MASTERCODE 464

MACHINE CODE SYSTEM EDITOR
ASSEMBLER DISASSEMBLER &
MONITOR DI PROGRAMMAZIONE
PER AMSTRAD CPC 464

MASTERCODE 464

MICRODATA 1985

MASTERCODE

ASSEMBLER + EDITOR

INTRODUZIONE

MASTERCODE ASSEMBLER e' il migliore assembler per l'AMSTRAD CPC 64 disponibile sul mercato.
L'ASSEMBLER puo'essere caricato in qualunque zona della memoria RAM permettendo cosi' la compilazione di qualsiasi programma.
La lunghezza del codice macchina e' circa 7K e viene utilizato uno stack interno. Il testo dei Vostri programmi viene memorizzato subito dopo la fine dell'ASSEMBLER espandendosi verso l'alto, per cui conviene caricare l'ASSEMBLER nelle zone basse di memoria per poter avere a disposizione piu' spazio possibile.

CARICAMENTO

Per caricare l'ASSEMBLER in memoria digitate: RUN" [ENTER]
e premete [PLAY] sul registratore.
Vi sara' quindi chiesto l'indirizzo di partenza a cui potete rispondere con un numero compreso tra 1000 e 30000.
Se volete avere in memoria contemporaneamente l'ASSEMBLER e il MONITOR dovete PRIMA caricare il MONITOR ad un indirizzo alto (per esempio 30000) e POI l'ASSEMBLER ad un indirizzo basso (per esempio 1000).
Se ritornate al BASIC tramite l'opzione B dell'EDITOR e quindi volete tornare di nuovo nell'ASSEMBLER dovete digitare CALL xxxx+2 se volete cancellare il testo presente in memoria (Cold Start), oppure CALL xxxx+4 se volete che il testo venga mantenuto in memoria (Warm Start), dove xxxx e' l'indirizzo di partenza da cui avevate caricato l'ASSEMBLER.

L'ASSEMBLER

Dopo che avete inserito un testo attraverso i comandi dell'EDITOR (vedi oltre) potete chiedere la compilazione attraverso il comando A.
La compilazione traduce nel corrispondente codice macchina le istruzioni Assembler da voi inserite nel testo.
Durante questa fase viene inoltre controllata la corretta sintassi del testo e nel caso venga incontrato un errore la compilazione verra' momentaneamente sospesa mostrando il punto in cui si e' verificato l'errore ed il codice numerico dell'errore stesso relativo alla tabella riportata in appendice.
In caso di errore premete E per tornare all'EDITOR o un altro tasto per continuare la compilazione.

NOTA: Le istruzioni Assembler devono essere inserite esclusivamente in maiuscolo.

Quando usate l'ASSEMBLER attraverso il comando A dell'EDITOR vi sara' chiesto: 'Dim. LABEL ?'
Dovete inserire un numero decimale che rappresenta la dimensione in bytes della tabella delle LABEL. Se premete solamente [ENTER] sara' assunto un valore normalmente accettabile per la dimensione del testo. Se invece usate l'opzione di inclusione oppure caricate un testo generato dal MASTERCODE MONITOR, probabilmente dovrete inserire una dimensione piu' grande del normale.
Quindi vi sara' chiesto: 'Opzioni ?'
Inserite un numero decimale corrispondente alla tabella seguente.

Opzione 1: Produce una tabella coi valori delle LABEL alla fine del testo.
Opzione 2: Non genera il corrispondente codice macchina.
Opzione 4: Non produce il listato del programma.
Opzione 8: Produce il listato sulla stampante.
Opzione 16: Dispone il codice macchina, se generato, dopo la tabella delle LABEL. Tuttavia il contatore di indirizzo e' aggiornato in base all' ORG, in modo che il codice macchina puo' essere sistemato in una sezione di memoria ma destinato a girare in qualsiasi altra zona.
Opzione 32: Non esegue alcun controllo sulla destinazione del codice macchina (utile per velocizzare).

Se desiderate piu' di un'opzione inserite il numero risultante dalla somma delle varie opzioni.
Ad esempio l'opzione 36 produce una notevole velocita' in quanto spegne ogni controllo e non produce il listato.

Il programma viene compilato in due passaggi consecutivi: nel primo viene controllato il testo e formata la tabella delle LABEL, nel secondo viene generato il corrispondente codice macchina e il listato se richiesto.
Il listato ha normalmente il seguente formato:

```
                C000 210100     25  label   LD    HL,1

colonna:        1    6          15  21      28    33
```

Alla colonna 1 abbiamo il valore del contatore di indirizzo e cioe' l'indirizzo di memoria a partire dal quale verra' assemblata l'istruzione; alla colonna 6 il codice esadecimale corrispondente all'istruzione presente nella linea; alla colonna 15 il numero della linea; alla colonna 21 il nome della LABEL se presente; alla colonna 28 il mnemonico dell'istruzione e alla colonna 33 l'operando.
Se non vi sono errori, dopo i due passaggi verra' specificato lo spazio ancora libero a disposizione delle LABEL e se si e' fatto uso della direttiva ENT verra' specificato l'indirizzo dal quale si puo' eseguire il programma .
NOTA: potete sospendere il listato premendo un tasto. Per continuare premete di nuovo un tasto diverso da E.

CARATTERI RISERVATI

; dopo questo simbolo qualsiasi carattere viene ignorato e cioe' trattato come commento.

* dopo questo simbolo l'ASSEMBLER si aspetta un'opzione del compilatore (vedi oltre).

Tutti gli spazi superflui all'interno di una linea verranno eliminati per una migliore leggibilita' del listato.

LABEL
Una LABEL puo' essere formata da qualsiasi numero di caratteri anche se vengono riconosciuti solo i primi 6.
Una LABEL deve iniziare con un carattere non numerico e non puo' essere uguale a una delle seguenti parole riservate:

```
A  B  C  D  E  H  L  I  R  $  Z  M  P
AF AF'BC DE HL IX IY SP NC NZ PE PO
```

Alcuni esempi di LABEL valide:
LOOP , loop , una-label-lunga , L[1] , a , LDIR , due^5

CONTATORE DI INDIRIZZO

Il contatore di indirizzo viene aggiornato dopo ogni istruzione e puo' essere posizionato al valore desiderato attraverso la direttiva ORG. Il simbolo $ puo' essere usato per riferirsi al corrente valore del contatore di indirizzo; per esempio LD HL,$+5 significa carica il registro HL con l'indirzzo di partenza di questa istruzione maggiorato di 5.

ESPRESSIONI

Come operandi sono possibili i seguenti termini:

```
          COSTANTE DECIMALE  -> 1029
       COSTANTE ESADECIMALE  -> #E0F1
         COSTANTE BINARIA    -> %10001101
                  LABEL      -> L1029
CONTATORE DI INDIRIZZO       -> $
          COSTANTE ASCII     -> "A"
```

e i seguenti operatori:

```
                ADDIZIONE  -> +
             SOTTRAZIONE   -> -
             AND LOGICO    -> &
              OR LOGICO    -> @
             XOR LOGICO    -> !
MOLTIPLICAZIONE INTERA     -> *
       DIVISIONE INTERA    -> /
FUNZIONE MOD (A MOD B=A-(A/B)*B) -> ?
```

E' possibile inoltre qualsiasi combinazione di termini e operatori ricordando pero' che non esiste priorita' nelle operazioni e cioe' i calcoli vengono eseguiti sequenzialmente da sinistra a destra. Se un'espressione e' racchiusa tra parentesi viene valutata come un indirizzo di memoria a 16 bit.

DIRETTIVE ASSEMBLER

Vengono riconosciuti i seguenti pseudomnemonici:

ORG espressione

posiziona il contatore di indirizzo al valore dell'espressione. Se le opzioni 2 e 16 non sono inserite e il codice generato andrebbe a corrompere l'ASSEMBLER allora viene sopesa l'esecuzione della compilazione e i comandi ritornano all'EDITOR.

EQU espressione

deve essere preceduta da una LABEL e assegna alla LABEL il valore dell'espresione.

DEFB espressione, espressione, ...

ciascuna espressione deve avere un valore tra 0 e 255 e a partire dal corrente indirizzo verrano inseriti i valori delle espressioni e il contatore di indirizzo incrementato conseguentemente.

DEFW espressione, espressione, ...

come DEFB tranne che le espressioni sono valutate a 16 bit (valori tra 0 e 65535) e il contatore di indirizzo viene incrementato di 2 per ogni espressione. (i valori vengono inseriti con il byte meno significativo che precede il piu' significativo).

DEFS espressione

aumenta il contatore di indirizzo del valore dell'espressione (per riservare parti di memoria).

DEFM "s"

inserisce la stringa s (codici ASCII) a partire dal corrente indirizzo e incrementa di conseguenza il contatore di indirizzo.

ENT espressione

posiziona l'indirizzo di esecuzione (attraverso il comando R dell'EDITOR) al valore dell'espressione.

PSEUDO-MNEMONICI CONDIZIONALI

Servono ad includere o ad escludere la compilazione di certe sezioni del testo.

IF espressione

se il risultato dell'espressione e' Ø allora la compilazione e' disattivata fino a che non viene incontrato un ELSE o un END, altrimenti la compilazione continua normalmente.

ELSE

attiva e disattiva alternatamente la compilazione, cioe' se viene incontrato mentre la compilazione e' attiva allora la disattiva, altrimenti la riattiva.

END

attiva in ogni caso la compilazione.

IMPORTANTE: NON nidificate MAI questi comandi .

COMANDI ASSEMBLER

I seguenti comandi devono essere inseriti all'INIZIO della linea e il resto della linea viene ignorato.

*E

(eject) manda tre linee vuote sullo schermo o sulla stampante (utile per separare dei blocchi).

*Hs

la stringa s viene stampata come titolo dopo ciascun eject (*E). *H viene automaticamente seguito da un *E.

*S

ferma il listato durante la compilazione (anche dopo un *L-). La compilazione puo' essere continuata premendo un tasto. *S non ha alcun effetto quando l'output e' diretto verso la stampante..

*L-

ferma la stampa del listato sullo schermo o sulla stampante fino a che non viene incontrato *L+.

*L+

riattiva la stampa del listato.

*D+

converte in decimale la stampa del contatore di indirizzo.

*D-

ritorna a stampare il contatore di indirizzo in esadecimale.

*T+

salva su nastro un blocco di codice macchina durante la compilazione.

*T-

interrompe l'invio del codice macchina al nastro.

*F s

Questo comando serve per assemblare delle routines in codice macchina, precedentemente salvate su nastro attraverso il comando P dell'EDITOR all'interno del vostro programma. Se durante la compilazione viene incontrato questo comando Vi sara' chiesto di avviare il nastro per caricare la routine precedentemente salvata con il nome specificato dalla stringa s, oppure la prima che incontra se omettete il nome. Questo processo viene ripetuto una volta per ogni passaggio durante la compilazione, quindi conviene avere a disposizione due copie di fila della routine per evitare di dover riavvolgere il nastro al secondo passaggio.

NOTA: Se la compilazione e' stata disattivata attraverso un IF o un ELSE non verranno riconosciuti i sopracitati comandi.

L'EDITOR

INTRODUZIONE

In questa sezione verranno usate le seguenti abbreviazioni:

ENTER [ENTER]. Per inserire un testo o un comando.

CC [CTRL/C] Per sospendere la corrente linea.

CH [DELETE] Per cancellare l'ultimo carattere.

CI [TAB] Per avanzare alla seguente posizione TAB.

CX [CTRL/X] Per cancellare l'ultima linea battuta.

L'EDITOR e' disponibile quando appare a sinistra del video il simbolo '>'.
In risposta potete inserire un comando nella forma:

C N1,N2,S1,S2 seguito da ENTER

C e' il comando da eseguire.
N1 e N2 sono numeri tra 0 e 32767.
S1 e S2 sono stringhe fino a 20 caratteri.

In certi comandi, come vedremo oltre, alcuni campi non sono richiesti o sono opzionali.
I comandi possono essere inseriti sia in maiuscolo che in minuscolo.

COMANDI DELL'EDITOR

INSERIMENTO DEL TESTO

Il testo puo' essere inserito battendo il numero di linea, uno spazio, e quindi il testo desiderato; oppure attraverso il comando I.
Se battete un numero di linea seguito da ENTER la linea corrispondente verra' cancellata dalla memoria.
Durante l'inserimento di un testo potete usare i controlli specificati nell'introduzione.

I n,m

Inserendo questo comando entrerete nel modo di inserimento automatico. Le linee verranno inserite partendo dalla linea con un incremento di m. Per uscire usate il controllo CC.

LISTATI

L n,m

Fornisce il listato delle linee comprese tra n e m inclusi.
Per listare l'intero programma usate soltanto L.
Dopo che avrete premuto ENTER verranno listate un certo numero di linee; Premete quindi un tasto qualunque per proseguire oppure usate il controllo CC per ritornare all'EDITOR.

K n

Stabilisce il numero di linee mostrate alla volta prima che il listato venga sospeso in attesa di premere un tasto per continuare o CC per smettere.

FASE DI EDITING

D n,m

Cancella tutte le linee comprese tra n e m inclusi.

M n,m

Esegue una copia della linea n con nuovo numero di linea m.

N n,m

Esegue la rinumerazione di tutto il listato con nuova base n e incremento m.

F n,m,f,s
Viene eseguita una ricerca tra le linee comprese tra n e m per trovare la stringa di ricerca f. Trovata la stringa viene editata la linea che la contiene; potete quindi usare i comandi relativi al modo di EDIT (vedi oltre) per sostituire la stringa f con la stringa s di sostituzione oppure proseguire la ricerca.

E n (modo di EDIT)
Edita la linea con numero n entrando nel modo di EDIT nel quale sono disponibili i seguenti comandi:

' ' (spazio)-incrementa il cursore di una unita' fino alla fine della linea.
CH (DELETE) -decrementa il cursore di una unita' fino all'inizio della linea.
CI -sposta il cursore in avanti fino alla prossima psizione TAB.
ENTER -per inserire la linea dopo averla modificata.
Q -per dimenticare ogni modifica e reinserire la linea come era originalmente.
R -per riavere in fase di EDIT la linea come era originalmente.
L -per mostrare tutta la linea.
K -per cancellare il carattere alla corrente posizione del cursore.
Z -per cancellare la linea dalla posizione del cursore in avanti.
F -per cercare la prossima occorrenza della stringa di ricerca (definita dal comando F n,m,f,s).
S -per sostituire la stringa di ricerca f con la stringa di sostituzione s (definite dal comando F n,m,f,s) e proseguire la ricerca.
I -per inserire del testo alla corrente posizione del cursore (ENTER per terminare). In questa fase il cursore mostrera il simbolo '*'.
X -per inserire del testo al fondo di una linea.
C -per sostituire del testo dalla corrente posizione del cursore in avanti (ENTER per terminare). In questa fase il cursore mostrera' il simbolo '+'.

COMANDI PER LA CASSETTA

P n,m,s
Il testo compreso tra la linea n e la linea m viene salvato su nastro con il nome definito dalla stringa s. Assicuratevi che entrambi i tasti [REC] e [PLAY] siano pigiati prima di eseguire.

G,,s
Viene cercato su nastro il testo definito dalla stringa s e quando viene incontrato sara' caricato in memoria. Se la stringa s e' vuota verra' caricato il primo testo trovato. Se un testo e' gia' presente in memoria quello caricato da nastro verra' inserito in coda e tutto il testo verra' rinumerato con base 1 e incremento di 1.
NOTA: con questo comando e' impossibile caricare un testo salvato con il comando T dell'ASSEMBLER.

V,,s
Viene verificato sul nastro il testo salvato con nome definito dalla stringa s.
Il testo su nastro viene comparato con quello presente in memoria.
Se il testo risulta uguale apparira' il messaggio 'O.K.', altrimenti verra' segnalato l'errore e dovrete riprovare a salvare il testo nuovamente.

O,,s
Salva il codice macchina prodotto dalla compilazione con inizio definito da ORG e titolo definito dalla stringa s. Se piu' di un ORG viene specificato nel programma sara' salvato solo il codice macchina relativo all'ultimo ORG incontrato.

T n
Cambia la velocita' di trasferimento dati su nastro.
T seguito da [ENTER] seleziona la velocita' bassa, mentre T seguito da un numero maggiore di 0 seleziona la velocita' piu' alta.

COMANDI PER IL DISPLAY

W
Questo comando, usato alternativamente, vi permette di passare dal display a 40 colonne al display a 80 colonne e viceversa.
Se utilizzate il monitor AMSTRAD a fosfori verdi potete tranquillamente lavorare a 80 colonne mentre col monitor AMSTRAD a colori conviene selezionare il display a 40 colonne.

COMANDI PER LA COMPILAZIONE E L'ESECUZIONE

A
Compila il testo dalla prima linea (vedi la sezione riguardante l'ASSEMBLER).

R
Se la compilazione viene terminata senza errori e' la direttiva ENT e' stata usata correttamente allora potete eseguire il corrispondente codice macchina all'interno dell'EDITOR con questo comando.
NOTA: Al termine del codice macchina deve essere presente l'istruzione RET per poter ritornare correttamente all'EDITOR.
Inoltre la direttiva ENT non ha alcun effetto se e' stata usata l'opzione 16 durante la compilazione.

COMANDI VARI

B
Ritorna al BASIC dell'AMSTRAD. Per ritornare nell'EDITOR eseguite un warm start o un cold start (vedi l'introduzione dell'ASSEMBLER).

S,,d
Permette di cambiare il separatore degli argomenti nei comandi.
All'inizio il separatore viene rappresentato da una virgola.
Utilizzando questo comando verra' assunto come separatore il carattere d inserito all'interno del comando.
Il separatore non puo' essere uno spazio.

C
Mostra sullo schermo il separatore corrente, i numeri N1 e N2 e le stringhe S1 e S2 attualmente memorizzati. E' utile controllarli prima di inserire un comando senza gli argomenti , nel qual caso verranno assunti come argomenti quelli attualmente memorizzati.

Z n,m
Manda alla stampante il testo compreso tra le linee n e m. Se inserite solo W sara' stampato l'intero testo (come per il listato su video dovrete premere un tasto per continuare dopo che saranno state stampate un certo numero di linee - vedi comando K).

J
Passa il controllo al MONITOR (se presente).

X
Mostra in decimale l'indirizzo di partenza e di fine del file di testo presente in memoria. Utile per controllare la quantita' di memoria ancora a disposizione.
L'ASSEMBLER memorizza in una locazione di memoria il valore dell'indirizzo di fine testo. Se volete quindi inserire in memoria un file di testo generato dal MONITOR dovete prima annotarvi l'indirizzo di partenza fornitovi dal comando X, quindi generare il file partendo da questa locazione, annotarvi l'indirizzo di fine testo fornitovi dal MONITOR e inserirlo con due POKE agli indirizzi xxxx+7 (byte meno significativo) e xxxx+8 (dove xxxx e' l'indirizzo da cui e' stato caricato l'ASSEMBLER). Per finire entrate nell'EDITOR con un warm start.
A questo punto il testo puo' essere comodamente manipolato attraverso i comandi dell'EDITOR.

UN ESEMPIO DI UTILIZZO DEL'EDITOR

Supponiamo abbiate battuto il seguente programma attraverso l'uso del comando I 10,10:

```
 10 *h    NUMERI CASUALI A SEDICI BIT
 20
 30 ;INPUT :HL contiene il precedente numero casuale.
 40 ;OUTPUT:HL contiene il nuovo numero caduale.
 50
 60 Inizio PUSH AF      ;salva i registri
 70 PUSH BC
 80         PUSH HL
 90         ADD  HL,HL  ;*2
100         ADD  HL,HL  ;*4
110         ADD  HL,HL  ;*8
120         ADD  HL,HL  ;*16
130         ADD  HL,HL  ;*32
140         ADD  HL,HL  ;*64
150         PIP  BC     ;numero precedente
160         ADD  HL,DE
170         LD   DE,41
180         ADD  HL,DE
190         POP  BC     ;risistema i registri
200         POP  AF
210         REY
```

Il programma contiene i seguenti errori:
Linea 10 : h minuscola invece di maiuscola.
Linea 40 : caduale invece di casuale.
Linea 70 : PUSH BC inizia nel campo delle LABEL.
Linea 150: PIP invece di POP.
Linea 160: volete aggiungere un commento (non e' proprio un errore).
Linea 210: REY invece di RET.

Inoltre dovete aggiungere 2 linee di ADD HL,HL trà 140 e 150 e dovete cambiare DE con BC dalla linea 160 alla 180.

Per fare tutti questi cambiamenti dovete procedere come segue:

```
E10 ENTER                  poi (space) C H ENTER ENTER
F40,40,caduale,casuale     ENTER  poi S
E70 ENTER                  poi I (7 spazi) ENTER ENTER
I142,2                     poi (space) ADD HL,HL    ;*128  ENTER
                           poi (space) ADD HL,HL    ;*256  ENTER
                           poi cc per uscire
F150,150,PIP,POP ENTER     poi S
E160 ENTER                 poi X ;*257+41 ENTER ENTER
F160,180,DE,BC ENTER       poi S S S
E210 ENTER                 poi CI CI C T ENTER ENTER
N10,10                     per rinumerare il testo
```

ERRORI E LORO SIGNIFICATO

ERRORE 1	Errore nel contesto di questa linea.
ERRORE 2	Mnemonico non riconosciuto.
ERRORE 3	Istruzione scritta in maniera errata.
ERRORE 4	Simbolo definito piu' di una volta.
ERRORE 5	Questa linea contiene un carattere illegale.
ERRORE 6	Uno degli operandi in questa linea e' illegale.
ERRORE 7	Un simbolo in questa linea e' una parola riservata.
ERRORE 8	Registri sbagliati.
ERRORE 9	Troppi registri nella linea.
ERRORE 10	Un'espressione che doveva essere valutata a 8 bit e' stata valutata a piu' di 8 bit.
ERRORE 11	Le istruzioni JP (IX+n) e JP (IY+n) sono illegali.
ERRORE 12	Errore in una direttiva dell'ASSEMBLER.
ERRORE 13	Riferimento in avanti illegale. Per esempio assegnare a una LABEL un valore non ancora definito.
ERRORE 14	Divisione per 0.
ERRORE 15	Overflow in un'operazione di moltiplicazione.

ORG !
L'indirizzo di partenza del codice macchina corromperebbe il compilatore. I comandi ritornano all'EDITOR

Spazio Label !
Capita durante il 1' passaggio se non e' stata specificata una quantita' sufficiente di memoria per le Label. I comandi ritornano all'EDITOR. Ripetere la compilazione specificando una maggiore quantita' di memoria per le Label.

Memoria !
Non vi e' piu' spazio in memoria per poter operare con sicurezza. Salvate parte del testo e compilate in fasi successive.

ATTENZIONE Label assente
Questo non e' proprio un errore ma significa che una Label e' stata dichiarata ma non e' quindi stata usata all'interno del programma.

INTRODUZIONE

Per caricare il monitor digitate: RUN" [ENTER]
e premete [PLAY] sul registratore.
Vi sara' quindi chiesto l'indirizzo di partenza a cui dovete rispondere con un numero compreso tra 1000 e 30000.
Una volta caricato il programma potete in qualunque momento ritornare al BASIC premendo [CTRL/X]. Se quindi desiderate ritornare al MONITOR dovrete inserire da BASIC: CALL xxxx+2 , dove xxxx e' l'indirizzo di partenza da cui avevate caricato il programma.

Dopo il messaggio di copyright apparira' sul video una tabella contenente il corrente indirizzo di memoria (inizialmente 0) e i contenuti di tutti i registri dello Z 80 oltre a una sezione di 24 bytes di memoria centrata attorno al corrente indirizzo segnalato dai simboli '>' e '<'.
I comandi vengono inseriti tramite tastiera e non e' necessario premere ENTER dato che il comando viene eseguito direttamente alla pressione del tasto.
Molti comandi richiedono degli input che dovranno essere inseriti ESCLUSIVAMENTE in esadecimale (tranne che per la conversione di un numero decimale in esadecimale).
Se alla fine del numero aggiungete un '-' il valore sara' assunto come negativo (nella forma del complemento a 2);per esempio 1800- da' E800. Se inserite piu' di 4 cifre saranno considerate solo le ultime 4.

NOTA: Potete in qualunque momento passare all'ASSEMBLER (se presente in memoria) premendo [CTRL] J (Vedi l'introduzione dell'ASSEMBLER.

COMANDI DISPONIBILI

[CTRL] D
Converte la base in cui sono mostrati gli indirizzi di memoria da esadecimale a decimale e vice-versa.

[CTRL] A
Mostra una pagina di disassemblato dal corrente indirizzo di memoria in avanti. Premere qualsiasi tasto per continuare il disassemblato o [CTRL] A per smettere.

'>' (Cursore a destra)
Incrementa il corrente indirizzo di memoria di 1.

'<' (Cursore a sinistra)
Decrementa il corrente indirizzo di memoria di 1.

'^' (Cursore in alto)
Decrementa il corrente indirizzo di memoria di 8 (utile per retrocedere velocemente).

' ' (Cursore in basso)
Incrementa il corrente indirizzo di memoria di 8 (utile per avanzare velocemente).

S
Posiziona il corrente indirizzo di memoria al corrispondente indirizzo attualmente sullo stack (SP). Utile per cercare l'indirizzo di ritorno di una routine chiamata.

G
Ricerca in memoria di una stringa.
Quando usate questo comando apparira' il simbolo ':' al quale dovete rispondere col primo byte che volete cercare seguito da ENTER e quindi inserire i successivi bytes fino a definire l'intera stringa. Quando avete finito di inserire i dati premete semplicemente ENTER per iniziare la ricerca dal corrente indirizzo in avanti.
Per esempio se volete cercare due byte consecutivi coi valori #3E e #FF a partire dall' indirizzo #8000 dovete premere:

```
M 8000 ENTER    posiziona l'indirizzo a #8000
G 3E ENTER      inserisce il primo byte della stringa
FF ENTER        inserisce il secondo byte
ENTER           inizia la ricerca
```

Se la stringa viene trovata il corrente indirizzo di memoria sara' aggiornato in base al risultato della ricerca.
Per trovare le stringhe seguenti usare il comando N.

H
Converte un numero decimale in esadecimale. Quando usate questo comando inserite quindi il numero decimale seguito da ENTER.

I
Copia un blocco di memoria in un'altra zona di memoria. Un blocco puo' essere copiato anche se si sovrappone con se stesso. Dovrete inserire nell'ordine gli indirizzi di inizio e di fine del blocco e quindi l'indirizzo di destinazione.

J
Esegue le istruzioni dall'indirizzo specificato in avanti. Se durante l'esecuzione volete tornare al monitor inserite un breakpoint (vedi comando !).
Potete abortire il comando in fase di inserimento dell'indirizzo con [ESC]ape.
Nota che prima dell'esecuzione viene resettato lo stack e corrotti i registri.

[CTRL] C
Continua l'esecuzione dall'indirizzo contenuto nel Program Counter (PC) in avanti. A differenza del comando J i registri e lo stack vengono conservati come mostrato sul video.

L
Lista il contenuto esadecimale della memoria e i corrispondenti caratteri ASCII dal corrente indirizzo in avanti.
Per continuare premere un tasto,per smettere [ESC]ape.

M
Posiziona il corrente indirizzo di memoria al valore inserito.

N
Cerca la prossima occorenza della stringa di ricerca definita attraverso il comando G.

O
Posiziona il corrente indirizzo alla destinazione di un salto relativo. Il comando prende il byte al corrente indirizzo di memoria, lo considera come il valore di un salto relativo e aggiorna il corrente indirizzo di conseguenza.
Vedi anche il comando U.

P
Inserisce in una zona di memoria un determinato byte.
Per esempio se volete riempire la memoria tra #7000 e #77FF con il byte #55 ('U') dovete premere:
P 7000 ENTER 77FF ENTER 55 ENTER.

[CTRL] L
Uguale al comando L tranne che l'output viene diretto sulla stampante.
Alla fine di ogni pagina potete premere [ESC]ape per ritornare al display principale, oppure un altro tasto (tranne [CTRL] X) per continuare con la pagina successiva.

R
Serve a leggere un blocco di codice macchina dal nastro. Vi sara' chiesto di inserire il nome (facoltativo) e l'indirizzo da cui desiderate sia caricato il codice macchina (obbligatorio).

> (Maggiore)
Posiziona un breakpoint dopo la corrente istruzione ed esegue il comando [CTRL]C. Utile quando usate il single-stepping per chiamare una subroutine senza effettuare il single-stepping di quella subroutine.

T
Disassembla un blocco di memoria sullo schermo o sulla stampante. Permette inoltre di generare dei file di testo da poter essere utilizzati con l'ASSEMBLER.
Prima Vi saranno chiesti gli indirizzi di inizio e di fine del blocco quindi dovrete rispondere se desiderate l'output su stampante (rispondete Y per la stampante o un altro tasto per il video).
Quindi vi sara' chiesto ('File : ') l'indirizzo di partenza del file di testo da generare (premete solo ENTER se non volete generare alcun file di testo). Se avete risposto con un indirizzo alla domanda precedente, oppure avete richiesto l'output su stampante, allora dovrete inserire alla richiesta 'Spazio ?' l'indirizzo di una zona di memoria sufficientemente libera per la tabella delle label presenti nel testo da generare,tenendo presente che servono due bytes per ogni label generata. Se premete semplicemente ENTER sara' assunto come valore #6000.
Dopo tutto questo vi saranno chiesti gli indirizzi di inizio e di fine di eventuali blocchi di dati (DEFB) all'interno del blocco da disassemblare (premete semplicemente ENTER quando avete finito di inserire i dati oppure se non avete blocchi di dati all'interno del blocco da disassemblare).
Terminate queste operazioni il blocco verra' disassemblato e potete interrompere il listato in qualsiasi momento premendo un tasto, quindi [ESC] per smettere o un tasto per continuare.
Se viene incontrato un op-code mon valido sara' segnalato con un NOP seguito da '*'.
Se avete richiesto il file di testo, alla fine apparira' il messaggio 'Fine testo yyyy' dove yyyy e' l'indirizzo che dovete inserire nell'ASSEMBLER prima di poter trattare correttamente il testo (vedi comando X dell'EDITOR).
NOTA: Le label possono essere generate correttamente solo se si riferiscono all'interno del blocco disassemblato, altrimenti verra' fornito solamente l'indirizzo esadecimale assoluto.
ESEMPIO:
Se volete disassemblare su stampante (senza generare file di testo) il blocco compreso tra #8B e #9E con all'interno un blocco di dati tra #95 e #9E dovete premere:
T 8B ENTER 9E ENTER Y ENTER 95 ENTER 9E ENTER ENTER ENTER

W
Serve per salvare su nastro un blocco di codice macchina. Vi sara' richiesto il nome e gli indirizzi di inizio e fine.

U
Utilizzato insieme al comando O, serve per ritornare al punto in cui era sato utilizzato il comando O e cioe' dove si trovava l'istruzione di salto relativo.

V
Utilizzato insieme al comando X, ha la stessa funzione del comando U e ritorna alla relativa istruzione CALL o JP.

! (Punto esclamativo)
Posiziona un breakpoint al corrente indirizzo.
Utile per far girare una routine, interromperla e controllare sul display i risultati. Ogni volta che viene incontrato un breakpoint sentirete un segnale sonoro e l'esecuzione verra' sospesa in attesa della pressione di un tasto per tornare al display principale. RicordateVi che il breakpoint viene cancellato ogni volta che viene incontrato.
NOTA: non e' possibile inserire dei breakpoints nella ROM.

X
Come il comando O,tranne che viene considerato un indirizzo a 16 bit con il primo byte puntato dal corrente indirizzo. Vedi anche il comando V.

Y
Inserisce caratteri ASCII dal corrente indirizzo in avanti.
Quando usate questo comando potete inserire una stringa che deve essere terminata con [ESC]ape. I codici ASCII corrispondenti verranno inseriti in memoria e il corrente indirizzo aggiornato di conseguenza.

[CTRL] S
Single-step.
Serve ad eseguire le istruzioni assembler una alla volta dal corrente indirizzo in avanti e poter cosi' controllarne gli effetti.
Prima di utilizzare questo comando bisogna posizionare il corrente indirizzo e il Program Counter (PC) all'indirizzo dell'istruzione che si desidera eseguire.
Il single-stepping puo' essere eseguito in qualunque zona di memoria, anche nella ROM.

COME MODIFICARE LA MEMORIA
Per modificare il contenuto di un byte basta farlo puntare dal corrente indirizzo e inserire direttamente il numero desiderato seguito da ENTER. Il byte sara' cosi' modificato e il corrente indirizzo aumentato di uno.

COME MODIFICARE I REGISTRI
Per modificare un registro bisogna prima spostare la freccia a sinistra del display sul registro desiderato premendo ogni volta '.' (punto), quindi quando la freccia indica il registro giusto inserire direttamente il numero seguito da un punto (es. 710C.). E' possibile modificare sia i registri normali che quelli alternati ma non e' possibile alterare lo Stack Pointer (SP) o il registro IR.

UN ESEMPIO DI UTILIZZO DEL MONITOR

Supponiamo di avere i sottostanti 3 blocchi di istruzioni : Il primo carica HL e DE con due numeri e quindi chiama una routine (secondo blocco) per moltiplicarli tra loro con il risultato in HL e infine chiama due volte una routine (terzo blocco) per stampare il risultato sullo schermo.

```
7080          LD   HL,(#7200)
7083          LD   DE,(#7202)
7087          CALL Molt
708A          LD   A,H
708B          CALL Out
708E          LD   A,L
708F          CALL Out
7092          LD   HL,0
.
.
.
7100 Molt     XOR  A
7101          SBC  HL,DE
7103          ADD  HL,DE
7104          JR   NC,Mo1
7106          EX   DE,HL
7107 Mo1      OR   D
7108          SCF
7109          RET  NZ
710A          OR   E
710B          LD   E,D
710C          JR   NZ,Mo4
710E          EX   DE,HL
710F          RET
7110 Mo2      EX   DE,HL
7111          ADD  HL,DE
7112          EX   DE,HL
7113 Mo3      ADD  HL,HL
7114          RET  C
7115 Mo4      RRA
7116          JR   NC,Mu3
7118          OR   A
7119          JR   NZ,Mu2
711B          ADD  HL,DE
711C          RET

711D Out      PUSH AF
711E          RRCA
711F          RRCA
7120          RRCA
7121          RRCA
7122          CALL Nibble
7125          POP  AF
7126 Nibble   AND  %1111
7128          ADD  A,#90
712A          DAA
712B          ADC  A,#40
712D          DAA
712E          LD   IY,#5C3A
7132          RST  #10
7133          RET
.
.
7200          DEFW 10779
7202          DEFW 3
```

Ora supponiamo di investigare il programma per vedere se funziona o il modo in cui funziona. Possiamo fare cio' con i seguenti comandi (si tratta di un esempio che non e' necessariamente efficiente ma serve a illustrare l'uso dei comandi) :

Nell'esempio il simbolo ^ sostituisce [CTRL].

```
M 7080 ENTER   Posiziona il corrente indirizzo a #7080.
7080.          Posizion il Program Counter a #7080.
^S             Single-step.
^S             Single-step.
^S             Segui la CALL.
M 7115 ENTER   Salta il pre-trattamento dei numeri.
!              Posiziona un breakpoint.
^C             Continua l'esecuzione da #7100 al breakpoint.
^S             Single-step.
^S             Segui il salto relativo.
^S             Single-step.
^S               "
^S               "
^S               "
^S               "
^S               "
^S               "
^S             Ritorna dalla routine di moltiplicazione.
^S             Single-step.
^S             Segui la CALL.
M 7128 ENTER   Posiziona il corrente indirizzo al bit voluto.
!              Posiziona un breakpoint.
^C             Continua l'esecuzione da #711D al breakpoint.
^S             Single-step.
^S               "
^S               "
^S               "
S              Controlla l'indirizzo di ritorno.
!              Inserisce in quel punto un breakpoint
^C             e continua.
^S             Single-step.
S              Ritorna dalla routine Out
!
^C
^S             Single-step.
>              Esegue l'intera CALL fino a Out.
```

Potete provare a seguire questo esempio inserendo il programma attraverso l'ASSEMBLER e quindi eseguendo i comandi spiegati sopra.
Vi consigliamo vivamente di seguire l'esempio per meglio capire il funzionamento dei vari comandi del MONITOR.

UN ESEMPIO DEL DISPLAY PRINCIPALE

```
710C  2007      JR   NZ,#7115
>PC 710C   20 07 EB C9 EB 19 EB
 SP D0AF   8A 70 06 03 0A 03 0D
 IY CF6A   0D 11 0C 0F 09 18 18
 IX D09F   04 04 04 00 00 00 1B
 HL 2A1B   DF FE 29 28 02 CF 02
 DE 0000   F3 AF 11 FF FF C3 CB
 BC 0004   FF C3 CB 11 2A 5D 5C
 AF 0304       V
 IR 3F7C


7100  AF   7108  37   7110  EB
7101  ED   7109  C0   7111  19
7102  52   710A  B3   7112  EB
7103  19   710B  3A   7113  29
7104  30  >710C  20<  7114  D8
7105  01   710D  07   7115  1F
7106  EB   710E  EB   7116  30
7107  B2   710F  C9   7117  FB

>
```

Ecco un tipico esempio del display principale (ottenuto durante il single-stepping del precedente esempio).
In alto a sinistra abbiamo il corrente indirizzo, quindi il codice e il mnemonico dell'istruzione corrispondente a quell'indirizzo.
Le 9 linee seguenti contengono i registri dello Z 80 e il loro contenuto. Sulla stessa linea del registro sono inoltre mostrati i contenuti delle 7 locazioni di memoria che hanno come inizio l'indirizzo contenuto del registro.
Il registro dei Flag (F) e' decodificato per mostrare quali flag sono settati, nell'ordine in cui sono disposti all'interno del registro-(se F contiene #FF e cioe' tutti i registri sono settati, vedrete sulla stessa linea: 'S Z H V N C'; i flag sono nell'ordine: Sign, Zero, Half Carry, Parity/Overflow, Add/Subtract e Carry.
Una freccia sulla sinistra indica il registro attualmente puntato (vedi COME MODIFICARE I REGISTRI).
Il display di memoria a 24 bytes sotto al display dei registri mostra i 24 indirizzi e i loro contenuti attorno al corrente indirizzo (compreso tra due frecce).
I comandi vengono inseriti nella linea al fondo in risposta al simbolo '>'. Il display viene aggiornato dopo che viene eseguito ogni singolo comando.

Manuale e programma a cura di Enrico Piacentini

MICRODATA 1985