Università degli Studi di Pisa

# CMS

## modalità di uso

F. Carreras

34

CNUCE

# INTRODUZIONE

Un sistema time-sharing quale il CP/CMS, per sua natura gia' estremamente flessibile, calato in un ambiente di ricerca non poteva sottrarsi alle continue tentazioni di trasformazioni, talvolta di dettaglio talvolta profonde, che le necessita' diverse degli utilizzatori imponevano, al punto che alla lunga la sua fisionomia originaria risultava profondamente alterata.
Tali modifiche, esternamente almeno, si presentano sotto forma di un arricchimento delle possibilita' operative del sistema, sia attraverso nuovi comandi sia attraverso l'aggiunta di ulteriori opzioni a comandi gia' esistenti; internamente invece alcune modifiche intervengono direttamente sulla logica di funzionamento del sistema e il loro effetto si avverte nelle piu' soddisfacenti prestazioni del sistema nel suo complesso.
Questo manuale si propone pertanto di fornire una descrizione dettagliata e svelta allo stesso tempo del sistema CP/CMS del CNUCE che, come sopra accennato, su molti punti presenta diversita' notevoli rispetto al CP/CMS originale.
I criteri di descrizione si avvicinano a quelli del manuale di CP/CMS IBM H20-0859, anche per non disorientare utenti abituati a consultare tale manuale.
Per ragioni di spazio ho omesso per lo piu' i messaggi di errore dei comandi i quali tuttavia sono generalmente chiari di per se'; in caso di incertezza si rimanda al manuale IBM sopra citato.
Per rendere piu' agevole la lettura del manuale tutti i comandi, di CP e CMS senza distinzione, sono stati inseriti in ordine alfabetico, cosi' come i paragrafi riguardanti la programmazione in vari linguaggi. In testa e' stato inserito un indice alfabetico e un indice per "funzione" il cui scopo e' di aiutare ad inquadrare l'insieme dei comandi CP e CMS che possono essere utilizzati per sfruttare le potenzialita' del sistema CP/CMS relative alle funzioni specificate.
Aggiornamenti periodici a questo manuale saranno disponibili secondo una procedura descritta in una comunicazione a tutti gli utenti e consisteranno in pagine aggiuntive o in sostituzione di pagine esistenti, tutte comunque stampabili direttamente su terminale nello stesso formato del manuale.
E' chiaro percio' che il contenuto di questo manuale non restera' fisso, ma anzi ci sara' un impegno costante a mantenerlo il piu' possibile aderente alla situazione del CP/CMS; ogni critica o suggerimento saranno sempre benvenuti.

Francesco Carreras

# INDICE

## CONVENZIONI DI SCRITTURA

- Il CP e il CMS, tranne i files di tipo SCRIPT e MEMO, traducono tutti i caratteri letterali in maiuscolo: le lettere possono quindi essere battute a terminale indifferentemente in maiuscolo o minuscolo

- una riga fisica battuta a terminale puo' contenere piu' righe logiche, ciascuna separata dalla precedente dal segno '#'

- il simbolo di cancellazione di caratteri '@', cancella il carattere che lo precede; se ripetuto n volte cancella n caratteri precedenti il primo @

- il simbolo di cancellazione di riga '¢' cancella tutti i caratteri che lo precedono sulla riga

- ogni comando di CP e di CMS deve essere contenuto su una sola riga

- il nome del comando e i suoi parametri devono essere separati da uno o piu' spazi bianchi

- i parametri sottolineati sono quelli default, che cioe' il sistema sceglie quando non siano esplicitamente specificati

- un asterisco al posto di un parametro significa la totalita' degli articoli che il parametro puo' rappresentare

- i parametri scritti in lettere maiuscole devono essere battuti esattamente come scritti (anche in caratteri minuscoli)

- quando piu' parametri di un comando compaiono scritti, in verticale, uno sopra l'altro, si richiede la scelta di uno solo di questi parametri

- i parametri racchiusi tra parentesi < > sono opzionali e possono non essere specificati. Generalmente per essi esiste un valore default. Le parentesi non vanno scritte

- i parametri racchiusi tra coppie di parentesi < > successive possono essere specificati in qualunque ordine

- i parametri racchiusi tra coppie di parentesi < > a loro volta racchiuse in altre coppie di parentesi < >, se

specificati, devono essere battuti nell'ordine dato nel comando

- quando si specificano parametri compresi entro parentesi tonde, le parentesi devono essere scritte.

NORME DESCRITTIVE PER L'USO DEI TERMINALI

L'utente che voglia collegarsi al sistema 360/67 tramite terminale deve attenersi alle procedure sotto descritte.

Procedura di 'LOGIN':

1) Stabilire una linea di comunicazione col sistema. Per i terminali connessi direttamente al calcolatore e' sufficiente mettere in "on" l'interruttore di accensione del terminale.

2) Il sistema risponde scrivendo il messaggio:

CP-67 ONLINE xxxxxxxxxxxx

dove xxxxxxxxxxxx rappresenta una sequenza di caratteri privi di significato e quindi da ignorare.

3) Notificare a questo punto il sistema che si vuole usare il terminale premendo il tasto ATTN, 'attention' (o il tasto equivalente).

4) Il sistema risponde sbloccando la tastiera.

5) Comunicare al sistema il nome della macchina virtuale che si intende collegare scrivendo:

LOGIN nome

seguito da un ritorno carrello; 'nome' specifica il nome della macchina virtuale da collegare.

Per es. LOGIN sys01

6) Il sistema risponde con uno dei messaggi seguenti:

a) ENTER PASSWORD:

e sovrappone tre linee di caratteri posizionandosi poi all'inizio della linea risultante.
Questo messaggio indica che l'identificazione 'nome' e' stata accettata dal sistema. Proseguire col passo 7).

b) RESTART xxxxxxxx (xx caratteri senza significato)
Questo significa che la parola 'LOGIN' e' stata battuta in modo incorretto. Ritornare al passo 5) e ribattere la riga.

c) LOGGED IN ON DEV xxx
RESTART xxxxxxxx

Un altro utente sta usando la macchina virtuale prima specificata al terminale di indirizzo xxx. Sara' possibile collegarsi a tale macchina solo quando l'utente al terminale xxx si sara' scollegato

d) USER NOT IN DIRECTORY
RESTART xxxxxxxx

E' stato comunicato al sistema il nome di una macchina non esistente, o comunque scritto in modo incorretto. Ritornare al passo 5).

e) MAX NO. OF USERS EXCEEDED
LOGGED OUT AT xx.xx.zz on xx/xx/xx *** BY OPERATOR ***

Se la tastiera si sblocca tornare al passo 5); altrimenti, e questo significa che la macchina sta servendo il numero massimo di utenti permesso, aspettare qualche minuto e riprovare il punto ).

f) UPDATING DIRECTORY
RESTART xxxxxxxx

E' in atto l'aggiornamento del 'directory' del CP/67. Attendere qualche minuto e riprendere dal passo 5).

7) scrivere la parola di identificazione (password) seguita da un ritorno carrello.

8) A questo punto se ci sono schede nel lettore virtuale o output per la stampante o il perforatore viene scritto il messaggio:

FILES:- xx RDR,xx PRT,xx PUN

dove xx indica il numero dei files; viene anche scritto un eventuale messaggio del giorno de l'operatore.

9) Il sistema risponde alla scrittura della 'password' con uno dei messaggi seguenti:

READY AT xx.xx.xx ON xx/xx/xx

dove xx.xx.xx specifica l'ora e xx/xx/xx la data. Questo messaggio indica che la password e' stata riconosciuta e accettata, e che la procedura d 'LOGIN' e' cosi'

terminata. Il controllo passa ora al CP/67 e si puo' richiedere qualunque funzione di consolle.

10) Inizializzare il CMS mediante la funzione di consolle

IPL 190 oppure IPL CMS

seguita da un ritorno di carrello. Il secondo modo e' piu' rapido.

11) Il sistema risponde scrivendo un messaggio del tipo:

CMS...VERSION n.n (xx xx xx)

e la tastiera si sblocca. Il sistema accetta ora qualunque comando CMS.

Procedura di 'LOGOUT'

Quando l'utente desidera disconnetersi dalla macchina virtuale su cui sta lavorando deve fare 'logout' dal CP/67. Se non e' gia' in 'ambiente' CP vi puo' entrare premendo una volta il tasto ATTN. La tastiera si sblocca e l'utente deve battere LOGOUT seguito da un ritorno carrello. Il sistema risponde:

CONNECT=hh.mm.ss VIRTCPU=mm.ss.hs TOTCPU=mmm.ss.hs
LOGOUT AT xx.xx.xx ON xx/xx/xx

e viene persa la connessione col calcolatore. Il tempo di connessione e' in ore, minuti e secondi; il tempo di CPU virtuale e quello totale sono in minuti, secondi e centesimi di secondo.
A questo punto si puo' spegnere il terminale.

Se l'utente desidera disconnettere dal sistema la sua macchina virtuale senza perdere la connessione col calcolatore puo' battere la parola LOGOUT seguita da almeno un carattere qualsiasi.

## CLASSIFICAZIONE DEI COMANDI PER FUNZIONE

### CONTROLLO DI UNA SESSIONE DI TERMINALE

| | | |
|---|---|---|
| BLIP | CMS | Permette di controllare il tempo di CPU consumato |
| CSET CP | CMS | Controllo dei comandi CP da CMS |
| CSET IMPEX | CMS | Controlla l'esecuzione di files di tipo EXEC |
| CSET RDYMSG | CMS | Controllo dei messaggi di READY |
| DIAL | C"P | Connette un terminale ad una macchina virtuale a piu' accessi |
| DISCONN | CP | Disconnette il terminale dalla macchina virtuale |
| LOGIN | CP | Inizio sessione di terminale |
| LOGOUT | CMS | Termina la sessione di CMS senza uscire dal CP |
| LOGOUT | CP | Termine sessione di terminale |
| MSG | CP | Messaggi ad altre M.V. o all'operatore |
| QUERY LOGMSG | CP | Stampa a terminale il messaggio del giorno |
| QUERY NAMES | CP | Da' i nomi e indirizzi delle M.V. collegate |
| QUERY RESIDUAL | CP | Da' il tempo di CPU residuo per M.V. a tempo limitato |
| QUERY TIME | CP | Da' il tempo di CPU consumato fino a quel momento |
| QUERY USER | CP | Da' il numero di utenti connessi al sistema |
| QUERY userid | CP | Da' l'indirizzo del terminale dove l'utente "userid" e' collegato |
| SET APLBALL | CP | Controllo codifica dei caratteri per uso APL |
| SET LINEDIT | CP | Controllo sui caratteri di edizione |
| SET MSG | CP | Controlla la stampa dei messaggi al terminale |
| SET WNG | CP | Controlla la stampa degli avvertimenti al terminale |
| SLEEP | CP | Permette di ricevere messaggi senza premere il tasto di ATTN, mentre la macchina e' ferma |
| 3.0 | CMS | Riporta il controllo al CMS 3.0 |
| 3.2 | CMS | Da' il controllo al CMS 3.2 |

## CONTROLLO DELLA MACCHINA VIRTUALE

| | | |
|---|---|---|
| BEGIN | CP | Inizia l'esecuzione, simula tasto START |
| DETACH | CP | Stacca una unita' dalla configurazione della M.V. |
| DISPLAY | CP | Da' il contenuto della memoria e registri |
| ECHO | CMS | Permette il controllo della qualita' della trasmissione del terminale |
| EXTERNAL | CP | Simula la funzione del tasto EXTERNAL |
| IPL | CMS | Simula la sequenza di IPL, azzerando la memoria |
| IPL | CP | Simula sequenza di IPL, azzerando la memoria |
| IPLSAVE | CP | Simula la sequenza IPL senza azzerare la memoria |
| LINK | CP | Collega un disco alla macchina virtuale |
| PSWRESTART | CP | Simula la funzione PSW RESTART |
| QUERY V <ccu> | CP | Da' le caratteristiche dell'unita' di indirizzo ccu |
| QUERY VIRTUAL | CP | Da' la configurazione della macchina virtuale |
| RESET | CP | Simula la funzione del tasto SYSTEM RESET |
| SET ATTN | CP | Controlla la funzione del tasto ATTN |
| SET PASSWORD | CP | Permette di ridefinire la password della macchina virtuale |
| SET RUN | CP | Controlla il funzionamento della macchina virtuale |
| STORE | CP | Cambia il contenuto dei registri e memoria |

## CONTROLLO DELLO SPOOL

| | | |
|---|---|---|
| CLOSE | CP | Invia i files sull'unita' specificata |
| CLOSIO | CMS | Permette di creare un solo file da piu' files in spool |
| CSET READ | CMS | Cambia l'unita' di lettura da terminale a file su disco |
| CSET TYPE | CMS | Cambia l'unita' di scrittura da terminale a file su disco |
| DETACH | CP | Stacca una unita' dalla configurazione della macchina virtuale |
| MOVEFILE | CMS | Permette lo spostamento di files fra unita' uguali o diverse |
| OFFLINE | CMS | Permette di inviare files all'unita' periferica indicata |
| PURGE | CP | Cancella dalle code di spool i files sull'unita' specificata |
| QUERY FILES | CP | Da' il numero dei files sulle unita' di spool della M.V. |
| READY | CP | Simula un End-of-file sull'unita' specificata |
| SET CARDSAVE | CP | Permette di mantenere o meno dei files sul lettore |
| SPOOL | CP | Indirizza ad una particolare unita' dei files |
| XFER | CP | Indirizza lo spool di un perforatore o stampante al lettore di una macchina virtuale |

## CONTROLLO DEI DISCHI

| | | |
|---|---|---|
| DETACH | CP | Disconnette un disco dalla configurazione di una acchina virtuale |
| DUMPD | CMS | Produce un dump esadecimale del contenuto di un disco |
| DUMPREST | CMS | Dump e restore di minidischi |
| FORMAT | CMS | Inizializza dischi in formato CMS |
| LINK | CP | Connette logicamente un disco alla configurazione di una macchina virtuale |
| LOGRN | CMS | Regola i rapporti tra i dischi di una macchina virtuale |
| RELEASE | CMS | Disattiva un disco |
| SOS | CMS | Permette di leggere o scrivere su disco records CMS |
| SOSUFD | CMS | Individua i records CMS di formato UFD su un disco |
| STATD | CMS | Produce dati statistici sull'utilizzo di un disco |

## MANTENIMENTO DEI FILES

| | | |
|---|---|---|
| ALTER | CMS | Cambia il nome, tipo, modo di un file |
| CNVT26 | CMS | Traduce un file da BCDIC in EBCDIC |
| COMBINE | CMS | Crea un file da files esistenti |
| COMPARE | CMS | Confronta i records corrispondenti di due files |
| CVTFV | CMS | Trasforma il formato di un file da fisso a variabile |
| CVTVF | CMS | Trasforma il formato di un file da variabile a fisso |
| DISK | CMS | Perfora o legge dal lettore files di formato qualunque |
| DUMPD | CMS | Stampa in esadecimale il contenuto di un disco |
| DUMPF | CMS | Stampa in esadecimale il contenuto di un file |
| EDIT,CEDIT | CMS | Scrittura ed edizione di files |
| ERASE | CMS | Cancella files |
| FST | CMS | Stampa informazioni su un file |
| LISTF | CMS | Lista dei nomi e altre caratteristiche di files |
| LOGIN | CMS | Regola i rapporti tra i dischi di una macchina virtuale |
| MACLIB | CMS | Crea e aggiorna librerie di files di tipo MACRO |
| MOVEFILE | CMS | Permette lo spostamento di files tra unita' uguali o diverse |
| NASTRI | CMS | Esame delle caratteristiche e contenuto di un nastro |
| OFFLINE | CMS | Invia files da disco a unita' periferiche o legge files su disco |
| OSTAPE | CMS | Lettura di un nastro OS |
| PACK | CMS | Compatta files |
| PRINTF | CMS | Stampa a terminale il contenuto di un file |
| READTAPE | CMS | Legge su disco un file su nastro formato OS |
| SALVA | CMS | Copia files fra dischi |
| SCRIPT | CMS | Permette la scrittura formattata di testi |
| SORT | CMS | Sort di un file |
| SPLIT | CMS | Crea piu' files da un file esistente |
| STAT | CMS | Produce dati statistici sull'utilizzo di un disco |
| STATE | CMS | Controlla l'esistenza di un file |
| SYN | CMS | Regola l'uso dei sinonimi di |

comandi CMS

| | | |
|---|---|---|
| TAPE | CMS | Scrive e legge su nastro files CMS |
| TAPEIO | CMS | Esegue funzioni di I/O su nastro |
| TAPRINT | CMS | Scrive e legge su nastro files di tipo LISTING |
| TPCOPY | CMS | Copia files fra nastri |
| TXTLIB | CMS | Crea e aggiorna librerie di files di tipo TEXT |
| UNPACK | CMS | Riporta files compattati al formato originario |
| UPDATE | CMS | Aggiorna files |
| WRTAPE | CMS | Copia files da disco a nastro |
| WTAPOS | CMS | Scrive un file CMS su nastro formato OS |

## SVILUPPO ED ELABORAZIONE DEI PROGRAMMI

| | | |
|---|---|---|
| BLIP | CMS | Definisce caratteri di controllo del tempo di CPU usato |
| CEDIT | CMS | Edizione di programmi di grandi dimensioni |
| CHARDEF | CMS | Ridefinisce i caratteri speciali, per es. di tabulazione |
| CPFUNCTION | CMS | Permette l'esecuzione di comandi CP dal CMS |
| CSET | CMS | Permette la ridefinizione delle unita' di input/output |
| DSNAME | CMS | Ridefinizione delle unita' di input/output |
| EDIT | CMS | Creazione e modifiche di un programma |
| EXEC | CMS | Permette l'esecuzione di piu' comandi mediante un solo comando |
| FILEDEF | CMS | Ridefinizioni delle unita' di input/output |
| FINIS | CMS | Chiude i files specificati |
| GENMOD | CMS | Crea un file immagine-di-memoria |
| GLOBAL | CMS | Definisce le librerie macro o text da usare per il caricamento di un programma |
| KT | CMS | Sopprime la scrittura di un file a terminale |
| KX | CMS | Interrompe l'esecuzione di un programma |
| LINEND | CMS | Ridefinisce il carattere di fine linea logica |
| LOAD | CMS | Carica in memoria i programmi specificati |
| LOADMOD | CMS | Carica in memoria un file immagine-di-memoria |
| MACLIB | CMS | Creazione e aggiornamento di librerie di macro |
| MACRO | CMS | Lista una macro |
| MODMAP | CMS | Stampa a terminale le mappe di caricamento di un file |
| MAPPRT | CMS | Crea e stampa una mappa dei punti di ingresso del nucleo |
| REUSE | CMS | Carica in memoria i files specificati |
| RT | CMS | Ripristina la scrittura di un file a terminale |
| START | CMS | Inizia l'esecuzione di un programma |
| TXTLIB | CMS | Creazione e aggiornamento di librerie di files text |

| | | |
|---|---|---|
| USE | CMS | Carica in memoria i programmi specificati |
| $ | CMS | Carica in memoria e inizia l'esecuzione di un programma |

## PROVA E CORREZIONE DI PROGRAMMI

| | | |
|---|---|---|
| BATCHCMS | CMS | Crea le schede di controllo e invia files al BATCH CMS |
| BEGIN | CP | Riavvia l'esecuzione in una macchina virtuale |
| CLROVER | CMS | Termina l'azione di SETERR e CLROVER |
| DEBUG | CMS | Ambiente utile per la correzione dei programmi |
| DISAS | CMS | Ricostruisce in ASSEMBLER programmi assemblati |
| DISPLAY | CP | Stampa il contenuto di parti di memoria etc |
| DUMP | CP | Stampa in esadecimale il contenuto di aree di memoria |
| FORTBUG | CMS | Funzione utile per la correzione di programmi FORTRAN |
| KO | CMS | Disattiva la registrazione di informazioni di traccia |
| OSBATCH | CMS | Crea le schede di controllo e invia files al BATCH OS |
| SET ADSTOP | CP | Arresta l'esecuzione all'indirizzo specificato |
| SETERR | CMS | Attiva la registrazione di informazioni di traccia |
| SETOVER | CMS | Attiva la registrazione di informazioni di traccia |
| SET TRACE | CP | Attiva la traccia di istruzioni etc |
| STORE | CP | Permette di cambiare il contenuto di registri, aree di memoria etc |

| | | |
|---|---|---|
| ALGOL | CMS | Compila programmi in ALGOL 60 |
| APL | CMS | Fa entrare in ambiente APL |
| ASSEMBLE | CMS | Assembla programmi scritti in ASSEMBLER |
| AWX | CMS | Compila programmi scritti in ALGOLW |
| BRUIN | CMS | Fa entrare in ambiente BRUIN |
| COBOL | CMS | Compila programmi in COBOL U |
| FORDECAL | CMS | Fa entrare in ambiente FORMAC conversazionale |
| FORM | CMS | Compila ed esegue programmi scritti in FORMAC |
| FORSTOP | CMS | Fa uscire dall'ambiente FORMAC conversazionale |
| FORTRAN | CMS | Compila programmi scritti in FORTRAN |
| GDYN | CMS | Compila ed inizia l'esecuzione conversazionale di programmi scritti in DYNAMO |
| GPSS | CMS | Compila ed esegue programmi scritti in GPSS |
| LISPCMS | CMS | Compila programmi scritti in LISP |
| PLI | CMS | Compila programmi scritti in PLI |
| SNOBOL | CMS | Compila programmi scitti in SNOBOL |

Funzioni del CMS

Programmazione in ASSEMBLER
Programmazione in FORTRAN
Programmazione in PLI

| | |
|---|---|
| ALGOL | COMPILA PROGRAMMI SCRITTI IN ALGOL-60. |
| ALGOL | filename1...filenameN<(opz1...opzM)> |
| filename | nomi del files da compilare, il tipo deve essere ALGOL ed i records di 80 caratteri. I nomi dei files sono separati da uno o piu' spazi bianchi. |
| opz | opzioni; vi possono essere fino a 8 opzioni, separate da spazi bianchi; quelle default sono sottolineate |
| PROGRAM PG | il file da compilare e' un programma ALGOL |
| PROCEDURE PC | il file da compilare e' una procedura ALGOL |
| SHORT SP | la rappresentazione interna dei reali e' in semplice precisione (1 voce) |
| LONG LP | la rappresentazione interna dei reali e' in doppia precisione (2 voci) |
| DECK D | genera il file tipo TEXT del codice oggetto |
| NODECK ND | non genera il file tipo TEXT |
| SOURCE S | aggiunge la lista del programma sorgente al file LISTING |
| NOSOURCE NS | non aggiunge la lista del programma sorgente |
| TEST T | controlla il valore degli indici delle matrici durante l'esecuzione |
| NOTEST NT | non controlla il valore degli indici |
| EBCIDIC EB | il programma sorgente e' in codice EBCIDIC |
| ISO I | il programma sorgente e' in codice ISO |
| PRINT P | stampa il file LISTING sulla stampante e lo cancella su disco |
| NOPRINT NP | memorizza il file LISTING su disco senza stamparlo |
| DIAG DG | lista i messaggi d'errore di compilazione sul terminale |
| NODIAG NDG | non lista i messaggi d'errore |

NOTE - Le opzioni di compilazione, scritte entro parentesi, possono essere specificate in un ordine qualunque e valgono per tutti i files da compilare col comando.C Le

opzioni errate non vengono prese in considerazione.

- La diagnostica d'errore viene inclusa nel file LISTING e stampata sul terminale (se non vi e' l'opzione NODIAG). Per la sua interpretazione vedere ALGOL PROGRAMMER'S GUIDE IBM C33-4000.

Programmazione ALGOL sotto CMS

Il risultato della compilazione di un filename ALGOL e' la creazione di un filename TEXT e di un filename LISTING che cancellano i files con lo stesso nome e tipo eventualmente presenti sul disco P. (A meno che non si specifichino le opzioni NODECK e/o PRINT)

Tutti i files sequenziali utilizzati o creati dai programmi ALGOL vengono identificati come

FILE ALGLDDXX

dove XX e' un numero da 02 a 15 che identifica il Data Set. Il numero 00 provoca la lettura da terminale sotto forma di records "formato scheda". I dati devono avere la stessa struttura che in OS. La fine della scheda e' specificata con un ritorno di carrello, e la fine dei dati con l'invio di una riga vuota (due ritorni di carrello consecutivi).
Il numero 01 provoca la scrittura su terminale e contemporaneamente la creazione del file FILE ALGLDD01. Tutti i data sets di ingresso devono essere presenti sul disco prima dell'inizio dell'esecuzione del programma ALGOL (salvo il caso del numero 00) ed avere records di 80 caratteri; i files in uscita vengono creati sul disco P e possono avere records lunghi fino a 132 caratteri. Per l'esecuzione di un file ALGOL filename TEXT occorre richiamare prima del caricamneto in memoria, la libreria TXTLIB dell'ALGOL col comando

GLOBAL T ALGLIB

E' necessario pertanto la sequenza di comandi

```
GLOBAL T ALGLIB
LOAD filename
START  (IHIFSAIN opz1,..opzN)
```

dove filename e' il nome del file ALGOL compilato e IHIFSAIN e' il punto d'ingresso per l'esecuzione solo nel caso che siano presenti opzioni

Opzioni:

TRACE da' la lista dei numeri delle istruzioni ALGOL eseguite
TRBEG xxxxx numero d'istruzione da cui iniziare la traccia
TREND yyyyy numero d'istruzione su cui terminare la traccia
DUMP dump parziale della memoria in caso di errore grave
TTY l'uscita su terminale produrra' records di 72 caratteri

Per ogni informazione sul linguaggio si rimanda ai manuali

ALGOL LANGUAGE IBM C28-6615
ALGOL PROGRAMMER'S GUIDE IBM C33-4000

Va osservato che i riferimenti al "Data management" e al JCL dell'OS non sono applicabili in CMS.

ALTER — CAMBIA TUTTO O IN PARTE L'IDENTIFICATORE (NOME, TIPO, MODO) DI UN FILE MEMORIZZATO SU UN DISCO IN LETTURA E SCRITTURA SENZA ALTERARE IL CONTENUTO DEL FILE.

| ALTER / AL | ofn | oft | ofm | nfn | nft | nfm |
|---|---|---|---|---|---|---|
| | * | * | * | * | * | * |
| | | | | = | = | = |

ofn oft ofm — sono rispettivamente il nome tipo e modo originali del file.
nfn nft nfm — sono rispettivamente il nuovo nome, tipo e modo del file.
* — un asterisco per ofn oft ofm significa rispettivamente tutti i files con quel nome, tipo e ogni disco read-write. Un asterisco o un (=) uguale per nfn nft nfm significa nessun cambiamento.

NOTA - Non si puo' cambiare la lettera del modo, ne' specificare come nuovo identificatore, l'identificatore di un file gia' esistente.

ESEMPI:
1) ALTER FILE SYSIN P1 = ASSEMBLE P1
Il file identificato come FILE SYSIN P1 viene cambiato

in file ASSEMBLE P1.

APL/CMS

L' APL/CMS e' in sostanza il programma IBM APL/360-DOS con alcune modifiche per permetterne l'utilizzazione da macchina virtuale sotto il controllo del CP/CMS. Per informazioni sul linguaggio APL si rimanda ai manuali:

| | |
|---|---|
| INTRODUZIONE ALL'APL CNUCE n.17 | |
| APL/360 PRIMER | IBM H20-0689 |
| APL/360 USER'S MANUAL | IBM H20-0683 |
| APL/360 GENERAL INFORMATION MANUAL | IBM H20-0850 |

mentre per l'uso dell'APL/CMS si rimanda al manuale APL(CMS) PROGRAM DESCRIPTION/OPERATION MANUAL IBM H20-1088.

I vantaggi dell'APL/CMS sono principalmente:

- Ampiezza degli spazi di lavoro, che e' pari alla dimensione della memoria virtuale della macchina meno circa 147K bytes
- Possibilita' di file I/O cioe' e' possibile trasferire su files CMS il contenuto di variabili APL e viceversa, creare variabili APL contenenti dati presi da un file CMS
- Possibilita' di usare in APL un certo numero di comandi di CP e CMS.

Per lavorare in APL/CMS occorre, da ambiente CMS, battere il comando:

```
APL
```

L'APL e' inizializzato quando l'elemento stampato si sposta di sei spazi e la tastiera si sblocca. Da questo momento in poi la tastiera sara' controllata dall'APL. Per terminare la seduta APL occorre emettere uno dei seguenti comandi APL:

```
)OFF
```

oppure

```
)CONTINUE
```

In questo caso si esce dall'APL e contemporaneamente si provoca il LOGOUT della macchina virtuale, mentre con l'opzione CMS cioe'

```
)OFF CMS
)CONTINUE CMS
```

si rientra in ambiente CMS.

Elemento stampante

Va a questo proposito notato che in CP esiste il comando:

SET APLBALL ON

che traduce le risposte al terminale del calcolatore in caratteri APL; questo comando e' automaticamente attivato allorquando in CMS si batte APL. Per contro esiste il comando

SET APLBALL OFF

che riporta la scrittura del terminale a caratteri CMS. Questo comando viene automaticamente attivato quando in APL si batte )OFF CMS oppure )CONTINUE CMS. Va notato infine che non vi e' completa corrispondenza tra APL e CMS per cui se si vuole lavorare in CMS con il

SET APLBALL ON

si perde un certo numero di caratteri di stampa.

Tasto di ATTENTION (ATTN)

In ambiente CMS il premere il tasto ATTN provoca il passaggio del controllo al CP, mentre in ambiente APL questo provoca

- sospensione dell'esecuzione di una funzione in esecuzione
- sospensione della stampa di funzione o variabili
- soppressione delle righe di ingresso di una funzione in fase di definizione

Il comando di CP

SET ATTN ON

permette di avere a disposizione le due possibilita' sopra menzionate, vale a dire fa in modo che premendo una prima volta il tasto ATTN si passi il controllo al CP, mentre premendo il tasto ATTN una seconda volta si provochino gli effetti normali dell'APL. La situazione normale per l'APL puo' essere ripristinata col comando di CP

SET ATTN OFF

Funzioni di I/O

Nella libreria pubblica n.1 oltre agli spazi di lavoro dell'APL/DOS si trova lo spazio di lavoro IOFNS contenente le funzioni che permettono all'APL/CMS di scambiare files col CMS. Per i dettagli si rimanda al manuale APL(CMS) Program description/Operation manual. Viene qui acclusa la documentazione delle funzioni conversazionali LEGGI e SCRIVI scritte presso il CNUCE. Queste funzioni facilitano il passaggio di dati dall'APL al CMS in un formato tipo FORTRAN.

Lo spazio di lavoro IOFNS contiene inoltre le due funzioni CP e CMS che permettono di utilizzare un sotto - insieme di comandi di CP e CMS dall'APL come per esempio:

```
CP 'SET ATTN ON'
CMS 'STATE nome tipo'
```

Per i ettagli vedere il manuale APL(CMS)

LEGGI

L'utente APL-CMS ha la possibilita' di trasferire il contenuto di files CMS in variabili APL mediante l'uso della funzione LEGGI. Il valore della funzione LEGGI deve essere assegnato alla variabile APL che si vuole creare; in caso contrario si ottiene la stampa del file CMS secondo il formato APL. La funzione chiede all'utente il nome ed il tipo del file CMS in cui risiedono i dati.
Se il contenuto del file e' numerico la variabile APL che risulta e' un vettore avente dimensione pari al numero di cifre contenute nel file CMS. Occorre rimarcare che il formato del singolo record deve essere tale che tutti i numeri risultino separati da almeno un bianco. Se il contenuto del file CMS e' alfanumerico la variabile APL risulta essere un vettore di dimensioni pari alla lunghezza del record moltiplicato il numero di record in quanto sono trasferiti nella variabile APL anche i bianchi.

SCRIVI

L'utente dell'APL-CMS ha la possibilita' di trasferire il contenuto di variabili APL in file CMS e viceversa. La funzione SCRIVI permette di creare files CMS aventi record

di lunghezza fissa pari a 80 caratteri con codifica EBCDIC; per quanto riguarda la possibilita' di scrivere e leggere files con records a lunghezza variabile e con codifica binaria, si rimanda al manuale.
SCRIVI e' una funzione monadica il cui argomento e' il nome della variabile APL contenente i dati da riportare sul file CMS. Per potere operare la funzione chiede all'utente le specifiche del file CMS; tali specifiche sono del tipo:

```
A

   nome tipo formato

W
```

Le specifiche sono costituite da 4 campi separati da almeno uno spazio. Il secondo e il terzo campo indicano il nome ed il tipo del file CMS cui si fa riferimento. Se il primo campo e' W viene creato un nuovo file (un eventuale file preesistente viene cancellato); se il primo campo e' A i dati vengono aggiunti in coda al file se esso esiste gia', oppure viene creato un nuovo file come nel caso precedente. Il quarto campo e' relativo al formato del record CMS; la simbologia usata si attiene alle specifiche di formato del FORTRAN. I formati possibili sono:

```
I  (numeri interi)
A  (caratteri alfanumerici)
F  (numeri con parte decimale)
E  (numeri in forma esponenziale)
```

formato I

```
N I W     oppure
  I W
```

dove N specifica i numeri da scrivere per ogni record e W e' il numero di cifre per ciascun intero. Se N e' omesso viene posto: N=1.
N e W devono soddisfare la relazione:

$$N \times W \leq 72$$

formato A

```
N A W     oppure
  A W
```

dove N e' il numero di campi alfanumerici per ogni record e W e' il numero di caratteri di ogni campo. Se N e' omesso

viene posto: N=1.

$$N \times W \leq 72$$

formato F

```
N F W.D  oppure
  F W.D
```

dove N specifica i numeri da scrivere per ogni record, W e' il numero di caratteri riservato per ogni numero e D il numero di cifre dopo la virgola. Poiche' W include anche il punto decimale e il segno, deve risultare:

$$W-D \geq 2$$

se N e' omesso viene posto: N=1. N e W devono soddisfare la relazione:

$$N \times W \leq 72$$

formato E

```
N E W.D  oppure
  E W.D
```

dove N specifica i numeri da scrivere per ogni record, W e' il numero di caratteri riservato per ogni numero e D il numero di cifre significative prima dell'esponente. L'aspetto del numero scritto col formato E e' il seguente:

```
+C.CC...CE+CC
```

dove il numero di cifre tra il punto decimale e l'esponente e' D-1. Poiche' W include anche l'esponente, deve risultare:

$$W-D \geq 6$$

se N e' omesso viene posto: N=1. N e W devono soddisfare la relazione:

$$N \times W \leq 72$$

## Librerie

Le librerie private in APL/CMS vengono memorizzate su disco col tipo APLWS e il nome imposto dall'utente. I records dei files di tipo APLWS sono di lunghezza variabile. Uno spazio di lavoro vuoto, se memorizzato su disco, occupa sei records. Le librerie pubbliche risiedono su un disco del

sistema CMS e sono accessibili solo in lettura. Esse hanno numeri di identificazione da 1 a 999. Se una libreria pubblica viene copiata in memoria e poi memorizzata nel proprio P-disk con un comando )SAVE essa viene copiata con filetype APLWSxxx dove xxx e' il numero della libreria. A partire da questo momento ogni volta che la libreria viene caricata in memoria con un comando )LOAD, e' la copia sul P-disk che viene caricata ed il comando )SAVE rimpiazza questa copia.

Passaggio di spazi di lavoro tra differenti istallazioni APL.

Tra istallazioni APL differenti gli spazi di lavoro devono essere necessariamente trasferiti mediante nastro (anche tra APL/360 DOS e APL/CMS). Sono a disposizione dell'utilizzatore APL (CMS) due comandi CMS chiamati APLUT1 ed APLUT2; per dettagli particolareggiati su dette utilities si rimanda a : APL (CMS) USER'S MANUAL.
La procedura APLUT1 serve per fare il restore da nastro di spazi di lavoro APL ivi contenuti; il nastro deve essere stato precedentemente attaccato alla macchina virtuale come 181.

ES.

APLUT1 180073 URCA

lo spazio di lavoro URCA della libreria 180073 viene caricato se esiste e copiato sul P-disk con filetype APLWS073
Per caricare questo spazio di lavoro in APL (CMS) occorrera' battere il comando APL. )LOAD 73 URCA.
La procedura APLUT2 serve per fare il dump su nastro di spazi di lavoro APL contenuti sul proprio P-disk; il nastro deve essere stato precedentemente attaccato alla macchina virtuale come 181.

Es.

APLUT2 333 SCREW

lo spazio di lavoro SCREW viene copiato su nastro con un numero di librerie fittizio 333 che verra' usato solo per operazioni di restore su APL/OS o APL/DOS.

Passaggio di librerie da una macchina virtuale all'altra

Valgono le procedure note per trasferire i files n CMS.

Occorre solo ricordare che si tratta di files aventi record a lunghezza variabile per cui non sono applicabili i comandi OFFLINE PUNCH e OFFLINE READ.
Dopo aver battuto in CP il comando

XFER D TO NOME

in CMS si batte il comando

DISK DUMP filename APLWS

Entrati nella macchina NOME, e' sufficiente scrivere

DISK LOAD

per avere memorizzato sul P-disk il file filename APLWS.

ASSEMBLE — CONVERTE CODICE SORGENTE, SCRITTO IN LINGUAGGIO ASSEMBLER 360, IN CODICE OGGETTO RILOCABILE.

ASSEMBLE fn1...fnN (opz1...opzN)
A

fn — specifica il file di tipo SYSIN da assemblare. Si possono specificare quanti files si vuole, purche' entrino in una riga.

Le opzioni sono:

| | |
|---|---|
| DECK | crea un file TEXT col modulo rilocabile |
| NODECK | sopprime la creazione del file TEXT |
| LIST | crea un file LISTING del programma assemblato |
| NOLIST | sopprime la creazione del file LISTING e la stampa dei messaggi diagnostici sul terminale |
| XREF | include la tabella delle cross-references nel file LISTING |
| NOXREF | sopprime la tabella delle cross-references |
| DIAG | stampa a terminale le istruzioni errate del codice sorgente, insieme ai messaggi diagnostici e di errore |

| | |
|---|---|
| NODIAG | sopprime la stampa di istruzioni errate e messaggi diagnostici |
| LTAPn | scrive il file LISTING sul nastro di indirizzo simbolico TAPn |
| LDISK | scrive il file LISTING sul disco permanente |
| PRINT | scrive il file LISTING sulla stampante offline |
| RENT | controlla la rientrabilita' del programma |
| NORENT | sopprime il controllo della rientrabilita' |

NOTE - I files trattati dal comando ASSEMBLE devono essere di tipo SYSIN.

- Ogni combinazioni delle opzioni puo' essere specificata, ma usando la NOLIST le opzioni XREF, DIAG, LDISK, PRINT, saranno ignorate.

- L'assembler ricerca le macro definizioni nelle librerie del sistema (SYSLIB MACLIB SY e OSLIB MACLIB SY). La ricerca puo' essere estesa a librerie addizionali col comando GLOBAL ASSEMBLER MACLIB.

- Il file TEXT e' un file in linguaggio di macchina che puo' essere caricato per l'esecuzione mediante i comandi LOAD o $.

- Un insieme di opzioni e' valido per tutti i files trattati dal comando.

- I files TEXT e LISTING creati dall'ASSEMBLE cancellano eventuali files preesistenti con lo stesso filename.

- Normalmente il nome dell'entry point e' uguale al filename del file TEXT, in tal caso il programma puo' essere chiamato per l'esecuzione mediante il comando $; se invece il nome dell'entry point ed il filename sono diversi, la sequenza di chiamata deve essere:

```
LOAD   filename
START  nome entry-point
```

ESEMPI:

1) ASSEMBLE FILO (PRINT)

Il file FILO SYSIN sara' assemblato e il file FILO LISTING creato verra' inviato sulla stampante; verra' inoltre creato su disco il file FILO TEXT.

AWX — COMPILA PROGRAMMI SCRITTI IN ALGOLW.

AWX filename1...filenameN <(opt1...optM)>

filename — nomi dei files da compilare. Il tipo deve essere ALGOLW e i records di 80 caratteri.

opzioni:

PRINT, P — fa stampare il file LISTING sulla stampante senza memorizzarlo su disco

NODECK, ND — impedisce la creazione del file TEXT

NODIAG, ND — impedisce la scrittura a terminale dei messaggi d'errore

Le opzioni default sono NOPRINT DECK DIAG.
Le opzioni non riconosciute sono ignorate.

Le opzioni del compilatore hanno la precedenza su quelle del comando. Esse devono apparire all'inizio del programma a colonna 1. Esse sono:

$LIST — stampa il programma sorgente (default)

$NOLIST — non stampa il programma sorgente ne' i messaggi d'errore.

$TRACEXY — permette di ottenere delle informazioni sullo svolgimento della compilazione e sul codice generato

Per il caricamento del programma compilato e' necessario prima richiamare la libreria dell'ALGOLW col comando

```
GLOBAL T AWXLIB
```

caricare il programma con

```
LOAD AWXRUN01 nome-del-programma
```

e infine iniziare l'esecuzione col comando

START

L' ingresso/uscita avviene di regola dal terminale. E' pero' possibile utilizzare due files di ingresso/uscita su disco. Occorre emettere il comando:

START AWXRUN01 DISK

invece del semplice START. Allora il file di lettura sara' FILE DONNEES, di modo qualunque qualunque e il file di uscita FILE RESULTAT, di modo P1. Se quest'ultimo gia' esiste gia' esiste viene cancellato.

Per ogni dettaglio si rimanda ai manuali:

INTRODUCTION A LA PROGRAMMATION EN ALGOLW
Universita' di Grenoble feb 1970

LE LANGUAGE ALGOLW (Danile Syty-IMAG)

| | |
|---|---|
| BATCHCMS | PERMETTE DI INVIARE PER L'ESECUZIONE FILES DA MACCHINE VIRTUALI ALLA MACCHINA BATCH CMS. |
| BATCHCMS BATCH | filename filetype accnt <DEF> |
| filename | nome del file contenente il programma sorgente |
| filetype | tipo del file contenente il programma sorgente; deve essere uno dei seguenti cinque: FORTRAN, FORMAC, PLI, ALGOL, SYSIN; ogni tipo diverso sara' interpretato come SYSIN e quindi il contenuto del file sara' trattato come ASSEMBLER. |
| accnt | codice di accounting dell'utente (4 carat.) |
| DEF | questo parametro se incluso nella lista specifica la validita' dei valori default di tempo CPU (15 sec.) e di righe di output (1000 righe) |

NOTA - Il file "filename filetype" non deve contenere schede di controllo: alla loro creazione provvede il comando BATCH.

Dopo aver battuto il comando BATCH coi parametri scelti il sistema pone le domande seguenti necessarie alla definizione completa del job:

RITORNO DELL'OUTPUT? BATTERE NO OPPURE USERID C( NOME DELLA MACCHINA VIRTUALE).

Risposta: NO
userid

userid — userid e' il nome della macchina virtuale sul cui lettore l'utente desidera che sia inviato l'output.

LETTORE REMOTO? (SI/NO)

Questa domanda viene posta solo se viene dato un userid nella precedente.

Risposta: SI
NO

Nel caso SI il nome userid della risposta precedente deve specificare un remoto (REMXXX); nel caso NO il nome di una macchina virtuale.

SE TEMPO > 15 SEC. E/O RIGHE > 1000 BATTERE XX (SEC.) E/O XX (MIGLIAIA DI RIGHE), BATTERE NO PER VALORI DEFAULT.

Risposta: XX XX
XX NO
NO XX
NO NO
NO

questo messaggio appare solo se e' stato omesso il parametro facoltatico DEF. La risposta NO e NO NO sono equivalenti, e valgono come DEF.

CI SONO DATI? BATTERE NO OPPURE NOME E TIPO DEL FILE DEI DATI

NO

Risposta:

filename filetype

NOTA - Al termine dell'esecuzione del comando BATCH i files del programma sorgente e dei dati rimangono immutati. L' output sulla stampante ha come prima pagina una mascherina contenente:

```
filename
accnt
0000
```

Questa mascherina non compare nel caso di risultati sul lettore di una macchina virtuale.

ESEMPI:

1) si vuole inviare alla macchina BATCH CMS il file PROVA1 FORTRAN con i dati contenuti nel file DATI DATA da una macchina virtuale con codice di accounting Y327.

```
batch prova1 fortran y327 def

RITORNO DELL'OUTPUT? BATTERE NO OPPURE USERID (NOME
DELLA MACCHINA VIRTUALE)

no

CI SONO DATI? BATTERE NO OPPURE NOME E TIPO DEL FILE

dati data

** CARDS XFERED TO BATCH **

R;T=.....
```

2) si vuole inviare alla macchina BATCH CMS il file PROVA2 PLI da macchina virtuale con codice di accounting Y327 e si desidera l'output nel lettore della macchina di nome PIPPO. Si desidera inoltre modoficare il valore default di CPU.

```
batch prova2 pli y327

RITORNO DELL'OUTPUT? BATTERE NO OPPURE USERID (NOME
DELLA MACCHINA VIRTUALE).

pippo

LETTORE REMOTO? (SI/NO)

no
```

```
SE TEMPO >15 SEC. E/O RIGHE >1000 BATTERE XX (SEC.) E/O
XX (MIGLIAIA DI RIGHE), BATTERE NO PER VALORI DEFAULT.

25  no

CI SONO  DATI? BATTERE NO OPPURE  NOME E TIPO  DEL FILE
DEI DATI.

no

** CARDS XFERED TO BATCH **

R;T=.....
```

**BEGIN** — FA RIPRENDERE L'ESECUZIONE DI UN PROGRAMMA TRASFERENDO IL CONTROLLO ALLA LOCAZIONE DI MEMORIA SPECIFICATA O ALLA LOCAZIONE CONTENUTA NELLA PSW CORRENTE.

```
BEGIN
  B                <hexloc>
```

hexloc — e' la locazione di memoria (esadecimale) da cui comincia l'esecuzione.

NOTA - L'esecuzione di un programma in una macchina virtuale puo' essere arrestata premendo il tasto "ATTENTION", il che equivale a premere il tasto "STOP" in una macchina reale. La funzione BEGIN equivale al tasto START, che permette di riprendere l'esecuzione.

**BLIP** — PROVOCA LA STAMPA A TERMINALE, A PARTIRE DAL MOMENTO DELL' EMISSIONE DEL COMANDO, DI UNA STRINGA DI CARATTERI (DA 1 A 8), OGNI 2 SECONDI DI TEMPO DI CPU; O FA TORNARE ALLA STAMPA DEI CARATTERI DEFAULT.

```
BLIP             caratteri    numero
                              (OFF)  1
```

caratteri — sono i caratteri da stampare
numero — e' un numero da 1 a 8 indicante il numero dei caratteri
(OFF) — mette il BLIP in una condizione che elimina qualsiasi indicazione ogni due secondi di tempo CPU.

NOTE - I caratteri default sono una sequenza di caratteri "non-stampanti" maiuscoli e minuscoli.

- Se il comando e' emesso senza parametri, si ritorna ai caratteri default.

- Se si desidera introdurre caratteri non presenti nella tastiera occorre usare BLIP in un programma assembler e specificare i caratteri col loro codice esadecimale.

ESEMPI:
1) BLIP +
Ogni due secondi di CPU verra' stampato a terminale il carattere +.

BRUIN — CHIAMA IL BROWN UNIVERSITY INTERPRETE, IN CUI L'UTENTE PUO' SCEGLIERE UN MODO ''CALCOLATORE DA TAVOLO'' E UN MODO '' PROGRAMMA MEMORIZZATO''.

BRUIN

NOTE - Questo comando fa passare dal CMS all'ambiente BRUIN; non appena a terminale viene stampato il carattere > e sbloccata la tastiera, possono essere emessi i comandi del BRUIN; per tornare nell'ambiente CMS si deve emettere il comando BRUIN: CANCEL.

Per ogni dettaglio sul linguaggio fare riferimento al manuale

IBM 360D-03.3.013 BROWN UNIVERSITY INTERPRETER.

CHARDEF — RIDEFINISCE I SIMBOLI DI CANCELLAZIONE DI CARATTERE E DI LINEA, DI TABULAZIONE LOGICA, E DI SPAZIO INDIETRO LOGICO.

CHARDEF codice <carattere-sostitutivo>

codice
- B significa carattere di spazio indietro logico dell'EDIT
- C significa simbolo di cancella-carattere
- L denota il simbolo di cancella-linea
- T significa carattere di tabulazione logica dell' EDIT

carattere sostitutivo — e' il carattere da usare per la funzione specificata; se omesso, non ci sara' alcun simbolo per quella funzione.

NOTA - Queste definizioni rimangono valide fino a quando non
1) si rifa' IPL del CMS
2) si emette un altro comando di CHARDEF
3) l'utente si stacca dal CP.

ESEMPI:
1) CHARDEF C T
Il carattere T da questo momento sara' interpretato come simbolo cancella carattere.

CLOSE — FA CESSARE L'ATTIVITA' DI SPOOLING SULL'UNITA' SPECIFICATA.

CLOSE indirizzo dell'unita'
C

indirizzo — e' l'indirizzo virtuale dell'unita' I/O.

NOTA - Nel caso del lettore di schede, il file in lettura viene cancellato; nel caso di stampante e perforatore

Il file viene inviato sulle unita' reali.

CLOSIO — SEGNALA AL CP CHE SONO STATE COMPLETATE LE OPERAZIONI DI I/O SU UNA O PIU' UNITA' (LETTORE, PERFORATORE DI SCHEDE, STAMPANTE) OFFLINE E CHE POSSONO INIZIARE LE OPERAZIONI DI OUTPUT SULLE UNITA' REALI OPPURE CHE L'OUTPUT ALL'UNITA' SPECIFICATA DEVE ESSERE RACCOLTO IN UN UNICO FILE.

```
          READER
                  ON
CLOSIO    PRINTER
                  OFF
                  ON
          PUNCH
                  OFF
```

OFF — raccoglie i files di output dell'unita' specificata nella relativa area di spooling, senza farli uscire sul dispositivo fisico

ON — segnala al CP che sono terminate le operazioni di I/O sul dispositivo specificato, e che devono avere inizio le operazioni di uscita sul dispositivo fisico.

NOTE - CLOSIO e' generalmente emesso automaticamente dai comandi che trattano lettore, perforatore di schede e stampante.

- Si puo' specificare una qualsiasi combinazione dei tre parametri che identificano i dispositivi di I/O.

- Tutti i dispositivi di I/O sopra specificati sono chiusi dal CP quando un utente fa LOGOUT dal sistema.

- Se non viene specificato alcun parametro sono chiusi tutti e tre i dispositivi come se fosse specificato ON.

- Nel caso di CLOSIO READER viene cancellata l'area di input del lettore.

ESEMPI:
1) CLOSIO PRINTER OFF

Tutti i files inviati alla stampante da ora fino all'emissione di CLOSIO PRINTER ON, sono raccolti in un unico file.

CLROVER — TERMINA L'AZIONE DEI COMANDI SETOVER E/O SETERR EMESSI DOPO L'ULTIMO COMANDO KO O CLROVER, O DOPO L'ULTIMO LOGIN DI CMS DELL'UTENTE, SE NON SONO STATI EMESSI COMANDI KO O CLROVER. TRASFERISCE ALLA STAMPANTE OFFLINE TUTTE LE INFORMAZIONI DI TRACCIA REGISTRATE FINO AL MOMENTO DELL'EMISSIONE DEL COMANDO.

CLROVER

NOTE - Questo comando fa cessare la registrazione di informazioni di traccia iniziata dai comandi SETOVER e/o SETERR.

CNVT26 — CONVERTE FILES FORMATO SCHEDA CODIFICATI IN BCDIC (026) IN CODICE EBCDIC (029).

CNVT26 filename filetype

NOTA - Il file originale (026) viene cancellato ed il nuovo prende lo stesso identificatore.

COBOL — COMPILA PROGRAMMI SCRITTI IN COBOL U.

COBOL nome1...nomen <(opz1...opzn)>

nome — nomi dei files da compilare il cui tipo deve essere COBOL. La lunghezza dei records deve essere di 80 caratteri.

opz — opzioni che sostituiscono quella default

Le opzioni sono:

DIAG PRINT NOLIST SOURCE APOST NOCLIST SUPMAP NODMAP NOTRUNC LINECNT=57 DECK SEQ FLAGE SPACE1

NOTA - Il caricamento di un programma COBOL compilato deve essere preceduto dal comando GLOBAL T COBLIB che attiva la libreria di routines del COBOL.

Per ogni dettaglio si rimanda alla dispensa COBOL U stampata

presso il CNUCE.

| | |
|---|---|
| COMBINE | COPIA I FILES SPECIFICATI CONCATENANDOLI NELL'ORDINE DATO IN UN NUOVO FILE, CHE VIENE SCRITTO SU DISCO IN LETTURA-SCRITTURA CON L'IDENTIFICATORE SPECIFICATO. |
| COMBINE | nfn nft nfm ofn1 oft1 ofm1...ofnN oftN ofmN |
| nfn nft nfm | sono il nome, tipo e modo rispettivamente del file da creare |
| ofn oft ofm | sono il nome, tipo e modo rispettivamente dei files esistenti che saranno inclusi nel nuovo file. |

NOTE - Per ogni file va specificato nome, tipo e modo.

- I files di input devono avere lo stesso formato di record (fisso o variabile) e se fisso la stessa lunghezza di record.

- I files di input possono essere su un disco qualsiasi, mentre il nuovo file puo' essere creato solo su un disco in lettura-scrittura.

- Se il file creato ha nome, tipo e modo di un file esistente, quest'ultimo viene cancellato.

- Specificando uguale fn e ft ma fm diverso questo comando ricopia files da un disco anche solo in lettura ad un disco in lettura-scrittura.

ESEMPI:
1) COMBINE AB SYSIN P1 AB SYSIN A1 AC SYSIN P1
Sul disco P viene creato il file AB SYSIN composto dei

files AB SYSIN del disco A e AC SYSIN del disco P.

COMPARE ESEGUE IL CONFRONTO TRA DUE FILES SU DISCO.

COMPARE filename1 filetype1 filename2 filetype2

NOTA - Il confronto viene effettuato tra records corrispondenti dei due files.

- Se i due files sono identici, viene stampato un messaggio a terminale.

- Quando vengono trovati due records corrispondenti diversi, viene stampato a terminale il record del primo file, seguito da quello corrispondente del secondo file.

- Un messaggio avverte se viene raggiunta la fine di un file , prima della fine dell'altro file.

- I records da confrontare possono avere la lunghezza massima di 256 caratteri.

- I due files devono avere le stesse caratteristiche, vale a dire stessa lunghezza e formato dei records.

ESEMPI:
1) COMPARE UNO SYSIN DUE SYSIN
Verranno stampate a terminale le coppie di records corrispondenti diversi. Se in un file si raggiunge la fine prima che nell' altro sara' stampato il messaggio:

EOF FILE n

dove n sara' 1 o 2.

**CPFUNCTN** PERMETTE DI EMETTERE FUNZIONI DI CONSOLE DEL CP SENZA LASCIARE IL CMS.

```
CPFUNCTN        NOMSG     stringa
CP
```

NOMSG — argomento opzionale che impedisce la stampa di messaggi CP per argomenti e richieste non valide

stringa — una funzione di console del CP

NOTA - Con questo comando e' possibile incorporare funzioni di console del CP in files EXEC.

ESEMPI:

1) CP XFER 00D TO *

files AB SYSIN del disco A e AC SYSIN del disco P.

COMPARE ESEGUE IL CONFRONTO TRA DUE FILES SU DISCO.

COMPARE filename1 filetype1 filename2 filetype2

NOTA - Il confronto viene effettuato tra records corrispondenti dei due files.

- Se i due files sono identici, viene stampato un messaggio a terminale.

- Quando vengono trovati due records corrispondenti diversi, viene stampato a terminale il record del primo file, seguito da quello corrispondente del secondo file.

- Un messaggio avverte se viene raggiunta la fine di un file , prima della fine dell'altro file.

- I records da confrontare possono avere la lunghezza massima di 256 caratteri.

- I due files devono avere le stesse caratteristiche, vale a dire stessa lunghezza e formato dei records.

ESEMPI:
1) COMPARE UNO SYSIN DUE SYSIN
Verranno stampate a terminale le coppie di records corrispondenti diversi. Se in un file si raggiunge la fine prima che nell' altro sara' stampato il messaggio:

EOF FILE n

dove n sara' 1 o 2.

| | |
|---|---|
| CPFUNCTN | PERMETTE DI EMETTERE FUNZIONI DI CONSOLE DEL CP SENZA LASCIARE IL CMS. |

CPFUNCTN NOMSG stringa
CP

| | |
|---|---|
| NOMSG | argomento opzionale che impedisce la stampa di messaggi CP per argomenti e richieste non valide |
| stringa | una funzione di console del CP |

NOTA - Con questo comando e' possibile incorporare funzioni di console del CP in files EXEC.

ESEMPI:
1) CP XFER 00D TO *

CSET — PRIMA FUNZIONE:
LEGGE DA DISCO O SCRIVE SU DISCO CIO' CHE DOVREBBE ESSERE LETTO O SCRITTO SU UNA UNITA' DIVERSA (TERMINALE, LETTORE, PERFORATORE O STAMPANTE).

```
                PRINT   ON
                        OFF   <nome<tipo>>
                              FILE  PRINTOUT
                PUNCH   ON
                        OFF   <nome<tipo>>
                              FILE  CARDOUT
                RDR     ON
CSET                    OFF   <nome<tipo>>
                              FILE  CARDIN
                READ    ON
                        OFF   <nome<tipo>>
                              FILE  CONSIN
                TYPE    ON
                        OFF   <nome<tipo>>
                              FILE  CONSOUT
```

| | |
|---|---|
| PRINT | indica le scritture su stampante |
| PUNCH | indica le scritture su perforatore di schede |
| RDR | indica le letture dal lettore di schede |
| READ | indica le letture da terminale |
| TYPE | indica le scritture su terminale |

NOTE -Il comando CSET XXXXXXXX OFF nome tipo fa si che l'operazione di I/O indicata da XXXXXXXX avvenga sul file nome tipo. Se il nome o il tipo non sono specificati vengono assunti i valori default specificati nello schema di sopra.

-Il comando CSET XXXXXXXX ON riporta l'operazione XXXXXXXX al modo normale.

-Il comando CSET TYPE OFF fa andare su file tutte le scritture destinate al terminale, eccetto i messaggi di CP, i messaggi:

```
R;.....
E(0000n);.....
```

alla fine di ogni comando ed il messaggio

XXX P-DISK(191) IS FULL XXX

-Il comando CSET READ OFF fa si che le letture da terminale vengano fatte da file. Sono pero' escluse le letture dei comandi di CMS. Alla fine del file viene automaticamente emesso un CSET READ ON in modo che da quel momento in poi le letture procedano in modo normale.
I formati dei files di scrittura sono i seguenti:

per CSET TYPE OFF
Variabile, lunghezza massima =65535

per CSET PRINT OFF
Fisso, lunghezza =133

per CSET PUNCH OFF
Fisso, lunghezza =80

I formati dei files di lettura possono essere qualunque, dal momento che la lista di chiamata per la lettura da disco viene sistemata a partire dalla FST al momento del comando CSET. Se pero' un comando CSET READ OFF o CSET RDR OFF viene emesso quando il file di lettura non esiste ancora sul disco, il file e' considerato di formato fisso e con records di 80 caratteri e viene stampato il messaggio WARNING:

INPUT FILE NOT FOUND

E' stato emesso un comando CSET READ OFF o CSET RDR OFF quando non esisteva ancora il file di lettura corrispondente. Si suppone allora che il file abbia records lunghi 80 caratteri.

SECONDA FUNZIONE:
PERMETTE DI SCRIVERE UN MESSAGGIO 'READY' BREVE AL TERMINE DI OGNI COMANDO, DI ESEGUIRE COMANDI DI CP DAL CMS E DI IMPEDIRE L'ESECUZIONE DEI FILES DI TIPO EXEC.

CSET { ON / OFF } funzione

la funzione puo' essere:

RDYMSG per provocare la stampa solo di R; o E(0000N); al termine di ogni comando

quando e' OFF.
Per stampare il messaggio completo quando e' ON.

CP — prima di scrivere il messaggio 'INVALID CMS COMMAND' tenta di eseguire tale comando come funzione di CP.

IMPEX — per sopprimere la ricerca del file di tipo EXEC, quando un comando e' introdotto da terminale (quando e' ON). Una procedura EXEC potra' essere eseguita in questo caso solo scrivendo EXEC nome.

Messaggi di ERRORE:
E(00003); un parametro e' sbagliato.

CVTFV — CONVERTE UN FILE CON RECORDS A LUNGHEZZA FISSA IN UN FILE CON RECORDS A LUNGHEZZA VARIABILE.

CVTFV filename filetype <(T)>

(T) specifica che devono essere cancellati tutti i blanks di coda.

NOTA - Se un file consiste di records di 80 caratteri ed e' specificata l'opzione (T), vengono cancellate le colonne 73-80, qualunque sia il loro contenuto.

- Il file originale viene cancellato ed il nuovo file prende l'identificatore del primo file.

- Questo comando e' utile per incorporare un file di formato fisso in un file SCRIPT.

CVTVF — COPIA IL CONTENUTO DI UN FILE SU UN ALTRO FILE CON RECORDS DI LUNGHEZZA SPECIFICATA.

CVTVF nome1 tipo1 modo1 nome2 tipo2 modo2 <nnnn>
80

dove nnnn e' un numero decimale.

NOTE -Il contenuto di nome2 tipo2 modo2 viene copiato su nome1 tipo1 modo1; i records di nome1 tipo1 modo1 sono di lunghezza nnnn se e' specificata, altrimenti sono di 80 caratteri.
I records del file di input piu' corti di nnnn caratteri vengono allungati con blanks, quelli piu' lunghi vengono troncati.
Se qualche record deve essere troncato viene stampato il messaggio:

SOME RECORDS WILL BE TRUNCATED.

Se il file nome2 tipo2 modo2 non esiste viene stampato il messaggio

INPUT FILE NOT FOUND.

N.B. -Se nome1 tipo1 modo1 gia' esiste, viene cancellato all'inizio del comando. Per questa ragione i due files devono essere diversi.

ESEMPI:

1) CVTVF B B P1 A A A1 25

Il contenuto del file A A A1 viene copiato su B B P1, dopo essere stato troncato alla venticinquesima colonna.

Messaggi di errore:

E(00021), ERRORS IN ITEM LENGTH INDICATION. Il settimo argomento non e' un numero decimale.

E(00022); INPUT FILE NOT FOUND. Il file nome2 tipo2 modo2 non esiste.

Ogni altro codice di errore corrisponde ad un errore nella lettura o nella scrittura dei files, ed ha lo stesso significato dei corrispondenti codici di errore di RDBUF e WRBUF.

**DEBUG** RENDE DISPONIBILI UN INSIEME DI FUNZIONI CHE PERMETTONO L'ESAME IN LINEA DEL CONTENUTO E FUNZIONAMENTO NONCHE' LA CORREZIONE DI UN PROGRAMMA.

DEBUG
DE

NOTE - Si entra nell'ambiente DEBUG
1) emettendo il comando DEBUG,
2) quando ha luogo una interruzione esterna,
3) quando durante l'esecuzione di un programma si incontra un punto di arresto,
4) quando ha luogo una interruzione di programma,
5) quando ha luogo un errore non rimediabile.

- In questo ambiente sono valide le richieste di DEBUG; cinque di queste GO, IPL, KX, RETURN, e RESTART, fanno uscire dall'ambiente di DEBUG.

- Il comando DEBUG, se emesso per l'esame di un certo programma, va dato dopo aver caricato in memoria tale programma.

RICHIESTE DI DEBUG:

- Tutte le richieste di DEBUG possono essere abbreviate ai primi due caratteri, eccetto X,RETURN e RESTART.

- Gli indirizzi specificati nelle richieste di DEBUG sono assoluti fintantoche' non si definisca una origine per gli indirizzi con la richiesta ORIGIN, ed e' in questo senso che si parla di indirizzi relativi all'origine corrente.

**BREAK** PERMETTE ALL'UTENTE DI FERMARE L'ESECUZIONE DI UN PROGRAMMA ALL'INDIRIZZO DI MEMORIA DI UNA SPECIFICA ISTRUZIONE.

BREAK id { simbolo / hexloc }

| | |
|---|---|
| id | e' un numero decimale fra 0 e 15 |
| simbolo | e' il nome assegnato (tramite la DEF) a un indirizzo di memoria |
| hexloc | e' la locazione di memoria in esadecimale (relativa all'origine corrente) in cui porre un punto di arresto. |

NOTE -Ogni richiesta di BREAK pone un solo punto di arresto identificato dal numero posto in id; si possono porre fino a 16 punti di arresto.

-Se il numero di identificazione e' uguale ad uno gia' definito, il vecchio punto di arresto corrispondente viene cancellato.

-I simboli e le locazioni esadecimali devono sempre riferirsi a locazioni contenenti codici di operazioni, cioe' al primo byte di una istruzione.

-Piu' punti di arresto possono essere assegnati a una medesima locazione di memoria. In questo caso l'esecuzione del programma si ferma a quell'istruzione tante volte quanti sono i punti di arresto ivi posti.

-Dopo aver caricato in memoria un programma e aver emesso il comando DEBUG, e con la richiesta BREAK aver posto un punto di arresto si puo' tornare al CMS con le richieste RETURN e iniziare l'esecuzione del programma col comando START.

-Si puo' riprendere l'esecuzione di un programma da un certo punto di arresto con la richiesta GO senza operandi.

-La richiesta BREAK sostituisce il byte del punto di arresto con un codice operativo invalido della forma EX, dove x rappresenta il numero d'ordine del punto di arresto.L'esecuzione di tale codice provoca l'ingresso nell'ambiente DEBUG nel quale tale codice e' riconosciuto e il byte prima sostituito viene ripristinato.

ESEMPI:

1) BREAK 5 12400

Si vuole arrestare l'esecuzione all'indirizzo 12400 con un punto d'arresto individuato come quinto.

CAW STAMPA A TERMINALE IN ESADECIMALE IL CONTENUTO DELLA CHANNEL ADDRESS WORD, COME E' STATO MEMORIZZATO QUANDO IL CONTROLLO E' PASSATO ALL'AMBIENTE DEBUG.

CAW

NOTA - Il contenuto della CAW ha questo significato

bits 0-3 chiave di protezione
bits 4-7 sempre zero
bits 8-31 indirizzo esadecimale della prima CCW associata con la prossima o la piu' recente START I/O.

CSW STAMPA A TERMINALE IN ESADECIMALE IL CONTENUTO DELLA CHANNEL STATUS WORD, COME E' STATO MEMORIZZATO QUANDO IL CONTROLLO E' PASSATO ALL'AMBIENTE DEBUG.

CSW

NOTA - Il contenuto della CSW ha questo significato

bits 0-3 chiave di protezione
bits 4-7 sempre zero
bits 8-31 indirizzo del prossimo CCW
bits32-47 stato del canale o dell'unita'
bits48-63 differenza tra il numero dei bytes specificati nell'ultima CCW eseguita e quelli veramente trasferiti.

DEF PERMETTE ALL'UTENTE DI ASSEGNARE UN NOME SIMBOLICO A UNO SPECIFICO INDIRIZZO DI MEMORIA E DI RIFERIRSI CON TALE NOME A QUELL'INDIRIZZO IN ALTRE RICHIESTE DI DEBUG.

DEF simbolo hexloc <numero di bytes>
4

| | |
|---|---|
| simbolo | e' il nome da assegnare all'indirizzo di memoria specificato nel secondo operando (nome da 1 a otto caratteri di cui uno almeno non numerico, e il primo diverso da asterisco) |
| hexloc | locazione di memoria, relativa all'origine corrente, a cui assegnare il nome specificato nel primo operando |
| numero-di-bytes | numero decimale fra 1 e 56 che specifica l'attributo lunghezza (in bytes) del simbolo specificato nel primo operando. Usato con la richiesta X. |

NOTE -Se il terzo operando e' omesso si assume un attributo di lunghezza di 4 bytes

-Si possono specificare fino a 16 simboli

- I simboli definiti rimangono validi fino a quando non vengono ridefiniti con un altro DEF oppure se non si rifa' IPL del CMS.

ESEMPI:
1) DEF LAURA 480
All'indirizzo ottenuto sommando l'origine corrente con 480 viene associato il simbolo LAURA.

DUMP — SI USA PER AVERE IL DUMP DEI CONTENUTI DI TUTTA O PARTE DELLA MEMORIA DELLA MACCHINA VIRTUALE SULLA STAMPANTE.

```
                 <simbolo1   simbolo2>
DUMP             ident  <hexloc1   hexloc2>
                          0          *
                                     3
```

| | |
|---|---|
| ident | specifica il nome che sara' stampato in testa al dump per identificarlo |
| simbolo1 | nome assegnato (mediante la DEF) all'indirizzo di memoria da cui deve iniziare il dump |
| simbolo2 | come sopra, per la locazione finale |
| hexloc1 | locazione esadecimale di memoria, relativa |

| | |
|---|---|
| | all'origine corrente, da cui iniziare il dump |
| hexloc2 | come sopra, per la locazione finale |
| * | indica che il dump deve terminare all'ultimo indirizzo della memoria virtuale dell'utente. |

NOTE -Se non si specificano il secondo e il terzo parametro, sara' usato l'indirizzo di memoria specificato nell'ultima richiesta di DUMP, oppure, se non ci sono state altre richieste di DUMP, si stampera' una parola a partire dalla locazione 0.
- Il dump porta l' intestazione ident FROM simbolo1 TO simbolo2.

ESEMPI:
1) DUMP ERRORE 0 1000
Si otterra' sulla stampante un dump della memoria della macchina virtuale dall'indirizzo 0 piu' origine corrente al 1000 piu' origine corrente.

GO — FA USCIRE DALL'AMBIENTE DEBUG E INIZIA L'ESECUZIONE DEL PROGRAMMA O AD UN INDIRIZZO SPECIFICATO DALL'UTENTE O ALL'INDIRIZZO CONTENUTO NEI BITS 40-63 DELLA VECCHIA PSW RELATIVA ALL'INTERRUZIONE CHE HA FATTO ENTRARE NELL'AMBIENTE DEBUG.

GO <simbolo>
GO <hexloc>

| | |
|---|---|
| simbolo | nome assegnato (mediante la DEF) all'indirizzo di memoria da cui deve iniziare l'esecuzione |
| hexloc | e' la locazione esadecimale di memoria, relativa all'origine corrente, da cui deve iniziare l'esecuzione. |

NOTA - Se non si specificano parametri l'esecuzione riprende dall'indirizzo contenuto nella vecchia PSW relativa all'interruzione che ha fatto entrare in DEBUG; questo vale anche per riprendere l'esecuzione dopo un punto d'arresto.

GPR — SI USA PER CONTROLLARE IL CONTENUTO DI UNO O PIU' REGISTRI, COME E' STATO MEMORIZZATO QUANDO IL CONTROLLO E' PASSATO ALL'AMBIENTE DEBUG. SE SI SPECIFICANO DUE OPERANDI , SARA' STAMPATO A TERMINALE IL CONTENUTO DI TUTTI I REGISTRI, A PARTIRE DAL PRIMO SPECIFICATO FINO AL SECONDO SPECIFICATO INCLUSO.

GPR reg1 <regN>

reg1 — numero decimale da 0 a 15 incluso, che indica il primo o solo, registro il cui contenuto sara' stampato a terminale

regN — numero decimale da 0 a 15 incluso (maggiore di reg1), che indica l'ultimo registro il cui contenuto sara' stampato a terminale.

IPL — PERMETTE DI USCIRE DALL'AMBIENTE DEBUG RICOPIANDO IN MEMORIA UNA NUOVA COPIA DEL NUCLEO CMS.

IPL

NOTA - Questa richiesta ripristina le condizioni iniziali di CMS, cancellando eventuali simboli definiti tramite la richiesta DEF.

KX — CHIUDE TUTTI I FILES APERTI E LE UNITA' DI I/O, AGGIORNA LA LISTA DEI FILES SU DISCO, E PORTA IN MEMORIA UNA NUOVA COPIA DEL NUCLEO CMS.

KX

NOTE - Questa richiesta, come IPL e RESTART, ricopia il nucleo del CMS nella memoria della macchina virtuale previo azzeramento di tutta la memoria, permettendo all'utente di ricominciare a lavorare con una macchina

virtuale "vergine".

- I simboli definiti con DEF vengono cancellati.

ORIGIN — PERMETTE ALL'UTENTE DI SPECIFICARE UNA "ORIGINE" O INDIRIZZO BASE, CHE SARA' AGGIUNTO ALLE LOCAZIONI ESADECIMALI SPECIFICATE IN ALTRE RICHIESTE DEBUG.

ORIGIN simbolo
hexloc

simbolo — nome che e' stato assegnato (tramite la DEF) a un indirizzo di memoria

hexloc — locazione esadecimale di memoria compresa tra 0 e la fine della memoria virtuale dell'utente.

NOTE - La richiesta ORIGIN permette di specificare indirizzi di istruzioni relativi a punti di caricamento del programma.

- In mancanza della richiesta ORIGIN, tutte le locazioni esadecimali specificate nel DEBUG sono relative a 0.

ESEMPI:
1) ORIGIN 12000
Permette di far riferimento in altre richieste di DEBUG all'indirizzo 12000 che viene generalmente usato dal CMS come punto di caricamento iniziale dei programmi.

PSW — SCRIVE A TERMINALE IL CONTENUTO DELLA VECCHIA PSW RELATIVA ALL'INTERRUZIONE CHE HA FATTO ENTRARE NELL'AMBIENTE DEBUG.

PSW

NOTA - Nel caso di interruzioni diverse da esterne e di programma, l'indirizzo di memoria contenuto nella PSW stampata e' l'indirizzo esadecimale del programma DEBUG.

- Per informazioni piu' dettagliate riferirsi al manuale IBM System 360 Principles of Operation.

RESTART — FA USCIRE DALL'AMBIENTE DEBUG RICOPIANDO IN MEMORIA UNA NUOVA COPIA DEL NUCLEO CMS.

RESTART

NOTA - Questa richiesta e' analoga alla richiesta IPL.

RETURN — FA USCIRE DALL'AMBIENTE DEBUG E ENTRARE NELL'AMBIENTE CMS.

RETURN

NOTA -La richiesta dovrebbe essere usata solo quando si e' entrati nell'ambiente DEBUG tramite il comando DEBUG.

SET — SERVE A CAMBIARE IL CONTENUTO DEI REGISTRI GENERALI E DI CONTROLLO.

```
SET   CAW  hexinfo
      CSW  hexinfo   <hexinfo>
      PSW  hexinfo   <hexinfo>
      GPR  reg       hexinfo  <hexinfo>
```

CAW — l'informazione specificata deve essere memorizzata nella CAW esistente al momento in cui siamo entrati nel DEBUG

CSW — come sopra per la CSW

PSW — come sopra per la PSW

GPR — l'informazione specificata deve essere memorizzata nel GPR indicato nel secondo operando

NOTA -Ogni operando hexinfo deve contenere da due a otto cifre esadecimali (da uno a quattro bytes). Per la CSW,

PSW, GPR e' possibile specificare fino a otto bytes di informazione, cioe' quando si specificano due operandi esadecimali, le informazioni vengono memorizzate in locazioni consecutive.
Vedere la nota relativa alla richiesta STORE per quanto riguarda gli allineamenti.

ESEMPI:
1) SET PSW 01000000 00016824
Il contenuto dell' intera PSW che ha causato l'entrata nell'ambiente DEBUG viene sostituito con le informazioni fornite.

STORE — PERMETTE DI MEMORIZZARE INFORMAZIONI IN UNA QUALSIASI LOCAZIONE DELLA MEMORIA VIRTUALE.

STORE { simbolo / hexloc } hexinfo <hexinfo> <hexinfo>>

simbolo — nome assegnato (usando la richiesta DEF) all'indirizzo di memoria a partire dal quale deve essere memorizzata l'informazione specificata

hexloc — indirizzo esadecimale (relativo all'origine corrente) da dove deve essere memorizzato il primo byte d'informazione

hexinfo — informazioni esadecimali, quattro bytes o meno di lunghezza da memorizzare a partire dall'indirizzo specificato nel primo operando.

NOTE - Se un operando e' lungo meno di quattro bytes e contiene un numero dispari di cifre esadecimali, l'informazione sara' allineata a destra e la meta' a sinistra del byte dispari (primo byte) conterra' zero; se si specificano piu' di otto cifre esadecimali in un singolo operando, l'informazione sara' allineata a sinistra e troncata a destra a otto cifre.

- Si possono storizzare fino a 12 bytes (24 cifre) usando una richiesta store.

**TIN** DETERMINA COME IL DEBUG DEVE TRATTARE LE OPERAZIONI DI I/O NELL'AMBIENTE DEBUG, O TRAMITE LE ROUTINES NORMALI DI I/O DEL CMS O TRAMITE IL DEBUG STESSO.

| | CMS |
|---|---|
| TIN | DEB |

CMS le operazioni di I/O nell'ambiente DEBUG saranno trattate dal CMS (quando si sia entrati di proposito in questo ambiente); e'cosi' possibile far eseguire richieste stacked in un file EXEC.

DEB accetta l'input all'ambiente DEBUG solo da terminale in quanto fa trattare dal DEBUG direttamente le operazioni di I/O.

**X** QUESTA RICHIESTA E' USATA PER ESAMINARE IL CONTENUTO DI LOCAZIONI DI MEMORIA VIRTUALE SPECIFICATE; IL CONTENUTO DELLE LOCAZIONI INDICATE VIENE SCRITTO A TERMINALE IN ESADECIMALE.

X simbolo <n> lunghezza
X hexloc <n>
4

simbolo nome assegnato (usando la richiesta DEF) all'indirizzo di memoria del primo byte da esaminare

hexloc locazione esadecimale di memoria (relativa all'origine corrente) del primo byte da esaminare

n numero decimale da 1 a 56 incluso, che specifica il numero di bytes da esaminare

lunghezza attributo lunghezza del simbolo specificato come primo operando

NOTA -Se il primo operando e' un simbolo e non e' specificato il secondo operando, si stampano tanti bytes quanti indicati dall'attributo lunghezza del simbolo definito nella richiesta DEF.

ESEMPI:
1) X 232 20
Saranno stampati venti caratteri a partire dall'indirizzo 232 piu' l'origine corrente.

DETACH — DISCONNETTE DALLA CONFIGURAZIONE DELLA MACCHINA VIRTUALE L'UNITA' INDICATA.

DETACH indirizzo
DE

indirizzo — e' l'indirizzo virtuale dell'unita' che si vuole staccare.

NOTA - Questa funzione serve a disconnettere qualunque tipo di unita'; mentre la funzione contraria dell'ATTACH e' privilegio dell'operatore il quale puo' dietro richiesta, per es. tramite MSG, attaccare alla configurazione di una macchina virtuale altre unita' come per esempio unita' nastro, dischi etc.

- Vedere anche la funzione di CP LINK.

**DIAL** PERMETTE DI CONNETTERE UN TERMINALE AD UN SISTEMA IN MACCHINA VIRTUALE CAPACE DI GESTIRE PIU' TERMINALI.

DIAL macchina virtuale
D

macch.virt. nome della macchina virtuale che gestisce un sistema a piu' terminali, per es.:APL/DOS.

NOTA - In questo caso il terminale e' sotto diretto controllo del sistema che lo gestisce.

**DISAS** PERMETTE DI RICOSTRUIRE IN ASSEMBLER UN FILE IN LINGUAGGIO OGGETTO RILOCABILE.

DISAS nome

nome nome del file il cui tipo deve essere TEXT

NOTE - Il comando DISAS richiede, per funzionare correntemente, un nastro scratch attaccato alla macchina virtuale con indirizzo 180

- Una volta terminata l'esecuzione del comando compare il messaggio di CP 'CP ENTERED' a cui occorre rispondere 'CLOSE 00E' per ottenere effettivamente i risultati sulla stampante.

- I risultati saranno una ricostruzione, per quanto possibile, del programma ASSEMBLER da cui e' derivato il file TEXT.

**DISCONN** DISCONNETTE IL TERMINALE DELLA MACCHINA VIRTUALE, MENTRE QUESTA RESTA IN FUNZIONE.

DISCONN <stringa>
DIS

stringa e' una qualsiasi stringa di caratteri non-blanchi; se specificata la linea di trasmissione non viene disattivata.

NOTA - La macchina virtuale disconnessa resta in funzione fino a quando non termina il programma in esecuzione, se questo non richiede di leggere da terminale; quando la macchina entra nello stato di "wait" disabilitato, cioe' quando termina di eseguire programmi, la macchina virtuale viene staccata dal sistema (LOGOUT)).

- Ogni scrittura a terminale quando la macchina e' disconnessa, viene ignorata mentre ogni richiesta di lettura provoca LOGOUT.

DISK TRASFERISCE AL PERFORATORE DI SCHEDE IL FILE SU DISCO SPECIFICATO, CONVERTENDO I RECORDS IN FORMATO SCHEDA CMS, OPPURE LEGGE SU DISCO DAL LETTORE DI SCHEDE UNO O PIU' FILES IN FORMATO SCHEDA CMS, SCRIVENDOLI IN RECORDS DI 800 BYTES.

DISK DUMP filename filetype<filemode>
LOAD

DUMP specifica un file da perforare identificato da filename, type e mode
LOAD specifica che uno o piu' files su schede devono essere letti sul disco.

Le schede hanno il formato seguente, formato CMS

| colonna | contenuto |
|---|---|
| 1 | perforazione 12-2-9 |
| 2-4 | CMS |
| 5 | blank, o N per EOF |
| 6-55 | dati del file |
| 56-57 | blank |
| 58-76 | filename, type e mode |
| 77-80 | numero di sequenza |

NOTE - Per l'operazione DISK DUMP occorre specificare nome tipo e modo del file da perforare; il formato del file puo' essere fisso o variabile; se il modo e' omesso, la ricerca e' effettuata nell'ordine normale. Quando tutti i dati sono stati perforati, viene perforata una scheda con "N" in colonna 5 e informazioni per il directory.

- Questo comando permette il trasferimento tra macchine virtuali o la perforazione di files di formato variabile per es. di tipo MODULE o SCRIPT.

- Il comando DISK LOAD legge tutti i files, che devono essere in formato CMS, che sono sul lettore.

ESEMPI:
1) DISK DUMP DOPO DOMANI
Il file DOPO DOMANI sara' perforato in formato CMS e rileggibile col comando DISK LOAD.

| | |
|---|---|
| DISPLAY | STAMPA A TERMINALE IN ESADECIMALE IL CONTENUTO DI AREE DI MEMORIA REGISTRI PSW DELLA MACCHINA VIRTUALE. |
| DISPLAY<br>D | <hexloc><<hexloc1><hexloc2>><br><L<hexloc>><L<hexloc1>-<hexloc2>><br><T<hexloc>><T<hexloc1>-<hexloc2>><br><K<hexloc>><K<hexloc1>-<hexloc2>><br><G<reg>><G<reg1>-<reg2>><br><Y<reg>><Y<reg1>-<reg2>><br><X<reg>><X<reg1>-<reg2>><br><PSW> |
| hexloc<br>hexloc1<br>-hexloc2 | Indirizzo in esadecimale. Se specificato solo un indirizzo sara' stampata una parola da quell'indirizzo. Se e' omesso il primo indirizzo si partira' dalla locazione zero, se e' omesso il secondo si andra' fino alla fine della memoria. (Vale anche per L,T,K) |
| L | stampa in esadecimale |
| T | stampa in esadecimale e in EBCDIC |
| K | stampa in esadecimale le chiavi di memoria relative agli indirizzi specificati. |
| Greg | stampa in esadecimale il contenuto dei |

registri generali specificati.

Greg1-reg2 Nel caso di piu' registri (reg1<reg2), sono considerati tutti i registri da reg1 a reg2.

Yreg come sopra per i registri floating-point.
Yreg1-reg2

Xreg come sopra per i registri di controllo (solo per CP/67 virtuale).
Xreg1-reg2

PSW STAMPA IN ESADECIMALE IL CONTENUTO DELLA PSW (DUE PAROLE).

NOTA - Per far cessare la stampa a terminale relativa a una richiesta di DISPLAY basta premere il tasto ATTENTION.

ESEMPI:

1) D 12000
Sara' stampato in esadecimale il contenuto della parola di indirizzo 12000

2) D 11600-2000
Sara' stampato in esadecimale il contenuto della memoria dall'indirizzo 1600 al 2000

3) D g0
sara' stampato in esadecimale il contenuto del registro zero.

DSNAME ASSOCIA AL NOME E TIPO DI UN FILE UN TERZO NOME, IN MODO CHE SIA POSSIBILE USARE IL COMANDO FILEDEF IN RIFERIMENTO A QUEL FILE.

DSNAME <nome <tipo <ddname <CLEAR>>>>
* * *

nome e' il nome del file che si vuole definire con un FILEDEF

tipo e' il tipo del file che si vuole definire con un FILEDEF

ddname e' il nome da usare nel comando FILEDEF per fare riferimento al file nome tipo

NOTE -1) Non e' presente l'opzione CLEAR

a) il terzo parametro non e' presente:
Se e' gia' stato attribuito un ddname al file 'nome tipo', viene stampata su terminale la terna

nome tipo ddname.

Se il nome e/o il tipo non sono specificati o in corrispondenza di loro c'e' un asterisco, le definizioni gia' fatte vengono stampate per ogni nome e/o tipo.

b) il terzo parametro e' presente:
Al file 'nome tipo' viene associato il ddname. Se gia' una definizione era stata data per quel nome e tipo viene stampato il messaggio

WAS nomevecchio tipovecchio ddnamevecchio.

2) E' presente l'opzione CLEAR

Una eventuale precedente definizione del file 'nome tipo' viene cancellata.
Gli asterischi in corrispondenza del nome e/o del tipo significano che la cancellazione deve avvenire per ogni nome e/o tipo.

N.B. le seguenti definizioni sono implicite e non possono essere cancellate:

DSNAME FILE tipo tipo

ESEMPI:

1) DSNAME
Vengono stampate tutte le precedenti definizioni date con il comando DSNAME.

2) DSNAME * FORTRAN Vengono stampate tutte le precedenti definizioni date con il comando DSNAME a files di tipo FORTRAN

3) DSNAME PROVA PLI READER
Al file PROVA PLI viene associato il ddname READER

4) DSNAME GAUSS * * CLEAR
Vengono cancellate tutte le precedenti definizioni di files di nome GAUSS.

Messaggi:

1) XXXXXXXXYYYYYYYYZZZZZZZZ
E' stato emesso il comando DSNAME con non piu' di 2 argomenti. La risposta indica che precedentemente era stato emesso il comando
DSNAME XXXXXXXX YYYYYYYY ZZZZZZZZ

2) WAS XXXXXXXXYYYYYYYYZZZZZZZZ
Il comando DSNAME ha associato al file XXXXXXXXYYYYYYYY un nuovo ddname. L'utente viene in questo modo avvisato che il precedente ddname era ZZZZZZZZ.

Messaggi di errore:
Nessuno.

DUMP — STAMPA SULLA STAMPANTE IN ESADECIMALE IL CONTENUTO DI AREE DI MEMORIA, REGISTRI, PSW DELLA MACCHINA VIRTUALE.

DUMP parametri
DU
parametri — come per il comando DISPLAY

NOTA - L'uscita sulla stampante reale avviene solo dopo aver dato un comando di CP CLOSE 00E.

DUMPD — TRASFERISCE ALLA STAMPANTE IL CONTENUTO (IN ESADECIMALE) DEL RECORD SU DASD SPECIFICATO DALL' INDIRIZZO CCHHRR.

DUMPD CCU CC <HH <RR>>

CCU — indirizzo virtuale del dispositivo ad accesso diretto (DASD)
CC HH RR — rispettivamente numero di cilindro, di traccia e di record che identificano il record da stampare.

NOTE - Se non e' specificato alcun indirizzo CCHHRR o dopo che un record e' stato stampato, DUMPD richiede a

terminale l'indirizzo di un altro record da stampare. Per terminare l'azione di DUMPD, emette un ritorno del carrello senza aver scritto alcun carattere.

- Se si specifica solo CC verra' stampato il contenuto dell'intero cilindro; specificando CC HH sara' stampato il contenuto di una traccia.

ESEMPI:
1) DUMPD 191 01 03
Il contenuto della terza traccia del primo cilindro sara' stampato in esadecimale sulla stampante.

DUMPF — STAMPA A TERMINALE IL CONTENUTO (IN ESADECIMALE) DEL FILE SPECIFICATO, O DI UNA SUA PARTE.

DUMPF filename filetype <n1 n2 <n3>>

* 80

n1,n2 — specificano nell'ordine il numero di linea della prima e dell'ultima linea da stampare

* — un asterisco al posto di n1 o n2 indica rispettivamente l'inizio o la fine del file. Due asterischi o l'omissione di questi due parametri provocano la stampa di tutto il file

n3 — specifica, se i records devono essere troncati, il numero massimo di caratteri da stampare per linea.

NOTA - La ricerca del file viene effettuata nell'ordine normale.

ESEMPI:
1) DUMPF STELLA SYSIN * 30
Saranno stampati a terminale in esadecimale i primi trenta records del file STELLA SYSIN.

DUMPREST — COPIA SU NASTRO (INDIRIZZO VIRTUALE 181) IL CONTENUTO DI UN DISCO (INDIRIZZO VIRTUALE 191) O RICOPIA SU DISCO IL CONTENUTO DI UN DISCO REGISTRATO SU NASTRO.

DUMPREST

NOTA - Dopo l'emissione del comando viene stampato a terminale un messaggio che richiede di specificare l'operazione; la risposta deve essere;

DUMP per copiare il contenuto del disco su nastro

RESTORE perche' venga eseguita l'operazione di "format" del disco, e vi venga copiato il contenuto del nastro.

- L'operazione di RESTORE va effettuata sullo stesso numero di cilindri che l'operazione di DUMP che ha creato il nastro.

ECHO CONTROLLA LA TRASMISSIONE DA TERMINALE RISCRIVENDO SUL TERMINALE LA LINEA BATTUTA.

ECHO <U|S|X> <nn>

Opzioni:

| | |
|---|---|
| U | interpreta i caratteri di cancellazione e converte le lettere miniscole in maiuscole |
| S | interpreta i caratteri di cancellazione, ma non converte le lettere maiuscole in minuscole |
| X | lascia la linea inalterata |
| nn | numero di volte che la linea viene ristampata (default 1) |

NOTA - Questo comando fa entrare nell'ambiente ECHO in cui tutte le linee trasmesse vengono riflesse a terminale. Per ritornare all'ambiente CLI battere la parola RETURN a partire da colonna 1.

EDIT — SERVE A CREARE NUOVI FILES DAL TERMINALE O A MODIFICARE FILES GIA' ESISTENTI.

EDIT
E — nome tipo

nome — e' il nome del file da modificare o creare
tipo — e' il tipo del file da modificare o creare

NOTA - Per prima cosa viene esaminato il tipo del file ed in base a questo vengono stabiliti la lunghezza dei records del file, la posizione delle tabulazioni e varie altre caratteristiche del file. Quindi, se il file non esiste, si passa nell'ambiente di input, nel quale ogni riga battuta al terminale viene posta nel file; se invece il file esiste, si passa nell'ambiente di EDIT nel quale l'utente ha a disposizione un certo numero di "richieste" (Comandi in ambiente EDIT). In quest'ultimo caso, se il formato e/o la lunghezza dei records del file esistente non corrispondono a quelli determinati in base al tipo, il comando termina con il messaggio

ERROR IN CALL TO RDBUF

Si consiglia in questo caso di alterare il tipo del file. L'editore lavora su una copia del file residente in memoria. Il file viene riscritto su disco quando si da' la richiesta di edit SAVE o la richiesta FILE. Queste due richieste provocano la scrittura sul file

$INPUT:$ tipo P1

Al termine della scrittura viene cancellata l'eventuale precedente copia del file nome tipo e il $INPUT:$ tipo P1 viene alterato in nome tipo P1. Se pero' nel corso della scrittura sul file il disco P viene riempito tutto, viene stampato sul terminale il messaggio

P-DISK IS FULL

e viene fatto automaticamente un nuovo ipl del CMS. Al termine dell'IPL la parte di file trascritta su disco e' ancora disponibile sotto il nome

$INPUT:$ tipo P1

Quando si emette il comando di EDIT, la ricerca

dell'eventuale copia precedente avviene su tutti i dischi nell'ordine standard di ricerca; pero' al momento di una richiesta di FILE o SAVE, la scrittura avviene sempre sul disco P. Se questo e' in sola lettura o non e' attivo, si entra in DEBUG.
Il passaggio dall'ambiente di EDIT all'ambiente di INPUT avviene mediante le richieste INPUT e SAVE; il passaggio da INPUT a EDIT avviene scrivendo una linea vuota.(Ritorno di carrello senza battere caratteri.).
Si ritorna in ambiente CMS con le richieste

FILE e QUIT.

## LOGICA DELL'EDITORE

### Puntatore.

Al file in corso di edizione e' associato un puntatore che individua una linea, detta "linea corrente", Vale a dire la linea da esaminare o modificare. Molte richieste di EDIT provocano lo spostamento del puntatore.
Quando si passa dall' ambiente di INPUT a quello di EDIT, il puntatore si trova sull'ultima linea scritta da terminale. Quando si passa dall'EDIT all'INPUT il puntatore e' posizionato in modo che le linee introdotte successivamente dal terminale seguano la linea che era corrente al momento del passaggio all'INPUT. All'inizio del comando di EDIT il puntatore e' posizionato su una riga vuota introdotta automaticamente dall'EDIT in testa al file.

### Caratteristiche associate al "tipo" di un file

Durante l'esecuzione del comando di EDIT, ogni riga (o record) del file e' considerata suddivisa in intervalli. Se in fase di INPUT viene premuto il tasto di tabulazione (in alto a sinistra sulla tastiera) o il carattere di tabulazione logica (normalmente '>'), i caratteri successivi della riga vengono scritti nel file a partire dall'inizio del primo di tali intervalli che si trovi a destra della posizione in cui e' stato battuto il carattere di tabulazione.
E' chiaro quindi che per ogni file e' necessario conoscere le colonne in cui cominciano questi intervalli. Queste colonne sono dette posizioni di tabulazione. Le posizioni di tabulazione vengono assegnate in base al tipo del file. A files di tipo non riconosciuto vengono assegnate le posizioni

1,5,10,15,20,25,30,35,40,45,50

segue la lista dei tipi riconosciuti dall'EDIT con le relative posizioni di tabulazione:

| Tipo | Posizioni |
|---|---|
| FORTRAN<br>SYSIN | 1,7,10,15,20,25,30 |
| ASP360<br>FLOW | 1,10,16,31,36,41,46,51,56,71 |
| PLI | 2,10,15,20,25,30,35,40,45,50,55,60 |
| SPL1 | 1,7,17,30,40,50,60 |
| SNOBOL | 1,10,20,25,30,35,40,45,50,55,60 |
| EXEC | 1,5,8,17,27,31 |
| REPS | 7,17,31,36 |
| AED<br>MAD | 1,10,15,20,25,30,35,40,45 |
| AS1130 | 21,27,32,35,45,50,55,60 |
| COBOL | 1,8,12,16,20,24,28,32,36 |

Ogni altro tipo e' considerato non standard e provoca la stampa del messaggio

## DEFAULT TABS SET

Le posizioni di tabulazione possono essere cambiate con la richiesta TABSET.

### Altre caratteristiche determinate in base al tipo

Nella tabella che segue sono specificate le caratteristiche che l'editore assegna ad un file in base al tipo:

| Tipo | Lunghezza della riga | Formato F/V | Ser. | Col.Tronc. | Conv. Min/Maiu. |
|---|---|---|---|---|---|
| SYSIN | 80 | F | SI | 71 | SI |
| ASP360 | 80 | F | SI | 71 | SI |
| COPY | 80 | F | SI | 71 | SI |
| UPDATE | 80 | F | NO | 71 | SI |
| AED | 80 | F | SI | 72 | SI |
| MAD | 80 | F | SI | 72 | SI |
| AS1130 | 80 | F | SI | 72 | SI |
| REPS | 80 | F | SI | 72 | SI |
| COBOL | 80 | F | SI | 72 | SI |
| FORTRAN | 80 | F | SI | 72 | SI |
| PLI | 80 | F | SI | 72 | SI |
| SPL1 | 80 | F | SI | 72 | SI |
| SNOBOL | 80 | F | SI | 72 | SI |
| FLOW | 80 | F | SI | 72 | SI |
| BASIC | 80 | F | SI | 72 | SI |
| SCRIPT | 132 | V | NO | 132 | NO |

| | | | | | |
|---|---|---|---|---|---|
| LISTING | 132 | F | NO | 132 | SI |
| MEMO | 80 | F | NO | 80 | NO |
| DATA | 80 | F | NO | 80 | SI |
| altri | 80 | F | NO | 80 | SI |

Nei tipi per i quali e' prevista la serializzazione, le colonne da 73 a 80 non sono accessibili in fase di EDIT, mentre, al momento della memorizzazione del file su disco vi vengono scritte le seguenti informazioni:

| Colonne | Contenuto |
|---|---|
| 73-75 | Prime tre lettere del nome del file. |
| 76-80 | Un numero progressivo a partire da 00010 e con incrementi di 10. |

La richiesta SERIAL permette di modificare il trattamento della serializzazione.

La colonna di troncatura indica la colonna oltre la quale viene cancellata ogni informazione introdotta dal terminale. Quando una riga viene troncata viene stampato il messaggio:

TRUNCATED

e viene stampata la riga cosi' come e' stata memorizzata.

Caratteri speciali.

Oltre ai caratteri speciali interpretati dalle routines di I/O del CMS (cancella-riga , cancella carattere, fine-riga ' '), l'EDIT interpreta in modo particolare anche il carattere di tabulazione logica '>' e lo spazio indietro. Il carattere di tabulazione logica puo' essere ridefinito con la richiesta TABDEF; lo spazio-indietro logico puo' essere ridefinito con la richiesta BACKSPACE.

Il carattere di spazio-indietro viene cosi' interpretato:

a) files di tipo SCRIPT
IL tasto di spazio indietro fisico da una parte genera il carattere che si inserisce nella colonna corrente della linea in costruzione, dall'altro provoca l'indietreggiamento della pallina. Questo simbolo permette in particolare di sottolineare un testo.
Esempio: abcd (BACK)(BACK)(BACK).....
provoca sul terminale la scrittura di abcd, e in memoria la registrazione di a(BACK)-b(BACK)-c(BACK)-

Viceversa il simbolo di spazio indietro logico non

viene inserito in una colonna della riga in costruzione. E' immediatamente interpretato e indica un indietreggiamento di una o piu' colonne per modificare il o i caratteri che vi comparivano per mezzo di quelli (non bianchi) che seguono:

Esempio: la scrittura di adddd%%c da' la linea:
abcd.

b) files non di tipo SCRIPT
I due simboli hanno lo stesso effetto interno dello spazio indietro logico per i files di tipo SCRIPT.

Richieste di EDIT

Nell'ambiente di EDIT sono disponibili alcune richieste che permettono all'utente di manipolare i files. Il formato di queste richieste e' simile a quello dei comandi di CMS: vi e' prima il nome della richiesta, seguito da una lista di argomenti separati da spazi bianchi. Due argomenti peculiari delle richieste di EDIT sono le stringhe e le linee:

a) linea: quando una richiesta ammette per parametro una 'linea', il parametro e' unico. La linea comincia immediatamente dopo il primo spazio bianco che segue la richiesta. Una 'linea' viene interpretata dall'editore esattamente come una riga scritta in INPUT. Vengono quindi interpretati nel modo consueto i caratteri di tabulazione e gli spazi-indietro.

b) stringa: una stringa e' un insieme di caratteri delimitato da una coppia di caratteri che non compaiono nella stringa.
Esempi: xabcdx
/abcd/
5abcd5
Nelle pagine che seguono una stringa sara' sempre rappresentata delimitata da una coppia di '/'.

AGAIN SERVE A RIPETERE IL COMANDO PRECEDENTE UNA O PIU' VOLTE.

AGAIN <n>
A 1

n specifica quante volte il comando sara' ripetuto. Se non e' specificato e' preso uguale a 1.

BACKSPACE — PERMETTE DI RIDEFINIRE IL CARATTERE DI SPAZIO INDIETRO LOGICO.

BACKSPACE <carattere>
BA

carattere — e' il carattere che assumera' il significato di spazio indietro logico.L' assensa del parametro ridefinisce ".

BLANK — CANCELLA CARATTERI DELLA RIGA CORRENTE NELLE POSIZIONI SPECIFICATE.

BLANK linea
BL

NOTA - I caratteri della 'linea' diversi da spazio bianco cancellano quelli corrispondenti della riga corrente.

BOTTOM — POSIZIONA IL PUNTATORE SULL'ULTIMA RICA DEL FILE.

BOTTOM
B

BRIEF — SOPPRIME LA STAMPA A TERMINALE DELLE LINEE TROVATE O MODIFICATE PER IL RESTO DELL'EDIZIONE SE NON SI DA' LA RICHIESTA CONTRARIA VERIFY.

BRIEF
BR

NOTA - Per sopprimere la stampa per una sola richiesta basta battere un punto subito dopo il nome della richiesta.

CHANGE — SOSTITUTISCE UNA STRINGA CON UN'ALTRA.

CHANGE /stringa1/stringa2/ <*<G>>
C n
1 *

NOTE - In assenza dei parametri facoltativi la prima

occorrenza della 'stringa1' nella riga viene sostituita dalla 'stringa2'. Il primo parametro facoltativo indica il numero di righe a partire dalla corrente in cui la sostituzione deve avere luogo. Un asterisco significa fino alla fine del file. La presenza di un G o di un asterisco come secondo parametro facoltativo fa si' che la sostituzione avvenga per tutte le occorenze della stringa nella riga.

- Le due stringhe possono avere lunghezza diversa; in questo caso la richiesta sposta a destra o a sinistra tutti i caratteri che seguono la stringa inserita.

DELETE — CANCELLA UNA O PIU' RIGHE DEL FILE A PARTIRE DALLA CORRENTE COMPRESA, POSIZIONANDO IL PUNTATORE SULLA PRIMA NON CANCELLATA.

DELETE
D
</stringa/>
< n >
1

stringa — indica che la cancellazione deve avvenire fino alla riga contenente la stringa esclusa.

n — indica il numero di righe che devono essere cancellate (La corrente e le n-1 successive).

NOTA - Se si e' specificato '/stringa/' e la fine del file viene raggiunta prima dell'incontro della stringa, non viene cancellato niente.

ERASE — PERMETTE DI CANCELLARE UN FILE RESTANDO IN EDIT.

ERASE nome tipo

nome — e' il nome del file da cancellare.
tipo — e' il tipo del file da cancellare.

FIND — POSIZIONA IL PUNTATORE SULLA PRIMA RIGA CONTENENTE UNA DETERMINATA SEQUENZA POSIZIONALE DI CARATTERI CIOE' POSTI IN POSIZIONI DETERMINATE DENTRO LA RIGA.

FIND
F
linea

NOTA - Per ogni riga a partire dalla linea successiva alla corrente: caratteri non bianchi di 'linea' vengono confrontati con quelli nelle corrispondenti posizioni della riga. Se sono uguali la ricerca ha termine e il puntatore resta posizionato su questa riga.

**FILE** TERMINA IL COMANDO DI EDIT COPIANDO IL FILE DALLA MEMORIA SUL DISCO P, E FACENDO RIENTRARE IN AMBIENTE CMS.

FILE
FIL

**GOTO** POSIZIONA IL PUNTATORE SULLA N-ESIMA RIGA DEL FILE.

GOTO n
G

n e' il numero d'ordine della riga sulla quale ci si vuole posizionare.

**GET** COPIA SUL FILE SUCCESSIVAMENTE ALLA RIGA CORRENTE UNA O PIU' RIGHE DA UN ALTRO FILE O DA UN'AREA DI MEMORIA DOVE SIANO STATE POSTE CON UNA RICHIESTA PUT PRECEDENTE.

GET < nome tipo > <n1 <n2>>
GE * * 1 EOF

nome e' il nome del file da cui devono essere copiate le righe
tipo e' il tipo del file da cui devono essere copiate le righe
* significa che la richiesta si riferisce allo stesso file del precedente GET
n1 e' il numero d'ordine della prima riga che deve essere coplata
n2 e' il numero dell'ultima riga da copiare.

NOTE - Se i parametri mancano o se il primo e' un numero (che in questo caso indica il numero di righe da prelevare) le righe vengono cercate nell'area di memoria in cui vengono poste dai PUT. Altrimenti

vengono prese dal file specificato.

- L'inserimento di righe all'interno del file provoca lo spostamento in basso delle righe successive a quella corrente preesistenti alle richieste.

- Il puntatore alla fine dell'inserimento resta posizionato sull'ultima riga inserita.

| | |
|---|---|
| INSERT | INSERISCE NEL FILE SUCCESSIVAMENTE ALLA RIGA CORRENTE LA LINEA SPECIFICATA. |
| INSERT linea<br>I | |
| INPUT | PERMETTE DI PASSARE DALLA FASE DI 'EDIT' A QUELLA DI 'INPUT'. LE RIGHE SCRITTE IN FASE DI INPUT VENGONO SUCCESSIVAMENTE ALLA RIGA CORRENTE. |
| INPUT<br>I | |

NOTA - Le righe successive a quella corrente, esistente al momento della richiesta, vengono spostate verso il basso dall' inserimento delle righe scritte in INPUT.

| | |
|---|---|
| LINENO | DA' IL NUMERO D' ORDINE DELLA RIGA CORRENTE. |
| LINENO<br>LI | |
| LOCATE | POSIZIONA IL PUNTATORE SULLA PRIMA RIGA SUCCESSIVA ALLA CORRENTE, IN CUI COMPARE UNA DETERMINATA STRINGA. |
| LOCATE /stringa/<br>L | |
| MDEF | PERMETTE DI DEFINIRE UNA MACRO, OSSIA UNA PROCEDURA COMPOSTA DI RICHIESTE EDIT E PAROLE DI CONTROLLO. |

MDEF nome <M>
M

nome — e' il nome della macro-richiesta che si intende definire

M — la presenza di M come secondo parametro fa si' che da definizione della macro venga letta dal file Nome MACRO esistente su disco. Il sistema risponde dicendo:

MDEF FROM FILE -O.K.

NOTA - In assenza del parametro M, si entra in un ambiente analogo all'INPUT dopo il messaggio

MACRO-DEFINITION:

e le righe scritte a partire da quel momento vengono inserite nella macro. Il ritorno all'EDIT avviene per mezzo della scrittura di una riga vuota. Una riga di una macro-richiesta puo' contenere qualunque richiesta di EDIT, oltre alle due seguenti richieste particolari:

& EXIT < TOP / EOF >.
Provoca l'arresto dell'esecuzione delle richieste contenute nelle macro e il ritorno in EDIT se durante uno spostamento del puntatore viene raggiunta una delle due estremita' del file. Specificando TOP o EOF l'uscita dalla macro avviene solo se il limite specificato e' raggiunto.

&GOTO n < TOP / EOF >. Fa eseguire la n-esima riga della macro. Specificando TOP o EOF il salto ha luogo solo se il puntatore non si trova all'estremita' specificata, altrimenti si precede in sequenza.
E' possibile inoltre nelle macro usare fino a due parametri (& 1 e & 2) in modo analogo a quanto avviene per le procedure EXEC. Al termine del comando di EDIT tutte le definizioni di macro date nel corso di esso vengono cancellate, a meno che non si siano salvate con la richiesta MSAVE.

MLIST — SERVE AD AVERE LA LISTA DELLE MACRO-RICHIESTE

ATTIVE O LA STAMPA DEL LORO CONTENUTO.

MLIST
ML

```
      nome
<      >
   *
```

nome — indica il nome della macro-richiesta di cui si vuole conoscere il contenuto

* — provoca la stampa dei contenuti di tutte le macro rese disponibili all'editore fino a quel momento.

NOTA - In assenza di parametri viene stampata una lista dei nomi delle macro attive.

MSAVE — COPIA UNA MACRO IN UN FILE SU DISCO CHE AVRA' COME NOME IL NOME DELLA MACRO E COME TIPO MACRO.

MSAVE
MS

```
nome
```

nome — e' il nome della macro da salvare e diviene il nome del file su disco.

NOTA - Se il file Nome MACRO gia' esiste viene cancellato per essere sostituito dal nuovo e viene dato il messaggio:

REPLACE OLD MACRO FILE

MUNDEF — SOPPRIME LA VALIDITA' DI UNA O TUTTE LE MACRO NEL CORSO DELLA PRESENTE FASE DI EDIT.

MUNDEF
MU

```
Nome
 *
```

Nome — e' il nome della macro da sopprimere

* — provoca la soppressione di tutte le macro.

NEXT — POSIZIONA IL PUNTATORE SULLA N-ESIMA RIGA DEL FILE SUCCESSIVA ALLA CORRENTE, O SULLA

PRIMA RIGA IN CUI SI TROVI UNA DATA STRINGA.

NEXT
N

/stringa/
< n >
1

n — specifica di quante righe si vuole abbassare il puntatore

stringa — fa posizionare il puntatore sulla prima riga in cui compaia tale stringa.

OVERLAY — SOSTITUISCE I CARATTERI IN DETERMINATE POSIZIONI DELLA RIGA CORRENTE CON ALTRI CARATTERI.

OVERLAY
O

linea

NOTA - I caratteri della 'linea' diversi da spazio bianco vanno a sostituire quelli nelle corrispondenti posizioni della riga corrente.

PRINT — SCRIVE SU TERMINALE N RIGHE DEL FILE A PARTIRE DALLA RIGA CORRENTE.

PRINT
P

<n <LINENO>>
1 L

n — e' il numero di righe che devono essere stampate

LINENO — provoca la scrittura delle colonne 73-80 anche per i tipi per cui e' prevista la serializzazione.

PUT — SERVE A COPIARE UNA O PIU' RIGHE DEL FILE A PARTIRE DALLA CORRENTE SU UN ALTRO FILE O IN UN'AREA DI MEMORIA DA CUI POSSONO ESSERE PRELEVATE CON LA RICHIESTA GET.

PUT
PU

/stringa/
< n <nome<tipo>>>
1 *

n — e' il numero di righe da porre nell'area di memoria o sul file.

| | |
|---|---|
| stringa | indica che devono essere copiate tutte le righe fino alla prima che contenga 'stringa' esclusa. |
| nome | e' il nome del file in cui devono essere copiate le righe |
| * | indica che le righe devono essere copiate sullo stesso file interessato dalla precedente richiesta di PUT. |
| tipo | e' il tipo del file in cui devono essere copiate le righe. Se il terzo parametro manca mentre e' presente il secondo il tipo e' considerato lo stesso del file in corso di edizione. |

NOTA - Se il file gia' esiste le righe vengono aggiunte in fondo. Altrimenti viene dato il messaggio:

NEW FILE

Se mancano il secondo e il terzo parametro le righe vengono poste in un'area di memoria del sistema.

| | |
|---|---|
| PUTD | FUNZIONA COME IL PUT MA PROVOCA LA CANCELLAZIONE DELLE RIGHE INTERESSATE DALL'OPERAZIONE. |
| QUIT | TERMINA IL COMANDO DI EDIT MA DIVERSAMENTE DA FILE NON PROVOCA LA TRASCRIZIONE DEL FILE DA MEMORIA SU DISCO |
| QUIT<br>Q | |
| REPLACE | SOSTITUISCE LA RIGA CORRENTE CON UN'ALTRA RIGA. |
| REPLACE<br>R | linea |
| linea | e' la 'linea' che deve sostituire la riga corrente. |
| SAVE | PERMETTE DI PASSARE DALL'EDIT ALL'INPUT MA, DIVERSAMENTE DALLA RICHIESTA INPUT, TRASCRIVE SU DISCO IL FILE, PERMETTENDO DI |

SALVARE LE MODIFICHE O LE AGGIUNTE FATTE FINO A QUEL MOMENTO.

SAVE
S

SERIAL — PERMETTE ALL'UTENTE DI SPECIFICARE I TRE CARATTERI DI IDENTIFICAZIONE PER LE COLONNE 73-74-75 DI UN FILE COSI' COME L'INCREMENTO DELLA SERIALIZZAZIONE FRA UNA RIGA E L'ALTRA. PERMETTE ANCHE DI SOPPRIMERE LA SERIALIZZAZIONE.

```
                ident   n
SERIAL           (NO) < 10 >
```

ident — specifica i tre caratteri per le colonne 73-75

(NO) — sopprime la serializzazione

n — indica l'incremento fra una riga e la successiva

NOTA - Se per ident si da' 000 il numero di sequenza occupa tutte le otto colonne 73-80.

TABDEF — PERMETTE DI RIDEFINIRE IL CARATTERE DI TABULAZIONE LOGICA.

```
TABDEF          carattere
TA          <             >
                ≥
```

carattere — e' il carattere cui deve essere attribuito il significato di carattere di tabulazione logica.

TABSET — SERVE A DEFINIRE POSIZIONI DI TABULAZIONE DIVERSE DA QUELLE IMPLICITE NEL TIPO DEL FILE.

```
TABSET          <n1<n2<n3<n4...>>>>
 TABS
```

n1 n2 etc... — sono le colonne su cui si vogliono porre le posizioni di tabulazione . Se i parametri

mancano vengono ristabilite le posizioni default.

UP — FA SALIRE IL PUNTATORE DI N RIGHE O FINO ALLA PRIMA RIGA CONTENENTE UNA DATA STRINGA.

```
          /stringa/
UP      <    n    >
U            1
```

n — specifica di quante righe si deve alzare il puntatore

stringa — indica che il puntatore deve salire fino alla prima riga contenente tale stringa.

VERIFY — SERVE, DOPO CHE SI SIA DATA LA RICHIESTA BRIEF, A FARE IN MODO CHE L'EDITORE TORNI A STAMPARE LE RIGHE MODIFICATE DA UNA RICHIESTA O RAGGIUNTE CON UNA RICHIESTA CHE SPOSTA IL PUNTATORE.

```
VERIFY
  V
```

ZONE — LIMITA LA RICERCA DELLE STRINGHE ALLA PARTE DI UNA RIGA COMPRESA FRA DUE COLONNE SPECIFICATE.

```
ZONE          n1          n2
  Z           1        troncamento
```

n1 n2 — indicano le colonne entro le quali deve essere effettuata la ricerca delle stringhe.

NOTA - In generale n2 non puo' superare il valore default, che e' la colonna di troncamento. Pero', per i tipi per cui il troncamento ha luogo a colonna 71 mentre la serializzazione comincia a colonna 73, e' possibile porre n2 uguale a 72, in modo da potere inserire un carattere di continuazione in una riga.

Edizione di files molto lunghi

Se si emette il comando EDIT per un file troppo grande, viene dato il seguente messaggio:

FILE TOO LARGE, USE'CEDIT' OR 'SCRIPT EDIT'

Se invece la capacita' della memoria viene superata mentre il file e' in corso di edizione, il messaggio stampato e':

CORE OVERFLOW-FILE STORED AS'EDTMPTDR',

che indica che la piu' grossa parte possibile del file in corso di edizione e' stata sistemata sul disco in 'EDTMPTDR tipo', dove il tipo e' lo stesso di quello del file editato. Ora, essendo il comando CEDIT troppo lento, si consiglia di usare il seguente metodo:

1) Si emetta il comando EDIT per un nuovo file di lavoro

2) Mediante un GET, si portino in memoria la parte iniziale del file da editare

3) Si lavori come desiderato sulla parte cosi' ottenuta

4) Con un PUTD si scriva tutto il file di lavoro su un nuovo file

5) Con un altro GET si ottenga una parte successiva del file

6) Si lavori su questa parte

7) Con PUTD si aggiunga questa parte al nuovo file, etc.

Esempio:
Si voglia lavorare sul file A SYSIN. Converra' procedere nel seguente modo:

```
Edit bidone sysin
NEW FILE
INPUT:

EDIT:
get a sysin 1 1000
top
.
.
.
.
```

```
.  (elab. delle prime 1000 righe di a sysin)
.
.
.
.
.
top
putd 1000 nuovoa sysin
EOF:
NEW FILE
get a sysin 1001 2000
top
.
.
.
.
etc....
.
.
.
.
putd 1000 nuovoa sysin
EOF:
erase a sysin
quit
FILE EMPTY - EXIT TAKEN
R;....
alter nuovoa sysin p1 a sysin p1
R;......
```

ERASE — CANCELLA UN FILE (O GRUPPO DI FILES CORRELATI), TOGLIENDO L'ENTRATA DEL FILE DAL DIRECTORY APPROPRIATO E RENDENDO DI NUOVO DISPONIBILE PER L'ALLOCAZIONE L'AREA DI DISCO OCCUPATA DAL FILE.

| | filename | filetype | <filemode> |
|---|---|---|---|
| ERASE | * | * | * |
| | | | P |

NOTE - Occorre specificare filename e filetype o per nome o con asterisco.

- Se filemode e' omesso, si considera valido il modo P; se e' specificato * la ricerca e' effettuata su tutti i dischi in lettura e scrittura.

- Per cancellare tutti i files su un disco e' piu' rapido usare il comando LOGIN-NO-UFD.

ESEMPI:
1) ERASE AT *
Tutti i files di nome AT e tipo qualunque presenti sul disco P sono cancellati.

## EXEC: Procedure EXEC

Una procedura EXEC permette di definire come un unico comando CMS una combinazione di comandi CMS e CP e di istruzioni di controllo EXEC. Il file contenente le procedure EXEC deve essere di tipo EXEC ed avere records di 80 caratteri. Deve contenere un comando per record.

Richiamo della procedura

L'esecuzione di una procedura puo' essere iniziata tramite il comando EXEC. Se la procedura non e' richiamata dall'interno di un'altra procedura, basta battere solo il nome della procedura in quanto, in questo caso, il CMS controlla se esiste una procedura EXEC di tal nome prima di ricercare un comando di tal nome.

EXEC nome <arg1 arg2....argN>

nome nome del file EXEC contenente la procedura
args parametri da passare alla procedura

In una procedura EXEC possono venire inserite fino a trenta variabili numeriche della forma &1 &2...&3. Prima dell'esecuzione di ogni comando, le variabili presenti nel comando vengono sostituite con i parametri corrispondenti passati col nome della procedura, andando il primo a sostituire &1, il secondo &2 e cosi' via.
Se al posto di un parametro si scrive " (doppio apice), la variabile corrispondente sara' ignorata.
I parametri possono essere concatenati a destra di una parola in comandi EXEC. Per esempio, ponendo EDIT FILE SYS&4 e passando come parametro IN, si otterra' il comando EDIT FILE SYSIN. Nel caso della concatenazione se al posto di un parametro compare " (doppio apice) la variabile corrispondente sara' ignorata e soppressa dalla parola da concatenare; se compare ' (apice) l'intera parola sara' soppressa. Nell'esempio precedente nel primo caso otteremmo EDIT FILE SYS, nel secondo caso EDIT FILE.
Il comando EXEC e' ricorsivo nel senso che una procedura puo' richiamare un'altra procedura e cosi' via fino ad un livello di ricorsivita' di 16.

Variabili

Le parole che iniziano per & (e commerciale) vengono considerate come parole particolari dell'EXEC. Si distinguono in variabili numeriche, variabili simboliche (d'utente e di sistema), parole di controllo.

Le variabili numeriche come sopra specificato, vengono sostituite con gli argomenti passati con il comando EXEC. Le variabili accettate vanno da &1 a &30 Ogni altra variabile di numero negativo o maggiore di 30, sara' ignorata tranne le due variabili speciali &0 e &*. Alla prima verra' sostituito il nome della procedura EXEC corrente. La seconda variabile significa "tutti gli argomenti specificati". Se inclusa in un comando CMS, il comando sara' eseguito una volta per ciascun argomento passato col comando EXEC.

Le variabili simboliche d'utente sono nomi che iniziano con & e contengono fino a sette caratteri alfanumerici, di cui uno almeno non numerico. Queste variabili possono contenere dati numerici o alfanumerici e devono essere inizializzate dall'utente. Il loro uso e' simile a quello delle variabili in un linguaggio evoluto (per es. Fortran).

Per es.

```
&K7 = 12
&A112 = &A112+7
```

Queste variabili possono essere concatenate.
Per es:

```
&K2 = 24
&PRINT SERIE AC&K2
```

l'esecuzione di queste due istruzioni provoca la stampa a terminale di :

SERIE AC24

Le variabili simboliche di sistema sono variabili aventi speciale significato per il sistema. Esse sono:

| | |
|---|---|
| &LINENUM | contiene il numero della riga corrente del file EXEC piu' uno |
| &INDEX1 | sono usati come indici e inizialmente contengono +1. |
| &INDEX9 | Possono essere inizializzati ad un valore intero (diverso da +1) con un'assegnazione; possono essere incrementati o decrementati specificando l'incremento in un'istruzione di EXEC<br>Per es: &INDEX4 = 256<br>&INDEX4 + 13<br>Aggiunge 13 al valore di &INDEX4.<br>Il valore degli indici vale solo per il livello di ricorsivita' in cui sono definiti. |
| &INDEX0 | Il suo valore e' il contenuto del registro 15 dopo ogni comando CMS |
| &INDEX | Il suo valore e' il numero di argomenti passati con una procedura EXEC |
| &GLOBAL0 | funzionano come &INDEX1...&INDEX9 e restano validi |
| &GLOBAL9 | anche nel passaggio fra livelli di ricorsivita' diversi |
| &GLOBAL | Il suo valore e' il livello di ricorsivita' |

Parole di controllo

&ARGS <arg1...argN>

Serve a ridefinire le variabili numeriche &1...&N con i valori di arg1...argN. &INDEX e' ridefinito al numero N.

&BEGSTACK <FIFO>

```
riga1       LIFO
riga2
-----
rigaN
&END STACK
```

Le righe scritte sono memorizzate in "stack" senza troncamenti e sostituzioni di variabili.
FIFO significa che le linee sono memorizzate in modo che la prima ad uscire sia la prima memorizzata, la seconda la seconda memorizzata e cosi' via; LIFO funziona al contrario, la prima riga ad uscire e' l'ultima memorizzata .etc....

&COMMENT riga

E' un'istruzione non eseguibile. Serve ad annotare la procedura EXEC

&CONTINUE

E' usato per fornire un indirizzo di salto per ERROR, &GOTO etc...

```
&ERROR azione
       &CONTINUE
```

'azione' e' una qualunque riga EXEC senza etichetta di riga. Viene eseguita subito dopo un ritorno con errore da un comando CMS che segue &ERROR. Un errore durante l'esecuzione di 'azione' provoca un'uscita da questo livello di EXEC con codice di errore 11

&EXIT <n>

causa il ritorno al livello di ricorsivita' superiore con codice di errore n. Se n non e' specificato il ritorno avviene con codice 0. Se n e' negativo il codice di ritorno sara' negativo se l'EXEC che contiene &EXIT e' stato chiamato da un altro EXEC; avra' il valore assoluto di n se l'EXEC e' stato chiamato dall'ambiente CMS.

```
       TOP
&GOTO  etichetta
       EXIT
```

permette di modificare l'ordine di esecuzione dei comandi:
TOP fara' ricominciare l'esecuzione della prima istruzione

EXEC.
EXIT e' equivalente a &EXIT 0 e fa uscire dal livello di ricorsivita' corrente.
Specificando come parametro 'etichetta' si rinvia l'esecuzione a l'istruzione individuata da tale etichetta.

&IF condizione azione

la condizione puo' essere un'istruzione di confronto secondo:
EQ, NE, GT, LT, GE, LE
con termini di paragone &*, &$, 'stringa'
dove &* significa una qualunque delle variabili numeriche, &$ una qualunque delle variabili simboliche e 'stringa' significa variabile di qualunque genere o numero. Infine 'azione' e' una qualunque istruzione eseguibile.

&LOOP etichetta condizione
n1 n2 .

esegue le successive n1 istruzioni o le istruzioni che seguono fino a quella individuata dall'etichetta specificata, e questo per n2 volte o fino a quando la condizione indicata non e' soddisfatta. Per la condizione vale il formato indicato per &IF.

&PRINT riga

Stampa una riga sul terminale. Le variabili numeriche o simboliche vengono prima sostituite con i loro valori. Le parole vengono troncate a otto caratteri.

n
&QUIT ON
OFF

Col parametro n, se non e' stato emesso prima &QUIT ON, fa ritornare all'ambiente CMS, qualunque sia il livello di ricorsivita' da cui si emetta &QUIT, col codice di errore n. &QUIT ON ogni volta che viene emesso, fa salire di uno il livello di ricorsivita', a partire dal corrente, ad uso di un successivo comando di &QUIT n, che in questo caso salira' al livello ancora superiore. Per es. se si emette 4 volte &QUIT ON dal livello corrente 8, un comando &QUIT n successivo fara' entrare al livello 3. &QUIT OFF ripristina il modo normale per &QUIT n, che fara' quindi ritornare al livello 0, il livello di CMS.

```
      <n>
&READ ARGS
```

Fa leggere n righe da terminale (default 1). Ogni riga viene immediatamente eseguita appena letta come fosse una istruzione specificata nel file EXEC. Finita l'esecuzione la tastiera si sblocca per permettere l'entrata della riga successiva, e il processo riprende fino all'ennesima riga a meno che una riga battuta non sia &GOTO, &EXIT, eLOOP, &SKIP o &READ. Se invece il parametro e' ARGS le parole scritte vengono considerate come argomenti che ridefiniscono i valori correnti di &1 &2...&N; anche &INDEX viene ridefinito al numero delle parole specificate.

&SPACE <n>

Provoca n (default 1) ritorni di carrello sul terminale.

&STACK <FIFO> riga
LIFO

memorizza la riga specificata in un buffer di ingresso per restituirla secondo l'ordine FIFO/LIFO (vedi &BEGSTACK) quando vengono successivamente emessi comandi di lettura da terminale (per es. &READ, da EDIT, DEBUG etc).

```
      TYPE
&TIME ON
      OFF
```

Il parametro TYPE provoca la stampa a terminale del tempo trascorso dall'ultimo messaggio di tempo stampato.
Il parametro ON fa stampare un messaggio con l'ora per ogni comando di CMS eseguito. Il parametro OFF (azione default) sopprime la stampa del tempo per i comandi CMS eseguiti.

```
         ALL TIME PACK
&TYPEOUT ON NOTIME NOPACK
         ERROR
         OFF
```

ALL fa stampare a terminale tutti i comandi CMS e istruzioni di controllo EXEC eseguite
ON stampa solo i comandi CMS eseguiti

ERROR stampa solo i comandi di CMS la cui esecuzione avviene con errore
OFF sopprime la stampa a terminale di comandi CMS e istruzioni di controllo
TIME fa stampare l'ora per ogni comando di CMS eseguito
NOTIME contrario del precedente
PACK non stampa i bianchi di coda nelle linee stampate a terminale
NOPACK contrario del precedente.

EXTERNAL — SIMULA UNA INTERRUZIONE ESTERNA SULLA MACCHINA VIRTUALE E PASSA IL CONTROLLO ALLA MACCHINA STESSA.

```
EXTERNAL
    E
```

FILEDEF

IL COMANDO FILEDEF PERMETTE ALL'UTENTE DI FARE IN MODO CHE UN PROGRAMMA DI CMS O DI UTENTE CHE SCRIVE SU O LEGGE DA UN FILE SU DISCO SCRIVA O LEGGA DA UNA DETERMINATA UNITA' (STAMPANTE, LETTORE, PERFORATORE, NASTRO O TERMINALE).

FILEDEF ddname device
<dsrn <CLEAR <lista delle opzioni>>>
*

ddname — e' il ddname di un file, definito mediante il comando DSNAME, oppure e' il tipo di un file di nome FILE.

dsrn — e' un numero da 1 a 99. Se XX e' il numero, il dsrn viene interpretato come se fosse il ddname FTXXF001.

device — e' uno dei seguenti, e specifica su quale device deve essere effettuato l'I/O:

| | |
|---|---|
| RDR | lettore di schede |
| PCH | perforatore di schede |
| PTR | stampante |
| CON | terminale |
| TAP | nastro montato su l'unita' 180 |
| TAP2 | nastro montato su l'unita' 181 |

CLEAR — permette di annullare l'effetto di precedenti FILEDEF.

lista delle opz — per il termirale e' possibile specificare L (lower case) o U (upper case) per stabilire se in lettura le minuscole devono essere lasciate tali o convertite in maiuscole. Quest'ultima e' l'opzione default. Per i nastri sono disponibili

LRECL n — lunghezza del record logico. 80 e' default

BLKSIZ n — lunghezza del record fisico. 80 e' default

RECFM F / FB / V / VB — formato del record. F e' default

```
                U

                0    densita' default
        MODE    16   1600 b.p.i.
                17    800 b.p.i.
```

NOTE -Nella sua forma completa:

```
            ddname
FILEDEF                 device
            dsrn
```

Il comando FILEDEF provoca la creazione di una tabella in cui al ddname specificato nel comando viene associato un determinato device di I/O. Le routines di lettura e scrittura sul disco del CMS consultano le tabelle definite da FILEDEF per effettuare l'I/O sul device appropiato.

-L'effetto di un comando FILEDEF puo' essere annullato specificando:

```
            ddname
FILEDEF                 CLEAR
            dsrn
```

Il comando FILEDEF * CLEAR cancella tutte le precedenti tabelle costruite da FILEDEF.

-Per conoscere quali definizioni sono valide in un dato momento si puo' lanciare il comando

FILEDEF

si ottiene la stampa di tutte le precedenti definizioni comprese le
FT05F001 CON
FT06F001 CON
che sono implicite e non possono essere cancellate.

ESEMPI:

1) FILEDEF
Una lista di tutti i ddnames e devices precedentemente definiti viene stampata sul terminale dell'utente.

2) FILEDEF NFILE RDR.
Al ddname NFILE viene associato il lettore di schede.

Ogni successiva lettura dal file
FILE INFILE
o da un file per cui sia stata o emesso il comando
DSNAME nome tipo INFILE
viene effettuata dal lettore di schede.

3) FILEDEF 4 PCH
Al ddname FT04F001 viene associato il perforatore di schede. Dato che un' istruzione WRITE (4,X) in un programma FORTRAN provoca la scrittura sul file
FILE FT04F001,
si ottiene in questo modo la perforazione di una scheda.

4) FILEDEF INFILE CLEAR
Viene cancellata la definizione del file con ddname INFILE.

5) FILEDEF * CLEAR
Provoca l'annullamento di tutte le precedenti definizioni di files eccetto quelle per i ddnames FT05F001 e FT06F001.

6) FILEDEF 3 TAP2 BLKSIZ 800 LRECL 800 RECFM VB
Al DDNAME FT03F001 viene associato il nastro di indirizzo 181, secondo le specifiche indicate nelle opzioni.

Messaggi di errore:

E(00001) Primo parametro invalido.
Il primo parametro non e' ne' un asterisco, ne' un numero di una o due cifre, ne' un ddname.

E(00002) Cattivo ritorno da un programma chiamato.
Generalmente questo e' dovuto all'esaurimento della memoria disponibile per la creazione delle tabelle. Si puo' tentare di nuovo il comando, altrimenti occorre rifare IPL del CMS.

E(00003) Parametri mancanti dopo l'operando xxxxxxxx.
Il secondo argomento manca o non e' valido.

E(00006) Illegale richiesta di CLEAR.
E' stata richiesta la cancellazione di un ddname non definito dall'utente.

E(00008) ddname non specificato dopo FILEDEF.
Manca il primo argomento.

E(00004)-E(00005)-E(00007) Il secondo argomento non e' uno dei 5 specificati, oppure la lista delle opzioni e' sbagliata.

FINIS — CHIUDE IL FILE (O FILES) SPECIFICATO SCRIVENDONE L'ULTIMO RECORD SU DISCO, AGGIORNANDO LO USER-FILE-DIRECTORY E CANCELLANDO L'ENTRATA PER QUEL FILE DALLA TABELLA DELL'UTENTE DEI FILES ATTIVI (ACTIVE FILE TABLE).

FINIS filename filetype <filemode>
* * *

filename — nome del file da chiudere
filetype — tipo del file da chiudere
filemode — modo del file da chiudere
* — significa rispettivamente tutti i filenames, filetypes, e modes.

NOTE - I files trattati dai comandi CMS sono chiusi automaticamente.

- Se e' omesso il filemode, viene chiuso il primo file incontrato con il filename e filetype specificati. L'ordine di ricerca del file e' quello normale.

ESEMPI:
1) FINIS FILE DATA P1
Il file FILE DATA P1 viene chiuso.

FORDECAL — PERMETTE ALL'UTENTE DI ENTRARE NELL'AMBIENTE FORDECAL.

FORDECAL

Il comando ha per effetto la stampa del messaggio:

**FORDECAL**

VERSION 5.5 e data dell'ultima edizione

--->

Dopo la stampa della freccia la tastiera del terminale si sblocca permettendo

l'inizio della sessione di FORDECAL.
Vedere anche il comando FORSTOP.

Per ogni dettaglio sul linguaggio riferirsi ai manuali:

FORMAC 360 CNUCE 24
FORDECAL 360 CNUCE

FORM — PRECOMPILA, COMPILA ED ESEGUE PROGRAMMI SCRITTI IN FORMAC.

FORM — nome <opzioni>

nome — nome del file da compilare ed eseguire il cui tipo deve essere FORMAC

opzioni

SYS — memorizza su disco il file PLI risultato della precompilazione del programma FORMAC

PRINT — stampa sulla stampante il file LISTING risultato della compilazione senza memorizzarlo

Per ogni informazione sul linguaggio si rimanda al manuale
FORMAC 360 CNUCE 24

NOTE - Le opzioni default sono NOSYS e NOPRINT

- I files usati col comando FORM devono essere di tipo FORMAC ed avere records di 80 caratteri.

- L'opzione default NOPRINT fa memorizzare il file LISTING su disco.

FORMAT — INIZIALIZZA UN'AREA DI DISCO NEL FORMATO CMS, CONTROLLA IL NUMERO DEI CILINDRI DI UN DISCO, SCRIVE UN'ETICHETTA DI IDENTIFICAZIONE DI UN DISCO.

FORMAT m <ALL> <(NOTYPE)> C L

R <SYS>
nn

| | |
|---|---|
| m | specifica il modo del disco |
| ALL | specifica che devono essere inizializzate tutte le tracce del disco indicato. Se omesso vengono saltati i primi tre records del cilindro 0,traccia 0. |
| C | specifica che il comando deve solo contare i cilindri, senza cancellarne il contenuto |
| L | specifica che si vuole scrivere solo un'etichetta sul disco |
| R | come C, ma aumenta o riduce il numero dei cilindri disponibili sul disco sempre nei limiti di quelli fisicamente presenti |
| nn | specifica che saranno formattati solo i primi nn cilindri |
| SYS | e' usato nel FORMAT P R SYS per riservare spazio per il nucleo CMS come scritto da IPLDISK |
| (NOTYPE) | specifica che le operazioni di format non richiederanno risposte da terminale |

NOTE - L'area di disco e' inizializzata, secondo le opzioni, scrivendo un nuovo "home address" e 15 records ogni due tracce di 2314. Ogni dato preesistente sul disco e' cancellato.

- Se e' specificato L, si puo' scrivere un'etichetta sul cilindro 0, record 3 del disco, nella forma CMS=,dopo che e' stato richiesto a terminale: ENTER-6-BYTE LABEL.

- Se e' specificato nn saranno inizializzati solo nn cilindri

- Se FORMAT e' il primo comando emesso dopo IPL, i files vengono cancellati, qualunque siano l'opzioni specificate. Per evitarlo usare prima il comando LOGIN UFD, o semplicemente pigiare il tasto "ritorno carrello".

- Nel caso di inizializzazione di un disco, l'indirizzo deve essere 191. Se non lo fosse occorre staccare dalla configurazione virtuale l'eventuale 191 esistente e attaccare il disco Cnn alla configurazione come 191.

ESEMPI:
1) FORMAT P ALL
Il disco P sara' completamente formattato e persi tutti i files eventualmente presenti.

2) FORMAT A C
Verra' scritta sul terminale la situazione del disco A comprendente numero di cilindri, numero di records occupati etc...

FORSTOP — PERMETTE DI USCIRE DALL'AMBIENTE FORDECAL E RIENTRARE IN AMBIENTE CMS.

FORSTOP

Il comando ha per effetto la stampa del messaggio:

**FORDECAL END**

e il ritorno nell'ambiente CMS.

FORTBUG — FORNISCE UN INSIEME DI FUNZIONI PER L'ESAME SIMBOLICO IN LINEA DI PROGRAMMI FORTRAN.

Questo comando richiama un compilatore FORTRAN G modificato; e' necessaria l'opzione DEBUG. Pertanto, tutti i programmi e sottoprogrammi FORTRAN che si desidera analizzare vanno compilati col comando

FORTBUG nome1...nomen (DEBUG

nome1...nomen — nomi dei programmi e sottoprogrammi, che saranno files di tipo FORTRAN, da compilare. Tali programmi non devono contenere istruzioni debug.

Notare che il comando FORTBUG emette fra l'altro un comando GLOBAL T BIBLUP BIBLOS18 SYSLIB per attivare le librerie necessarie al caricamento dei programmi da testare; il caricamento e l'inizio dell'esecuzione si fanno regolarmente coi comandi LOAD e START. E' possibile caricare insieme routines compilate con FORTBUG e altre compilate normalmente, purche' non contengano istruzioni debug. Si potra' comunque intervenire interattivamente solo nelle prime.
Per ogni sottoprogramma compilato con l'opzione DEBUG viene creato un file con lo stesso nome e tipo DEBUG, mentre per un programma principale il nome sara' quello del programma sorgente solo se non vi sono piu' programmi principali nello stesso file, nel qual caso il nome sara' MAIN. Tali files sono utilizzati da FORTBUG per risolvere i riferimenti.
L'inizio dell'esecuzione di un programma compilato col FORTBUG provoca l'entrata in ambiente FORTBUG nel quale la stampa su terminale di un trattino e il successivo sblocco della tastiera indicano che il sistema e' in attesa di una richiesta, la quale fa riprendere l'esecuzione del programma Fortran fino al successivo punto d'arresto specificato.
L'esecuzione del programma termina col ritorno in ambiente CMS quando si emette la richiesta QUIT, o si esegue una istruzione STOP o si giunge all'ultima istruzione del programma principale.
L'ordine di esecuzione delle istruzioni Fortran segue la logica del programma e non puo' essere cambiato. Si puo' infatti soltanto seguire l'esecuzione del programma o istruzione dopo istruzione o da un'istruzione ad un'altra scelta.
Nelle richieste si fa riferimento alle variabili fortran col loro nome e alle istruzioni con l'etichetta o col numero d'ordine dell'istruzione. E' percio' utile avere una lista del programma fortran serializzato, ottenibile per es. col comando di CMS OFFLINE PRINT nome FORTRAN.

Va tenuto presente che le variabili, le etichette e i numeri d'ordine di linee sono relativi a una routine, ed occorre specificare sempre a quale routine si riferiscano le richieste facendole precedere dalla richiesta QUALIFY nome-della-routine. Quando si entra in ambiente FORTBUG automaticamente viene qualificata la routine che contiene la prima istruzione eseguita.

## RICHIESTE DI FORTBUG

Scelta della routine a cui riferirsi:

QUALIFY — FA IN MODO CHE LE SUCCESSIVE RICHIESTE OPERINO SULLE ISTRUZIONI DELLA ROUTINE SPECIFICATA.

QUALIFY nome-della-routine

Questa richiesta attiva il dizionario delle variabili simboliche, la tabella delle etichette relative alla routine indicata e cura l'aggancio col file sorgente relativo. Non occorre emettere questa richiesta per il programma che viene eseguito per primo.

Messaggi:

FILE DEBUG NOT FOUND

che puo' significare:

- routine non interattiva
- modulo compilato senza l'opzione DEBUG
- il file DEBUG associato alla routine e' stato cancellato.

Verifica della esecuzione del programma

VERIFY — STAMPA SUL TERMINALE L'ISTRUZIONE FORTRAN INTERESSATA DA UN PUNTO D'ARRESTO.

VERIFY

Questa richiesta fa stampare sul terminale l'istruzione successiva nel caso di arresto dell'esecuzione o l'istruzione su cui si vuole fermare l'esecuzione nel caso di inserimento di un punto d'arresto.

BRIEF — SOPPRIME LE STAMPE PROVOCATE DALLA RICHIESTA VERIFY.

BRIEF

SUBTRACE — FA STAMPARE SUL TERMINALE IL NOME DI UNA ROUTINE ALL'INGRESSO E ALL'USCITA.

SUBTRACE

Questa richiesta fornisce una traccia dei programmi eseguiti.

NOSUBTRACE — FA STAMPARE SUL TERMINALE IL NOME DI UNA ROUTINE ALL'INGRESSO E ALL'USCITA.

NOSUBTRACE

SOURCE — PROVOCA LA STAMPA SUL TERMINALE DELL'ISTRUZIONE FORTRAN CORRISPONDENTE ALL'ISTRUZIONE ESEGUITA.

SOURCE

Messaggi:

SOURCE NOT POSSIBLE

che puo' significare

- Il file FORTRAN e' stato cancellato
- la routine nella quale ci si trova non e' stata compilata con l'opzione DEBUG.

TRACE — PROVOCA LA STAMPA SUL TERMINALE DEI NOMI DI TUTTI I PROGRAMMI CHIAMATI DOPO CHE IL PROGRAMMA PRINCIPALE HA AVUTO IL CONTROLLO DAL SISTEMA.

TRACE

Verifica e modifica dei valori degli argomenti e delle variabili.

ARG — PERMETTE DI CONOSCERE I VALORI DEGLI ARGOMENTI ALL'ENTRATA DI UN SOTTOPROGRAMMA.

ARG numero descrizione formato

numero — intero che identifica l'argomento di cui si vuol conoscere il valore, nella lista degli

argomenti. Uno identifica il primo, due il secondo e cosi' via.

descrizione — descrive l'argomento. E' della forma <numero d'elementi> <tipo>*<lunghezza> dove il 'numero d'elementi' serve a delimitare la parte da stampare di un argomento dimensionato; il 'tipo' puo' essere I, R, C, L secondo le convenzioni FORTRAN e la 'lunghezza' 1, 2, 4 e 8. Per esempio una descrizione potrebbe essere 15R*4 per i primi quindici elementi di una matrice REAL*4

formato — descrive il formato secondo cui si vuole la stampa sul terminale dell'argomento prima individuato. E' identico all'istruzione FORMAT del Fortran senza la parola FORMAT e l'etichetta.

Esempio:

```
ARG 2 21*2 (4H I1=,I4,4H I2=,I4)
```

Messaggi:

```
ARGUMENT ERROR
```

puo' significare

- il formato della richiesta e' errato
- il sottoprogramma di cui si vuol conoscere il valore di un argomento non e' attivo. Utilizzare la richiesta TRACE: il nome del sottoprogramma deve apparire nella lista dei programmi in esecuzione.

DISPLAY — STAMPA SUL TERMINALE IL CONTENUTO DELLE VARIABILI SPECIFICATE.

DISPLAY lista-di-variabili

lista-di-variabili — lista dei nomi delle variabili separate da virgole. Specificando senza indici il nome di una variabile dimensionata si otterra' tutta la matrice; per ottenere un elemento occorre specificare gli indici con numeri interi.

Messaggi:

NAME NOT IN DICT

Il nome della variabile non e' nella tabella dei simboli associata al programma

INVALID SUBSCRIPT

L'indice specificato e' superiore a quello dichiarato.

ERROR OF DELIMITER oppure

ILLEGAL VARIABLE NAME

vi sono errori nella lista.

| | |
|---|---|
| SET | PERMETTE DI MODIFICARE IL CONTENUTO DELLE VARIABILI SPECIFICATE. |
| SET | lista |
| lista | per ogni variabile va indicato il nome seguito dal segno = e dal nuovo valore. Gli elementi della lista sono separati da virgole. Ogni elemento della lista contiene un valore nel caso di variabili semplici, piu' valori entro parentesi e separati da virgole nel caso di variabili dimensionate. Il tipo delle costanti-valore puo' essere diverso da quello della variabile associata; valgono le convenzioni dell'istruzione NAMELIST. |

Esempi:

```
SET LG=(15,0.6),A=.TRUE.,AS=42.3e-10
```

Controllo sull'esecuzione

| | |
|---|---|
| AT | FISSA LE ISTRUZIONI SULLE QUALI L'ESECUZIONE DEVE ARRESTARSI. |
| | etichetta |
| AT | LINE numero STOP |
| | ALL |
| | STOP |
| etichetta | individua l'istruzione dalla sua etichetta |

numero numero d'ordine dell'istruzione nel programma sorgente

L'esecuzione si arresta prima dell'esecuzione delle istruzioni individuate o mediante l'etichetta o dal numero d'ordine di lenea.
Specificando ALL l'esecuzione si arresta dopo l'esecuzione di ogni istruzione; specificando STOP si sopprime l'effetto di ALL.

NOTE - Non viene fatta alcuna verifica sul numero di linea: se a questo non corrisponde alcuna istruzione la richiesta non ha effetto.

- Se e' attiva la richiesta VERIFY saranno stampate le istruzioni Fortran interessate alla richiesta AT

- Si possono fissare al piu' 30 punti di arresto.

Messaggi:

ILLEGAL LABEL

L'etichetta specificata non esiste nella routine in cui siamo

RESET CANCELLA PUNTI D'ARRESTO FISSATI DA PRECEDENTI RICHIESTE AT.

RESET etichetta
LINE numero

Specificando RESET senza parametri si sopprimono tutti gli arresti precedentemente fissati

Messaggi:

ILLEGAL LABEL

L'etichetta specificata non esiste nella routine in cui siamo.

GO FA RIPRENDERE L'ESECUZIONE DOPO UN PUNTO D'ARRESTO.

GO

QUIT FA USCIRE DALL'AMBIENTE FORTBUG E RIENTRARE IN CMS.

QUIT

INTERRUZIONE ESTERNA

E' possibile arrestare l'esecuzione di un programma premendo il tasto ATTN (attention) ed entrare in ambiente CP. Battendo poi il comando di CP EXTERNAL si simula una interruzione esterna che fa entrare in ambiente FORTBUG interattivo mettendosi in attesa di una richiesta FORTBUG. Questo vale solo se la routine di cui si interrompe l'esecuzione e' stata compilata con FORTBUG.

FORTRAN — CONVERTE CODICE SORGENTE, SCRITTO IN LINGUAGGIO FORTRAN, IN CODICE OGGETTO RILOCABILE.

FORTRAN fn1...fnN <(opz1...opzN)>
F

fn — nome del file da compilare. Si possono specificare fino a 32 files.

Le opzioni sono:

| | |
|---|---|
| MAP | include nel file LISTING le tabelle delle variabili FORTRAN, NAMELIST, e gli statements FORMAT |
| NOMAP | sopprime la tabella delle variabili |
| DECK | genera il file TEXT |
| NODECK | sopprime il file TEXT |
| LIST | include una lista del modulo oggetto (in linguaggio assembler nel file) LISTING |
| NOLIST | sopprime la lista del modulo oggetto |
| SOURCE | include il programma sorgente nel file LISTING |
| NOSOURCE | sopprime la lista del programma sorgente dal file LISTING |
| BCD | e' usato quando il modulo sorgente e' perforato in BCD |
| EBCDIC | e' usato quando il modulo sorgente e' perforato in EBCDIC |
| GO | fa completare la compilazione del programma anche se vi sono delle istruzioni errate |
| NOGO | termina la compilazione del programma anche quando vengono trovati errori gravi |
| PRINT | stampa il file LISTING sulla stampante offline, e lo cancella da disco |

NOPRINT sopprime la stampa del file LISTING

DIAG stampa a terminale gli errori del programma sorgente ed i messaggi diagnostici

NODIAG sopprime la stampa degli errori e dei messaggi diagnostici

NOTE - I files trattati dal comando FORTRAN devono essere di tipo FORTRAN ed avere records di 80 caratteri.

- Si puo' specificare una combinazione qualsiasi delle opzioni, un insieme scelto delle opzioni vale per tutti i files compilati con quel comando.

- Il comando FORTRAN puo' creare, secondo le opzioni specificate, due files con lo stesso nome del file da compilare e tipo TEXT, per il file contenente il programma compilato, e tipo LISTING per quello contenente il codice sorgente, la mappa di caricamento etc secondo le opzioni scelte. Eventualmi files preesistenti con lo stesso nome e tipo vengono cancellati.

- La lista delle opzioni default e' la seguente: NOGO DIAG EBCDIC DECK NOSOURCE NOMAP NOLIST NOPRINT.

- Quando si vogliono compilare piu' files con un unico comando e' bene specificare l'opzione GO per evitare che un errore in un modulo sorgente termini la compilazione dei moduli sorgente successivi.

ESEMPI:

1) FORTRAN ACI AS (LIST MAP)
I files di nome ACI e AS e tipo FORTRAN saranno compilati con le opzioni non-default LIST e MAP.

FST STAMPA A TERMINALE INFORMAZIONI SULLE CARATTERISTICHE DI UN FILE.

FST nome tipo <carattere qualunque>

nome, tipo identificatore del file

NOTA - Specificando solo nome e tipo si ottiene per il file:
numero di records logici
lunghezza del record logico
formato del file (F o V)
Specificando anche un terzo argomento (un carattere qualunque) si ottiene inoltre:
data e ora dell'ultimo aggiornamento
puntatore di scrittura
puntatore di lettura
numero di blocchi (di 800 bytes)
indirizzo della First Chain Link
flag dello stato del file.

## FUNZIONI DEL CMS

Volendo chiamare una funzione del CMS senza fare uso di macro, si deve ricorrere alla chiamata via SVC, secondo lo schema.

```
          .
          .
          .
          LA    1,PLIST
          SVC   202
          DC    AL4(*+4)
        ritorno normale
          .
          .
          .
```

Saranno qui descritte le liste di chiamata delle funzioni piu' utili.

ATTN MEMORIZZA IN UNA CODA UNA RIGA PERCHE' SIA LETTA DA UN SUCCESSIVO COMANDO DI LETTURA DA TERMINALE.

```
LISTA   DS    0D
        DC    CL8'ATTN'
        DC    CL4'ordine'     lifo o fifo
```

```
          DC    AL1(LBUFF)      lunghezza del buffer
          DC    AL3(BUFF)       indirizzo del buffer
          .
          .
          .
BUFF      DC    C'riga'
LBUFF     EQU   *-BUFF
```

NOTA - Il programma per la lettura da terminale WAITRD, controlla, prima di emettere la SIO, che non ci siano righe in coda in attesa di essere lette. Se la coda non e' vuota, la lettura viene fatta dalla prima riga in coda anziche' da terminale.

CARDIO LEGGE O PERFORA UNA SCHEDA.

```
LISTA     DS    0D
          DC    CL8' '  CARDRD o CARDPH
          DC    X'flag' X'00': lettore normale
                        X'80': lettore non standard

          DC    AL3(BUFFER)
          DC    H'numero di bytes da leggere'
          DC    H'0' numero di bytes letti
        Necessari solo se flag=X'80'
```

NOTA - Flag = X'80' e' necessario leggere schede di lunghezza diversa da 80 caratteri. Per la perforazione il flag e' sempre X'00'. Al termine di un programma che perfori con la CARDPH il perforatore non viene automaticamente chiuso. Occorre quindi chiamare la CLOSIO dopo la ultima perforazione.

Codici di Errore:

```
E(00001)  End of File
E(00002)  Unit check
E(00003)  Errore sconosiuto
E(00004)  Not operational
E(00005)  Lunghezza diversa da quella specificata
```

CLOSIO CHIUDE O IMPEDISCE LA CHIUSURA DI UNA UNITA'.

```
LISTA     DS    0D
          DC    CL8'CLOSIO'
         <DC    CL8'READER'>
         <DC    CL8'PRINTER'>
         <DC    CL8'PUNCH'>
```

```
          <DC    CL8'OFF'>
          <DC    CL8'ON'>
           DC    X'FFFFFFFF'
```

NOTA - CLOSIO chiude tutte le unita' specificate nella lista, a meno che l'ultimo parametro non sia OFF. In questo caso, ogni chiamata di CLOSIO per quei parametri viene ignorata a meno che l'ultimo parametro non sia ON. Se nessun parametro e' specificato vengono chiusi tutti e tre le unita'.

CONWAIT — ASPETTA CHE TUTTE LE LETTURE E SCRITTURE SU TERMINALE SIANO FINITE.

```
  LISTA   DS   0D
          DC   CL8'CONWAIT'
          DC   CL4'CON1'
```

CPFUNCTN — CHIAMA IL CP PERCHE' ESEGUA UNA DATA FUNZIONE DI CONSOLE.

```
  LISTA  DS    0D
         DC    CL8'CPFUNCTN'
        <DC    CL8'NOMSG> sopprime la stampa su terminale di
                          messaggi di CP
         DC    C'comando di CP'
         DC    X'FF'
```

Codici di errore:

E(00001) Nessun comando di CP presente
E(00004) INVALID CP REQUEST
E(00008) BAD ARGUMENT
E(xxxxx) vedere la corrispondente funzione di CP

ERASE — CANCELLA I(L) FILE(S) SPECIFICATO(I).

```
          DS    0D
          DC    CL8'ERASE'
          DC    CL8' '          nome o *
          DC    CL8' '          tipo o *
          DC    CL2' '          modo o *
```

Codici di Errore:

E(00001) Il primo carattere del modo e' illegale
E(00002) File non trovato
E(00004) Errore di scrittura su disco.

FINIS CHIUDE IL FILE SPECIFICATO.

```
LISTA   DS   0D
        DC   CL8'FINIS'
        DC   CL8' '       nome o *
        DC   CL8' '       tipo o *
        DC   CL2' '       modo o *
```

NOTA - Se nel corso di uno stesso programma si passa dalla lettura alla scrittura su uno stesso file, e' obbligatorio chiamare FINIS.

Codici di Errore:

E(00001) Nome invalido
E(00002) Tipo invalido
E(00003) Errore di scrittura su disco
E(00004) Modo invalido
E(00006) File non aperto

PRINTR SCRIVE UNA RIGA SULLA STAMPANTE.

```
LISTA   DS   0D
        DC   CL8'PRINTR'
        DC   A(area)  indirizzo dell'area contenente
                      la riga
        DC   F(dimensione) LUNGHEZZA DELLA RIGA
```

NOTA - Per la chiusura della stampante vale quanto si e' detto per CARDIO. Il primo carattere e' interpretato come carattere di controllo del carrello; come caratteri di controllo possono essere usati i codici operativi delle CCW per il controllo del carrello o i seguenti:

```
CL1' '     spazio singolo
CL1'0'     spazio doppio
CL1'1'     nuova pagina
CL1'-'     spazio triplo
CL1'+'     scrivi senza spazio
CL1'n'     scrivi dal canale n
```

La lunghezza della riga non puo' superare i 133 bytes.

Codici di Errore:

E(00001) Unit Check
E(00002) Carattere di controllo illecita
E(00003) Lista di chiamata scorretta
E(00004) Not operational

E(00005) Errore sconosciuto

RDBUF LEGGE UNO O PIU' RECORDS DA UN FILE SU DISCO.

```
LISTA  DS  0D
       DC  CL8'RDBUF'
       DC  CL8'  '      nome
       DC  CL8'  '      tipo
       DC  CL2'  '      modo o *
       DC  H'  '        Numero del record
       DC  A(area)      INDIRIZZO DOVE LEGGERE IN MEMORIA
       DC  F'  '        DIMENSIONE DELL'AREA DI LETTURA
       DC  CL2'  '      'F' PER FISSO 'V' PER VARIABILE
       DC  H'  '        NUMERO DI RECORDS DA LEGGERE
       DC  F'0'         NUMERO DI BYTES DA LEGGERE
```

NOTA - Per files di lunghezza variabile si puo' leggere solo un record alla volta. Tutti gli errori tranne l'8 sopprimono l'esecuzione della lettura. Se il numero del record e' 0 la lettura e' sequenziale.

Codici di Errore:

E(00001) File non trovato
E(00002) L'indirizzo dell'area di memoria e' illegale cioe' fuori della memoria o nel nucleo o nel nucleo
E(00003) Guasto sul disco
E(00005) Il numero dei records da leggere e' uguale a 0
E(00007) Il file non e' stato scritto con WRBUF, cioe' non e' un file CMS
E(00008) L' area di lettura e' troppo piccola
E(00009) File aperto per scrivere (chiamare prima FINIS)
E(00011) piu' di record richiesta per un file a lunghezza variabile
E(00012) La riga richiesta non esiste (EOF)
E(00013) File variabile codificato in modo scorretto (errore di sistema)

STATE FORNISCE LA FILE STATUS TABLE DEL FILE SPECIFICATO.

```
LISTA  DS  0D
       DC  CL8'STATE'
       DC  CL8'    '    nome del file
       DC  CL8'    '    tipo
       DC  CL2' '       modo o *
       DC  CL2' '       senza significato
       DC  A( )         indirizzo della FST al ritorno
```

NOTA - Specificando come modo * vengono ricercati tutti i dischi nell'ordine standard di ricerca. La FST e' una tabella contenente tutte le informazioni necessarie sul file, come nome, tipo, modo lunghezza dei records, formato, numero di righe, data, etc.....

Codici di Errore:

E(00001) Il file specificato non esiste
E(00004) Il primo carattere del modo e' illegale.

TAPEIO LEGGE O SCRIVE UN RECORD SU NASTRO O POSIZIONA UN NASTRO.

```
   LISTA  DS  0D
          DC  CL8'TAPEIO'
          DC  CL8' '        funzione
          DC  CL4' '        TAP1 o TAP2
          DC  XL1' '        modo
          DC  AL3' '        indirizzo del
                            buffer
          DC  F' '          dimensione del
                            buffer
          DS  F' '          numero di bytes
                            letti (al ritorno)
```

Le funzioni sono:

```
  BSF      indietreggia di un file
  BSR      indietreggia di un record
  FSF      avanza di un file
  FSR      avanza di un record
  READ     leggi un record
  REWIND   riavvolgi il nastro
  RUN      riavvolgi il nastro e apri
           l'unita' (per toglierlo)
  WRITE    scrivi un record
  WRITEOF  scrivi un EOF
  WTM      scrivi un EOF
  ERG      cancella un gap
```

Il modo e' un byte della forma DDMMM011 con la seguente interpretazione:

| DD | tracce | densita' |
|---|---|---|
| 00 | 7 | 200 b.p.i. |
| 01 | 7 | 556 |
| 10 | 7 | 800 |
| 11 | 9 | - |

| MMM | tracce | densita' | parita' | convertitore | translator |
|---|---|---|---|---|---|
| 000 | 9 | 1600 b.p.i. | — | — | — |
| 001 | 9 | 800 b.p.i. | — | — | — |
| 010 | 7 | — | dispari | on | off |
| 011 | — | — | — | — | — |
| 100 | 7 | — | pari | off | off |
| 101 | 7 | — | pari | off | on |
| 110 | 7 | — | dispari | off | off |
| 111 | 7 | — | dispari | off | on |

NOTE - Il modo viene preso in considerazione solo nel caso di una scrittura quando il nastro e' posizionato al punto di caricamento (inizio). Altrimenti come modo viene scelto quello col quale e' stato gia' scritto il nastro.

- Il modo X'00' e' interpretato come modo default per l'unita' usata.

- Per unita' a 7 piste questo e' X'B3'.

- Per unita' a 9 piste a 1 velocita' il modo non ha nessun significato e viene sempre interpretato come X'CB' (800 b.p.i.). Questo vale per le 2400.

- Per unita' a 9 piste a 2 velocita' (come le 3420) il modo default e' X'C2' (1600 b.p.i.).

Codici di Errore:

E(00001) INVALID 'TAPEIO READ' PARAMETER-LIST. La lista di chiamata per la lettura non e' corretta.
E(00001) INVALID 'TAPEIO WRITE' PARAMETER-LIST. La lista di chiamata per la scrittura non e' corretta.
E(00002) Fine del file o del nastro.
E(00003) Errore permanente di I/O.
E(00004) L'indirizzo simbolico non e' valido.
E(00005) TAPn NOT ATTACHED. Alla macchina virtuale non e' stato attaccata l'unita' 180 (TAP1) o 181 (TAP2).
E(00006) TAPn IS FILE PROTECTED. Tentativo di scrivere su un nastro senza anello.
E(00007) TAPn-SERIOUS TAPE ERROR ATTEMPTING funzione. Un errore irrecuperabile e' avvenuto tentando la funzione specificata.

TYPE SCRIVE UN MESSAGGIO SUL TERMINALE SENZA CANCELLARE I BLANKS FINALI E SENZA AGGIUNGERE UN RITORNO CARRELLO AUTOMATICO.

```
LISTA  DS  0F
```

```
          DC   CL8'TYPE'
          DC   AL1(1)      numero del terminale
          DC   AL3(MSG)    indirizzo del messaggio da scrivere
          DC   C' '                    B  o  K
          DC   AL3(EMSG-MSG)     lunghezza del messaggio
          .
          .
          .
  MSG     DC   C' '
  EMSG    EQU  *
```

Il significato dei codici B e K e' il seguente:
B sposta la riga in memoria libera prima di scrivere
K scrive la riga dalla locazione specificata

NOTE - Il messaggio deve avere una lunghezza fra 1 e 130 bytes.

- Dato che la scrittura su terminale non e' sincrona, si consiglia di usare il codice B, altrimenti parti successive del programma potrebbero modificare il messaggio prima che questo venga scritto.

Codici di errore

E(00001) Il numero del terminale non e' 1
E(00002) La lunghezza del messaggio non e' fra 1 e 130 bytes.

TYPLIN — SCRIVE UNA RIGA SOPPRIMENDO I BLANKS FINALI E AGGIUNGENDO UN RITORNO CARRELLO.

```
  LISTA  DS    0F
         DC    CL8'TYPLIN'
         DC    AL1(1)
         DC    AL3(MSG)
         DC    C' '       B o K
         DC    AL3(EMSG-MSG)
         .
         .
         .
         .
  MSG    DC    C' '
  EMSG   EQU   *
```

NOTE - Vale quanto si e' detto per TYPE. Sono accettati anche messaggi di lunghezza 0. In tal caso, viene eseguito solo il ritorno-carrello.

WAIT — PONE LA MACCHINA IN WAIT FINO A QUANDO ARRIVA UNA INTERRUZIONE DA UNO DEI DUE

DISPOSITIVI SPECIFICATI.

```
LISTA  DS  0F
       DC  CL8'WAIT'
       DC  CL4' ' dispositivo 1
       DC  CL4' ' dispositivo 2
       .
       .
       .
       DC  F'0'
       DS  F     dispositivo che ha provocato
                 l'interruzione
```

I dispositivi ammessi sono CON1, DSK1, PCH1, RDR1, PRN1, TAP1 e TAP2.

NOTA - La voce F'0' chiude la lista dei dispositivi specificati.

- Dopo l'interruzione il controllo torna al programma chiamante e l'ultima voce della lista contiene il nome del dispositivo che ha provocato l'interruzione.

Codici di errore: E(00001) Il nome di un dispositivo non e' valido.

WAITRD LEGGE UNA RIGA DAL TERMINALE IN UNA DATA AREA ASPETTANDO FINO AL RITORNO-CARRELLO.

```
LISTA  DS  0D
       DC  CL8'WAITRD'
       DC  AL1(1)      numero del terminale
       DC  AL3(INPBUF) indirizzo dell'area di 130 bytes
       DC  C' '          u,v,s,t o x
       DS  AL3         numero di bytes letti, al ritorno
       .
       .
       .
       .
       .
INPBUF DS  CL130
```

Il significato dei codici U,V,S,T,X e' il seguente:

U interpreta i caratteri di cancellazione, converte in maiuscolo e aggiunge blanks in fondo
V interpreta i caratteri di cancellazione e converte in maiuscolo
S interpreta i caratteri di cancellazione e aggiunge blanks in fondo
T interpreta i caratteri di cancellazione

```
X  lascia la riga cosi' come e'
```

NOTA - Se alcune righe sono gia' state messe in coda per la lettura (per mezzo di ATTN, battendo due volte l'attention, scrivendo piu' di una riga grazie al carattere o con l'$STACK in una procedura EXEC), WAITRD trasmette al programma chiamante la prima in coda. L'area di input viene riempita di zeri per 130 bytes a partire da INPBUF; e' bene quindi riservare 130 bytes per la lettura anche se ne verranno letti meno .
Codici di errore:
E(00001) Il numero del terminale e' diverso da 1
E(00002) Il codice di lettura non e' U,V,S,T, o X.

WRBUF SCRIVE UNO O PIU' RECORDS SU UN FILE.

```
  LISTA  DS   0D
         DC   CL8'WRBUF'
         DC   CL8' '      nome del file
         DC   CL8' '      tipo
         DC   CL2' '      modo(* e' illegale)
         DC   H' '        numero del record
         DC   A( )        indirizzo del buffer
         DC   F' '        numero di bytes da scrivere
         DC   CL2' '      F o V
         DC   H' '        numero di records da scrivere
```

Codici di Errore:

E(00001) nome e tipo del file non specificati
E(00002) l'indirizzo dell'area e' illegale
E(00003) errore di I/O. Il disco potrebbe essere read-only
E(00004) il primo carattere del modo e' illegale
E(00005) il secondo carattere del modo e' illegale
E(00006) il numero di bytes di uno dei records da scrivere supera 65535
E(00007) tentativo di lasciare un record non scritto in un file a lunghezza variabile
E(00008) il numero di bytes da scrivere e' uguale a 0
E(00009) il file e' gia' attivo per leggere
E(00010) e' stato raggiunto il massimo numero di files CMS (3500)
E(00011) non e' stato specificato F o V come formato dei records
E(00012) il modo S e' illegale
E(00013) il disco e' gia' pieno. In questo caso viene scritto
**P-DISK (191) IS FULL**
e poi inizia la procedura del KX
E(00014) il disco non e' stato ancora formattato
E(00015) il file ha formato fisso, e i records da scrivere

non hanno la stessa lunghezza di quelli gia' scritti

E(00016) il formato specificato nella lista non e' quello del file su disco

E(00017) records piu' lungo di 65.535 bytes

E(00018) piu' di un record da scrivere per un file a formato variabile

E(00019) e' stato raggiunto ul massimo numero di blocchi per un file (16060)

GDYN — COMPILA ED INIZIA L'ESECUZIONE INTERATTIVA DI FILES SCRITTI IN DYNAMO.

GDYN nome

nome — nome del file contenente il programma in DYNAMO il cui tipo deve essere DYNAMO, ed avere records di 80 caratteri.

NOTA - Il GAMING DYNAMO e' una versione conversazionale del linguaggio di simulazione DYNAMO II. Per ogni ulteriore informazione si rimanda al manuale

GAMING DYNAMO manuale CNUCE

GENMOD — CREA SU DISCO PERMANENTE UN FILE IN FORMA IMMAGINE DI MEMORIA NON RILOCABILE COPIANDO IL CONTENUTO DELLA MEMORIA FRA DUE LOCAZIONI ASSEGNATE

GENMOD entry1 entry2 <(<opz1><opz2>)>
G

entry1 — indica un punto d'ingresso o il nome di una sezione di controllo da cui deve iniziare la copia immagine di memoria. Sara' anche il nome assegnato al file generato.

entry2 — come sopra per la locazione finale della copia immagine di memoria.

Opzioni:

NOMAP — specifica che non deve essere inclusa nel

file creato una mappa di caricamento
P2 specifica che il file MODULE deve avere modo P2

NOTE - Il file creato sara' posto nel disco permanente con nome uguale ad entry1, tipo MODULE, e modo P1 se non e' stata specificata l'opzione P2.

- Se non si specifica la seconda entrata (entry2), il nuovo file sara' costituito dal contenuto della memoria da entry1 fino al successivo punto di caricamento.

- Ogni file su disco permanente di tipo MODULE e nome uguale a quello del file che si crea, sara' cancellato.

- Per caricare in memoria il file creato si usa il comando LOADMOD; se pero' il file e' stato creato con l'opzione (NOMAP), per caricarlo in memoria ed eseguirlo, occorre scrivere il nome del file a terminale, come se fosse un comando CMS.

- Un file MODULE creato con l'opzione (NOMAP) richiede meno spazio su disco, in quanto non contiene mappa di caricamento.

ESEMPI:
1) GENMOD ECCO
Verra' creato su disco un file ECCO MODULE, comprendente l'immagine di memoria del file ECCO TEXT prima caricato, dall' entrata ECCO fino alla fine.

GLOBAL SPECIFICA LIBRERIE DI MACRO DEFINIZIONI IN CUI RICERCARE MACRO DURANTE IL PROCESSO DI ASSEMBLAGGIO O LIBRERIE DI TIPO TEXT IN CUI RICERCARE MODULI ALL'ATTO DEL CARICAMENTO DI FILES CONTENENTI CODICE RILOCABILE.

```
GLOBAL      ASSEMBLER MACLIB
               M          <lbn1...lbnN>
            LOADER TXTLIB
                   T
            PRINT
```

ASSEMBLER MACLIB specifica le librerie nelle quali cercare definizioni di macros durante gli assemblaggi successivi

LOADER TXTLIB — specifica le librerie in cui cercare moduli mancanti nelle operazioni di LOAD, USE e REUSE successive

PRINT — specifica che deve essere stampata una lista delle librerie attive al momento

lbn1...lbnN — specificano librerie il cui tipo e' MACLIB oppure TXTLIB

NOTE - Nella forma ASSEMBLER del comando GLOBAL si possono specificare fino a cinque librerie di tipo MACLIB; nella forma LOADER, fino a otto di tipo TXTLIB.

- Se si desidera che i moduli siano ricercati anche nelle librerie del sistema (SYSLIB MACLIB, OSMACRO MACLIB e SYSLIB TXTLIB), i nomi di queste vanno inclusi nella lista delle librerie di tipo MACLIB e TXTLIB rispettivamente.

- Il comando resta valido nella forma ASSEMBLER e/o LOADER fino a quando si emette un nuovo comando GLOBAL M e/o GLOBAL T rispettivamente, oppure si reinizializza il nucleo del CMS o l'utente si stacca dal CP. Se non si specificano nomi di librerie, si ripristina la ricerca dei moduli mancanti nelle librerie del sistema (SYSLIB MACLIB e OSMACRO MACLIB se si e' specificato GLOBAL M, e SYSLIB TXTLIB se si e' emesso-GLOBAL T)

ESEMPI:
1) GLOBAL T MYLIB SYSLIB
Ogni successivo comando di LOAD fara' riferimento alle due librerie TEXT di nome MYLIB e SYSLIB.

GPSS — COMPILA ED ESEGUE PROGRAMMI SCRITTI IN GPSS.

GPSS nome

nome — nome del programma da elaborare il cui tipo deve essere GPSS ed avere records di 80 caratteri.

Per ogni informazione sul linguaggio si rimanda al manuale

GPSS CNUCE

IPL — FA IN MODO CHE IL CP SIMULI UNA SEQUENZA DI IPL DALL'UNITA' SPECIFICATA.

```
IPL          CMS
 I           devadd
```

CMS — specifica che deve essere portata in memoria una nuova copia del nucleo CMS in precedenza "salvato"

devadd — specifica l'indirizzo del dispositivo da cui fare IPL.

NOTA - Questa funzione simula la sequenza di azzeramento della memoria, impostazione di un indirizzo di unita' e schiacciamento del tasto di LOAD necessario per fare IPL su una macchina reale.

- Il comando IPL CMS e' piu' rapido in quanto copia in memoria un nucleo CMS gia' 'immagine di memoria'.

IPL — AZZERA LA MEMORIA E VI CARICA UNA NUOVA COPIA DEL NUCLEO CMS.

```
IPL          <devadd>
```

devadd — e' l'indirizzo dell'unita' da cui deve essere caricata la nuova copia del nucleo.

NOTA - Se il parametro e' omesso, viene portata in memoria una copia del nucleo CMS identica a quella usata fino al momento dell'emissione del comando.

IPLSAVE — COME IPL, ECCETTO CHE NON AZZERA LA MEMORIA.

```
IPLSAVE      ccu
```

ccu — specifica l'indirizzo dell'unita' da cui

eseguire IPL.

KO — TERMINA L'AZIONE DEI COMANDI SETOVER E/O SETERR EMESSI DOPO L'ULTIMO COMANDO KO O CLROVER; O DOPO L'ULTIMO CMS LOGIN DELL'UTENTE, SE NON SONO STATI EMESSI COMANDI KO O CLROVER. TRASFERISCE ALLA STAMPANTE OFFLINE TUTTE LE INFORMAZIONI DI TRACCIA REGISTRATE FINO AL MOMENTO DELL'EMISSIONE DEL COMANDO.

KO

NOTA - Questo comando differisce dal comando CLROVER in quanto puo' essere usato durante la esecuzione di un programma. Deve essere emesso dopo aver premuto due volte il tasto interruzione per far accettare il comando dal CMS. (Comando ad azione immediata).

KT — IMPEDISCE LA STAMPA A TERMINALE PER LA DURATA DEL COMANDO CHE VIENE ESEGUITO O DEL PROGRAMMA DELL'UTENTE, OPPURE NON STAMPA UN NUMERO DI RIGHE PARI AD N.

KT <n>

n — numero di righe che non si vuole far stampare a terminale

NOTA - Il comando deve essere emesso dopo aver premuto due volte il tasto interruzione per fare accettare il comando dal CMS. (Comando ad azione immediata).

- Il comando va usato durante l'esecuzione di un programma.

ESEMPI:
1) KT 15
Verra' soppressa la stampa al terminale delle prossime 15 righe inviate dal programma al terminale.

KX — ARRESTA IMMEDIATAMENTE L'ESECUZIONE DEL COMANDO CMS O PROGRAMMA DELL'UTENTE IN CORSO, CHIUDE I FILES APERTI E I DISPOSITIVI DI I/O, AGGIORNA LO USER-FILE-DIRECTORY, FA NUOVAMENTE IPL DEL CMS.

KX

NOTA - Il comando deve essere emesso dopo aver premuto due volte il tasto interruzione per fare accettare il comando dal CMS. (Comando ad azione immediata).

LINEND — RIDEFINISCE IL CARATTERE DI FINE LINEA LOGICA CHE PERMETTE DI SCRIVERE PIU' COMANDI IN UNA SOLA LINEA. IL VALORE DEFAULT E' ;

LINEND c

c — e' il carattere logico di fine linea ridefinito. Se omesso, non ci sara' alcun carattere di fine linea logica e l' unico delimitatore sara' il ritorno carrello.

NOTA - Il carattere di fine linea ridefinito rimane valido fino a quando:
1) si rifa' IPL del CMS
2) si emette un altro comando di LINEND
3) si fa logout.

ESEMPI:
1) LINEND ?
Ove il carattere serva per altri scopi, questo comando lo sostituisce col carattere "?".

LINK — PERMETTE DI CONNETTERE UN DISCO DELLA MACCHINA VIRTUALE SPECIFICATA ALLA CONFIGURAZIONE DELLA MACCHINA VIRTUALE CHE EMETTE IL COMANDO SECONDO MODALITA' DIPENDENTI DALLA DEFINIZIONE DEL DISCO.

LINK userid xxx yyy <W> <(NOPASS)>
* R

| | |
|---|---|
| userid | nome della macchina virtuale a cui e' collegato il dispositivo xxx |
| * | significa nome della macchina virtuale che emette il comando. In questo caso non occorre parola chiave e il disco viene riconnesso col modo di accesso che ha definito. |
| xxx | indirizzo (hex.) del disco che viene richiesto |
| yyy | indirizzo (hex.) da assegnare al disco nella macchina virtuale richiedente |
| W | richiede l'accesso in scrittura |
| R | richiede l'accesso solo in lettura (default) |
| (NOPASS) | usato solo se non e' richiesta alcuna parola chiave per il modo di accesso desiderato. |

NOTE - Se la funzione viene usata dal CMS col comando CP LINK la parola chiave va specificata come parametro nella forma 'PASS= parola-chiave'

- Questa funzione ha effetto solo se il disco richiesto appartiene alla macchina virtuale che lo richiede (per es. puo' essere staccato con un DETACH precedente), oppure se nella configurazione della macchina che lo ha definito e' definito come condivisibile in lettura o in scrittura con parola chiave o no

- Il LINK usato col DETACH permette di cambiare l'indirizzo virtuale di un disco per es: con la sequenza seguente l'indirizzo e' cambiato da 193 a 195:
DETACH 193
LINK * 193 195

- Un disco in lettura scrittura connesso a una macchina virtuale attiva non viene connesso ad un'altra macchina virtuale che lo richiede

- Questa funzione permette a piu' macchine virtuali di accedere ad uno stesso disco.

ESEMPI:
1) LINK USER2 191 291 W
Si richiede la connessione in lettura scrittura come 291 del disco 191 della macchina USER2.

LISPCMS FA ENTRARE IN AMBIENTE LISP.

LISPCMS

Il LISP e' un 'interprete' di tipo 'evalquote', accetta cioe' per la valutazione coppie del tipo:

<definizione di funzione><lista di argomenti>

Il ritorno in ambiente CMS si ottiene mediante la funzione *FIN* senza argomenti.

NOTE - L'area 'free storage' e' composta di 15854 celle LISP
e di 4998 celle per lo 'stack'.

In ambiente LISP e' possibile trattare files (di tipo LISP)

Questa versione dell'interprete LISP e' incompleta, fra l'altro non tratta matrici e non ha un compilatore. Per ogni dettaglio sul linguaggio si rimanda al manuale:

LISPCMS CNUCE

LISTF — STAMPA A TERMINALE L'IDENTIFICATORE, LA DIMENSIONE (IN RECORDS) LA DATA E IL TEMPO DI CREAZIONE O CAMBIAMENTO DEI FILES SU DISCO SPECIFICATI, OPPURE CREA UN FILE SUL DISCO PERMANENTE DELL'UTENTE, CONTENENTE INFORMAZIONI DA USARSI IN PROCEDURE EXEC.

LISTF
L

<nome <tipo <modo>>> <(opz1,..,opzn)>
* * *

nome tipo modo
*

Identificatore del file da listare (e' il valore default) si intende rispettivamente: lista per tutti i nomi, tutti i tipi, tutti i modi; se il modo e' omesso, la ricerca e' effettuata su tutti i dischi in lettura e scrittura. Un * preceduto da caratteri (es. ABC*) al posto del nome e/o tipo fa effettuare la ricerca di tutti i files il cui nome e/o tipo iniziano per le letture specificate.

opzioni:

EXEC — crea il file CMS EXEC P1 sul disco permanente contenente la lista dei files richiesti, e cancella un eventuale file CMS EXEC gia' esistente. Ogni identificatore di file e' preceduto dai parametri &1 &2 propri delle procedure EXEC.

ITEM — stampa il numero dei records logici invece del numero dei records fisici di 800 bytes.

SORT — raggruppa per tipo i files specificati.

NAME — produce una lista dei soli nomi dei files.

TYPE — produce una lista contenente nome e tipo dei files.

MODE — produce una lista con nome, tipo e modo dei files.

REC — produce una lista con nome, tipo, modo e numero dei records.

DATE — come per REC con in piu' la data in cui il

file e' stato scritto l'ultima volta. (Questa e' la linea Default).

YEAR — include anche l'anno nella data (mm/gg/aa).

TIME — alla linea default sara' aggiunto il tempo (in ore e minuti) dell'ultimo aggiornamento dei files.

NOTE - Tutti gli operandi sono opzionali. Se non e' specificato alcun operando, viene stampata una lista (formato linea default) completa di tutti i files esistenti sui dischi in lettura e scrittura.

ESEMPI:
1) LISTF * FORTRAN
Viene stampata a terminale una lista di tutti i files di tipo FORTRAN.

LOAD — CARICA IN MEMORIA DA DISCO IL FILE(S) TEXT SPECIFICATO, CONTENENTE MODULI RILOCABILI, STABILENDO GLI OPPORTUNI AGGANCI; I MODULI MANCANTI SONO RICERCATI NELLE LIBRERIE APPROPIATE SECONDO QUANTO SPECIFICATO DALLE OPZIONI.

LOAD fn1...fnN <(opz1...opzN)><lbn1...lbnN>

fn1...fnN — specifica i files di tipo TEXT da caricare in memoria

lbn1...lbnN — specifica i nomi (max 8) delle librerie TXTLIB in cui ricercare le subroutines mancanti.

Opzioni :

CLEAR — azzera, prima del caricamento, l'area di memoria interessata

NOCLEAR — non azzera l'area di memoria prima del caricamento

SLCXXXXX — inizia il caricamento dalla locazione XXXXX(hex.)

SLC12000 — inizia il caricamento dalla locazione 12000 (hex.)

| | |
|---|---|
| NOMAP | non crea il file LOAD MAP |
| MAP | crea il file LOAD MAP |
| TYPE | stampa il file LOAD MAP a terminale |
| NOTYPE | non stampa il file LOAD MAP a terminale |
| SINV | sopprime le "invalid card images" dal file LOAD MAP |
| PINV | include le "invalid card images" nel file LOAD MAP |
| SREP | sopprime le "replace card images" dal file LOAD MAP |
| PREP | include le "replace card images" nel file LOAD MAP |
| LIBE | esegue la ricerca delle routines mancanti solo nelle librerie TXTLIB specificate nel comando LOAD |
| SLIBE | non esegue alcuna ricerca nelle librerie TXTLIB per i riferimenti non risolti |
| SAUTO | sopprime la ricerca automatica dei files TEXT |
| AUTO | esegue la ricerca dei files TEXT nei dischi P, T, e S, al fine di risolvere i riferimenti non definiti (AUTO e' default e non puo' essere specificato come opzione) |
| XEQ | inizia l'esecuzione dei files caricati subito alla fine del caricamento se non vi sono errori. |
| NOXEQ | non esegue i files caricati. |

NOTE - I files TEXT specificati nel comando LOAD devono essere moduli rilocabili, come quelli prodotti dai comandi: ASSEMBLE, FORTRAN o PLI.

- L'opzione LIBE permette la ricerca di moduli mancanti solo nelle librerie specificate nel comando LOAD, impedendo la ricerca in librerie specificate da eventuali comandi GLOBAL LOADER precedenti.

- Se SLCXXXXX compare come prima opzione deve essere preceduta da uno o piu' spazi vuoti.

- La ricerca di moduli mancanti durante il caricamento viene effettuata prima sui dischi nell'ordine di ricerca normale, dopo, se vi sono ancora simboli non risolti, nelle librerie TXTLIB appropriate.

ESEMPI:
1) LOAD PINCO ( SLC15000 ) MYLIB
Il file PINCO TEXT viene caricato a partire dalla locazione esadecimale 15000 e eventuali moduli mancanti vengono ricercati nella libreria MYLIB TXTLIB.

LOADMOD — CARICA IN MEMORIA, NELLE STESSE LOCAZIONI IN CUI E' STATO GENERATO, UN FILE DI TIPO MODULE (CONTENENTE CODICE IMMAGINE DI MEMORIA NON RILOCABILE)

LOADMOD filename <filemode>

filename — e' il nome del file da caricare in memoria, il cui tipo deve essere MODULE
filemode — e' il modo del file da caricare in memoria

NOTE - LOADMOD si usa per caricare un file creato dal comando GENMOD. Se il file e' stato generato senza mappa di caricamento, LOADMOD non deve essere usato perche' un successivo comando START non sarebbe in grado di inizlare l'esecuzione. In questo caso occorre scrivere il nome del file da solo, come se fosse un comando CMS.

- Se non si specifica il modo, la ricerca del file e' quella normale.

ESEMPI:
1) LOADMOD DETTO P2
Il file DETTO MODULE P2 viene caricato in memoria; viene ricercato solo sul disco P.

LOGIN — PERMETTE L'ATTIVAZIONE DI UNA MACCHINA VIRTUALE.

LOGIN userid
L

userid — nome della macchina virtuale da attivare.

NOTE - Il CP risponde alla richiesta chiedendo la parola di riconoscimento, scrivendo tre stringhe di caratteri sovrapposte e sbloccando la tastiera.

- Questa funzione permette di iniziare una seduta a terminale con una data macchina virtuale.

LOGIN — REGOLA L'ACCESSO AI FILES DEL DISCO SPECIFICATO ASSOCIANDOVI UN MODO.

LOGIN
NOPROF
UFD
<NO-UFD>
<NO_UFD>

ccu
<UFD>
<NO-UFD>
Z<,Y<fn<ft<fm>>>>
P

NOPROF — va emesso come primo comando dopo l'IPL del CMS e impedisce l'esecuzione del file PROFILE EXEC

UFD — porta in memoria la lista dei files del disco specificato (P default) rendendo accessibili tali files

NO-UFD
NO_UFD — cancella la lista dei files del disco liberando cosi' lo spazio disco da essi occupato

ccu — specifica l'indirizzo del disco da attivare

Z — specifica il modo del disco da attivare

,y — indica il modo del disco del quale il disco

Z deve essere una estensione in solo-lettura

fn ft fm — specifica il file(s) su disco solo-lettura di cui sara' permessa la lettura.

NOTE - Il file di nome PROFILE e tipo EXEC, quando presente, viene eseguito automaticamente dal CMS dopo l'IPL se il primo comando non e' LOGIN NOPROF, LOGIN-UFD o FORMAT. Tale file puo' contenere una sequenza qualunque di comandi CMS e parole di controllo EXEC.

- Se e' specificato un operando, questo deve essere NOPROF o UFD, o NO-UFD, o NO_UFD o ccu. Se non e' specificato alcun operando, viene attaccato alla macchina virtuale il disco 191 o l'ultimo disco usato come disco P.

- Se LOGIN ccu e' emesso come primo comando dopo IPL, ccu dovrebbe specificare il disco da usare come disco P.

- Il comando LOGIN NO-UFD non formatta fisicamente il disco, per cui risulta piu' veloce del comando FORMAT quando il disco sia gia' formattato e si voglia cancellare tutti i files.

- L'estensione di un disco in sola-lettura modifica l'ordine normale di ricerca dei files che e' P,T,A,B,S,C ma effettuato solo sui dischi attivi, spostando il disco estensione subito dopo il disco esteso nella catena sopra indicata

- Il modo dei files su qualunque disco e' sempre P. Se il disco viene attivato con un modo diverso, per es. B, sara' il CMS a tradurre P in B in tutti i messaggi a o da terminale.

ESEMPI:

1) LOGIN 193 NO-UFD
Il disco 193 viene attivato come disco P e i files eventualmente presenti su di esso cancellati

2) LOGIN 291 A,P
Il disco 291 viene attivato come disco A estensione in solo-lettura del disco P, cioe' il CMS avra' accesso a tutti i files dei dischi P e A ma potra' soltanto leggere, e non scrivere, sul disco A

3) LOGIN 196 B,B
Il disco B viene attivato come estensione in solo-lettura di se stesso.

LOGOUT — STACCA DAL SISTEMA LA MACCHINA VIRTUALE E LE UNITA' AD ESSA COLLEGATE, CHIUDE LE AREE DI SPOOLING RELATIVE ALLA MACCHINA.

LOGOUT <stringa>
LOG

stringa — e' una qualsiasi stringa di caratteri non-bianchi; se presente la linea di trasmissione non viene disattivata.

NOTA - Questa funzione viene usata per terminare una seduta a terminale con una data macchina virtuale.

LOGOUT — CHIUDE I FILES E I DISPOSITIVI DI I/O, COMPATTA LA USER'S FILE DIRECTORY E LO SCRIVE SUL DISCO APPROPRIATO; ESEGUE POI OGNI COMANDO CMS SPECIFICATO E ESCE DAL CMS, TRASFERENDO IL CONTROLLO AL CP.

LOGOUT <comando>

comando — qualsiasi comando CMS che non crei distrugga o modifichi files.

MACLIB — GENERA UNA LIBRERIA DI MACRODEFINIZIONI; AGGIUNGE, CANCELLA O SOSTITUISCE DELLE MACROS IN UNA LIBRERIA ESISTENTE, STAMPA A TERMINALE NOME, DIMENSIONE E LOCAZIONE DELLE MACRO DEFINIZIONI CONTENUTE IN UNA LIBRERIA; COMPATTA UNA LIBRERIA.

```
GEN libname   filename1...filenameN
ADD libname   filename1...filenameN
```

```
                REP libname   filename1...filenameN
                DEL libname   macroname1...macronameN
MACLIB          COMP libname
                PRINT libname
                LIST libname
```

| | |
|---|---|
| GEN | genera la libreria di macros "libname" dalle macros contenute nel file(s) specificato |
| ADD | aggiunge alla libreria "libname", le macro definizioni contenute nel file(s) specificato |
| REP | aggiunge una macro(s) alla libreria specificata, e cancella la macro (s) con lo stesso nome |
| DEL | cancella dal dizionario le entrate delle macros specificate |
| COMP | compatta la libreria specificata e cancella le macros le cui entrate sono state tolte dal dizionario |
| PRINT | crea un file "libname MAP P1", contenente nome, dimensione, e posizione delle macros della libreria, e lo stampa sulla stampante offline |
| LIST | scrive a terminale il dizionario della libreria di macros specificata dal "libname". |

NOTE - Il tipo dei files usati col comando MACLIB deve essere ASP360 o COPY.

- Il dizionario di una libreria comprende: nome, posizione relativa e dimensione delle macros.

- Le librerie macro, tranne le SYSLIB e OSMACRO del CMS, per essere utilizzate da comandi di caricamento devono essere rese attive con un comando di GLOBAL M.

- Per la stampa del contenuto di una macro vedere il comando MACRO.

ESEMPI:

1) MACLIB LIST OSMACRO
   Sara' stampato a terminale il dizionario della libreria OSMACRO

2) MACLIB ADD LORA MAC3 MAC12
   Saranno aggiunte le due macro MAC3 e MAC12 alla

libreria LORA.

MACRO PERMETTE DI CONOSCERE IL TESTO DI UNA MACRO.

```
                                    TYPE
                                    PRINT
MACRO  nome <lib      <modo>>       PUNCH
            SYSLIB    *             FILE   <fname <ftype>>
                                             nome   ASP360
```

nome nome della macro di cui si desidera avere il testo

lib nome della libreria (file di tipo MACLIB) nella quale deve essere ricercata la macro)

modo indica il disco su cui si trova la libreria

fname nome del file sul quale si vuole copiare il testo della macro

ftype tipo del file sul quale si vuole copiare il testo della macro.

Uso: La macro 'nome' viene ricercata nella libreria

libreria MACLIB modo

e ricopiata

1) su terminale, se si e' specificato TYPE
2) su stampante, se si e' specificato PRINT
3) su schede, se si e' specificato PUNCH
4) su file nome ASP360 se non si specificano parametri.

Si possono ricevere le seguenti risposte:

'FILE NOT FOUND'

se non e' stata trovata la libreria;

'MACRO NOT FOUND'

se non e' stata trovata la macro.

Esempi:

MACRO TYPE Il testo della macro 'TYPE' viene copiato dalla libreria SYSLIB sul file TYPE ASP360.

MACRO GET OSMACRO (TYPE) Il testo della macro GET viene copiato dalla libreria OSMACRO sul terminale.

MAPPRT — CREA SU DISCO E, SE SPECIFICATO, STAMPA UN FILE CONTENENTE UNA MAPPA DEI PUNTI DI ENTRATA DEL NUCLEO CMS.

```
              A      ON
MAPPRT       <N     <OFF>>
              C      NO
```

A — crea il file CMS-NUC ALPHABET P1, contenente i punti di entrata del nucleo in ordine alfabetico

N — crea il file CMS-NUC NUMERIC P1, contenente i punti di entrata del nucleo in ordine numerico

C — crea il file CMS-NUC ALPHANUM P1, contenente entrambi gli ordinamenti alfabetico e numerico

ON — stampa il file a terminale

OFF — stampa il file sulla stampante offline

NO — non esegue alcuna stampa del file

NOTA - Questo comando per avere effetto deve essere emesso come primo comando dopo un IPL di CMS.

MODMAP — STAMPA A TERMINALE LA MAPPA DI CARICAMENTO ASSOCIATA AL FILE MODULE SPECIFICATO.

MODMAP filename

filename — nome del file il cui tipo deve essere MODULE

NOTA -La mappa di caricamento e' creata dal comando GENMOD se non e' specificata l'opzione (NOMAP).

MOVEFILE — COPIA IL CONTENUTO DI UN FILE IN UNO O PIU'

FILES .

MOVEFILE nomein tipoin nomeout1 tipoout1 <etc
* *

nomein e' il nome del file il cui contenuto deve essere copiato.

tipoin e' il tipo del file il cui contenuto deve essere copiato.

nomeout1 e' il nome del file su cui deve essere copiato il contenuto di 'nomein tipoin'.

tipoout1 e' il tipo del file su cui deve essere copiato il contenuto di 'nomein tipoin'.

<etc si possono specificare i nomi e tipo di quanti altri files si voglia.

* equivale a scrivere il nome FILE.

NOTE -L'esecuzione del comando inizia con la sostituzione della parola 'FILE' a tutti gli asterischi in corrispondenza degli argomenti di posizione dispari. Successivamente il primo file della lista viene copiato su tutti gli altri files.
N.B. Per lo spostamento di files su disco e' conveniente usare il comando COMBINE. MOVEFILE e' particolarmente utile se usato insieme al comando FILEDEF.

ESEMPI:
1) filedef reader rdr
R;....
filedef console con
R;....
movefile * reader * console gauss fortran
R;....
Il file che si trova sul lettore di schede viene stampato sul terminale e contemporaneamente ricopiato nel file GAUSS FORTRAN.

2) filedef printer ptr
R;....
movefile gauss fortran * printer
R;....

Il contenuto del file gauss fortran viene stampato

sulla stampante veloce. Diversamente dal comando 'OFFLINE PRINT', MOVEFILE non provoca la stampa dell'intestazione 'CAMBRIDGE MONITOR SYSTEM', etc. all'inizio di ogni pagina e puo' quindi essere utile quando si e' fatto
XFER E TO XXXXXXXX.

-La combinazione dei comandi CSET, DSNAME e FILEDEF permette di fare in modo che ogni programma che esegua l' I/O su disco, lettore, perforatore, terminale o stampante sia costretto ad eseguirlo su un qualunque altro device fra questi 5.
Ad esempio, un programma che legga dal lettore di schede puo' essere costretto a leggere da console mediante la seguente combinazione di comandi:

```
CSET  RDR  OFF
R;
FILEDEF  CARDIN  CON
R;
```

Il primo comando dice che l'input dal lettore di schede deve in realta' avvenire dal file FILE CARDIN.
Con il secondo comando si fa in modo che l'input che deve avvenire dal file FILE CARDIN deve essere in realta' letto dal terminale. In altre parole, per passare dal lettore al terminale e' necessaria la seguente sequenza di devices:
lettore - disco - terminale
E' necessario comunque osservare la seguente norma:

PER PASSARE DA UN DEVICE AD UN ALTRO OCCORRE SEMPRE USARE LA SEQUENZA PIU' BREVE.

Per esempio, nel caso precedente non sarebbe stato permesso usare la sequenza:
lettore-disco-lettore-disco-terminale
Una conseguenza della precedente norma e' che non sono permesse sequenze in cui il primo e l'ultimo device siano gli stessi, nemmeno per il passaggio da disco a disco. Sempre per la stessa ragione non bisogna mai eseguire il comando
CSET READ OFF FILE FT05F001
oppure il comando
CSET TYPE OFF FILE FT06F001
Questo e' vero perche', essendo sempre valide le definizioni:
FT05F001 CON
FT06F001 CON
ciascuno dei due comandi provocherebbe una sequenza:
terminale-disco-terminale

MSG — TRASMETTE UN MESSAGGIO ALL'UTENTE SPECIFICATO.

```
MSG          userid      messaggio
 M             CP
```

| | |
|---|---|
| userid | nome della macchina virtuale a cui il messaggio deve essere trasmesso. |
| CP | il messaggio e' trasmesso all'operatore del sistema |
| messaggio | testo del messaggio da trasmettere. |

NOTA - Questa richiesta permette lo scambio di informazioni con tutte le macchine virtuali attive.

ESEMPI:

1) MSG CP prego montare nastro C40 come 181
   Richiesta all'operatore di montare un nastro con indirizzo di unita' 181.

NASTRI — PERMETTE DI EFFETTURE L'ANALISI DEL CONTENUTO DI UN NASTRO E DI TRASFERIRE FILES O BLOCCHI DA NASTRO A DISCO, STAMPANTE O TERMINALE.

NASTRI

Messaggi: 'DEV ID <180> OR <181>'
Rispondere 180 oppure 181

'TAPE IS WRITTEN IN x TRACKS DENS yyyy'
Dove x e' uguale a 7 oppure a 9, e yyyy sara' 200, 556, 800 o 1600.
Il nastro e l'unita' di lettura sono compatibili e il nastro e' scritto con densita' yyyy.

'TAPE AND UNIT ARE NOT COMPATIBLE'
Il nastro e l'unita' non sono compatibili, cioe'
- o nastro a 7 piste e unita' a 9 piste
- o nastro a 9 piste e unita' a 7 piste
- o nastro vergine

```
'AFTER EACH MESSAGE 'REPLY' YUO CAN USE'
'S N    -SCAN N FILES'
'L N    -LOCATE N FILES'
'LB N   -LOCATE BLOCK N'
'  '    -END                          linea vuota
'D N M     -DUMP M BLOCKS,LRECL=N'
'P N M     -PRINT M BLOCKS,LRECL=N'
'PCC N M   -PRINTCC M BLOCKS,LRECL=N'
'T N       -TYPE ONE BLOCK,LRECL=N'
'IF M IS OMITTED THE WHOLE FILE IS ASSUMED'
```

Descrizione dei diversi comandi nel programma .di utilita' NASTRI:

SCAN — LEGGE SUL NASTRO N FILES E SCRIVE A TERMINALE LA CONFIGURAZIONE DI CIASCUNO DI ESSI.

S N

Le informazioni ottenute per ciascun file sono:

- Il numero d'ordine (partendo dal load point)
- La lunghezza dei blocchi e il numero dei bloccchi di lunghezza uguale trovati durante la lettura

La posizione del nastro dopo l'esecuzione del comando SCAN, sara' l'inizio del file successivo

Messaggi: 'GAP NOT CORRECT'
Esiste un errore di scrittura sul nastro oppure il nastro e' vergine

'AFTER THIS POINT IS NOT POSSIBLE TO READ ANYTHING'
Termina l'azione del comando in esecuzione ed e' pronto ad accettare un nuovo comando

Nota: '0006 *EOF*
il file 6 e' costituito solo da un tape-mark

LOCATE — POSIZIONA IL NASTRO ALL'INIZIO DEL FILE N E NE SCRIVE A TERMINALE LA CONFIGURAZIONE

L N

Messaggi: Vedere il comando SCAN

LOCATE BLOCK  POSIZIONA IL NASTRO ALL'INIZIO DEL BLOCCO N DEL FILE CORRENTE

LB N
Messaggi: Vedere il comando SCAN

DUMP  CREA UN FILE SUL DISCO P.

D N <M>

Il file e' identificato da:

| filename | filetype | filemode |
|---|---|---|
| Dxxxxy | DUMP | P1 |

dove:
xxxx e' il numero d'ordine del file corrente sul nastro
y e' il numero delle volte (da 1 a 9) che il comando DUMP e' stato attivato sul file corrente.

Se gia' esisteva sul disco P un file di nome Dxxxxy DUMP P1 tale file e' cancellato.

N rappresenta la lunghezza dei record logici sul disco
M rappresenta il numero dei blocchi sul nastro che devono essere ricopiati sul disco P.
Se M viene omesso sul disco P sara' ricopiato l'intero file.

Messaggi: Vedere il comando SCAN
'GIVE LRECL,REPLY N'
Se N e' stato omesso nel comando, rispondere N, ed M se voluto

'THE FILE IS DUMPED AS Dxxxxy DUMP P1'
Il comando DUMP ha creato sul disco P un file di nome Dxxxxy DUMP P1

Alla fine del comando il nastro e' posizionato all'inizio del blocco che segue l'ultimo blocco su disco.

Errori: i codici di errore da 00001 a 00019 della funzione WRBUF

PRINT  FA USCIRE SULLA STAMPANTE IL CONTENUTO DI M BLOCCHI.

P N <M>

N e' il numero dei caratteri che si vogliono su ogni riga di stampa
M e' il numero dei blocchi scritti. Se M viene omesso sara' stampato l'intero file.

Messaggi: Vedere il comando SCAN
'GIVE LRECL,REPLY N N<=133'
Se N e' stato omesso nel comando.
Rispondere con N 133 e >0, ed M se voluto.

'FILE PRINTED'
Il comando PRINT e' regolarmente terminato.

Dopo l'esecuzione del comando il nastro rimane posizionato all'inizio del blocco che segue l'ultimo blocco stampato.

Errori: i codici di errore da 00030 a 00035 della funzione PRINT.

PRINTCC FUNZIONA COME IL COMANDO PRINT TRANNE CHE IL PRIMO CARATTERE DELLA LINEA E' UTILIZZATO COME CARATTERE DI CONTROLLO.

PCC N M

Messaggi: Vedere il comando PRINT

Errori: Vedere il comando PRINT

Dopo l'esecuzione del comando il nastro rimane posizionato all'inizio del blocco che segue l'ultimo blocco stampato.

TYPE STAMPA SUL TERMINALE IL CONTENUTO DEL BLOCCO CORRENTE.

T N

N e' il numero dei caratteri della riga di stampa a terminale

Messaggi: Vedere il comando SCAN

Dopo l'esecuzione del comando il nastro rimane posizionato all'inizio del blocco stampato sul terminale.

END IL COMANDO PERMETTE DI RITORNARE AL CMS.

IL NASTRO RIMANE POSIZIONATO DOVE SI TROVAVA DOPO L'ESECUZIONE DEL COMANDO PRECEDENTE.

Ritorno di carrello (linea vuota)

| | |
|---|---|
| OFFLINE | CREA FILES SU DISCO DA INPUT SU SCHEDE, STAMPA UN FILE RESIDENTE SU DISCO SULLA STAMPANTE OFFLINE O LO PERFORA SU SCHEDE. |

OFFLINE comando filename filetype (filemode)
0 *

comandi:

| | |
|---|---|
| READ | specifica che deve essere letto un deck dal lettore di schede |
| PRINT | il file specificato sara' stampato sulla stampante offline con spaziatura automatica singola. |
| PRINTCC | il file specificato deve essere stampato interpretando il primo carattere di ciascun record come carattere di controllo del carrello. |
| PRINTUPC | i records del file specificato saranno tradotti in caratteri maiuscoli, poi stampati. |
| PRINTVLR | sara' stampato il file specificato, con records di lunghezza variabile, prodotto da un compilatore di linguaggio OS/360. |
| PUNCH | il file specificato sara' perforato sulla perforatrice offline. |
| PUNCHCC | una scheda "OFFLINE READ filename filetype" sara' inserita come prima scheda, prima della perforazione del file specificato. |
| PUNCHDT | sara' inserita come prima scheda del file perforato una scheda controllo OFFLINE READ filename filetype filemode data e tempo in |

cui il file e' stato scritto l' ultima volta.

* specifica che filename, filetype e filemode saranno ottenuti da una scheda di controllo OFFLINE READ nel deck di schede in input (valido solo col comando READ).

NOTE - La lunghezza massima dei records in lettura e' di 132 caratteri.

- Se e' omesso il filemode, in lettura si considera che sia P1, in output la ricerca del file e' effettuata nell' ordine normale.

- Se in un comando OFFLINE READ si specifica filename e filetype, sara' letto tutto il deck nel lettore di schede come un solo file. Se si vogliono leggere piu' files da un unico deck di schede occorre specificare "OFFLINE READ *" (un solo asterisco) e il deck di schede deve contenere,come prima scheda di ogni file, una scheda controllo avente perforato da colonna 1 OFFLINE READ filename filetype <filemode> specificante l'identificatore del file che segue.

- Quando si opera con una macchina virtuale, i decks di schede dell'utente devono essere prima letti dal CP. E' necessario allora che la prima scheda del deck sia una scheda controllo contenende ID a colonna 1 e 2 e l'identificazione della macchina virtuale a partire da colonna 10.

- Caratteri di controllo del carrello: blank fa eseguire linea di spaziatura semplice; zero ('F0') fa precedere e seguire la linea stampata da spaziatura semplice; uno ('F1') fa saltare all'inizio della pagina seguente, prima che sia stampata la linea. Ogni altro carattere e' considerato un codice di operazione valido di CCW ed e' messo nella CCW associata con la SIO. La max lunghezza di una riga di stampa con controllo di carrello e' di 133 caratteri.

ESEMPI:

1) OFFLINE PRINTCC FRANCO LISTING
Il file FRANCO LISTING, con records quindi di 133 caratteri, viene inviato alla stampante (ma non cancellato dal disco) e il primo carattere di ogni record viene considerato come carattere di controllo

del carrello.

2) OFFLINE READ *
Se il primo file sul lettore di schede contiene come prima scheda OFFLINE READ AC FORTRAN e dopo il programma di nome AC un'altra scheda OFFLINE READ ANC SYSIN e un programma in assembler, saranno letti su disco i due programmi AC FORTRAN e ANC SYSIN e il messaggio:

OFFLINE READ AC FORTRAN
OFFLINE READ ANC SYSIN

sara' stampato sul terminale.

OSBATCH — PERMETTE DI INVIARE PER L'ELABORAZIONE FILES DA MACCHINE VIRTUALI AL SISTEMA OPERATIVO OS.

OSBATCH

Il comando richiede le informazioni necessarie alla definizione del job con le domande seguenti:

NOMP:CCCCCCCC PGRM:XXXX ACCNT:CXXX TEMPO:XXXX RIGHE:XX <VER>

| | |
|---|---|
| NOMP | nome del job |
| PGRM | codice numerico del programmatore |
| ACCNT | codice di accounting dell'utente |
| TEMPO | tempo di CPU in minuti |
| RIGHE | righe di stampa in migliaia |
| VER | se incluso nella lista stampa sul terminale ogni scheda di controllo (JCL) inserita nel job |
| Risposte possibili: | CCCCCCCC XXXX CXXX XXXX XX<br>CCCCCCCC XXXX CXXX XXXX XX VER |

NOTA - Il carattere C indica "carattere qualunque", mentre X indica "cifra decimale".

RITORNO DELL'OUTPUT? NO OPPURE USERID

| | |
|---|---|
| userid | nome della macchina virtuale sul cui |

lettore l'utente desidera che sia inviato l'output.

REGION XXXXK

REGION ampiezza dell'area di memoria prevista per compilazione ed esecuzione.

Risposta: XXXXK

PROCEDURA E LISTA DI OPZIONI

PROCEDURA le procedure ammesse sono: FORTG, FORTH, FORTX, ASMF, ASMH, PL1.

LISTA DI OPZIONI le opzioni default per le singole procedure sono:
FORTG:NODECK,NOLIST,MAP,SOURCENO,LET,CALL
FORTH:quelle di FORTG piu' NOXREF,(l'ultimo e' il valore default dell'opzione OPT)
FORTX: quelle di FORTH piu' NOFMT
ASMF:NODECK,NOLET,MAP,CALL
ASMH:le stesse di ASMF
PL1:NODECK,NOLIST,XREF,NOMACRO,MAP,NOLET, CALL,(l'ultimo e' il valore default dell'opzione OPT).

NOTA - Se la procedura non e' fra quelle ammesse viene segnalato l'errore e ripetuta la domanda. E' sufficiente indicare i valori non default delle opzioni. I valori delle opzioni inesistenti non ammessi per la procedura in oggetto vengono ignorati.

Risposta: nomproc opt1 opt2....optN

NOME E TIPO DEL FILE PROGRAMMA SORGENTE

Risposta: filename filetype

NOTA - Se il file dato non esiste viene segnalato l'errore e ripetuta la domanda.

ALTRI FILE DI USCITA? NO OPPURE NOME X
(X: A=STAMPANTE, B=PERFORATORE)

Risposta: NO
nome A
nome B

NOTA - Questo messaggio permette di specificare un altro file di uscita oltre a SYSPRINT.

DATI? NO OPPURE NOME E TIPO DEL FILE.

Risposta: NO
filename filetype

ESEMPI:

1)
```
osbatch
NOMP:CCCCCCCC PGRM:XXXX ACCNT:CXXX  TEMPO:XXXX RIGHE:XX
<VER>
roar 0000 k004 0001 01
RITORNO DELL'OUTPUT? NO OPPURE USERID
ugo
REGION:XXXXK
0128k
PROCEDURA E LISTA DI OPZIONI
fortg
NOME E TIPO DEL FILE PROGRAMMA SORGENTE
proval fortran
ALTRI FILES DI USCITA ? NO OPPURE NOME X
(X: A=STAMPANTE,B=PERFORATORE)
no
DATI ? NO OPPURE NOME E TIPO DEL FILE
dati data
** CARDS XFERED TO OS370 **
```

2)
```
osbatch
NOMP:CCCCCCCC PGRM:XXXX ACCNT:CXXX  TEMPO:XXXX RIGHE:XX
<VER>
roar 0000 k004 0001 01 ver
RITORNO DELL'OUTPUT? NO OPPURE USERID
remidi
REGION:XXXXK
0128k
//ROAR       JOB   (0000,K004,0001,01),'=REMISI    ROAR
',REGION=0128K
PROCEDURA E LISTA DI OPZIONI
fortg v call
// EXEC FORTG
//COMP.SYSIN DD *
NOME E TIPO DEL PROGRAMMA SORGENTE
proval fortran
/*
ALTRI FILES DI USCITA ? NO OPPURE NOME X
(X: A=STAMPANTE,B=PERFORATORE)
eco b
//GO.ECO      DD SYSOUT=B
```

```
//GO.SYSIN DD *
DATI ? NO OPPURE NOME E TIPO DEL FILE
dati data
/*
** CARDS XFERED TO OS370 **
```

OSTAPE — CREA FILES SU DISCO DA FILES FORMATO SCHEDA SU NASTRO CREATI DA ALTRI SISTEMI OPERATIVI.

OSTAPE <ft <fn>> (opzione1...opzioneN)
SYSIN OSTAPE

ft — tipo di file creato (default SYSIN)

fn — nome del file creato (default OSTAPE). Solo per files NPDS.

Opzioni:

| | |
|---|---|
| PDS | specifica che il nastro contiene membri di un partitioned data set ciascuno preceduto da "member name XX" |
| NPDS | specifica che il nastro non contiene membri di un partitioned data set |
| COL1 | specifica che la colonna 1 contiene dati |
| NCOL1 | specifica che la colonna 1 contiene un carattere di controllo del carrello |
| TAPX | X indica il numero dell'unita' nastro (TAP2 default) |
| END | specifica che la scheda END significa fine del membro di un partitioned data set |
| NEND | la scheda END non significa fine del membro di un partitioned data set |
| MAXTEN | indica che possono essere letti contemporaneamente dieci data sets al massimo |
| NMAXTEN | possono essere letti piu' di |

dieci date set contemporaneamente.

NOTA - Puo' essere letto un nastro prodotto dal programma di utilita' dell'OS/360 IEBPTPCH.

PACK — CREA UNA COPIA COMPATTATA DI UN FILE.

PACK nome1 tipo1 <nome2><tipo2>
*

nome1 — nome del file da compattare
tipo1 — tipo del file da compattare. Puo' essere omesso se il file e' di tipo FORTRAN.
nome2 — nome del file compattato creato; se * si intende uguale a nome1
tipo2 — tipo del file compattato. Se omesso, al file nome2 sara' assegnato un tipo creato con le regole della seconda nota.

NOTE - Questo comando crèa un nuovo file compattando tutte le sequenze di piu' di tre caratteri uguali.

- Il file di partenza non viene cancellato. Il file creato avra' lo stesso nome del file d'origine e tipo cosi' fatto
1) nel caso di tipi di fino a sette caratteri sara' aggiunta una P in coda
2) nel caso di tipi di otto caratteri di cui gli ultimi sette non siano P sara' aggiunta una P in coda e tolto il primo carattere (Questo carattere sara' ripristinato automaticamente nel processo di scompattamento)
3) nel caso che gli ultimi sette caratteri siano tutti P il tipo creato sara' PPPACKED.

- Un file compattato non puo' essere utilizzato dai comandi CMS se prima non viene scompattato.

- Questo comando, insieme al contrario UNPACK, e' utile per ridurre le dimensioni di files su un disco, i quali temporaneamente non servano, per guadagnare spazio disco.

PLI — CONVERTE CODICE SORGENTE, SCRITTO IN LINGUAGGIO PL/1, IN CODICE OGGETTO RILOCABILE.

PLI fn1...fnN <(opz1...opzm)>

fn — nomi dei files PLI da compilare. Ogni file sara' compilato separatamente. Le opzioni specificate saranno valide per la compilazione di tutti i files specificati.

Le opzioni possibili sono:

SIZE nnnnnn (SIZE 999999 il compilatore otterra' quante memorie possibile)

OPT n (def OPT 1)

STMT (def STMT)
NOSTMT

OBJIN
OBJOUT (def OBJOUT)

EXTDIC (def NOEXTDIC)
NOEXTDIC

SYNCHKE (def SYNCHKT)
SYNCHKS
SYNCHKT

MACRO (def NOMACRO)
NOMACRO

COMP
NOCOMP (def COMP)

MACDCK (def NOMACDCK)
NOMACDCK

opzioni per l'ingresso:

CHAR60 (def CHAR60)
CHAR48

BCD (def EBCDIC)
EBCDIC

SORMGIN mm,nn,cc (def SORMGIN 2,72,1)

opzioni per l'uscita:

DECK (def DECK)
NODECK

opzioni per il Listing:

LINECNT xxx (def LINECNT 55)

OPLIST (def OPLIST)
NOOPLIST

SOURCE2 (def SOURCE2)
NOSOURCE2

SOURCE (def SOURCE)
NOSOURCE

NEST (def NEST)
NONEST

ATR (def ATR)
NOATR

XREF (def XREF)
NOXREF

EXTREF (def NOEXTREF)
NOEXTREF

LIST
NOLIST (def NOLIST)

FLAGS (def FLAGW)
FLAGW
FLAGE

DUMP

opzioni di CMS:

PUNCH specifica che il file TEXT prodotto dal compilatore deve essere perforato
PU

TDECK specifica che il file TEXT prodotto dal compilatore deve essere scritto su un nastro montato su una unita' con indirizzo 181
TD

PRINT — specifica che il file LISTING prodotto dal compilatore deve essere stampato sulla stampante

P

NOPRINT — specifica che il compilatore non deve produrre LISTING

NP

TLIST — specifica che il file LISTING prodotto dal compilatore deve essere scritto su un nastro montato su una unita' con indirizzo 181.

TL

NODIAG — specifica che i diagnostici del compilatore non devono essere stampati sul terminale.

NDG

NOTE - I files trattati dal comando PLI devono essere di tipo PLI ed avere records di 80 caratteri.

- E' possibile specificare una qualunque combinazione di opzioni; se due o piu' di esse sono contraddittorie viene ritenuta valida l'ultima opzione.

- Prima di caricare in memoria un file per l'esecuzione occorre attivare le librerie TEXT del PLI mediante il comando CMS.

GLOBAL T PLILIB

NOTA IMPORTANTE - Questo manuale fa generalmente riferimento alla versione 3.0 del CMS. Comunque, essendo decisamente migliore la versione del PL/1 del CMS 3.2, e' opportuno, prima di ogni compilazione, caricamento ed esecuzione di un programma PL/1 passare al CMS 3.2 mediante il comando 3.2 col comando 3.0; evitare di rientrare in CMS 3.0 mediante IPL CMS.

PRINTF — STAMPA A TERMINALE TUTTO IL CONTENUTO, O UNA PARTE SPECIFICATA, DEL FILE INDICATO.

| | | | | | |
|---|---|---|---|---|---|
| PRINTF<br>P | filename | filetype | n1<br>* | n2<br>* | n 3 |

n1 — indica il numero d'ordine della prima linea da stampare

n2 — indica il numero d'ordine dell'ultima linea da stampare

n3 — numero massimo di caratteri per linea di stampa, se le linee devono essere troncate.

NOTE - Un asterisco in luogo di n1 o n2 specifica, rispettivamente, l'inizio o la fine del file.

- Se n1 e n2 sono omessi o e' specificato *, sara' stampato l'intero file.

- Le righe di stampa sono troncate al valore n3, se indicato; altrimenti a 133 caratteri per i files LISTING, a 120 per i files SCRIPT, a 80 per i files MEMO, a 72 negli altri casi.

- Nel caso di files di tipo LISTING il primo carattere (di controllo del carrello) non viene stampato.

ESEMPI:

1) PRINTF MARCO PLI * 10 50
Il file MARCO PLI viene stampato dalla prima riga alla decima e ogni record e' troncato a colonna cinquanta.

## PROGRAMMAZIONE IN ASSEMBLER

### Nome del programma

Se il punto d'ingresso del programma (entry point) ha lo stesso nome del file TEXT contenente il programma questo puo' essere eseguito per mezzo del comando

```
$ nome
```

Altrimenti e' necessario ricorrere alla sequenza:

```
LOAD nomefile
START nomeprog
```

### Inizio e fine del programma

Per iniziare l'esecuzione di un programma occorre dunque emettere uno dei due comandi

```
$ nome <par1<par2<par3...<parn>...>>>
```

oppure

```
START nome <par1<par2<par3...<parn>...>>>
```

Le parole di ogni comando CMS vengono poste dal CMS in doppie voci consecutive allineate a sinistra, troncate all'ottavo carattere; per esempio il comando START viene ristrutturato nel seguente modo:

```
DC    0D
DC    CL8'START'
DC    CL8'nome'
DC    CL8'par1'
DC    CL8'par2'
DC    CL8'par3'
.
.
.
DC    CL8'parn'
```

quando parte l'esecuzione del programma, nel registro 1 viene caricato l'indirizzo della seconda doppia voce della lista (che contiene il nome del programma).Cosi' e' possibile passare al programma dell'utente dell'informazione parametrizzata servendosi del registro 1. Il CMS carica quindi nel registro 15 l'indirizzo iniziale del programma

dell'utente e vi salta con una

```
BALR   14,15
```

E' bene quindi che il programma termini l'esecuzione con una

```
BR     14
```

per ritornare al nucleo del CMS, e terminare anche l'esecuzione del comando $ o START, provocando la stampa di "R; T=n.nn/x.xx xx/xx/xx" se il registro 15 contiene uno 0 o di "E (nnnnn);T=n.nn/x.xx xx/xx/xx" se il registro 15 contiene nnnnn. Per evitare la stampa del messaggio d'errore al termine del comando occorre azzerare il registro 15 prima del BR 14.

Chiamata dei comandi e dei programmi del CMS

Ogni comando di CMS puo' essere richiamato dall'interno di un programma ponendo le parole componenti il comando in doppie voci consecutive, caricando nel registro 1 l'indirizzo di questa lista ed emettendo una SVC 202.

Es.:

```
          LA     1,PLIST
          SVC    202
           .
           .
           .
PLIST     DC     0D
          DC     CL8'ERASE'
          DC     CL8'*'
          DC     CL8'WORKFILE'
          DC     X'FFFFFFFF'
```

L'ultima doppia voce deve essere X'FFFFFFFF' per individuare fine lista. Il ritorno con errore da un comando chiamato via SVC provoca l'entrata in DEBUG a meno che non si specifichi un indirizzo di rotorno-in-caso-di-errore nel seguente modo:

```
          LA     1,PLIST
          SVC    202
          DC     AL4(ERRET)
          LH     3,26(1)
```

Cosi', in caso di errore, al ritorno dalla SVC si salta a ERRET, altrimenti l'esecuzione continua dall'istruzione successiva (LH in questo caso). Volendo che gli errori siano ignorati occorre scrivere:

```
          SVC    202
```

```
        DC      AL4(*-4)
```

La chiamata via SVC e' valida per tutti i programmi residenti su disco e per quelli del nucleo elencati nella tabella FUNCTAB (tutti i comandi di CMS rientrano in queste due categorie). Per eseguire i programmi del nucleo che non sono elencati in FUNCTAB occore usare una

```
        BALR 14,15
```

Questo presuppone la conoscenza dell'indirizzo del programma da eseguire. A partire dall'entry-point SYSREF del nucleo e' elencata una lista degli indirizzi delle routines del nucleo piu' usate. Tramite la macro CMSYREF della libreria SYSLIB, (la cui espansione produce statements del tipo:

```
        Dnome EQU disp
```

dove nome e' il nome di una routine e disp e' il displacement a partire da SYSREF della locazione in cui si trova l'indirizzo di nome) e' possibile richiamare qualunque routine elencata in SYSREF. Per esempio volendo richiamare SCAN occorre scrivere:

```
     L      3,=V(SYSREF)     R3=IND.DI SYSREF
     L      15,DSCAN(3)    R15=IND. DI SCAN
     BALR   14,15
       .
       .
       .
       .
       .
     CMSYSREF         MACRO  CMSYSREF
```

Una lista dell'espansione della macro CMSYREF puo' essere ottenuta col comando MACRO.

Per la descrizione di tutti i programmi del CMS si rimanda comunque al "CMS Program Logic Manual IBM Y20-0591."

## MACRO DEL CMS

Nelle pagine che seguono vengono descritte le macro proprie del CMS.Generalmente la loro espansione consiste in una chiamata della SVC 202 dopo aver caricato nel registro 1 l'indirizzo di una lista di parametri. Occorre osservare che una macro del CMS provoca generalmente la distruzione del contenuto dei registri 1 e 15.
Diamo l'esempio di un programma che copia il contenuto del file FILE1 DCL sul file FILE2 DCL.

```
BEGIN   CSECT
        BALR    12,0
        USING   *,12
        STATE   SINPUT,ERR1
        SETUP   INPUT
RD      RDBUF   INPUT,ERROR=EOF
        WRBUF   OUTPUT,ERROR=ERR2
        B       RD
EOF     CKEOF   ERR3
        SR      15,15
DONE    BR      14
ERR1    TYPE    'FILE NON TROVATO'
ERRET   LA      15,1
        B       DONE
ERR2    TYPE    'ERRORE DI SCRITTURA'
        B       ERRET
ERR3    TYPE    'ERRORE DI LETTURA'
        B       ERRET
INPUT   FCB     (FILE1,DCL),BUFF1
OUTPUT  FCB     (FILE2,DCL),BUFF1
BUFF1   DS      CL80
        END
```

La lettura dal file avviene con la macro RDBUF; la scrittura con la WRBUF.Entrambe le macro richiedono come primo operando l'indirizzo del File Control Block (FCB) contenente il nome e tipo del file da cui deve avvenire la lettura o scrittura e l'indirizzo dell'area di memoria (BUFF1) su cui leggere la riga del file o da cui prendere la riga da copiare. Ogni chiamata di RDBUF provoca la lettura di una nuova riga del file, mentre WRBUF aggiunge la riga dopo l'ultima del file di scrittura. Il programma eseguendo ad ogni ciclo una RDBUF e una WRBUF, fino a quando non capita un errore in lettura o in scrittura; per un errore in lettura si salta a EOF (come specificando dall'operando ERROR=EOF), per un errore in scrittura si salta a ERR2. In condizioni normali l'uscita dal ciclo avviene solo incontrando l'End-of-File sul file di entrata; per questo saltando a EOF occorre eseguire la macro CKEOF per accertarsi che l'errore in lettura sia stato veramente provocato dalla fine del file. In caso affermativo l'esecuzione procede in sequenza con le

```
SR    15,15
BR    14;
```

altrimenti si salta a ERR3 dove con la macro TYPE si scrive sul terminale ERRORE IN LETTURA. Nel caso di un errore di scrittura si sarebbe senz'altro scritto ERRORE DI SCRITTURA.

Le due macro iniziali STATE e SETUP vanno sempre emesse per i files in lettura e hanno la seguente funzione:
la STATE interpreta il primo operando, come nome di un FCB preceduto da S e controlla l'esistenza del file descritto dall'FCB. Se il file non esiste l'esecuzione passa all'indirizzo indicato dal secondo operando (ERR1 in questo caso), altrimenti la File Status Table del file ricercato viene posta in una particolare locazione di memoria. La SETUP copia quindi nell'FCB del file in lettura vari dati presi dalla File Status Table, come il modo del file, il formato dei records, la loro lunghezza, eccetera.

CKEOF — CONTROLLA SE IL CODICE DI RITORNO DALLA RDBUF CORRISPONDE A UN "END OF FILE". OGNI VALORE DIVERSO E' CONSIDERATO ERRORE.

```
etichetta   CKEOF   errname
```

etichetta — e' normalmente il parametro ERROR della macro RDBUF
errname — specifica una routine che tratta gli errori

NOTA -La macro CKEOF controlla se il valore nel registro 15 e' 12 (eof). Se il valore e' diverso passa il controllo alla routine di errore specificata.

CMSREG — DEFINISCE NOMI SIMBOLICI PER I REGISTRI GENERALI E FLOATING POINT.

```
CMSREG
```

NOTA -A ciascun registro viene associato un nome simbolico (vedi tabella).
Questo permette di usare i nomi simbolici al posto delle designazioni normali.

```
Registri generali          Registri floating point
R0   EQU  0                FR0   EQU  0
R1   EQU  1                FR2   EQU  2
R2   EQU  2                FR4   EQU  4
.............              FR6   EQU  6
R15  EQU 15
```

CMSYSREF — PERMETTE ALL'UTENTE DI SALTARE ALLE ROUTINES DEL NUCLEO CMS CHE NON POSSONO ESSERE CHIAMATE VIA UN SUPERVISOR CALL DEL CMS.

CMSYSREF

NOTA -La macro CMSYSREF genera una serie di istruzioni EQU che forniscono, per ciascun nome di routine del nucleo, il displacement, relativo al punto di entrata SYSREF, dell'indirizzo della routine nel nucleo. Infatti gli indirizzi delle routines del nucleo sono elencati in un modulo del nucleo il cui punto di entrata e' SYSREF.

-Il nome di ciascuna routine del nucleo e' scritto dalla CMSYSREF preceduto da una "D"

-Se un programma usa la macro CMSYSREF deve dichiarare il nome SYSREF come EXTRN.

ERASE — PROVVEDE UN AGGANCIO AL COMANDO ERASE CHE CANCELLA DA UN DISCO IN LETTURA SCRITTURA IL FILE SPECIFICATO.

<etichetta> ERASE fcb

etichetta — etichetta della macro-istruzione
fcb — etichetta della macro FCB che identifica il file che deve essere cancellato

NOTA -La macro ERASE permette di cancellare qualsiasi file, su un disco in lettura e scrittura.

FCB — GENERA UN FILE CONTROL BLOCK CHE SERVE COME LISTA DEI PARAMETRI PER IDENTIFICARE E DESCRIVERE UN FILE SU DISCO

etichetta FCB (filename, filetype), area

etichetta — etichetta della macro-istruzione (fino a 7 caratteri)

filename
filetype — specificano il nome e il tipo del file; si assume il modo *
area — e' il nome del buffer in lettura o scrittura

NOTE -Un File Control Block e' necessario per ciascun file a cui si fa riferimento in un programma.

-Prima di usare la macro FCB per i files in lettura si devono eseguire le macros STATE e SETUP

FINIS — PROVVEDE UN AGGANCIO AL COMANDO FINIS CHE CHIUDERA' IL FILE SPECIFICATO CANCELLANDO LA RELATIVA ENTRATA DALLA TABELLA DEI FILES ATTIVI.

<etichetta> FINIS fcb

etichetta — etichetta della macro-istruzione
fcb — nome della macro FCB che identifica il file da chiudere

RDBUF — LEGGE UN RECORD DA UN FILE SU DISCO

<etichetta> RDBUF fcb,<AREA=anome><,ERROR=enome>

etichetta — etichetta della macro-istruzione

fcb — nome della macro FCB che identifica il file da leggere

anome — nome del buffer di lettura; se omesso viene usato il buffer indicato nella macro FCB

enome — punto di entrata di una routine che tratta gli errori; se omessa, tutti gli errori vengono ignorati (compreso l'end of file)

NOTE -La macro RDBUF legge un record ogni volta che e' eseguita

-Il numero del record letto e' specificato nel campo del blocco FCB alla locazione fcb + 26; se fcb + 26

contiene 0 la lettura avviene sequenzialmente. Se contiene un numero diverso da 0 viene letto quel record, ma il programmatore stesso deve provvedere a incrementare il numero per la lettura dei records successivi.

-I files possono contenere records di lunghezza fissa o variabile, il buffer di entrata deve avere dimensione sufficiente per contenere il record piu' lungo

-Se il nome del buffer di lettura specificato nella macro FCB e' diverso da quello nella macro RDBUF, viene considerato valido il nome specificato nella RDBUF

-La macro RDBUF chiama (via SVC) una routine di lettura, questa gli ritorna il controllo ad esecuzione avvenuta, dopo aver impostato un codice di ritorno nel reg. 15. In caso di errore (codice di ritorno diverso da zero) il controllo va alla routine che tratta gli errori

-Se nella RDBUF non e' stata specificata una routine per trattare gli errori l'esecuzione del programma procede in sequenza, indipendentemente dagli eventuali errori

-Una routine che tratta gli errori dovrebbe cominciare con una macro CKEOF per controllare se si tratta di end-of-file (codice di ritorno uguale a 12)

-La macro RDBUF deve essere preceduta dalle macros STATE e SETUP

SETUP — INIZIALIZZA IL FILE CONTROL BLOCK CON INFORMAZIONI CONCERNENTI LE CARATTERISTICHE DEL FILE.

etichetta SETUP fcb

etichetta — etichetta della macro-istruzione

fcb — nome del FCB da inizializzare

NOTA -La macro SETUP deve seguire la STATE ma non deve essere eseguita se quest'ultima ha dato un codice di errore nel registro 15.

- Se il file control block non viene inizializzato con la SETUP si suppone che il file abbia records fissi di 80 caratteri.

**STATE** PROVVEDE UN AGGANCIO AD UNA ROUTINE DEL CMS CHE CONTROLLA L'ESISTENZA ED EVENTUALMENTE FORNISCE L'INDIRIZZO DEL FILE SPECIFICATO.

<etichetta> STATE Sfcb, ename

| | |
|---|---|
| etichetta | etichetta della macro-istruzione |
| Sfcb | nome della macro FCB a cui si fa riferimento, preceduta da una "S" |
| ename | etichetta di una routine di errore |

NOTE -La STATE pone nel File Control Block l'indirizzo della FST del file specificato;questa informazione viene poi utilizzata dalla macro SETUP per inizializzare il File Control Block.

-Se il file specificato non viene trovato nello User's File Directory, la STATE passa il controllo alla routine di errore indicata.

**TYPE** PERMETTE LA STAMPA SUL TERMINALE DI UNA LINEA SPECIFICATA.

```
                  'messaggio'
etichetta  TYPE
                                (r)
                    msgnome     ,n
                                 l
```

| | |
|---|---|
| etichetta | etichetta della macro-istruzione |
| 'messaggio' | messaggio da stampare (tra apici) |
| msgnome | etichetta dell'area contenente il messaggio da stampare |
| (r) | registro contenente la lunghezza dell'area che contiene il messaggio |
| n | lunghezza, come self-defining term, dell'area che contiene il messaggio |
| l | etichetta di una voce, o mezza voce, di memoria contenente la lunghezza dell'area che contiene il messaggio |

NOTE -La lunghezza massima del messaggio e' 130 caratteri.

-Se il messaggio da stampare viene specificato come parametro (tra apici), il parametro che esprime la lunghezza puo' essere omesso.

TYPIN PERMETTE DI LEGGERE UNA LINEA DA TERMINALE.

<etichetta> TYPIN area<,lung><,ED=c>

| | |
|---|---|
| etichetta | etichetta delle macro-istruzione |
| area | etichetta di un buffer d'entrata di 130 caratteri |
| lung | etichetta assegnata ad un campo di tre bytes della lista di parametri, dove viene posto il numero di caratteri letti |
| c | e' un carattere che specifica le funzioni di edit che il CMS deve eseguire sulla linea di entrata:<br>U interpreta i caratteri di cancellazione, converte in maiuscolo, completa la linea con blanks fino a 130 caratteri<br>V interpreta i caratteri di cancellazione e converte in maiuscolo<br>S interpreta i caratteri di cancellazione e completa la linea con blanks fino a 130 caratteri,<br>T interpreta i caratteri di cancellazione,<br>X non esegue alcuna funzione di edit sulla linea. |

NOTA - Il parametro 'lung', se specificato, permette di conoscere il numero di caratteri effettivamente letti, indipendentemente dall'eventuale completamento della linea con caratteri bianchi.

WRBUF PERMETTE DI SCRIVERE RECORDS SU DISCO.

<etichetta> WRBUF fcb<,AREA=anome><,ERROR=enome>

| | |
|---|---|
| etichetta | etichetta della macro-istruzione |

| | |
|---|---|
| fcb | etichetta della macro FCB che descrive il file su cui scrivere |
| anome | etichetta del buffer di uscita |
| enome | etichetta di una routine che tratta gli errori |

NOTE -La macro WRBUF viene normalmente usata per creare un nuovo file sequenziale o per aggiungere record(s) alla fine di un file esistente; tuttavia e' possibile rimpiazzare qualsiasi record di un file con records a lunghezza fissa, cambiando opportunamente il contenuto del campo "numero di record" alla locazione di memoria fcb+26.

-Se il parametro AREA= e' omesso viene considerato valido il parametro specificato nella macro FCB; se e' specificato in entrambe le macros e le due etichette sono diverse, vale quello specificato nella WRBUF.

-Se la macro WRBUF viene eseguita senza errori nel registro 15 viene impostato il codice di ritorno zero, altrimenti tale registro contiene un codice indicante la natura dell'errore; se nella macro e' specificata una routine di errore, il controllo e' passato ad essa, altrimenti gli errori vengono ignorati.

## PROGRAMMAZIONE FORTRAN

Il compilatore Fortran del CMS e' identico a quello usato dall'OS, per cui sono valide le specifiche del linguaggio riportate nei manuali:

IBM System/360 FORTRAN IV LANGUAGE C28-6515
IBM System/360 FORTRAN IV Library C28-6810

Il CMS assegna ai files utilizzati nei programmi FORTRAN delle caratteristiche ben precise, diverse secondo che l'accesso sia sequenziale o diretto.

A) INGRESSO/USCITA SEQUENZIALE

In questo caso tutti i files sono individuati come:
FILE FTxxFyyy
dove xx (limitato tra 01 e 12) e' il numero dell'unita' logica di ingresso/uscita, e yyy e' il numero di sequenza (a partire da 001) necessario ad identificare files multipli con lo stesso numero di unita' logica.

I numeri logici 1,2,3,4,7 e 10 possono essere referenziati nelle istruzioni READ, WRITE, END FILE e REWIND; i files corrispondenti hanno records di 80 caratteri. I numeri logici 8 e 9 si usano con le stesse istruzioni FORTRAN ma i files corrispondenti hanno records rispettivamente di 133 (il primo carattere e' di controllo della stampante e va stampato col comando OFFLINE PRINTCC) e 140 caratteri.
Il numero logico 5 puo' essere usato in scrittura sul FILE FT05Fyyy con records di 80 caratteri e in lettura per leggere dati (fino a 80 caratteri) da terminale. La scrittura di dati sul terminale si effettua col numero logico 6, con records di fino a 120 caratteri.
L'istruzione PUNCH scrive sul FILE FT07Fyyy su disco; la perforazione attuale va richiesta poi col comando OFFLINE PUNCH.
I numeri logici 11 e 12 indirizzano le unita' nastro di indirizzo 180 e 181 rispettivamente e sono usati con le istruzioni READ, WRITE, END FILE, REWIND e BACKSPACE. I records sono di 80 caratteri per il numero 11 e di 133 per il numero 12.

Il numero di sequenza e' sempre 001, a meno che non si chiuda un file, coll'istruzione END FILE n, dove n e' il numero dell'unita' logica; ogni successiva istruzione di lettura o scrittura su quel numero logico fara' riferimento ad un file con lo stesso numero di unita' logica ma con numero di sequenza aumentato di uno. L'istruzione REWIND n

fa in modo che la successiva lettura scrittura al numero logico n siano di nuovo sul FILE FTnF001.

Riassumendo le caratteristiche dei files FORTRAN sequenziali abbiamo:

| Numero di unita' logica | Tipo del file | Caratteri per record |
|---|---|---|
| 1 | FT01Fyyy | 80 |
| 2 | FT02Fyyy | 80 |
| 3 | FT03Fyyy | 80 |
| 4 | FT04Fyyy | 80 |
| 5 | FT05Fyyy | 80 |
| 5 | lettura da console | 80 |
| 6 | scrittura su console | 120 |
| 7 | FT07Fyyy | 80 |
| 8 | FT08Fyyy | 133 |
| 9 | FT09Fyyy | 140 |
| 10 | FT10Fyyy | 80 |
| 11 | unita' nastro 180 | 80 |
| 12 | unita' nastro 181 | 133 |

B) INGRESSO/USCITA AD ACCESSO DIRETTO

In questo caso i files sono individuati come
FILE DAxx
dove xx e' il numero dell'unita' logica (limitato tra 01 e 08). L'accesso diretto funziona solo per i files definiti con l'istruzione fortran DEFINE FILE, nella quale si stabilisce tra l'altro la lunghezza dei records e il numero di records che il file avra'. Quando il file viene creato, il numero di records specificato viene allocato su disco. Tutti i files ad accesso diretto sono su disco.
In uno stesso programma un numero di unita' logico se usato con files ad accesso diretto non puo' essere usato per files sequenziali. Comunque si possono usare in uno stesso programma sia files sequenziali che files ad accesso diretto. Uno stesso file su disco, purche' venga alterato convenientemente il nome, puo' essere usato in programmi diversi sequenzialmente o direttamente.

La libreria di sistema SYSLIB TXTLIB, sempre attiva se non viene emesso un comando di GLOBAL T che non la specifichi, comprende, oltre ai sottoprogrammi della FORTRAN LIBRARY dell'OS, anche un insieme di sottoprogrammi (richiamabili con l'istruzione CALL) propri del CMS di cui viene data una descrizione qui di seguito.

BLIP — FA STAMPARE A TERMINALE UNA SEQUENZA (DA 1 A 8 CARATTERI) OGNI DUE SECONDI DI TEMPO DI CPU, PERMETTENDO COSI' UN CONTROLLO VISIVO DEL TEMPO DI CPU CONSUMATO.

CALL BLIP ('sequenza' <,numero>)

| | |
|---|---|
| sequenza | insieme dei caratteri scelti |
| numero | numero (intero) dei caratteri componenti la sequenza. Se omesso viene considerato uguale a 1. |

Scrivendo CALL BLIP (0) si ripristina la sequenza default, mentre CALL BLIP (OFF) fa cessare la stampa di qualunque carattere, anche "non stampante".

CPFUNC — PERMETTE DI ESEGUIRE FUNZIONI DI CP DURANTE L'ESECUZIONE DI UN PROGRAMMA.

CALL CPFUNC (I,J,K)

| | |
|---|---|
| I | funzione di CP richiesta che deve essere passata come stringa alfanumerica compresa tra apici |
| J | lunghezza in caratteri della stringa alfanumerica |
| K | codice errore di ritorno, che deve essere una voce intera definita nel programma chiamante. I valori del codice sono 0 per nessun errore, 1 per errore. |

Esempio:
CALL CPFUNC ('MSG CP MONTARE NASTRO XXY COME 180',34,K)
l'esecuzione di questa subroutine provoca l'invio del messaggio all'operatore.

CLOCK — PRELEVA IL CONTENUTO DELL'OROLOGIO DELLA MACCHINA VIRTUALE.

CALL CLOCK (N)

N — voce intera in cui viene scritto in sessantesimi di secondo il contenuto dell'orologio.

DATE — PRELEVA DAL CMS LA DATA DEL GIORNO.

CALL DATE (NN)

NN — doppia voce nella quale la data viene scritta come MM/GG/AA

DEFINE — CREA UNA CORRISPONDENZA TRA UN NUMERO DI UNITA' LOGICO E UN FILE CMS E PERMETTE UN ACCESSO DIRETTO A TALE FILE.

CALL DEFINE (nul, nome, tipo, recno, recsiz)

nul — numero di unita' logica

nome tipo — nome e tipo del file CMS, devono essere 8 caratteri, anche con bianchi in coda, fra apici.

recno — variabile intera che indica quale record del file deve essere letto o scritto

recsiz — lunghezza del record. Tutti i records devono avere la stessa lunghezza. Se un file gia' esiste la lunghezza di record ora specificata sostituisce quella preesistente. Il recsiz non puo' superare 256.

NOTE - In questo caso le istruzioni di I/O del Fortran fanno riferimento al file di nome e tipo specificati e non al FILE FTxxFyyy. Dopo l'esecuzione di un READ o WRITE il numero di record viene automaticamente incrementato di uno per cui e' possibile anche l'I/O sequenziale purche' a recno si dia un valore intero prima della prima operazione di READ o WRITE. Se il valore assegnato a recno e' uno sara' possibile operare dal primo record del file. Assegnando a recno valori diversi sara' possibile operare ad accesso diretto.
Se nell'istruzione FORMAT si specificano piu' records da leggere o scrivere il numero di record verra' incrementato corrispondentemente, mentre per le istruzioni di READ o WRITE unformatted ogni operazione incrementa recno di uno.
Una seconda chiamata a DEFINE con lo stesso numero di unita' logica ma con riferimento ad un file CMS diverso

svincola il file prima definito da quel numero di unita' logica.
Si possono definire fino a 20 files, con i numeri logici da 1 a 99 esclusi 5,6 e 7.

DSDSET — PERMETTE DI CAMBIARE LA LUNGHEZZA DEL RECORD LOGICO E IL FORMATO DI FILES FORTRAN SU DISCO.

CALL DSDSET (nul,blksiz,recm,lrecl)

| | |
|---|---|
| nul | numero di unita' logica del file |
| blksiz | lunghezza in bytes del record fisico |
| recfm | numero da 1 a 5 specificante il formato del record secondo la tabella:<br>1 lunghezza fissa, non bloccato<br>2 lunghezza fissa, bloccato<br>3 lunghezza variabile, non bloccato<br>4 lunghezza variabile, bloccato<br>5 indefinito, senza bloccaggio |
| lrecl | lunghezza in bytes del record logico, usato con recfm da 1 a 4; per recfm = 2 lrecl deve essere sottomultiplo di blksiz. |

NOTE - Questo sottoprogramma permette di cambiare le caratteristiche default dei files con numero logico da 1 a 12 eccetto il 5, 6 e 7.

ERASE — PERMETTE DI CANCELLARE UN FILE DA UN DISCO.

CALL ERASE (nome,tipo <,modo>)

| | |
|---|---|
| nome,tipo,modo | identificatori del file da cancellare; il nome e tipo devono essere stringhe di otto caratteri, anche bianchi in coda, fra apici. |

GETPAR — PERMETTE DI RECUPERARE I PARAMETRI SPECIFICATI COL COMANDO DI CHIAMATA DEL PROGRAMMA.

CALL GETPAR (nome,numero<,'RITE'><,&fine>)

| | |
|---|---|
| nome | variabile (real*8) in cui andra' il parametro |
| numero | numero d'ordine del parametro (integer*4) nella lista dei parametri. Specificando zero si ottiene il nome del programma. Se |

il numero e' superiore a quello dei parametri passati il controllo va all'indirizzo specificato in &fine, se presente, altrimenti va all'istruzione successiva; non viene comunque passato alcun parametro.

'RITE' fa in modo che il parametro sia allineato a destra nel campo di otto caratteri fornito.

&fine fine e' l'etichetta di un'istruzione a cui passare il controllo se 'numero' e' superiore a quello dei parametri presenti.

HOUR PRELEVA DAL CMS L'ORA.

CALL HOUR (NN)

NN doppia voce nella quale l'ora viene scritta nelle forme HH.MM.SS

LOGDSK CHIUDE TUTTI I FILES APERTI.

CALL LOGDSK

Questo sottoprogramma permette di scrivere permanentemente su disco tutti i files usati nel programma.

NLSTON/NLTOF PERMETTE L'INGRESSO DI VARIABILI IN UN PROGRAMMA IN UN FORMATO LIBERO SENZA SPECIFICARE IL NOME DELLE VARIABILI COME RICHIESTO DALL'ISTRUZIONE NAMELIST.

CALL NLSTON attiva questo modo di operare
CALL NLSTOF disattiva tale modo

Dopo il richiamo a NLSTON e' possibile specificare le variabili nello stesso ordine indicato in NAMELIST ciascuna separata da una virgola senza la specifica del nome della variabile.

OPRINT PERMETTE DI STAMPARE FILES FORTRAN E DI CANCELLARLI DA DISCO.

CALL OPRINT (N,K<,M>)

N numero di unita' logica (tra 1 e 9) che individua il FILE FTONFyyy
K voce in cui andra' il codice di ritorno coi valori:
0 - nessun errore
1 - numero del file illegale
2 - tipo funzione illegale
4 - file non esistente
5 - errore su disco (anche per file non chiuso)
6 - richiesta illegale
M Se vale 1 il file sara' stampato con OFFLINE PRINT se vale 2 con OFFLINE PRINTCC. Quest'ultima e' l'opzione default.

NOTE - Questo sottoprogramma e' utile quando si voglia creare files di grosse dimensioni e si disponga di poco spazio su disco.

- Prima di ogni richiamo alla routine il file interessato deve essere chiuso con l'istruzione fortran END FILE.

- Per avere su un'unica uscita su stampante tutti i files inviati in stampa occorre emettere il comando

CLOSIO PRINTER OFF prima dell'esecuzione del programma e CLOSIO PRINTER ON alla fine dell'esecuzione.

- La routine puo' essere richiamata fino a 999 volte per ogni file; il numero logico di sequenza sara' incrementato dopo ogni END FILE.

PLOTTER — PERMETTE LA CREAZIONE DI FILES SU NASTRO DESTINATI AL PLOTTER FUORI LINEA.

Per una descrizione dettagliata delle routines richiamabili si rimanda alla pubblicazione CNUCE sul PLOTTER.

RENAME — PERMETTE DI CAMBIARE L'IDENTIFICATORE DI UN FILE.

CALL RENAME (vnome,vtipo,nnome,ntipo)

vnome,vtipo — identificatore del file originale
nnome,ntipo — nuovo identificatore assegnato al file

Tutti i nomi e tipi indicati devono essere stringhe di otto caratteri, eventualmente con bianchi in coda, fra apici.

REREAD — PERMETTE LA RILETTURA IN MEMORIA CON FORMATO DIVERSO DI UN RECORD.

CALL REREAD (nu1 <,blksiz>)

nu1 — numero di unita' logica da usare per le opzioni di rilettura (da 1 a 99 escluso 5,6 e 7)

blksiz — bloccaggio del record da rileggere. Se omesso viene preso 140 bytes.

NOTE - La rilettura sulla stessa unita' puo' avvenire quante volte si voglia purche' dopo ogni lettura si esegua una istruzione di REWIND su quell'unita'.

TAPFUN — PERMETTE ALCUNE OPERAZIONI DI CONTROLLO SU UN'UNITA' NASTRO.

CALL TAPFUN (I,J,L<,M>)

I — 1 corrisponde all'unita' 180, 2 all'unita' 181
J — Codice della funzione richiesta:
1 BACKSPACE FILE

2 BACKSPACE RECORD
3 SKIP FILE
4 SKIP RECORD
5 REWIND UNLOAD

L Codici di errore al ritorno. Deve essere una voce intera
0 - nessun errore
1 - funzione invalida
2 - end-of-file
3 - errore su nastro
4 - numero di nastro invalido
5 - unita' nastro non attaccata

M fattore di ripetizione della funzione (preso uguale ad uno in caso di omissione)

TAPSET — PERMETTE DI CAMBIARE LE DEFINIZIONI DEFAULT ASSOCIATE ALLE UNITA' LOGICHE FORTRAN DEI NASTRI.

CALL TAPSET (nastro,nul<,blksiz><,modo><,recfun><,lrecl>)

nastro — 1 corrisponde all'unita' 180, 2 all'unita' 181

nul — numero di unita' logica usato per riferirsi a questa unita' nastro (da 1 a 12 eccettuato 5,6 e 7)

blksiz — lunghezza massima dei records fisici da leggere o scrivere

modo — 0 - nessun cambiamento
16 - 9 tracce, 1600 bpi
17 - 9 tracce, 800 bpi

recfun — numero da 1 a 5 specificante il formato del record
1 - fisso, non bloccato
2 - fisso, bloccato
3 - variabile, non bloccato
4 - variabile, bloccato
5 - undefined, senza bloccaggio

lrecl — lunghezza del record logico (solo per recfun 1-4) Per recfun 2 deve essere sottomultiplo di blksiz.

# PROGRAMMAZIONE PL/1 IN CMS

Caricamento ed esecuzione di programmi PL/1.

Compilato il programma PL/1 e caricate in memoria le librerie necessarie, si puo' passare alla fase di caricamento ed esecuzione, secondo una delle tre tecniche:

1) LOAD prog (XEQ)

2) LOAD prog

3) LOAD prog GENMOD prog prog

Per mezzo del comando START e' possibile richiamare la routine di inizializzazione IXECMS, che trasferisce il controllo al programma principale, e che permette di passare una sequenza di caratteri (MAX 8) come parametro al programma principale PL/1.

Si consideri, ad esempio, il programma:

```
ES1: PROC (PAROLE) OPTIONS (MAIN);
     DCL PAROLE CHAR (100) VARYING;
     DISPLAY (PAROLE);
     END ES1;
```

Lo schema di compilazione, caricamento ed esecuzione e' il seguente:

```
PLI ES1
LOAD ES1
START IXECMS ABCDEFGH
```

A seguito dell'istruzione DISPLAY si ottiene la stampa della sequenza

```
ABCDEFGH
```

Ingresso / uscita di dati tramite files

Le istruzioni di ingresso / uscita del PL/1 GET/PUT e

READ/WRITE fanno sempre riferimento a files del CMS di nome FILE e di tipo corrispondente al nome del file usato nel programma. Ad esempio le istruzioni di ingresso:

```
GET FILE (SISIN) DATA; GET FILE (IN) LIST (A,B);

READ FILE (DATI) INTO (A);
```

leggono dati rispettivamente dai files FILE SYSIN, FILE IN, FILE DATI, mentre le istruzioni di uscita:

```
PUT FILE (SYSPRINT) EDIT (A,B,C); WRITE FILE (RIS) FROM
(RISULT);
```

scrivono dati rispettivamente nei files SYSPRINT e RIS. Ad eccezione del file SYSPRINT, se un file di uscita gia' esiste gia' il suo contenuto precedente non viene automaticamente distrutto, cioe' il nuovo contenuto puo' essere scritto di seguito o puo' essere sostituito al precedente a seconda del valore del puntatore di WRITE.
Di tutti i files usati devono essere specificate le dimensioni dei records e dei blocchi, in uno dei due modi seguenti:

1) tramite l'istruzione DECLARE inserita nel programma PL/1

2) tramite il comando CMS FILEDEF.

Quest'ultima soluzione evita di dover ricompilare il programma PL/1 nel caso in cui si debba solo modificare la descrizione dei files. Ad esempio:

```
DECLARE IN FILE STREAM INPUT ENVIRONMENT (F(80));
```

oppure:

```
FILEDEF OUT DSK FILE OUTPUT RECFM F LRECL 100
```

Solo il file SYSPRINT ha una dichiarazione per default che e':

```
DECLARE SYSPRINT FILE STREAM PRINT ENVIRONMENT (V(129));
```

Pertanto non e' necessario ridefinirlo, a meno che non si voglia cambiare definizione.
Le istruzioni GET/PUT devono fare riferimento a files di tipo STREAM. Non vi sono limitazioni agli attributi del file, ne' alle opzioni dell'istruzione di ingresso / uscita.
Le istruzioni READ/WRITE devono fare riferimento a files di tipo RECORD SEQUENTIAL. Non e' possibile usare l'attributo di accesso diretto. E' opportuno, anche se non strettamente

richiesto, che tutti i files usati nel programma vengano aperti esplicitamente all'inizio con un'unica istruzione OPEN. Esempio:

```
OPEN FILE (SYSIN), FILE (DATI), FILE (RIS);
```

Ingresso / uscita dei dati tramite terminale.

L'istruzione DISPLAY del PL/1 permette la stampa a terminale di sequenza di caratteri. L'opzione REPLY permette la lettura di caratteri battuti a terminale. Esempio:

```
DISPLAY ('INIZIO PASSO' // I) REPLY (A);
```

Le variabili I ed A dovranno essere state dichiarate CHARACTER.
Se si vuole ottenere stampa o lettura di dati numerici si puo' seguire una delle due vie:

1) fare ricorso alle routines di conversione del PL/1.
   Ad esempio:

```
DECLARE VALORE CHAR (3), N FIXED (5);
DISPLAY ('N=?') REPLY (VALORE); N=VALORE;
```

2) fare ricorso all'opzione STRING delle istruzioni GET/PUT.
   Ad esempio:

```
DECLARE (IN, OUT) CHAR (80) VARYING;
PUT STRING (OUT) EDIT ('A=',A,'N=?') (A,F(6,3),A);
DISPLAY (OUT) REPLY (IN);
GET STRING (IN) EDIT (N) (F(3));
```

E' poi possibile ottenere direttamente la stampa del file SYSPRINT o la lettura del file SYSIN direttamente da terminale in fase di esecuzione usando il comando del CMS FILEDEF prima di caricare il programma PL/1 prog nel modo seguente:

```
FILEDEF SYSPRINT con

LOAD prog

START
```

Si noti pero' che il file SYSPRINT passa attraverso un buffer, e quindi la stampa a terminale non si avra' se non quando il buffer e' pieno o non viene eseguita un'opzione SKIP.

Per ogni informazione sul linguaggio PL/1 si rimanda ai manuali

O.Menchi Manuali di PL/1 CNUCE 27 e 28

PL/1 (F) Language Reference Manual
IBM GC28-8201

PL/1 (F) Programmer's Guide
IBM GC28-6594

PSWRESTART — SIMULA L'OPERAZIONE DI PSWRESTART DI UNA MACCHINA REALE.

PSWRESTART

PURGE — PERMETTE ALL'UTENTE DI CANCELLARE TUTTI I FILES CHE SI TROVANO NELLE AREE DI SPOOLING DELLE PROPRIE UNITA' VIRTUALI DI INPUT O DI OUTPUT.

PURGE
P

READER
R
PRINTER
P
PUNCH
PU

ESEMPI:
1) P R

risposta del CP
04 FILES PURGED
Quattro files presenti sul lettore virtuale vengono cancellati.

**QUERY** — STAMPA A TERMINALE INFORMAZIONI SULLO STATO DEL SISTEMA E DELLA MACCHINA VIRTUALE.

```
QUERY
Q
        FILES
        LOGMSG
        NAMES
        USER
        userid
        RESIDUAL
        TIME
        VIRTUAL <ccu>
                CORE
```

FILES
F — stampa il numero dei files presenti nelle varie unita' di spooling della macchina virtuale.

LOGMSG
L — stampa il messaggio del giorno.

NAMES
N — stampa il nome e l'indirizzo di linea delle macchine virtuali connesse al sistema.

USER
U — stampa il numero degli utenti connessi al sistema.

userid — comunica nome e indirizzo, se l'utente e' attivo.

TIME
T — da' il tempo virtuale, totale e di connesssione consumato dall'indizio della seduta.

RESIDUAL
R — solo per le macchine virtuali con limitazioni sul tempo di CPU consumabile, da' il tempo di CPU ancora disponibile.

VIRTUAL
V — da' la configurazione della macchina virtuale, se non vi sono parametri. Se e' specificato CCU da' informazioni sull'unita' di indirizzo CCU, se CORE da' la taglia di memoria della macchina virtuale.

**RDCMS** — PERMETTE DI RECUPERARE FINO AD UN MASSIMO DI 10 FILES, DI CUI SIANO NOTI IL NOME E IL TIPO, QUANDO NON SIANO PIU' PRESENTI NEL DIRECTORY DEL LORO DISCO (AD ESEMPIO, DOPO IL COMANDO ERASE).

Uso: prima di emettere il comando occorre essersi procurati un disco di indirizzo virtuale 193 ed averne fatto LOGIN come disco P:

LOGIN 193 P

Se non interessa conservare il contenuto del disco 193 e' consigliabile dare il comando:

LOGIN 193 P (NO-UFD)

Il disco dal quale si vogliono recuperare i files deve avere indirizzo 191.
Dato il comando RDCMS l'utente ottiene il messaggio:

FILENAME FILETYPE PLEASE:

La risposta puo' essere il nome e tipo del file desiderato o una linea vuota. RDCMS ripete la domanda fino a quando l'utente non da' una riga vuota e comunque non oltre dieci volte. Una volta che l'utente ha specificato tutti i files di cui desidera il recupero, RDCMS cerca di copiarli dal disco 191 al disco 193. Nel corso dell'esecuzione possono essere stampati i seguenti messaggi:

CCHHR OUT OF DISK. Da ignorare

'FILENAME' 'FILETYPE' P1 COPIED, Segnala la fine del recupero di un file.

'FILENAME' 'FILETYPE' NOT FOUND. Il file richiesto non e' stato trovato.

'FILENAME' 'FILETYPE' CANNOT BE READ. Non e' stato possibile il recupero dell'intero file.

ERROR DURING WRITE ON P DISK. Errore sul disco 193. Puo' essere necessario riformattarlo.

ERROR DURING LOGIN BIDON. Il contenuto del disco 191 deve considerarsi perduto. Puo' essere necessario riformattarlo.

ERROR DURING RELEASE BIDON. Non dovrebbe verificarsi; in ogni caso non implica il fallimento del recupero.

READTAP LEGGE NASTRI OS STANDARD LABEL O NOLABEL DI FORMATO FISSO E BLOCCAGGIO FISSO O VARIABILE E CREA DEI FILES CMS SU DISCO.

READTAP

Utilizzo:

* READTAP legge da nastro attaccato come 181 (tap2).

* Il nastro e' riavvolto alla fine del comando.

* Il comando chiede all'utente:

1/ LABEL=

la risposta e': SL o NL

caso SL (standard label)

Il comando prosegue dal numero 3.

caso NL (no label).

Il comando effettua le operazioni del numero 2.

2/

2-1 SKIP=

L'utente precisa il numero di EOF cioe' i files da saltare (da 0 a 9) per posizionarsi sul file cercato.

2-2 BLKSIZE=XXXXX:

La risposta e' un numero di cinque cifre che indica la lunghezza massima in caratteri del record fisico; e' obbligatoriamente un multiplo di LRECL.

2-3 LRECL=XXXXX:

La risposta e' un numero di cinque cifre specificante la lunghezza del record logico.

Il comando prosegue al numero 4.

3/

3-1 VOL=

La risposta e' una stringa di sei caratteri identificanti

Il nastro.

3-2 DSNAME=

E' il nome, su 17 caratteri al piu', del file da leggere.

Il comando prosegue al numero seguente.

4/ PDS=

La risposta e': YES o NO

4-1 CASO YES:

Il file fisico e' considerato come piu' files logici. Ciascun file logico e' preceduto da un record logico contenente il nome del membro di questo PDS.

4-1-1 Il comando domanda:

FILETYPE=

la risposta e' il tipo del file CMS da creare (il nome e' quello del membro).

4-1-2 L'utente e' avvisato che il file e' in corso di copiatura dal messaggio :

FILE BEING COPIED: XXX---XXX

e questo messaggio apparira' per ogni membro del PDS. Alla fine del file il comando continua al numero 5.

4-2 CASO NO: Il file da copiare e' un file sequenziale.

4-2-1 Il comando domanda:

FILENAME=

La risposta e' il nome del file CMS da creare.

FILETYPE=

La risposta e' il tipo di questo file.

4-2-2 Alla fine del file il comando continua al numero seguente.

5/ La fine copiatura e' segnalata all'utente dal messaggio:

*EOF*

Quello che fara' in seguito il comando dipende dalla risposta a LABEL=(numero 1).

Se SL, il comando riprende al numero 3-2 se NL al numero 2-1.

NOTE - Quando la risposta a una qualsiasi delle domande e' una riga vuota (RC) il comando termina immediatamente, generalmente su DSNAME= o su SKIP=

- Questo comando permette la lettura di un file sequenziale e di un PDS uno di seguito all'altro.

Errori

Errori segnalati:

a/ INVALID MOUNTING: VOL=XXXXXX
DO YOU WANT TO CONTINUE.REPLY YES OR NO:

Il numero e' quello del nastro realmente montato.
Questo messaggio appare dopo una risposta sbagliata alla domanda VOL=,oppure dopo un errore di montaggio del nastro.

b/ FILE ALREADY EXISTS.REPLY ERASE OR KEEP:

questo messaggio appare dopo le risposte a

FILENAME=
FILETYPE=

per informare l'utente che il file CMS esiste gia'.
Se la risposta e' ERASE, il vecchio file e' cancellato e il lavoro continua.
Se la risposta e' KEEP il lavoro riprende al numero 4-2-1.

c/ DSNAME NOT FOUND ON TAPE

Il nome dato in risposta a DSNAME= non e' stato trovato sulla parte del nastro che seguiva l'ultimo file trattato; il nastro e' riavvolto e il lavoro riprende al numero 3-2.

d/ TRAILER LABEL ERROR

Il comando finisce. Questo errore sopraggiunge se un nastro SL e' stato scritto male.

e/ BLOCK LENGTH IS NOT LRECL MULTIPLE

Il risultato della divisione della lunghezza del record fisico per la lunghezza del record logico non e' un numero intero. Il comando finisce.

f/ FILE filename filetype ALREADY EXISTS: REPLY ERASE OR QUIT

Questo messaggio appare nel caso di un file PDS; il file CMS da creare (filename=nome del membro) esiste gia'.
Se la risposta e' ERASE questo file e' cancellato e il lavoro continua.
Se la risposta e' QUIT il comando finisce.

g/ "V" FILES NOT SUPPORTED

I files di formato variabile non sono trattati; il comando finisce.

h/ DISK ERROR

Errori di scrittura sul P-disk; il comando finisce.

i/ TAPE ERROR

Errore di lettura su nastro; il comando finisce.

READY — SIMULA UN 'DEVICE END' PER L'UNITA' VIRTUALE SPECIFICATA.

READY Indirizzo
R

Indirizzo — Indirizzo dell'unita' virtuale della forma ccu.

RELEASE — DISATTIVA UN DISCO QUANDO L'UTENTE NON NE HA PIU' BISOGNO.

RELEASE ccu modo <(DETACH)>

ccu — e' l'indirizzo virtuale del disco
modo — e' il modo del disco
(DETACH) — indica che il disco disattivato deve essere anche staccato dalla configurazione della macchina virtuale.

ESEMPI:
1) RELEASE 193 A (DETACH)
Il disco A di indirizzo 193 viene disattivato e staccato dalla configurazione della macchina virtuale.

RESET — SIMULA IL TASTO RESET DELLA CONSOLE.

RESET
RES

NOTA - Il sistema e' resettato automaticamente dal comando IPL.

| | |
|---|---|
| REUSE | CARICA IN MEMORIA DA DISCO I FILES DI TIPO TEXT SPECIFICATI (CONTENENTI CODICE RILOCABILE) STABILENDO AGGANCI CON FILES CARICATI IN PRECEDENZA E CAMBIANDO IL PUNTO DI ENTRATA DEFAULT DI QUESTI FILES IN QUELLO DEL PRIMO FILE SPECIFICATO NEL COMANDO REUSE. RIFERIMENTI NON RISOLTI VENGONO RICERCATI IN LIBRERIE TXTLIB EVENTUALMENTE SPECIFICATE. |

REUSE fn1...fnN <(opz1...opzN)<lbn1...lbnN>>

| | |
|---|---|
| fn1...fnN | specifica i nomi dei files TEXT da caricare in memoria |
| opz1...opzN | specifica le opzioni che saranno in effetto durante il caricamento |
| lbn1...lbnN | specifica i nomi di fino a 8 librerie TXTLIB in cui cercare le routines mancanti durante il caricamento. |

NOTE - Le opzioni sono quelle usate dal comando LOAD.

- REUSE carica i files specificati in posizioni di memoria crescenti a partire dal punto in cui precedenti comandi LOAD, USE e REUSE hanno terminato il caricamento.

- Le opzioni specificate da precedenti comandi LOAD, USE, REUSE rimangono valide se non vengono rispecificate.

- REUSE ha la stessa funzione di USE, salvo che cambia il punto di entrata default.

ESEMPI:

1) REUSE MANCA FINE
I files MANCA TEXT e FINE TEXT sono caricati in memoria dopo l'ultimo file caricato in precedenza e il punto d'entrata e' cambiato nel primo punto d'entrata in MANCA.

RT — RIATTIVA LA STAMPA A TERMINALE CHE ERA STATA SOPPRESSA DA UN COMANDO KT PRECEDENTE.

RT

NOTA - Il comando deve essere emesso dopo aver pigiato due volte il tasto interruzione per fare accettare il comando dal CMS. (comando immediato).

- Il comando agisce solo sul programma in esecuzione al momento della sua emissione.

SALVA — COPIA SUL FILE1 IL FILE2 PURCHE' ABBIA RECORDS LOGICI DI 80 CARATTERI USANDO UN BUFFER DI LETTURA DI nnnn SCHEDE.

SALVA fn1 ft1 fm1 fn2 ft2 fm2 <nnnn>

NOTE - Per ogni file va specificato nome, tipo, e modo.

- I due files devono avere records logici di 80 caratteri.

- Se FILE1 esiste, FILE2 sara' aggiunto alla fine, altrimenti FILE1 verra' creato.

- Se nnnn non e' specificato, viene usato un buffer di 50 schede (circa una pagina).

- Questo comando risulta piu' veloce del COMBINE nel copiare files su uno stesso disco o su dischi diversi.

ESEMPI:
1) SALVA A TEXT P1 A TEXT A1
Il file A TEXT e' copiato dal disco A al disco P.

SAVCMS — PERMETTE DI RECUPERARE UN NUMERO SPECIFICATO DI FILES QUANDO NON SONO PIU' PRESENTI NEL DIRECTORY DEL LORO DISCO (AD ESEMPIO, DOPO IL COMANDO ERASE).

Uso: Prima di emettere il comando occorre essersi procurati un disco di indirizzo virtuale 193 e averne fatto login come disco P:

LOGIN 193 P

Se non interessa conservare il contenuto del disco 193 e' consigliabile dare il comando:

LOGIN 193 P (NO-UFD)

Il disco dal quale si vogliono recuperare i files deve avere indirizzo 191. Dato il comando SAVCMS l'utente ottiene il messaggio:

GIVE MAX OF FILES IN DECIMAL YOU WISH COPIED:

La risposta puo' essere un numero decimale o 'ALL' (se si vuole che tutti i files siano copiati). SAVCMS cerca quindi di copiare sul disco 193 tutti i files che trova sul 191, fino a quando il numero specificato e' stato raggiunto.
Per il significato dei messaggi dati da SAVCMS, vedere il comando RDCMS.

SCRIPT — PERMETTE DI STAMPARE AL TERMINALE O SULLA STAMPANTE UN FILE CON RECORDS A LUNGHEZZA VARIABILE, NEL FORMATO SPECIFICATO DA PAROLE DI CONTROLLO INCLUSE NEL FILE STESSO.

SCRIPT filename (opzione1 ... opzioneN)
SC

Lo SCRIPT permette, mediante l'inserimento di parole di controllo, di regolare la stampa di un testo battuto senza preoccupazione di lunghezza di riga o impaginazione; piu' precisamente permette, per esempio, di scegliere la lunghezza della riga e della pagina, il salto a nuova riga o pagina, la spaziatura delle righe, lo spostamento a destra o sinistra di parti del testo e altro.

Per una descrizione dettagliata di questo comando, fare riferimento al manuale CNUCE numero 30.

SET

PERMETTE DI INTERVENIRE SULLO STATO DELLA MACCHINA VIRTUALE MEDIANTE ALCUNE FUNZIONI DI CONTROLLO.

SET

```
ADSTOP     xxxxxx
AD          OFF

APLBALL     OFF
            ON

ATTN ON
AT   OFF

CARDSAVE OFF
C        ON

LINEDIT OFF
L       ON

MSG ON
M   OFF

PASSWORD
PASS

RUN OFF
R   ON

TRACE ON trdev1 trtyp1...trdevN trtypN
T    OFF

TRACE END

WNG ON
W   OFF
```

ADSTOP xxxxxx — specifica l'indirizzo virtuale a cui si deve arrestare l'esecuzione.

ADSTOP OFF — cancella l'azione di precedenti comandi.

ATTN ON — default. Specifica che la macchina virtuale deve entrare in ambiente CP quando si preme il tasto ATTENTION.

ATT OFF — specifica che l'interruzione causata dal tasto ATTENTION deve essere riflessa alla macchina virtuale.

APLBALL OFF — default. Specifica che il CP/67 usera' tabelle di tabulazione normali per il terminale

| | |
|---|---|
| APLBALL ON | saranno usate tabelle speciali per APL |
| CARDSAVE OFF | default. Cancella un file dal lettore una volta letto |
| CARDSAVE ON | lascia un file nel lettore una volta letto |
| LINEDIT OFF | default. Specifica che non valgono caratteri di edizione della linea scritta da terminale |
| LINEDIT ON | valgono i caratteri di edizione della linea |
| MSG ON | default. Specifica che saranno scritti a terminale i messaggi |
| MSG OFF | impedisce la stampa a terminale dei messaggi |
| PASSWORD | permette di ridefinire la password associata alla macchina virtuale |
| RUN OFF | default. Pone la macchina virtuale nello stato normale in cui lo schiacciamento del tasto ATTENTION ne provoca l'arresto |
| RON ON | sblocca la tastiera quando si preme l'ATTENTION e legge una funzione di CP senza arrestare l'esecuzione della macchina virtuale. La funzione viene eseguita appena letta. |
| TRACE ON | inizia le funzioni di traccia |
| TRACE OFF | termina una o piu' funzioni di traccia |
| trdev | unita' su cui scivere i risultati della traccia: |
| CNSL | uscita sul terminale |
| PRT | uscita sulla stampante |
| BOTH | uscita sia sul terminale che sulla stampante |
| trtyp | tipo di informazione da registrare con la traccia: |
| SVC | interruzioni di SVC |
| I/O | interruzioni di I/O |
| PRG | interruzioni di programma |
| EXT | interruzioni esterne |
| BR | interruzioni e salti |
| ALL | tutte le istruzioni (privilegio operatore) |
| SIO | tutte le operazioni di I/O |
| CCW | insieme a SIO da' le CCW virtuali e reali delle operazioni di I/O |
| TRACE END | termina tutte le funzioni di traccia. |

SETERR — ATTIVA UN "FLAG" CHE CAUSA LA REGISTRAZIONE DI INFORMAZIONI DI TRACCIA OGNI VOLTA CHE UN PROGRAMMA, CHIAMATO VIA SVC, RITORNA AL PROGRAMMA CHIAMANTE CON UN CODICE DI ERRORE NEL REGISTRO 15.

SETERR

NOTE - Saranno registrate le informazioni seguenti:
1) una riga contenente l'indirizzo della SVC e il nome del programma chiamato, il contenuto della vecchia PSW di SVC, e gli indirizzi a cui il programma chiamato dalla SVC ritorna in condizione normale e di errore.
2) il contenuto dei registri prima che il programma chiamato dalla SVC prenda il controllo e dopo che ha emesso un'istruzione di ritorno.
3) il contenuto dei registri floating-point prima che il programma chiamato dalla SVC prenda il controllo e dopo che ha emesso un'istruzione di ritorno.
4) due righe o 16 parole della lista dei parametri esistenti al momento dell'esecuzione della SVC.

- Le informazioni saranno stampate solo dopo un comando CLROVER o KO, o dopo una richiesta di RESTART di DEBUG, o quando un untente fa logout dal CP.

SETOVER — ATTIVA UN "FLAG" CHE CAUSA LA REGISTRAZIONE DI INFORMAZIONI DI TRACCIA PER TUTTI I PROGRAMMI CHIAMATI VIA SVC, SIA CHE SIANO ESEGUITI SENZA ERRORE, SIA CHE TORNINO AL PROGRAMMA CHIAMANTE CON UN CODICE DI ERRORE NEL REGISTRO 15.

SETOVER <SAMELAST> <opzioni 1 ... opzione N>

| | |
|---|---|
| SEMELAST | lascia le opzioni stabilite dall'ultimo comando SETOVER. Se non e' specificato le opzioni saranno quelle default |
| opzioni: | |
| GPRS | da' il contenuto dei registri prima della chiamata e dopo il ritorno del programma chiamato via SVC |
| GPRSB | come sopra solo prima della chiamata |
| GPRSA | come sopra solo dopo il ritorno |
| FPRS<br>FPRSB<br>FPRSA | come sopra ma per i registri floating-point |
| NOPARM | non sara' registrata la lista dei parametri ad ogni chiamata via SVC |
| PARM1 | sara' registrata solo una riga (8 parole) della lista dei parametri ad ogni chiamata via SVC |
| NOWAIT | non saranno registrate informazioni quando si chiama WAIT via una SVC |
| WAITSAME | registra informazioni anche quando si chiama WAIT |
| WAIT1 | registra una riga (8 parole) della lista dei parametri anche per WAIT |
| WAIT2 | registra due righe (16 parole) della lista dei parametri anche per WAIT |

NOTE - Le opzioni default sono:
1) non vengono registrate informazioni riguardanti i registri generali, ne' i registri floating point.

2) vengono registrate due righe (16 parole) della lista dei parametri quando un programma e' chiamato da una istruzione SVC.
3) quando si chiama la funzione WAIT non vengono registrate informazioni relative alla lista dei parametri ne' ai registri generali e floating point, anche se sono state scelte opzioni perche' tali informazioni siano registrate negli altri casi.

SLEEP — PERMETTE ALL'UTENTE DI PORRE IL SUO TERMINALE NELLA CONDIZIONE DI RICEVERE MESSAGGI SENZA PREMERE IL TASTO ATTENTION.

SLEEP

NOTA - Questa funzione puo' essere utile se si e' in attesa di messaggi e non si usi la macchina virtuale per qualche tempo. Per tornare nell' ambiente CP premere il tasto ATTENTION.

SNOBOL — CONVERTE I MODULI SORGENTI SCRITTI IN LINGUAGGIO SNOBOL IN PROGRAMMI SCRITTI IN LINGUAGGIO SPL/1 ED ESEGUE QUESTI PROGRAMMI.

SNOBOL filename (opzione1.....opzioneN)

filename — specifica un file con filetype SNOBOL da compilare ed eseguire, o con filetype SPL1 da eseguire.

Le opzioni sono:

OFFLINE — specifica che le informazioni del file LISTING devono anche essere stampate sulla stampante offline

ONLINE — come sopra, sul terminale

NOLIST — sopprime il file LISTING senza pero' annullare le opzioni precedenti

SPL1 — indica che il file specificato

e' di tipo SPL1 e deve essere eseguito

L'esecuzione e' inoltre governata dalle schede controllo inserite nel modulo sorgente; queste schede devono iniziare col carattere ''-'' (trattino) a colonna 1:

| | |
|---|---|
| -LIST ON | riprende a listare il modulo sorgente SNOBOL nel file LISTING |
| -LIST OFF | interrompe la lista del modulo SNOBOL nel file LISTING |
| -SEQUENCE | fa ignorare il contenuto delle colonne 73-80 del modulo sorgente, permettendo la numerazione delle schede. Questa scheda di controllo deve apparire all'inizio di ogni modulo sorgente. |
| -EJECT | inserisce un carattere di controllo del carrello nel file LISTING, per saltare a nuova pagina |
| -DECK | genera un file "filename PUNCH P1", contenenete l' output SPL/1, in uno speciale formato abbreviato |
| -NOGO | sopprime l' esecuzione del programma compilato |
| -MEMBER=nome | identifica le schede seguenti come una routine SPL/1 |
| -DATA | segna la fine del deck di ingresso; questa scheda deve essere sempre presente anche se non seguono dati |
| -ASSEMBLY OFF | sopprime nel file LISTING la lista dei programmi SPL/1 quando vengono caricati per l' esecuzione |
| -ASSEMBLY ON | include nel file LISTING la lista di programmi SPL/1 quando vengono caricati per l' esecuzione |
| -TRACE | tutte le stringhe cui fa riferimento il programma di utente vengono listate nel file LISTING |

NOTE - Per una descrizione del linguaggio fare riferimento al manuale: SNOBOL (STRING-ORIENTED SYMBOLIC LANGUAGE) IBM 360D-03.2.016

SORT — DISPONE I RECORDS DEL FILE 1 NEL FILE 2 IN ORDINE CRESCENTE RELATIVAMENTE AI CAMPI DI SORT SPECIFICATI.

SORT filename1 filetype1 filename2 filetype2

filename1 nome del file su cui effettuare il sort
filename2 nome (diverso da filename1) del file che conterra' i records ordinati

NOTE - I records sono ordinati secondo il codice EBCDIC. Se vengono trovati records uguali, sono scritti nel file 2 nell' ordine in cui sono incontrati. Tutti devono essere di lunghezza fissa.

- Emesso il comando, viene stampato a terminale un messaggio che richiede di definire il campo del sort. Il campo del sort e' definito scrivendo una coppia di numeri, separati da blanks, che specificano la posizione nel record del primo e ultimo carattere del campo sort.Possono essere specificate tante coppie di numeri quante permesso dalla lunghezza della riga di input a terminale e dei records su cui fare il sort. La lunghezza dei records su cui fare il sort non deve superare i 254 caratteri.

ESEMPI:
1) SORT LAURA SYSIN AD SYSIN
Il CMS scrive a terminale:
*ENTER SORT FIELD DEFINITIONS
Si puo' battere allora per esempio
10 12 20 23 71 72
Il sort sara' effettuato considerando i tre campi definiti e i records sorteggiati saranno scritti nel file AD SYSIN.

SOS PERMETTE DI LEGGERE, SCRIVERE O RISCRIVERE N RECORD-CMS SUL DISCO P.

SOS

Dopo aver battuto il comando SOS viene richiesto a terminale:
DARE IL NUMERO (DECIMALE DI 5 CIFRE) DEL RECORD CMS
Tale record sara' caricato in memoria alla locazione X'13000' Entrando in CP si puo' allora, tramite le funzioni DISPLAY e STORE, controllarne e modificarne il contenuto.
Tornando in CMS il comando chiede:
RISPONDERE: GO, PER RISCRIVERE IL RECORD, O QUIT
Rispondendo QUIT si ritorna in CMS, rispondendo GO il

comando chiede ancora:
DARE IL NUMERO DI RECORD SUL QUALE SCRIVERE
che permette di ricopiare su disco all'indirizzo voluto il record modificato.

SOS1 — PERMETTE DI RITROVARE TUTTI I RECORDS CMS SUL DISCO P CONTENENTI IDENTIFICATORI DI FILES (FSTB), CHE APPARTENGONO O SONO APPARTENUTI AL MASTER FILE DIRECTORY.

SOS1

Emesso il comando SOS1 viene richiesto a terminale:
DARE IL NUMERO DEI REC. DA ESAMINARE (5 CIFRE)
Tale numero devé essere stabilito considerando che ogni cilindro di 2314 contiene 150 records CMS.
Il risultato sara' una lista, sulla stampante, di tutti i records trovati, fra quelli esaminati, di tipo FSTB, contenenti cioe' identificatore di files.

NOTA - Lo scopo di questo comando, e' di permettere il ripristino, insieme al SOS, dei records FSTB che rispecchiano la situazione del disco prima di qualche evento che l'ha modificata ( per es. un comando ERASE, o errori che alterino la catena degli FSTB).

SPLIT — COPIA UNA PARTE SPECIFICATA DI UN FILE E LO AGGIUNGE ALLA FINE DI UN SECONDO FILE, SE ESISTE, O NE CREA UNO NUOVO.

SPLIT fn1 ft1 fn2 ft2 label1 label2
n1 <n2>
eof

fn1 ft1 — identifica il file di cui sara' copiata una parte
fn2 ft2 — identifica il file a cui parte del file1 sara' aggiunta
label1 — una etichetta alfanumerica di 8 caratteri con il primo carattere non numerico, che individua il primo record da copiare
label2 — formato come label1 , individua il record successivo all'ultimo da copiare

n1 numero decimale che spefica il numero di sequenza del primo record da copiare

n2 numero decimale specificante il numero di records da copiare.

NOTE - File1 e File2 non possono indicare lo stesso file.

- Se non viene specificato il secondo parametro, il file verra' copiato fino all'end-of-file.

- Il file1 rimane invariato.

- L'etichetta deve comparire nei primi otto caratteri del record.

ESEMPI:
1) SPLIT ASP FORTRAN APP FORTRAN 20 60
I sessanta records del file ASP FORTRAN a partire dal ventesimo sono copiati nel file APP FORTRAN che sara' creato se non esisteva gia', altrimenti appesi alla fine.

SPOOL — PERMETTE ALL'UTENTE DI INDIRIZZARE I FILES ALL'UNITA' REALE SPECIFICA O DI OTTENERE UNA LETTURA CONTINUA DEI FILES DI INGRESSO.

SPOOL
SP

xxx ON yyy
OFF

ccc CONT
OFF

xxx indirizzo dell'unita' virtuale le cui uscite vanno dirette all'unita' reale yyy

yyy indirizzo dell'unita' reale a cui si vuole indirizzare l'uscita (puo' essere soltanto l'indirizzo di una stampante o di un perforatore)

ccc Indirizzo del lettore virtuale su cui si vuole avere una lettura continua dei files di ingresso.

CONT specifica che tutti i files in arrivo sul lettore vanno considerati appartenenti ad un solo file, fino a quando non si da' l'OFF.

NOTE-Questa funzione puo' essere usata per indirizzare la

stampa ad una stampante con una catena speciale, quando siano disponibili piu' stampanti.

ESEMPI:
1) SPOOL 00E ON 00F
Tutte le stampe successive andranno alla stampante di indirizzo reale 00F.

START — INIZIA L'ESECUZIONE DEL PROGRAMMA O PROGRAMMI CARICATI, A PARTIRE DAL PUNTO DI ENTRATA SPECIFICATO O DEFAULT, E PASSA AL PROGRAMMA L'INDIRIZZO DI UNA LISTA DI ARGOMENTI DELL'UTENTE.

START <entrata <arg1...argN>>

entrata — specifica il nome della sezione di controllo o punto di entrata a cui sara' passato il controllo al momento dell'esecuzione

* — specifica che il controllo deve essere passato al punto di entrata default e deve essere presente se non si specifica il punto d'entrata ma vi sono degli argomenti

arg1...argN — specifica informazioni da passare al programma (max 8 caratteri ogni argomento)

NOTE - Il punto d'entrata default e' l'indirizzo specificato nel campo dell'operando della prima istruzione END avente il campo dell'operando non vuoto, oppure l'inizio del primo file caricato. Gli argomenti specificati vengono passati in una lista il cui indirizzo e' nel registro 1, e sono posti uno per ogni doppia parola.

ESEMPI:
1) START * 65 80 13
Inizia k'esecuzione del programma caricato in precedenza e il registro 1 punta ad una lista contenente in doppie parole i numeri 65 80 13.

STAT — STAMPA A TERMINALE STATISTICHE SULL' UTILIZZO DEI DISCHI DELLA MACCHINA VIRTUALE, QUALI: IL NUMERO DI RECORDS IN USO, IL TIPO DEI DISCHI, LO STATO DEI DISCHI.

STAT <modo <?>>
S *
?

modo specifica che vanno stampate statistiche solo per il disco del modo indicato

* specifica che vanno stampate statistiche per tutti i dischi in lettura e scrittura e in lettura solo

se non e' specificato alcun operando vengono stampate statistiche per tutti i dischi in lettura e scrittura.

? come secondo parametro specifica che si vogliono informazioni addizionali, cioe' il tipo del disco (2311 o 2314), e il suo stato (solo lettura o lettura e scrittura). Come primo parametro provoca la stampa per tutti i dischi attivi di queste informazioni: indirizzo del disco, e modo del disco.

NOTA - Le statistiche fornite danno: il numero dei records in uso, il numero dei records liberi che restano, la percentuale dei records utilizzati e il numero dei cilindri del disco. I records su disco sono sempre di 800 bytes.

ESEMPI:
1) STAT A
Risposta:
A (191): 6 FILES; 98 REC IN USE,1398 LEFT (of 1496", 7%
FULL (10 CYL)
Il disco A di 5 cilindri e' occupato al 7%.

STATE ACCERTA L'ESISTENZA DI UN FILE.

STATE filename filetype (filemode)

NOTE - Se filemode non e' specificato il file e' ricercato

nell'ordine normale

- Se il file non esiste la risposta sara' E(00001).

ESEMPI:
1) STATE PIPPO SYSIN
Se il file esiste la risposta al terminale sara' R;.....

STORE — SOSTITUISCE IL CONTENUTO DI AREE DI MEMORIA, REGISTRI, PROGRAM STATUS WORD, CON LE INFORMAZIONI ESADECIMALI FORNITE.

STORE
<Lhexloc hexinfo1...hexinfoN>
<Shexloc hexstring>
<Greg hexnofo1...hexinfoN>
<Yreg hexinfo1...hexinfoN>
<Xreg hexinfo1...hexinfoN>
<PSW<hexinfo1>hexinfo2>

L — memorizza le informazioni esadecimali hexinfo in parole successive a partire dall'indirizzo esadecimale hexloc se allineato alla parola, altrimenti a partire dal piu' vicino indirizzo allineato alla parola. La quantita' minima di informazione memorizzabile e' una parola.

S — memorizza le informazioni esadecimali hexstring in bytes successivi a partire dall'indirizzo hexloc. La quantita' minima di informazione memorizzabile e' un byte.

G, Y, X — memorizza informazioni esadecimali in registri (generali, floating-point, di controllo rispettivamente) successivi a partire dal registro indicato. Il parametro reg puo' essere un valore decimale da 0 a 15, o esadecimale da 0 a F per i casi G e X e un valore da 0 a 6 per il caso Y. In quest'ultimo caso se reg e' un numero dispari viene cambiato al pari precedente.

PSW — memorizza le informazioni esadecimali hexinfo1 e hexinfo2 nella prima e seconda parola della PSW. Se e' specificata solo

una parola viene memorizzata nella seconda parola della PSW e il codice di interruzione viene messo a zero.

NOTE - Le opzioni possono essere combinate in qualunque ordine purche' entrino su una stessa riga.

- I parametri hexinfo devono essere separati da spazi bianchi.

ESEMPI:
1) STORE 112321 24A12C21 1273
Le informazioni 24A12C21 e 00001273 vengono memorizzate a partire dall'indirizzo 12320 (allineato alla parola)

2) STORE G2 41 127 302A
Le informazioni 00000041 00000127 e 0000302A vengono memorizzate nei registri generali 2,3 e 4.

SYN — PERMETTE , ALL'UTENTE DI SPECIFICARE LE PROPRIE ABBREVIAZIONI DEI COMANDI DA USARE INSIEME O AL POSTO DELLE ABBREVIAZIONI STANDARD DEL SISTEMA.

SYN <fn<ft<fm>><(opz1...opzN)>>
SYN *

fn ft fm — identificatori del file contenente i sinonimi definiti dall'utente.

opzioni:

| | |
|---|---|
| P | stampa le abbreviazioni standard del sistema e i sinonimi dell'utente |
| PUSER | stampa solo i sinonimi dell'utente definiti al momento |
| STD | devono essere usate le abbreviazioni standard del sistema |
| NOSTD | le abbreviazioni standard del sistema non devono essere usate |
| MIN | va usato il numero minimo di caratteri specificati per identificare i comandi |
| EXACT | va usato il numero esatto di caratteri specificati per identificare i comandi |

CLEAR cancella ogni precedente insieme di sinonimi definiti tramite SYN

NOTE - Il file,se specificato, deve contenere records di 80 caratteri nel formato libero:

Comando sinonimo numero

Le colonne 73-80 sono ignorate.Il numero stabilisce il numero di caratteri minimo perche' il sinonimo sia accettato. Se non viene specificato un file, le abbreviazioni del sistema vengomo usate come descritto delle opzioni. Emettere SYN senza parametri d' equivalente a SYN (P).

IAPE

COPIA FILES DA DISCO SU NASTRO IN UN FORMATO CON RECORDS DI 800 BYTES; RICOPIA SU DISCO FILES SCRITTI SU NASTRO IN TALE FORMATO; SCRIVE TAPE MARKS, POSIZIONA IL NASTRO E RICERCA SU NASTRO IDENTIFICATORI DI FILES.

TAPE
T

DUMP filename filetype <filemode>
<n> <1600BPI>
LOAD <800BPI>
REWIND
SCAN <n> <TAPn>
SKIP <n> IAP2
SLOAD filename filetype <n>
WRITEOF <m>

DUMP precisa che il file specificato deve essere scritto su nastro a partire dal punto in cui e' posizionato il nastro. Puo' essere usato "*" al posto di filename e/o filetype per indicare tutti i nomi e/o tutti i tipi.

LOAD n specifica che devono essere copiati su disco tutti i files prima degli n tapemarks che seguono. Il valore default e' 1. Per ogni file copiato viene scritto a terminale nome e tipo

REWIND posiziona il nastro al punto di caricamento

SCAN n lista a terminale gli identificatori di tutti i files su nastro prima degli n tapemarks che seguono (il valore default d' 1), a partire dal punto in cui e' posizionato il nastro al momento del

| | |
|---|---|
| | comando. |
| SKIP n | posiziona il nastro dopo l'ennesimo tapemark (default 1) dal punto in cui e' attualmente posizionato il nastro |
| SLOAD | cerca tra i files che precedono l'ennesimo tapemark dalla posizione corrente del nastro il file specificato e lo carica su disco (default n=1) |
| WRITEOF | scrive n tapemarks dal punto in cui il nastro e' posizionato (default n)1) |
| TAPn | specifica l'indirizzo simbolico dell'unita' nastro (TAP2 default) |
| 800BPI | specifica che il nastro deve essere scritto a 800 BPI su nastro a 9 tracce |
| 1600BPI | specifica che il nastro deve essere scritto a 1600 BPI su nastro a 9 tracce. Questa e' l'opzione default |

NOTA -I files possono contenere records di lunghezza fissa o variabile; i records scritti da TAPE DUMP sono di 805 bytes, di cui i primi 5 di controllo.

- Questo comando e' utile per breare librerie di programmi su nastro.

- Prima dell'dmissione del comando occorre farsi attaccare una unita' nastro alla macchina virtuale.

- TAP2 corrisponde all'unita' nastro di indirizzo 181, TAP1 al 180.

- Nel caso di lettura o di scrittura da un punto diverso dall'inizio del nastro le unita' a doppia densita' (800/1600bpi) leggono e scrivono automaticamente alla densita' a cui e' gia' stata scritta il nastro.

ESEMPI:

1) TATE DUMP * SYSIN
Tutti i files su disco di tipo SYSIN saranno scritti su nastro dalla posizione corrente. Il comando listera' a terminale gli identificatori dei files che scrive su nastro

2) TAPE LOAD 3
Saranno caricati su disco tutti i files esistenti su nastro dalla posizione corrente fino al terzo tapemark seguente. Saranno listati a terminale gli identificatori dei files caricati

3) TAPE SLOAD GASBA FORTRAN 2
A partire dalla posizione corrente e fino al secondo tapemark seguente viene ricercato il file GASBA FORTRAN e caricato su disco se trovato.

TAPEIO — ESEGUE UNA SERIE DI FUNZIONI DI CONTROLLO E DI I/O SU NASTRO.

TAPEIO funzione <TAP1>
TAP2

funzioni:

| | |
|---|---|
| BSF | torna indietro di un file |
| BSR | torna indietro di un record |
| ERG | cancella una gap |
| FSF | avanza di un file |
| FSR | avanza di un record |
| REWIND | riavvolge il nastro |
| RUN | riavvolge e scarica il nastro |
| WRITEOF | scrive un tape mark |
| WTM | scrive un tape mark |

TAP1 e' l'indirizzo simbolico dell'unita' 180
TAP2 e' l'indirizzo simbolico dell'unita' 181 (default)

NOTE -Il comando puo' essere usato con unita' nastro a 9 tracce, 800 o 1600 bpi.

-Su altri tipi di unita' nastro TAPEIO deve essere chiamato come funzione da un programma assembler.

- WTM o WRITEOF, se il nastro e' il punto di caricamento, scriveranno a 1600bpi altrimenti alla densita' a cui e' stato scritto il nastro.

ESEMPI:
1) TAPEIO BSF
Il nastro viene riavvolto per la lunghezza del file che precede la posizione corrente.

TAPRINT — COPIA FILES DI TIPO LISTING DIRETTAMENTE DA NASTRO A STAMPANTE.

TAPRINT <TAP1>
TAP2

TAP1/TAP2 specificano l'indirizzo simbolico dell'unita' nastro da usare. TAP2 default

NOTE -I files devono essere stati creati dal comando ASSEMBLE (opzione LTAPn) o dal comando WRTAPE.

-La stampa inizia dal punto in cui e' posizionato il nastro e prosegue finche' non vengono trovati due end of file consecutivi.

-Al termine il nastro viene riavvolto.

TPCOPY COPIA FILES DA NASTRO A NASTRO.

```
                TAPI    TAPo    n     YES
TPCOPY          <TAP1   <TAP2   <1    NO>>>>
                 *       *      *     *
```

dove:

| | |
|---|---|
| TAPI | e' l'indirizzo simbolico del nastro contenenti i files da copiare |
| TAPo | e' l'indirizzo simbolico del nastro su cui vanno copiati i files |
| n | e' il numero di files da copiare (da 1 a 9, default 1) |
| YES | specifica che un sommario di ciascun file copiato deve essere stampato a terminale (se omesso il sommario non viene stampato) |
| * | indica che devono essere presi i valori per default |

NOTE -La massima dimensione dei records e' 4096 caratteri.

-Il sommario di un file contiene queste informazioni: numero di sequenza del file sul nastro; numero dei suoi records e loro lunghezza in bytes (o lunghezza massima), numero totale dei records copiati.

TXTLIB — E' USATO PER GENERARE UNA LIBRERIA DI FILES TEXT; AGGIUNGERE FILES AD UNA LIBRERIA TEXT ESISTENTE; CANCELLARE FILES DA UNA LIBRERIA TEXT ESISTENTE; LISTARE I PUNTI DI ENTRATA, I NOMI DELLE SEZIONI DI CONTROLLO E LA DIMENSIONE DEI FILES TEXT INCLUSI NELLA LIBRERIA SPECIFICATA.

```
TXTLIB    GENERATE   libname   filename1...filenameN
          G
          ADD        libname   filename1...filenameN
          A
          DELETE     libname   csectname1..csectnameN
          D
          PRINT      libname
          P
          LIST       libname
          L
```

| | |
|---|---|
| GENERATE | crea la libreria 'libname' dal file(s) specificato |
| ADD | aggiunge il contenuto del file(s) specificato alla libreria text 'libname' |
| DELETE | cancella dalla libreria text 'libname' le sezioni di controllo specificate |
| PRINT | crea il file 'libname MAP P1' contenente una lista dei nomi dei punti di entrata e delle sezioni di controllo dei files TEXT nella libreria, la loro posizione nel file e la loro dimensione (in numero di records), e la stampa sulla stampante offline |
| LIST | le stesse informazioni raccolte da PRINT sono stampate a terminale |
| csectname | nome di sezione di controllo |

NOTE -Il tipo delle librerie (libnames) deve essere TXTLIB.

-Il tipo dei files (filenames) specificati deve essere TEXT.

- Le librerie TEXT, tranne SYSLIB, devono essere attivate da un comando GLOBAL T per essere utilizzate dai comandi di caricamento.

ESEMPI:
1) TXTLIB G NEWLIB A1 A5 A9
Viene creata la libreria NEWLIB TXTLIB dai files TEXT di nome A1 A5 A9.

UNPACK — RIPORTA UN FILE COMPATTATO ALLA SUA FORMA ORIGINALE.

UNPACK nome tipo

nome tipo — identificatore del file compattato

NOTE - Il file di partenza (compattato) non viene cancellato

- Il tipo del file ricreato (non compattato) verra' ripristinato automaticamente uguale a quello originale se e' stato creato come descritto nella seconda nota di PACK; altrimenti, se il tipo e' definito in PACK diverso da quello del file da compattare, il tipo del file scompattato sara' ottenuto cambiando in U l'ultimo carattere.

UPDATE — AGGIORNA IL CONTENUTO DI UN FILE SECONDO QUANTO SPECIFICATO NELLE SCHEDE DI CONTROLLO CONTENUTE IN UN SECONDO FILE.

UPDATE
U fn1 <ft1 <fn2 <ft2>>><(opzioni)>

fn1 ft1 — identificano il file da modificare; se ft1 e' omesso, il file e' considerato di tipo SYSIN

fn2 ft2 — identificano il file contenente le schede di controllo; se fn2 e' omesso, al file e' assegnato il nome fn1; se ft2 e' omesso, il file e' considerato di tipo UPDATE

Opzioni:

P — specifica che il file creato, incorporante i cambiamenti deve sostituire il file

originale. Se omesso, il file originale e' conservato e al file creato viene assegnato un nome della forma:punto (.) seguito dai primi sette caratteri del nome del file originale.

SEQ8 specifica che i numeri di sequenza saranno scritti nelle colonne 73-80, saranno cioe' di otto cifre decimali

INC specifica che i numeri di sequenza nelle colonne 73-80 del deck UPDATE devono essere posti nel nuovo deck SYSIN.

Schede di controllo:

./ S segno1 incremento etichetta

Specifica che il nuovo file avra' un numero di sequenza nelle colonne 76-80. Questa scheda, se inclusa deve essere la prima del file UPDATE.

segno1 specifica il primo numero di sequenza

incremento indica l'incremento da aggiungere ogni volta al numero di sequneza per ottenere il successivo

etichetta specifica tre caratteri che saranno posti nelle colonne 73-75.

./ D segno1 segno2

specifica che devono essere cancellate delle schede dal file originale.

segno1 e' il numero di sequenza (originale) della prima scheda da cancellare

segno2 e' il numero di sequenza dell'ultima scheda da cancellare. Se omesso, viene cancellata una sola scheda.

./ I segno1

Indica che devono essere inserite delle schede nel file originale.Queste schede devono seguire questa scheda di controllo. Sono inserite tutte le schede fino alla successiva scheda di controllo.

segno1 specifica il numero di sequenza della scheda dopo la quale saranno inserite le schede che seguono ./ I

./ R segno1 segno2

Specifica che devono essere inserite delle schede nel file originale al posto di schede esistenti.

| | |
|---|---|
| segno1 | e' il numero di sequenza della prima scheda da sostituire |
| segno2 | e' il numero di sequenza dell'ultima scheda da sostituire. Le schede da inserire possono essere in numero diverso di quelle da sostituire, e devono seguire immediatamente la scheda ./ R |

NOTE - Se non e' specificata l'opzione INC e nel file UPDATE non c'e' la scheda di controllo ./S, nelle colonne 73-80 di tutte le schede aggiunte o sostituite saranno scritti asterischi.

- I cambiamenti devono essere specificati relativamente a numeri di sequenza crescenti ordinatamente.Un cambiamento di un record precedente uno gia' cambiato provoca errore.

ESEMPI:
1) UPDATE PINCO
Si presuppone che il file PINCO sia di tipo SYSIN e che i cambiamenti siano nel file PINCO UPDATE.

USE — CARICA IN MEMORIA IL FILE(S) SPECIFICATO (CONTENENTE CODICE RILOCABILE), STABILENDO AGGANCI CON FILES CARICATI IN PRECEDENZA. RIFERIMENTI NON RISOLTI VENGONO RICERCATI IN LIBRERIE TXTLIB EVENTUALMENTE SPECIFICATE.

USE fn1...fnN <(opz1...opzN)<lbn1...lbnN>>

| | |
|---|---|
| fn1 fnN | specifica i nomi dei files TEXT da caricare in memoria |
| opz1.. opzN | specifica le opzioni che saranno in effetto durante il caricamento |
| lbn1...lbnN | specifica i nomi di fino a 8 librerie TXTLIB in cui cercare moduli mancanti durante il caricamento. |

NOTE - Le opzioni che si possono specificare con USE sono le stesse specificabili con LOAD.

- Il comando USE carica i files specificati in posizioni di memoria crescenti, a partire dal punto in cui hanno terminato il caricamento precedenti i comandi LOAD, USE o REUSE.

- USE deve essere preceduto da LOAD; si usa infatti normalmente per risolvere riferimenti non definiti in un precedente LOAD.

ESEMPI:
1) USE COS
Il file COS TEXT e' caricato in memoria risolvendo gli agganci coi files caricati precedentemente.

WRTAPE — COPIA FILES CON RECORDS DI LUNGHEZZA FISSA DA DISCO SU NASTRO (TAP1).

WRTAPE filemane filetype

NOTE -I records vengono bloccati in gruppi di dieci.

-La lunghezza massima dei records e' di 256 caratteri.

-Sul nastro non viene scritto alcun end of file.

-Il nastro non viene riavvolto.

-WRTAPE e' specialmente indicato per trattare i files LISTING creati dai comandi ASSEMBLE e FORTRAN.

-I files scritti su nastro dal comando WRTAPE hanno lo stesso formato di quelli scritti dal comando ASSEMBLE opzione LTAPn.

-Questo formato e' accettabile dal comando TAPRINT.

-I files su nastro scritti col comando WRTAPE non possono essere letti col comando TAPE LOAD.

-Questo comando e' utile per creare librerie su nastro di files di tipo LISTING. In questo caso i files contengono un comando di controllo del carrello della stampante come primo byte di ogni record.

WTAPOS — WTAPOS CREA SOPRA UN NASTRO "STANDARD LABEL" DEI FILES OS A PARTIRE DA FILES CMS.

WTAPOS

Utilizzazione

* WTAPOS crea dei files OS sopra un nastro con label standard attaccato come 181 'TAP2).

* I records sono bloccati a 40 e la lunghezza di ciascun record logico e' quella del file CMS. Il solo formato accettato e' 'F'.

* Il nastro e' riavvolto solamente all'emissione del comando.

* Il comando domanda all'utente:

1/ VOL=

La risposta e' il numero del nastro in 6 caratteri.

Per es.: VOL=000275

Se la label non corrisponde a questo numero viene emesso il messaggio seguente:

INVALID MOUNTING:VOL=002340 CONTINUING ? REPLY YES OR NO

Il numero e' quello del nastro realmente montato.

Se la risposta e' : YES la label e' considerata come buona ed il processo continua al numero 3 altrimenti il lavoro e' abbandonato.

2/ SKIP DATASET=

La risposta e' una cifra da 0 a 9 che e' il numero dei files da saltare prima della scrittura (caso di un nastro gia' utilizzato). Se la risposta e' 0 la scrittura comincia all'inizio del nastro.

3/ DSNAME=

La risposta e' il nome desiderato per il file OS su 17 caratteri al massimo senza bianchi intermedi (convenzione OS).

4/ FILENAME=

La risposta e' il nome del file CMS.

5/ FILETYPE=

La risposta e' il tipo del file CMS.

Se il file CMS non esiste, l'utente ne e' avvisato con il messaggio:

FILE NOT FOUND.TRY A NEW FILENAME,FILETYPE

e il processo riprende al numero 4. Se il file CMS esiste, allora e' copiato sul nastro e l'utente riceve le informazioni seguenti:

a/ LRECL=XXXXX cioe' la lunghezza del record logico del file CMS.

b/ BLKSIZE=XXXXX cioe' la lunghezza del record fisico sul nastro.

c/ END OF COPY quando il file e' ricopiato interamente.

6/ Il processo ricomincia allora al numero 3 permettendo cosi' di copiare piu' files uno di seguito all'altro sullo stesso nastro.

NOTE - Se la risposta a una delle domande e' una linea vuota (ritorno carrello), il comando termina normalmente; la linea vuota e' generalmente inviata su la richiesta DSNAME=

- La struttura del nastro e' una struttura OS; cioe':
  VOL1 HDR1 HDR2*FILE1*EOF1 EOF2*HDR1 HDR2*
  FILE2*EOF1 EOF2*----------

  ---------- HDR1 HDR2*FILEn*EOF1 EOF2**

in cui l'asterisco rappresenta qui un TAPE MARK.

- Se lo sticker di fine nastro e' incontrato durante la copia di un file , il messaggio:

END OF TAPE

e' stampato sul terminale; il comando ignora questo file in corso e da' all'utente il DSNAME dell'ultimo

file copiato correttamente con il messaggio:

LAST FILE AVAILABLE: XXXXXX---XX

Se questo ultimo messaggio non e' stampato nessun file e' utilizzabile (caso particolare dove un solo file ha dimensione superiore alla capacita' del nastro).

Errori

a/ TAPE ERROR

Ci sono degli errori di scrittura su nastro; provare di nuovo il comando; se l'errore continua far cambiare il nastro.

b/ "V" FILES NOT SUPPORTED

Il file CMS da copiare e' di formato variabile.

c/ D SK ERROR

Errori di lettura su disco.
In questi casi di errore il comando termina nello stesso modo del fine-nastro. (vedi nota 3).

d/ BANDE NON STANDARD o ERREUR DE SKIP

Il lavoro riprende al numero 2 (SKIP DATASET=).

XFER — CONTROLLA IL TRASFERIMENTO DI FILES TRA MACCHINE VIRTUALI.

XFER indirizzo TO userid
X OFF *

indirizzo — specifica l'indirizzo virtuale dell'unita' le cui uscite da questo momento saranno o trasferite all'unita' specificata o fatte uscire normalmente.

TO userid — attiva il modo trasferimento. Userid e' il nome della macchina virtuale a cui vanno trasferite le uscite

TO * — specifica che la macchina virtuale a cui si deve trasferire e' quella stessa che ha emesso il comando

OFF — termina il trasferimento delle successive uscite dell'unita' "indirizzo" alla macchina virtuale 'userid'.

NOTE - E' permesso trasferire solo le uscite del perforatore e della stampante (indirizzi normali 00D e 00E rispettivamente).

- I files trasferiti ad una macchina virtuale vengono posti nel lettore virtuale di tale macchina. Il lettore virtuale accetta anche files con records di 132 caratteri. Tali files possono essere letti su disco coi normali comandi CMS OFFLINE READ o DISK LOAD.

- Il comando evita il passaggio su schede in ogni trasferimento di programmi tra macchine virtuali. In particolare una macchina virtuale puo' inviare nel proprio lettore virtuale un programma senza far intervenire un operatore.

- Ogni comando di XFER, anche se con userid errata, cancella gli effetti di un comando precedente.

ESEMPI:

1) XFER 00D TO LARA
Ogni successiva uscita nel perforatore sara' trasferitta al lettore virtuale della macchina LARA.

2) XFER 00E TO *
Ogni successiva stampa sara' dirottata sul lettore virtuale della macchina che richiama la funzione XFER.

3) XFER 00D OFF

Ogni uscita al perforatore verra' effettivamente perforata e non piu' trasferita.

$ ESEGUE UN FILE CONTENENTE UNO O PIU' COMANDI CMS (FILE DI TIPO EXEC), O CARICA IN MEMORIA UN FILE CONTENENTE CODICE IMMAGINE DI MEMORIA (FILE DI TIPO MODULE) O CODICE RILOCABILE (FILE DI TIPO TEXT) E NE INIZIA L'ESECUZIONE.

$ filename (arg1 ... argN)

filename e' il nome del file il cui tipo deve essere EXEC, MODULE, o TEXT

arg1 ... argN sono uno o piu' argomenti che l'utente vuole passare al programma

NOTE - Il file specificato e' ricercato sui dischi nell'ordine normale secondo i tipi EXEC, MODULE e TEXT nell'ordine dato.

- Se per il dato nome si trova un file di tipo EXEC, si assume che il file sia composto di comandi CMS, e si chiamera' il comando EXEC ad eseguirlo. In questo caso gli argomenti del comando hanno la stessa funzione degli argomenti nel comando EXEC.

- I files di tipo MODULE e TEXT sono caricati in memoria chiamando rispettivamente i comandi LOADMOD e LOAD. Gli argomenti sono passati al programma caricato in una lista , ciascuno in una doppia parola, allineati a sinistra, il cui indirizzo e' posto nel registro 1. Il comando START inizia infine l'esecuzione.

3.0 PERMETTE DI TORNARE ALLA VERSIONE 3.0 DEL CMS DOPO CHE SE NE SIA USATA UN'ALTRA.

Uso: Trovandosi in ambiente CMS (non importa quale versione), dare il comando 3.0. Il sistema risponde con il messaggio

CMS..VERSION 3.0 (GG mese AA)

Il comando 3.0 e' l'unico modo per tornare alla versione 3.0 dal CMS 3.2. Dopo aver dato il comando

3.0 e' nuovamente possibile emettere da CP il comando IPL CMS.

## 3.2 FA L'IPL DELLA VERSIONE 3.2 DEL CMS.

Uso: Dare da CMS il comando 3.2. Il sistema risponde con il messaggio

CMS..VERSION 3.2 -MM/DD/YY-

Il comando 3.2 manda in esecuzione una procedura EXEC che esegue le seguenti azioni:
1) stacca il disco 190.
2) si attacca (con un LINK) il disco 290 di DUMPRES come 190.
3) fa IPL 190.
Una volta entrati in ambiente CMS 3.2, ogni successivo KX provoca nuovamente l'IPL del CMS 3.2.
Dopo aver dato il comando 3.2, e' importante ricordarsi di NON DARE MAI IL COMANDO DI CP IPL CMS. E' possibile invece dare nuovamente da CP il comando IPL 190, che fornisce una nuova copia del CMS 3.2. Volendo ritornare dal CMS 3.2 al CMS 3.0, si dia il comando 3.0.
I comandi della versione 3.2 del CMS sono generalmente gli stessi della versione 3.0. Le principali restrizioni riguardano il comando EDIT.
Si raccomanda di usare il CMS 3.2 in particolare a coloro che devono compilare o eseguire programmi in PL/1.