Programmazione dello Z80

- Il microprocessore si incarica di:
- gestire il programma e i suoi dati
- eseguire i calcoli richiesti.

Le azioni appena elencate rendono necessario che il microprocessore abbia da qualche parte, al suo interno, qualcosa che gli consenta di prendere nota di ciò che sta facendo e di trascrivere i risultati parziali dei suoi calcoli.

Ed infatti all'interno del microprocesore ci sono una serie di registri , impiegati per tutta quella serie di operazioni che devono essere svolte con velocità, dati i frequenti accessi richiesti, oppure che servono alla definizione dello stato del microprocessore stesso.

- Tali registri sono collocati all'interno del microprocessore, e ciò per risparmiare il tempo che sarebbe richiesto se si impiegassero i bus esterni per raggiungerli.

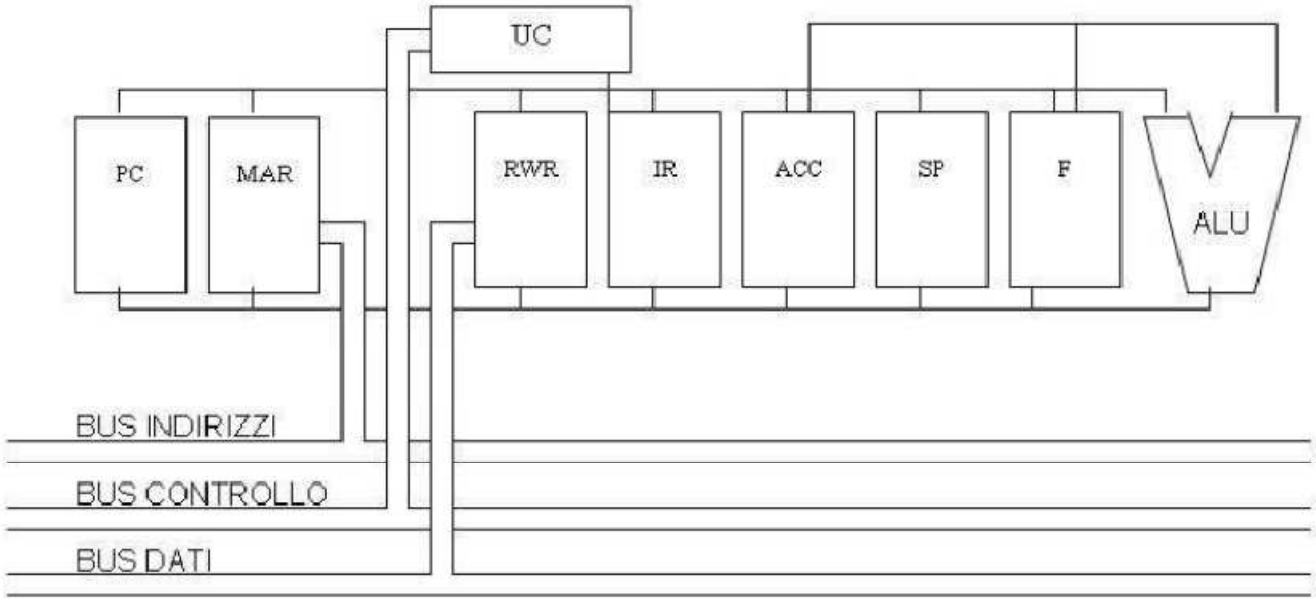
I registri si dividono in due categorie:

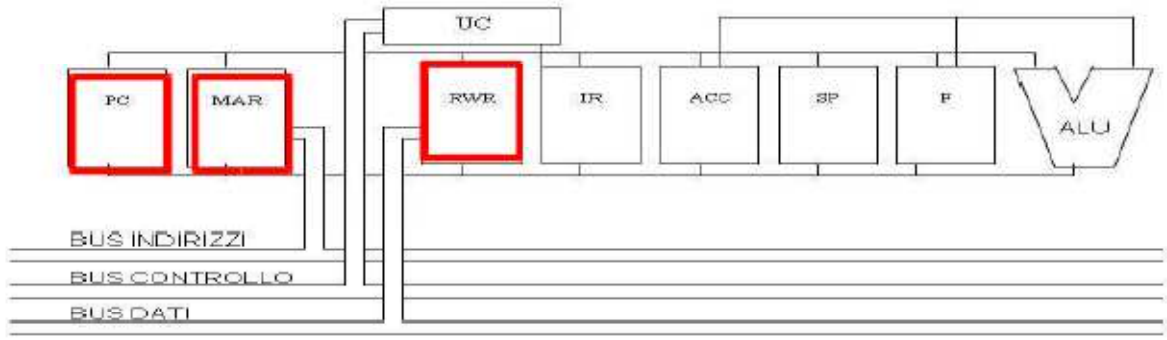
- i registri generali
- i registri speciali.

I registri generali non hanno uno scopo specifico, ma servono per mantenere traccia del lavoro in corso.

I registri speciali, invece, servono a prendere nota di un particolare aspetto o evento relativo allo stato complessivo del microprocessore.

- **Nella figura successiva è riportato un esempio di blocchi interni di un microprocessore.**

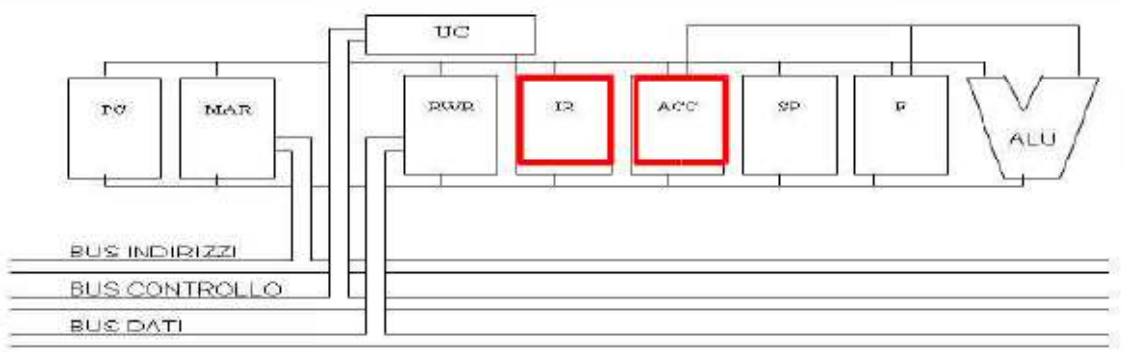




Il registro PC (program counter) è il registro contatore di programma, esso serve per conservare l'indirizzo della locazione successiva di memoria da analizzare.

Il registro MAR (memory address register) è il registro indirizzi, esso è un registro di parcheggio contenente l'indirizzo che è stato inviato sul bus indirizzi

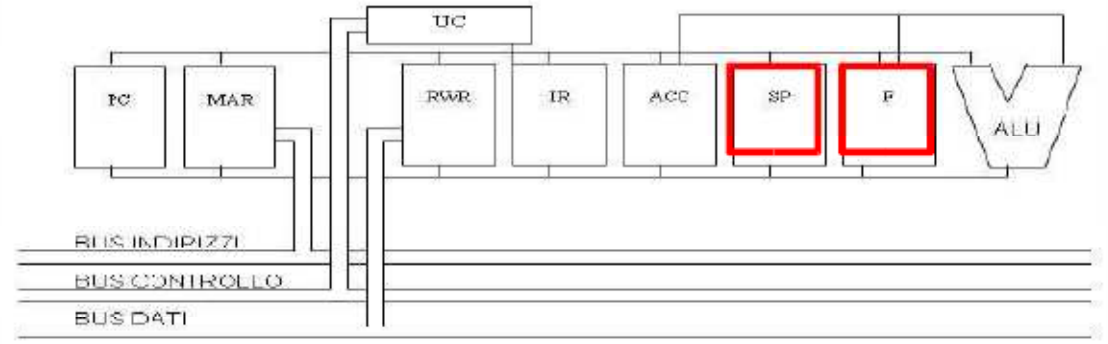
Il registro RWR (read write register) è il registro di lettura e scrittura, il suo scopo è di fungere da area di parcheggio dei dati
- o che devono essere posti sul bus dati verso l'esterno del microprocessore
- o che sono appena arrivati all'interno del microprocessore per essere manipolati secondo quanto indicato dal programma.



IR (instruction register) è il registro di istruzione, la sua funzione è di conservare il codice operativo dell'istruzione in corso di esecuzione.

ACC (accumulator) è il registro accumulatore, esso è necessario nelle operazioni logiche aritmetiche perché ;

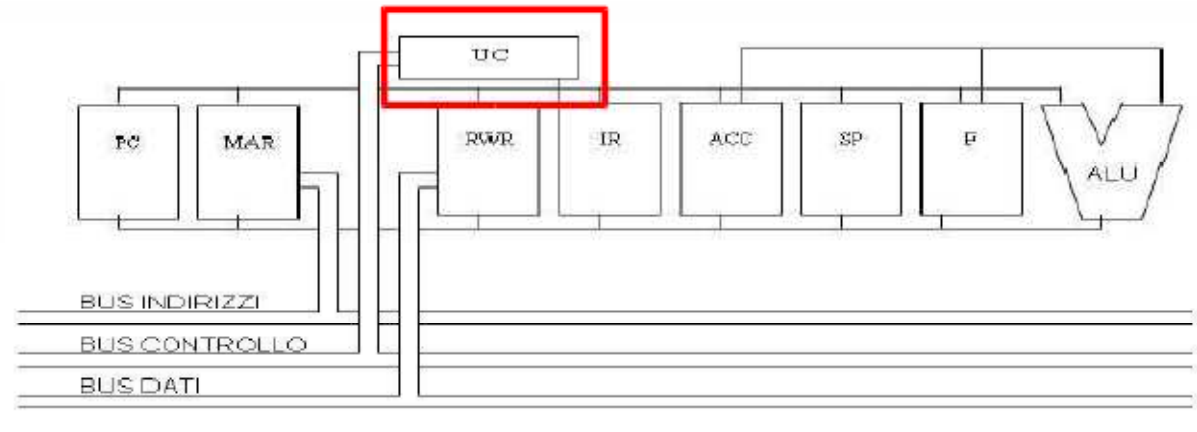
- o fa da sorgente di uno degli operandi dell'operazione da svolgere all'interno dell'ALU
- o fa da destinazione del risultato dell'operazione svolta all'interno dell'ALU.



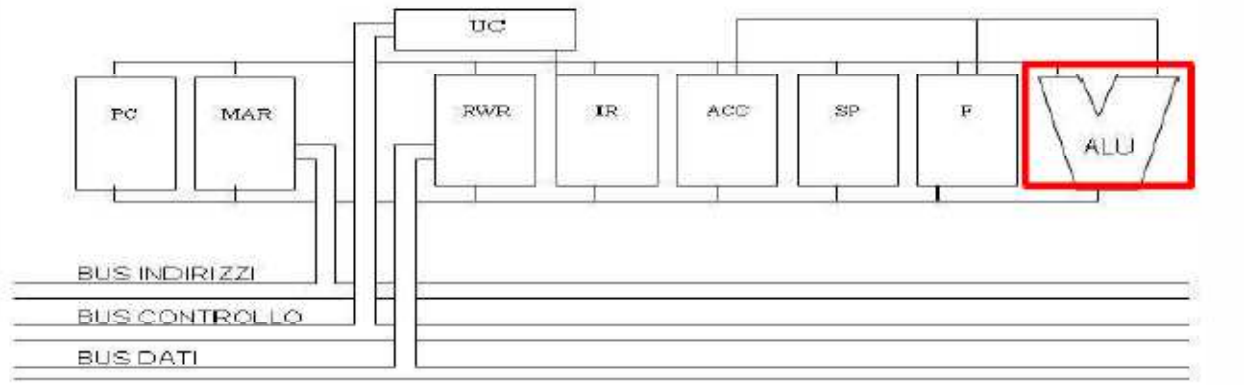
SP (stack pointer) è il puntatore all'area di stack, questo registro contiene l'indirizzo dell'ultima locazione occupata dall'area di stack o pila.

F (flag) è il registro dei flag, in questo registro ogni bit ha un significato specifico visto che ognuno indica il verificarsi o meno di un evento specifico, la sua consultazione è fondamentale ogni volta che il microprocessore deve prendere una decisione sul flusso di azioni da eseguire, con i flag che sono modificati solo da operazioni di tipo logico o aritmetico.

B7	B6	B5	B4	B3	B2	B1	B0
S	*Z*	*C*	*H*	*O*	*P*	*>*	*IE*



L'unità di controllo UC è il referente unico del bus di controllo; in base, infatti, ai segnali di controllo eventualmente attivati, o alle istruzioni date da programma, l'unità di controllo deve prendere le sue decisioni e attivare le parti coinvolte del processore, con l'unità di controllo che dunque si preoccupa di identificare e decodificare l'istruzione e di conseguenza eseguire l'azione relativa.



L'ALU lavora con la maggior parte dei registri indicati nella figura sopra riportata, tuttavia il suo lavoro ha effetto immediato sicuramente sull'accumulatore ACC (che le fornirà sicuramente uno degli operandi), e sul flag F (che conterrà indicazioni sul risultato delle operazioni).

SET DI ISTRUZIONI

La CPU Z80 ha uno dei più potenti e versatili set di istruzioni disponibile in un microprocessore a 8 bit. Tale set include operazioni particolari come movimento di blocchi (block move) per un trasferimento veloce ed efficiente di dati all'interno della memoria o tra memoria e dispositivi di I/O. Inoltre consente operazioni logiche su qualunque bit in qualunque locazione di memoria. Le istruzioni possono essere suddivise nelle seguenti categorie

- caricamento a 8 bit
- caricamento a 16 bit
- scambio, ricerca e trasferimento di blocchi
- operazioni aritmetiche di uso generale
- operazioni aritmetiche a 16 bit
- rotazione e spostamenti
- operazioni sul bit di set, reset, e test
- salti
- operazioni di ingresso/uscita

SET DI ISTRUZIONI

Per effettuare trasferimenti efficienti e veloci di dati tra registri, locazioni di memoria e dispositivi di I/O, sono implementati una grande varietà di modi di indirizzamento:

- immediato
- immediato esteso
- diretto
- modificato in pagina zero
- relativo
- indicizzato
- a registri
- indiretto a registri
- implicito

Introduzione ai tipi di istruzioni

- ### Caricamento e scambio

Le istruzioni di Load (caricamento) muovono i dati internamente tra i vari registri della CPU, oppure tra i registri e la memoria esterna. Tutte queste istruzioni devono precisare una locazione sorgente (source location), dalla quale il dato deve essere prelevato, ed una posizione destinazione (destination). La sorgente non viene modificata dalle istruzioni di Load. Le istruzioni di Load comprendono, per esempio, lo spostamento di un dato tra due qualsiasi dei registri di uso generale. Questo gruppo di istruzioni comprende pure le Load immediate del dato specificato nell'istruzione in un qualsiasi registro della CPU o in una qualsiasi posizione di memoria. Altri tipi di istruzioni di Load permettono il trasferimento di dati tra i registri della CPU e la memoria. Le istruzioni in Exchange (scambio) possono effettuare lo scambio tra il contenuto di due registri.

- ### Trasferimento e ricerca di blocchi di dati

Nello Z80 é disponibile un set di istruzioni di Block Transfer (trasferimento di blocchi di dati). Con una sola istruzione é possibile trasferire un blocco di dati di qualsiasi dimensione da una posizione di memoria ad una qualsiasi altra. Inoltre mediante una sola istruzione di Block Search (ricerca in un blocco di dati) é possibile analizzare un blocco di memoria di qualsiasi dimensione per ricercare un carattere ad 8 bit. Una volta che il carattere é stato trovato, o che si é giunti alla fine del blocco di memoria l'istruzione termina automaticamente. Sia le istruzioni di Block Transfer che quelle di Block Search possono essere interrotte durante la loro esecuzione in modo da non occupare per troppo tempo la CPU.

- ### Istruzioni aritmetiche e logiche

Le istruzioni aritmetiche e logiche operano sui dati posti in accumulatore e in un altro qualsiasi dei registri di uso generale della CPU, oppure sui dati posti in accumulatore ed in una posizione della memoria esterna. Il risultato dell'operazione é posto in accumulatore ed i flag sono posizionati in accordo con il risultato stesso. Un esempio di operazione aritmetica può essere la somma dell'accumulatore con il contenuto di una locazione della memoria esterna. Il risultato dell'addizione viene posto in accumulatore.
In questo gruppo di istruzioni sono comprese la somma e la sottrazione tra due registri a 16 bit della CPU.

- ## Istruzioni di rotazione e scorrimento

Il gruppo delle istruzioni di Rotate e Shift (rotazione e scorrimento) permette di far ruotare verso destra o verso sinistra il contenuto di un qualsiasi registro o locazione di memoria, con o senza l'utilizzo del bit di riporto (Carry), in modo aritmetico o logico.Inoltre il nibble (gruppo di 4 bit) di ordine più basso posto in accumulatore può essere fatto ruotare a destra o a sinistra congiuntamente a due altri nibble posti in una qualunque posizione di memoria.

- ## Istruzioni di manipolazione del bit

Le istruzioni di manipolazione del bit permettono di porre ad 1 logico (set), e 0 logico (reset), o di esaminare (test) un qualsiasi bit dell'accumulatore, di un registro di uso generale, o di una posizione qualunque di memoria, mediante una singola istruzione.

- ## Istruzioni di salto, chiamata e ritorno

Le istruzioni di Jump (salto), Call (chiamata) e di ritorno sono utilizzate per passare da una istruzione di programma ad un'altra. Questo gruppo utilizza varie tecniche per ottenere il nuovo valore del program counter a partire da una specifica posizione di memoria.

- ## Ingresso/uscita

Le istruzioni di ingresso/uscita dello Z80 consentono molte possibilità di trasferimento tra la memoria esterna ed i registri di uso generale della CPU ed i dispositivi esterni di ingresso/uscita. In qualunque caso, il numero della porta é fornito dagli 8 bit meno significativi del bus degli indirizzi durante il ciclo di I/O. Per esempio, una istruzione consente di indicare questo numero di porta come secondo byte dell'istruzione, mentre un'altra consente di indicarlo mediante il contenuto del registro C.

La PROGRAMMAZIONE CON LO Z-80 consiste nella scrittura , in memoria programma , di tutte le istruzioni che si rendono necessarie per ottenere un determinato funzionamento.
Ogni istruzione è idealmente composta da :

-**INDIRIZZO** ovvero la locazione di memoria che conterrà il codice operativo mnemonico dell'istruzione;

-**CODICE OPERATIVO MNEMONICO**, detto anche codice macchina, indica l'operazione che la CPU deve compiere;

-**OPERANDO**, può assumere valori e significati diversi a secondo del tipo d'istruzione;

-**COMMENTO**, aggiunta di frasi e spiegazioni delle soluzioni adottate solo per rendere meglio comprensibili i programmi scritti. Queste frasi sono totalmente ignorate dall'assemblatore durante la fase di traduzione

Una volta scritto il programma sorgente, esso viene tradotto in programma oggetto, ovvero in una serie d'istruzioni scritte in linguaggio macchina (codice binario).

Tuttavia il programma scritto in esadecimale è ancora poco leggibile, pertanto per rendere più agevole il compito del programmatore le case costruttrici forniscono per ogni istruzione un codice simbolico o mnemonico, formato da due, tre o quattro lettere che richiamano il tipo di operazione svolta da quella determinata istruzione.

L'insieme delle istruzioni espresse mediante i codici mnemonici costituiscono il linguaggio assembly del microprocessore.

Esempio: **01111000** (che potrebbe essere l'istruzione di caricamento dell'accumulatore con il contenuto del registro B) e una istruzione scritta in codice binario, sicuramente è più semplice scriverla in codice esadecimale: **78**, ancora più semplice in codice mnemonico o assembly:

LD A,B

BINARIO	HEX
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

ISTRUZIONI DI CARICAMENTO A 8 BIT

Sono utilizzate per trasferire un dato ad 8 bit da un registro sorgente ad un registro destinazione, oppure da una locazione di memoria a un registro della CPU e viceversa.

Le lettere **LD**, sono state utilizzate per definire i codici mnemonici delle istruzioni di caricamento.

Quindi ogni istruzione di caricamento ha la seguente forma sintetica:

LD destinazione, sorgente

La sorgente indica la locazione di memoria o il registro contenente il dato che deve essere trasferito, mentre la destinazione è la locazione di memoria o il registro sul quale deve essere copiato (trasferito) il dato.

Ad esempio l'istruzione **LD A,B**

Trasferisce nel registro accumulatore **A** il dato contenuto nel registro sorgente **B**

Al termine dell'istruzione il registro accumulatore **A** e il registro **B** contengono lo stesso dato.

Altro Esempio: **LD (4000 H),A**

Trasferisce nella locazione di memoria di indirizzo 4000 H il contenuto del registro A. Da ricordare che l'indirizzo della locazione di memoria nella quale è contenuto il dato, deve essere racchiuso tra parentesi tonda, questo indica che il dato è contenuto in una locazione di memoria.

Questa è un'istruzione di scrittura in memoria

METODI DI INDIRIZZAMENTO DELLO Z80

Indirizzamento diretto tra registri

Operano il trasferimento di un dato dal registro (sorgente) al registro (destinazione) mantenendo inalterato il contenuto della sorgente.

LOAD DEST, SORG **(trasferisci nel registro r il dato memorizzato in r')**
LD r , r'

E ' un esempio di trasferimento dati tra registri perché la sorgente e la destinazione sono due registri interni della CPU

Ad esempio l'istruzione **LD A,B**

Trasferisce nel registro accumulatore **A** il dato contenuto nel registro sorgente **B**

Al termine dell'istruzione il registro accumulatore **A** e il registro **B** contengono lo stesso dato.

REGISTRO A PRIMA	REGISTRO B PRIMA
XX	09
REGISTRO A DOPO	REGISTRO B DOPO
09	09

Se andiamo a cercare l'operando dell'istruzione, verrà fuori:

Locazione	Codice Mnemonico	Codice oggetto (operando)	Commento
F000	LD A,B	78	Trasferisci nel registro A il dato 09 memorizzato in B

Indirizzamento diretto

Operano il trasferimento tra il registro ACCUMULATORE e la MEMORIA. Una Locazione di memoria è dunque sorgente o destinatario. Notare l'uso delle parentesi: il dato sorgente è contenuto in una locazione di memoria ,non si trasferisce l'indirizzo ma il dato contenuto in quell'indirizzo di memoria.

LOAD DEST, SORG **(trasferisci nel registro A il dato memorizzato nella locazione di memoria "hhll")**

LD A , (hhll)

hhll è una stringa di 16 bit, in questo caso un indirizzo

Ad esempio l'istruzione **LD A,(F000)** **(lettura in memoria)**

E' un esempio di trasferimento di un dato in modo diretto perché l'istruzione contiene l'indirizzo della locazione di memoria nella quale è contenuto il dato che deve essere trasferito. **(in questo esempio il dato è 09H)**

REGISTRO A PRIMA	F000 PRIMA
XX	09
REGISTRO A DOPO	F000 DOPO
09	09

N.B. si deve caricare prima il dato 09 nella **locazione F000H**

Locazione	Codice Mnemonico	Codice oggetto (operando)	Commento
FB00	LD A,(F000)	3A 00 F0	Trasferisci nel registro A il dato 09 memorizzato nella locazione di memoria F000

Invece **LD (F000),A** **(scrittura in memoria)**

è un altro esempio di trasferimento di un dato in modo diretto perché trasferisce nella locazione di memoria di indirizzo F000 il dato memorizzato nel registro accumulatore

Indirizzamento immediato

Caricano un dato in un registro (destinazione).

Il dato su cui operare (sorgente) risiede nella memoria programma in un byte di istruzione

LOAD DEST, SORG **(trasferisci il dato nel registro r)**

LD r , D8

D8 è un dato simbolico di un byte

r *un registro simbolico a 8 bit*

Es :

LD A,05H

Locazione	Codice Mnemonico	Codice oggetto (operando)	Commento
F000	LD A,05H	3E 05	Trasferisci nel registro A il dato 05H

Tale istruzione occupa solo due locazioni di memoria: la prima è riservata al codice oggetto o operando dell'istruzione (3E) e la seconda, al dato che deve essere trasferito nel registro A (05).

Il caricamento immediato come si può vedere occupa una minore quantità di memoria e richiede minore tempo di esecuzione rispetto a quello diretto.

Non sempre però si conosce a priori il dato che deve essere trasferito perché esso deriva da elaborazioni e, nella maggior parte dei casi, è memorizzato in locazioni di memoria.

Indirizzamento indiretto tramite registri

Consente il trasferimento dati tra registri e memoria, in quanto da una parte sono coinvolti tutti i registri generali e dall'altra il dato in memoria è individuato attraverso un indirizzo in tre possibili coppie di registri (BC, DE, HL).

Anche in questo caso la sorgente è racchiusa entro parentesi tonde per indicare che il dato da trasferire è contenuto in una locazione di memoria il cui indirizzo, è memorizzato in una coppia di registri.

LOAD DEST, SORG

LD r , (HL)
(BC)
(DE)

(trasferisce nel registro r, il dato contenuto nella locazione di memoria il cui indirizzo è memorizzato nella coppia di registri HL o BC o DE.)

Esempio: **LD A, (HL)**

N.B. bisogna caricare prima il dato nella locazione F000

Locazione	Codice Mnemonico	Codice oggetto (operando)	Commento
FB00	LD HL,F000	21 00 F0	Trasferisce l'indirizzo della locazione di memoria, F000, nella coppia di registri HL
FB03	LD A, (HL)	7E	Trasferisco nel registro A il dato contenuto nella locazione di memoria il cui indirizzo è memorizzato nella coppia di registri HL.

	PRIMA	
H	L	A
F0	00	XX
	DOPO	
H	L	A
F0	00	07

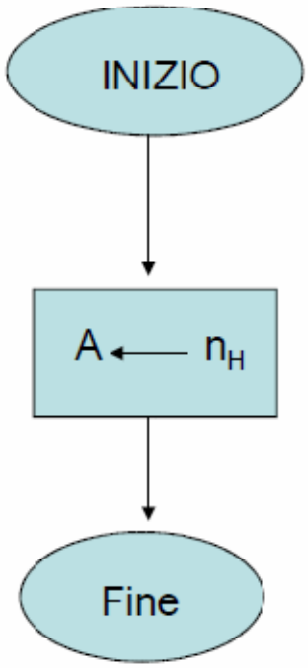
In questo esempio il dato è 07 H

La procedura da seguire per la stesura di un programma in assembler per lo Z80 è analoga a quella utilizzata per un qualunque linguaggio di programmazione. Si tratta cioè di analizzare il problema, individuarne un algoritmo risolutivo e tradurlo nel linguaggio di programmazione prescelto.
Per la codifica in Assembler Z80 è necessario:

- fissare l'indirizzo di mermoria di inizio programma;
- scrivere il listato in codice mnemonico;
- tradurre il listato in codice oggetto;
- riportare le note di commento.

Esempio 1: Caricamento di un byte di generico valore n_H nel registro accumulatore A.

Il Flow-Chart del programma è il seguente:



Dalla carta di riferimento si ricava che l'istruzione che esegue tale operazione (operazione simbolica: A← n_H) ha codice mnemonico:

LD r,n_H

e codice operativo:

00 r 110

con **r** registro generico

Il codice operativo dell'istruzione si ottiene sostituendo nella combinazione:

00r110

il valore di **r** corrispondente al registro **A**, vale a dire **111**.

Pertanto si ottiene:

0011	1110	binario
↓	↓	
3	E	esadecimale

Supponendo che il programma sia caricato in memoria a partire dalla locazione di memoria 0100_H, e posto ad esempio: $n_H=30_H$

Locazione	Codice Mnemonico	Codice oggetto (operando)	Commento
0100	LD A,30_H	3E 30_H	Trasferisci nel registro A il dato 30_H

Il tempo necessario per eseguire questa istruzione , composta da 2 cicli macchina (CM1 + lettura in memoria programma del dato 30H) e 7 cicli CK, nell' ipotesi di F_{CK} = 2 [MHz], sarà di 7*0,5 = 3,5 [µs]

*Esempio 2: Esecuzione della somma dei byte contenuti nei registri **B** e **C**, riportando il risultato nel registro accumulatore **A***

Poiché non è possibile sommare i contenuti dei registri B e C e porre il risultato in A mediante una unica istruzione, si può impostare il Flow-chart seguente:

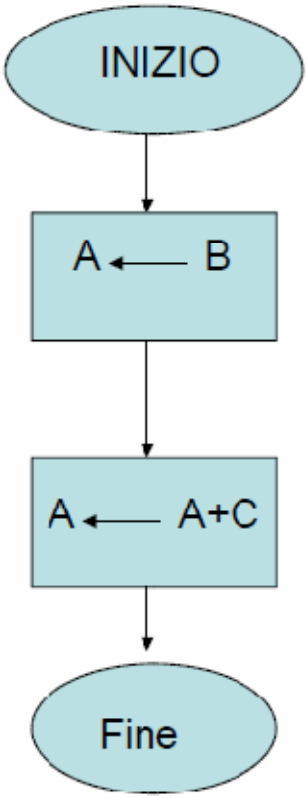
La prima operazione consiste nel trasferire il contenuto del registro B nell'accumulatore A.

LD r,r'

e codice operativo:

01 r r'

con **r ed r'** registro generico, tale operazione richiede 4 cicli macchina.

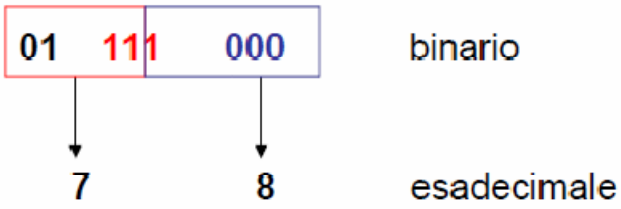


Il codice operativo dell'istruzione si ottiene sostituendo nella combinazione:

01rr'

il valore di **r** corrispondente al registro **A**, vale a dire **111**, il valore di **B** ossia **000**

Pertanto si ottiene il codice operativo:



La seconda operazione viene svolta dall'istruzione aritmetica:

ADD A, r

operazione simbolica

A ← A + r

codice operativo **10000 r**

con 4 periodi di clock per l'esecuzione.

Con tale istruzione il contenuto del registro r viene sommato a quello dell'accumulatore.

Ponendo: **r = C**, (dove **C** vale **001**)il codice operativo risulta:

10000r = 1000 0001 = 8 1 esadecimale

Il programma caricato ad esempio a partire dalla cella di memoria di indirizzo 0200, risulta:

Locazione	Codice Mnemonico	Codice oggetto (operando)	Commento
0200	LD A,B	78	Trasferisci nel registro A il contenuto di B
0201	ADD A,C	81	Somma il contenuto di C con quello di A, risultato in A

Le 2 istruzioni richiedono ciascuna 1 ciclo macchina da 4 cicli CK , per cui la durata è :

8* 0,5 = 4 [µs]