestazioni complessive dell'apparecchio, portano a notevoli riduzioni delle dimensioni e del prezzo.

Infine, i progettisti di microelaboratori semplificano al massimo questi processi per cui tutto il circuito può essere realizzato con un solo circuito integrato.

I progressi nella tecnologia di fabbricazione dei circuiti integrati, che permettono di conglobare in un solo chip una gran quantità di circuiti attivi, con-

sente a un microprocessore di effettuare prestazioni che superano quelle dei minielaboratori di qualche anno fa. Lo stesso fatto accade per i minielaboratori rispetto ai mainframes.

## *Glossario*

La forma con cui i temi sono stati esposti è stata seria e chiara, senza essere pedante, con un approfondimento progressivo per evitare paure infondate. Con la stessa finalità si inserisce in quest'ultimo capitolo un glossario, concepito per agevolare la comprensione dei termini meno conosciuti, cui riferirsi quando nel testo compare qualche termine nuovo.

**ACKNOWLEDGE:** risposta affermativa. Segnale di controllo utilizzato per completare una sequenza di conferma di uno scambio di indicazioni e segnali di controllo.

**ADDRESS:** indirizzo. Numero che indica la posizione di una parola in memoria.

**AND:** operazione logica definita dalla regola seguente: se A e B sono 1, C è 1; altrimenti è 0.

**BISTABILE:** dispositivo con due possibili stati che possono essere invertiti mediante un segnale esterno.

**BIT:** contrazione di binary digit (numero binario). Unità di informazione che può assumere due valori o stati distinti.

**BYTE:** otteetto. Termine che rappresenta una parte misurabile di numeri binari consecutivi.

**CARRY:** riporto delle varie operazioni aritmetiche che può effettuare un microprocessore.

**CHIP:** circuito integrato.

**CLOCK:** generatore di impulsi. Dispositivo che genera una base dei tempi utilizzata per fornire gli impulsi sequenziali fondamentali per le operazioni di un elaboratore, il cui modo di lavorare è sequenziale.

**CPU:** unità centrale di processo. Modulo che ha il compito di cercare, decodificare ed eseguire istruzioni. Comprende un'unità di controllo, una unità aritmetica e logica (ALU) e dispositivi affini (registri, clock, ecc.).

**DMA:** direct memory access. Accesso diretto alla memoria.

**FIRMWARE:** supporto logico inalterabile. Programma esistente in ROM.

**HARDWARE:** componenti, dispositivi fisici. Apparecchiatura fisica contrapposta ai programmi dell'elaboratore o ai metodi per il suo utilizzo.

**HMOS:** i progressi della tecnologia MOS hanno permesso di aumenta-



re la densità di integrazione della NMOS, e di quadruplicare il suo rapporto velocità/consumo.

LATCH: dispositivo fisico che cattura l'informazione e la trattiene.

KILOBYTE (K): 1024 bytes.

LSI (Large Scale Integration): integrazione a grande scala.

MEGABYTE (M): 1048576 bytes.

μP: microprocessore. E' fondamentalmente un circuito integrato che si può utilizzare come circuito logico programmabile, o come unità centrale di processo di un microelaboratore.

NAND: porta NON-AND. Operazione logica definita dalla regola seguente: se tutti gli ingressi sono 1, l'uscita è 0; altrimenti è 1.

NOT: funzione negazione, per cui tutti i bit o ingressi sono negati.

OR: porta OR. Elemento logico in cui la variabile presente nell'unico segnale binario di uscita è la disgiunzione delle variabili presenti nei due o più segnali binari di ingresso; cioè l'uscita sarà 1 quando almeno un ingresso è 1.

PIN: terminale di collegamento all'esterno dei circuiti integrati.

READY: Segnale utilizzato come ingresso o uscita nei microprocessori, per comunicare o ricevere la conferma che lo stesso o una periferica è pronto.

RESET: azzeramento. In programmazione, azione di mettere a zero un contatore o portare un indicatore a qualche condizione stabile.

SILICIO: elemento molto comune utilizzato come base per la costruzione dei circuiti integrati.

SOFTWARE: supporto logico, insieme di programmi. Complesso di programmi e procedimenti che si inseriscono in un apparecchio di elaborazione dati, e che ne rendono possibile l'utilizzazione.

STACK: catasta. Struttura LIFO che memorizza l'informazione in ordine cronologico.

VLSI (Very Large Scale Integration): Integrazione a scala molto grande.









# **progettare con L'ELETTRONICA**

## **costruire per conoscere**

il Gruppo Editoriale Jackson è lieto di annunciare la nascita di una nuova collana di libri di elettronica, naturale complemento di Libri di Base Elettronica:

PROGETTARE CON L'ELETTRONICA, un nuovo metodo per imparare l'elettronica attraverso la pratica.

PROGETTARE CON L'ELETTRONICA, 20 preziose guide ricche di progetti pratici idee e suggerimenti per chi ama l'elettronica e il fai da te. Argomenti approfonditi, circuiti collaudati e di facile realizzazione, fotografie e schemi ti permetteranno di approfondire le tue conoscenze e arricchire la tua esperienza.

PROGETTARE CON L'ELETTRONICA,  
ti da il primo appuntamento con:

**SISTEMI DI ALLARME**

**TV VIA SATELLITE**

**MULTIMETRI**

**PREAMPLIFICATORI AUDIO**

**QUATTRO VOLUMI  
NELLA TUA EDICOLA OGNI MESE**



**GRUPPO EDITORIALE  
JACKSON**

# **progettare con** **L' ELETTRONICA** **costruire per conoscere**

**QUATTRO VOLUMI NELLA TUA EDICOLA OGNI MESE:**

- 1 SISTEMI DI ALLARME
- 2 TV VIA SATELLITE
- 3 MULTIMETRI
- 4 PREAMPLIFICATORI AUDIO
  
- 5 TELECOMANDI
- 6 RADIOTELEFONI E RICETRASMITTENTI
- 7 ALIMENTATORI
- 8 AMPLIFICATORI DI POTENZA
  
- 9 CIRCUITI PER L'AUTOMOBILE
- 10 RICEVITORI E SINTONIZZATORI
- 11 ELETTRONICA DIGITALE CON ESPERIMENTI  
dalle porte logiche ai flip flop
- 12 MIXER AUDIO
  
- 13 MODELLISMO E ELETTRONICA
- 14 TELEFONI, CENTRALINE TELEFONICHE E MODEM
- 15 ELETTRONICA INTEGRATA DIGITALE  
CON ESPERIMENTI  
dai contatori alle memorie
- 16 EQUALIZZATORI AUDIO
  
- 17 LUCI PSICHEDELICHE
- 18 AMPLIFICATORI A CIRCUITI INTEGRATI
- 19 GENERATORI DI FUNZIONE
- 20 MUSICA ELETTRONICA



**GRUPPO EDITORIALE**  
**JACKSON**

## **CONTIENE LA GUIDA A:**

**SIMBOLI, UNITÀ DI MISURA,  
TABELLE DI CONVERSIONE  
E CODICI COLORE  
DEI COMPONENTI ELETTRONICI**

## **TABELLE E CODICI**

GUIDA AI SIMBOLI  
UNITÀ DI MISURA  
TABELLE DI CONVERSIONE  
E CODICI COLORE  
DEI COMPONENTI ELETTRONICI

**L**

*a parte hardware che costituisce il cuore effettivo di qualsiasi apparecchiatura di elaborazione è la cosiddetta CPU. Tale dispositivo, a sua volta, è gestito da un particolare componente integrato noto come MPU o, più generalmente, come micro-*

*processore.*

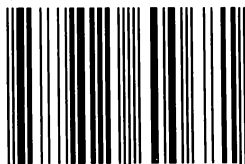
*I microprocessori altro non sono che dei chip integrati su larga scala dotati ognuno di una sua specificità costruttiva. Pur se diversi strutturalmente, i vari microprocessori si rifanno tutti ad un'unica logica funzionale che viene ampiamente descritta nel presente volume. Sono inoltre descritte le caratteristiche principali e la tecnologia di fabbricazione di vari tipi di microprocessori, sia a 8 che a 16 bit. Ciò, di primo acchito, permette di confrontare tra loro i vari componenti, consentendo così un'utile verifica atta a facilitarne l'utilizzo e la scelta, nel caso in cui si desideri effettuare qualche esperimento basato su di essi.*

**GRUPPO EDITORIALE JACKSON**

**L. 12.500**

**Cod. 072E**

ISBN 88-256-0068-2



9 788825 600681





**EH17**

**MICROPROCESORI**

