Consiglio Nazionale delle Ricerche

# La nuova versione del JES 2: Punzioni specifiche della installazione CNUCE

G. Mainetto

197

CNUCE

A cura di : Giovanni Mainetto

# LA NUOVA VERSIONE DEL JES2: FUNZIONI SPECIFICHE DELLA INSTALLAZIONE CNUCE

G.Mainetto

ZC-197
CNUCE

# LA NUOVA VERSIONE DEL JES2: FUNZIONI SPECIFICHE DELLA INSTALLAZIONE CNUCE

(G. Mainetto)

## 1. INTRODUZIONE.

L'installazione presso il centro di calcolo del CNUCE del nuovo elaboratore IBM 3081K ed il conseguente trasferimento su tale calcolatore del sistema operativo OS/VS2 MVS e' stata l'occasione per un allineamento del complesso del sistema MVS alle ultime versioni e per una revisione della politica di scheduling e di dispatching dei job operata dal sistema.

In particolare, per quanto riguarda il JES2, l'installazione della Versione 1 Release 3.4 ha determinato una completa riscrittura ed aggiornamento delle modifiche che nel tempo i sistemisti del CNUCE avevano apportato al sottosistema di immissione dei job.

La nuova versione del JES2 si caratterizza per la presenza nel suo codice di un insieme di punti strategici, detti "exit point", dove il JES2 puo' passare il controllo a delle "exit routine", e cioe' a routine scritte dall'utente-sistemista per realizzare funzioni specifiche di una particolare installazione.

Queste chiamate di funzioni rappresentano una interfaccia abbastanza stabile fra il sottosistema JES2 e le routine specifiche di una installazione: e' parso cosi' del tutto ragionevole aggiornare le modifiche apportate al JES2, riscrivendole in termini di exit routine allo scopo di rendere piu' facile la futura migrazione verso nuove versioni del JES2 e meno problematica la manutenzione di questo codice.

In questo scritto viene presentata una breve documentazione relativa al funzionamento delle funzioni specifiche del JES2 installato presso il CNUCE.

L'utenza puo' risultare particolarmente interessata al capitolo 2 in cui si descrivono, da un punto di vista funzionale, le modifiche apportate al JES2 "standard".
Il capitolo 3 contiene una documentazione non di dettaglio relativa alla implementazione delle funzioni specifiche introdotte nel capitolo precedente.
L'ultimo capitolo e' dedicato agli strumenti dell'ambiente VM-MVS utilizzati per lo sviluppo e il testing.
In appendice e' riportata la lista del codice sorgente che rappresenta la implementazione delle exit routine.

## 2. LE FUNZIONI SPECIFICHE DEL JES2 DEL CNUCE.

Questo capitolo presuppone la conoscenza da parte del lettore del formato e del significato di alcuni statement di controllo del JES2 e dei parametri in essi specificabili.

Possiamo cosi' classificare le funzioni aggiunte al JES2 installato al CNUCE:
- funzione di verifica della correttezza lessicale e sintattica di alcuni statement di controllo dei job inviati per l'esecuzione al sistema MVS;
- funzione di scheduling dei job;
- funzioni che realizzano comandi messi a disposizione degli operatori del sistema MVS.

### 2.1 VERIFICA CORRETTEZZA.

Le funzioni di controllo della correttezza lessicale e sintattica sono relative ai parametri degli statement JOB, JOBPARM e SETUP.

Il JOB statement e' il primo statement di controllo del job utilizzato per specificare determinate richieste di risorse necessarie alla elaborazione del job stesso. Non e' un vero e proprio statement di controllo del JES2, anche se il JES2 utilizza informazioni qui presenti durante varie fasi della sua elaborazione.
Ha come parametri: le cosiddette "informazioni di accounting" (codice del programmatore, codice ente, tempo in minuti stimato per l'esecuzione, ecc...), il nome del programmatore e parametri specificabili attraverso parole chiave.
Il controllo lessicale consiste nel verificare che il primo parametro delle informazioni di accounting, il codice del programmatore, sia costituito da una stringa composta esattamente da quattro caratteri alfanumerici (il JES2 standard permette a questo parametro di essere composto da un numero variabile di caratteri compreso fra 1 e 4).
Attraverso questa precisa definizione della lunghezza del codice del programmatore si ha la possibilita' di utilizzare package disponibili sul mercato (quale ad esempio il CA-JASPER) per effettuare l'addebito delle risorse consumate dai job.

Lo statement JOBPARM e' uno statement di controllo del JES2 che permette di ridefinire una serie di parametri caratteristici del job, alcuni dei quali specificabili anche nel JOB statement.
Per motivi legati all'addebito si effettuano dei controlli su questo statement volti ad impedire l'uso di alcuni particolari parametri.
In particolare, la presenza di uno qualsiasi dei parametri

aventi le seguenti parole chiavi:

| | |
|---|---|
| ROOM | abbreviazione R |
| TIME | abbreviazione T |
| LINES | abbreviazione L |
| COPIES | abbreviazione N |

avra' come conseguenza la cancellazione del job.

Lo statement JES2 SETUP serve per specificare le richieste dei nastri necessari alla esecuzione del job. Con lo statement SETUP il job informa il sistema MVS del tipo di nastri richiesti e del loro numero.
Il controllo effettuato sugli statement SETUP consiste nel verificarne la correttezza lessicale e sintattica e cioe' che il job richieda nastri di un certo tipo (TPE9, TPV9, TPE7, TPH9) ed in numero corretto.
Queste informazioni vengono utilizzate per realizzare la funzione di scheduling descritta nel prossimo capitolo.

Infine, fra le funzioni di controllo possiamo classificare quella volta a generare automaticamente il parametro con parola chiave TIME dello statement JOB.
Questa operazione si rende necessaria per la presenza di tale parametro nel JOB statement poiche' il tempo di esecuzione effettivamente a disposizione di un job e' rappresentato dal valore assunto dal parametro con parola chiave e non quello indicato dal terzo parametro delle informazioni di accounting.
La generazione automatica avviene qualora il parametro in questione non sia gia' specificato nel JOB statement dell'utente: a tal fine si utilizza il valore assunto dal parametro di accounting (al limite il suo valore di default) per determinare il valore che deve essere indicato nel parametro con parola chiave.
L'utente puo' cosi' specificare il tempo richiesto come parametro di accounting senza preoccuparsi del corrispettivo parametro con parola chiave; qualora cio' non avvenga, deve ricordarsi che e' la specifica del parametro con parola chiave ad essere significativa.

## 2.2 SCHEDULING DEI JOB.

I job in attesa di esecuzione presenti nella "job queue" devono essere associati ad un initiator per poter iniziare la loro elaborazione.
Lo scheduling consiste nel determinare quale fra tutti i job in attesa di esecuzione debba essere scelto dal JES2 per essere "dato in pasto" ad un initiator libero.
Per comprendere il funzionamento dello scheduling del JES2 installato al CNUCE, occorre sapere come avviene l'interazione JES2-initiator.
Un initiator e' un programma di sistema che viene fatto partire o dall'operatore, con un comando, o automaticamente dal JES2 in fase di inizializzazione del sistema. Un initiator fa partire un job permettendogli cosi' di competere per acquisire le risorse del sistema,

in concorrenza con gli altri job gia' in esecuzione.
L'initiator chiede al JES2 un job da mandare in esecuzione. Il JES2 sa le classi di job associati a ciascun initiator e quindi in quale ordine occorre scandire la coda dei job per individuarne uno. Se ad un initiator per esempio, sono assegnate due classi di job, il JES2 scandisce la job queue per determinare se esiste un job della prima classe in attesa di esecuzione; qualora non trovi alcun job di tale classe, riscandisce la coda alla ricerca di un job appartenente alla seconda classe. All'interno di una determinata classe, il JES2 seleziona job in base alla loro priorita'. Il JES2 seleziona i job a partire da quelli con priorita' piu' alta della prima classe, terminando con quelli a priorita' minore della seconda classe. La priorita' e' funzione del tempo stimato di esecuzione di un job. Quando il JES2 seleziona un job, lo passa all'initiator. Associare a ciascun initiator una o piu' classi, e' il modo per controllare la selezione dei job e quindi per controllare l'uso delle risorse del sistema da parte dei lavori.
Dopo che il JES2 ha selezionato il job da mandare in esecuzione e lo ha passato all'initiator, quest'ultimo provvede a costruire alcuni blocchi di controllo e ad allocare i device di input e di output specificati nel primo step del job. L'allocazione garantisce che i device siano disponibili prima che inizi il primo passo del job. L'initiator infine fa partire il programma richiesto dallo statement EXEC.

Con questo schema generale di funzionamento, l'installazione del CNUCE provvede a controllare lo scheduling attraverso la determinazione della classe di ciascun job e della associazione fra classi ed initiator.

L'installazione MVS del CNUCE ha sempre attivi 13 initiator dedicati agli utenti. La loro associazione alle classi varia durante l'orario di servizio in funzione del tariffario adottato.

Per quanto riguarda il periodo diurno la situazione e' la seguente:

| | | |
|---|---|---|
| I1 | CLASS=AB | INITIATOR 1 FOR JOB CLASS AB |
| I2 | CLASS=AB | INITIATOR 2 FOR JOB CLASS AB |
| I3 | CLASS=KBA | INITIATOR 3 FOR JOB CLASS KBA |
| I4 | CLASS=CBA | INITIATOR 4 FOR JOB CLASS CBA |
| I5 | CLASS=FBA | INITIATOR 5 FOR JOB CLASS FBA |
| I6 | CLASS=DBA | INITIATOR 6 FOR JOB CLASS DBA |
| I7 | CLASS=EDAB | INITIATOR 7 FOR JOB CLASS EDAB |
| I8 | CLASS=GHAB | INITIATOR 8 FOR JOB CLASS GHAB |
| I9 | CLASS=AB | INITIATOR 9 FOR JOB CLASS AB |
| I10 | CLASS=BAC | INITIATOR 10 FOR JOB CLASS BAC |
| I11 | CLASS=BA | INITIATOR 11 FOR JOB CLASS BA |
| I12 | CLASS=AB | INITIATOR 12 FOR JOB CLASS AB |
| I13 | CLASS=AB | INITIATOR 13 FOR JOB CLASS AB |

Allorche' entra in vigore il periodo di elaborazione notturna, vengono cambiate le associazioni initiator-classi ed attivate le classi T; il numero di classi T attivate dipende dalla quantita' di job in attesa di esecuzione presenti sullo SPOOL.

La determinazione della classe cui appartiene un job e' realizzata da un gruppo coordinato di exit routine. La seguente tabella mostra in base a quali criteri si stabilisce la classe di un job (le specifiche TPE9 e TPV9 sono eguivalenti):

| PLOTTER | NASTRI | MINUTI CPU RICHIESTI | CLASSE | HOLD |
|---|---|---|---|---|
| NO | NO | <=3 | A | NO |
| | | >3 E <=10 | B | NO |
| | | >10 | C | NO |
| | 1 O 2 TPE9 | <=10 | D | NO |
| | | > 10 | E | NO |
| | 1 TPH9 | <=10 | F | NO |
| | | > 10 | H | SI |
| | VARI (*) | <=10 | G | SI |
| | | > 10 | H | SI |
| SI | - | - | K | SI |

(*) TPE7 oppure TPH9>1 oppure TPE9>2
oppure 1 TPH9 e 1<=TPE9<=2

(-) non preso in considerazione

Nell'ultima colonna e' indicato se il job viene messo in HOLD e cioe' se si rende necessario un intervento dell'operatore per rilasciarlo e mandarlo in esecuzione.

Se invece un lavoro appartiene alla classe T (ha specificato il parametro CLASS=T nella scheda JOB), allora le exit routine stabiliscono solo se esso debba essere messo in hold:

| PLOTTER | NUMERO NASTRI | CLASSE | HOLD |
|---|---|---|---|
| NO | <3 | T | NO |
| | >=3 | T | SI |
| SI | - | T | SI |

## 2.3 COMANDI PER L'OPERATORE.

Attraverso le exit routine sono stati implementati due comandi che servono all'operatore della macchina MVS per avere informazioni sullo stato dei lavori in esecuzione e su alcuni tipi di richieste effettuate.

Nella descrizione della sintassi dei comandi useremo le seguenti convenzioni:

| | |
|---|---|
| \| \| | indica il fatto che uno solo dei caratteri racchiusi fra questi simboli puo' essere scelto |
| n m | rappresentano sequenze di cifre decimali composte al piu' da quattro cifre |
| ( ) | indica un parametro opzionale |
| ... | significa che il parametro opzionale che precede puo' essere ripetuto piu' volte |

Con il comando $Y, avente come parametri l'identificatore di un job in esecuzione o di uno started task o di un utente TSO, l'operatore viene a conoscenza della quantita' di righe di stampa e di schede per il punch gia' effettuata dal lavoro in questione.
Il comando $Y ha la seguente sintassi:

```
      | J |                  | J |
 $Y   | S |  n (-m) (, (|  S |) n (-m)) ...
      | T |                  | T |
```

Possibili invocazioni del comando $Y sono:

```
$YJ1234   $YS100-500,650-700   $YJ50,T23,S10-20,78
```

Vediamo ora un semplice esempio di utilizzo del comando $Y:

```
 $da
  15.13.31 JOB  424  $HASP608 MAINT1   EXECUTING A PRIO  9
  15.13.31 JOB  201  $HASP608 APR68    EXECUTING C PRIO  6
  15.13.31 JOB  207  $HASP608 APR68A   EXECUTING C PRIO  6
  15.13.31 JOB  217  $HASP608 P0330004 EXECUTING E PRIO  7

 $yj424
  15.13.40 JOB 0424  $HASP608 MAINT1   PR=000100  PU=00000

 $yj201-217
  15.13.55 JOB 0201  $HASP608 APR68    PR=000043  PU=00000
  15.13.55 JOB 0207  $HASP608 APR68A   PR=000046  PU=00000
  15.13.55 JOB 0217  $HASP608 P0330004 PR=000072  PU=00000
```

Attraverso il comando $X, avente come parametro l'identificatore di un job in esecuzione, l'operatore e' informato del codice, tempo, linee di stampa, schede per il punch, form, numero copie di output, e nastri richiesti dal/i job. Inoltre viene mostrato sulla consolle dell'operatore il testo dell'ultima /*MESSAGE presente nel job.
La sintassi del comando e' la seguente:

```
$X  Jn (-m) (, (J) n (-m)) ...
```

Esempi di utilizzo del comando $X:

```
 $da
  16.46.38 JOB  201  $HASP608 APR68     EXECUTING C PRIO  6
  16.46.38 JOB  207  $HASP608 APR68A    EXECUTING C PRIO  6
  16.46.38 JOB  740  $HASP608 P0330004 EXECUTING D PRIO  9
  16.46.38 JOB  685  $HASP608 COPIA99  EXECUTING D PRIO  8
  16.46.38 JOB 7249  $HASP608 BOCC2267 EXECUTING H PRIO  7

 $xj7249
  16.46.52 JOB 7249  $HASP608 A033 T=0012 PR=0008 PU=4000
                              F=STD  CPY=0001 1000
  16.46.53 JOB 7249  $HASP608 /*MSG TPH9=( E041=02224)

 $xj740,j200-300
  16.47.21 JOB 0740  $HASP608 P033 T=0001 PR=0001 PU=0100
                              F=STD  CPY=0001 0100
  16.47.21 JOB 0740  $HASP608 /*MSG NASTRO DA  PLOTTARE.
  16.47.21 JOB 0201  $HASP608 A028 T=0080 PR=0080 PU=0100
                              F=STD  CPY=0001 0000
  16.47.21 JOB 0201  $HASP608 NO /*MSG
  16.47.21 JOB 0207  $HASP608 A028 T=0080 PR=0080 PU=0100
                              F=STD  CPY=0001 0000
  16.47.21 JOB 0207  $HASP608 NO /*MSG
```

Per quanto riguarda i nastri richiesti dal job in esecuzione, le quattro cifre riportate rappresentano il numero di nastri richiesti nel seguente ordine: TPH9, TPE9/TPV9, TPE7. L'ultima cifra rappresenta un flag (assume solo valore 0 o 1) che attesta la richiesta di nastro per il plotter.
Come si puo' notare, il job P033004, avente identificatore 740, ha chiesto un nastro TPE9 che (come si deduce dal testo del messaggio) vuole rappresenti l'input del plotter: non avendo specificato sulla /*MESSAGE TPE9=($PLOT), al job e' stata assegnata la classe D anziche' quella riservata agli utilizzatori del plotter, determinando cosi' una inutile contesa in quella classe con il job COPIA99.

# 3. IMPLEMENTAZIONE.

Nel codice del JES2 esistono exit point predefiniti da cui e' possibile agganciare al piu' 25 exit routine specifiche dell'installazione. L'aggancio alle exit routine e' organizzato in modo che risulti semplice operare dal loro interno sia per alterare il funzionamento standard del JES2 sia per raccogliere informazioni. Infatti, gli exit point sono collocati in punti strategici del codice, laddove vengono svolte particolari funzioni, e i parametri passati alle exit routine sono tali da poter facilmente manipolare le funzioni del JES2 standard. Percio' ognuna delle 25 exit routine implementabili e' preposta ad una particolare funzione.

Inoltre e' prevista, per il blocco di controllo associato a ciascun job (JCT), la presenza di alcuni campi non utilizzati dal JES2 e riservati quindi alle necessita' delle exit routine. Un puntatore a questo blocco di controllo viene passato a quasi tutte le exit routine che sono state realizzate.

Solo quattro exit routine sono state implementate per la realizzazione delle funzioni specificate nel precedente capitolo:

- EXIT 2 scansione JOB statement
- EXIT 4 scansione statement di controllo del JES2 e del JCL
- EXIT 5 preprocessore comandi JES2, elabora i comandi ricevuti dall'interprete di comandi del JES2
- EXIT 20 fine del job in input, altera lo stato del job alla fine della sua elaborazione da parte del JES2.

Nella descrizione della implementazione sara' seguito lo stesso criterio di suddivisione funzionale che ci ha guidato finora, anche se nelle varie exit queste funzioni sono "mescolate" in un unico testo.

## 3.1 IMPLEMENTAZIONE VERIFICA CORRETTEZZA.

La verifica della correttezza del JOB statement viene realizzata dalla EXIT 2.
Alla EXIT 2 e' passato come parametro un puntatore al buffer contenente la scheda JOB: la exit routine scandisce la scheda verificando se il primo dei parametri di accounting e' composto da quattro caratteri; qualora non sia questo il caso, termina con un codice di ritorno che determina la cancellazione del job.

Simile il controllo operato sugli statement SETUP e JOBPARM dalla EXIT 4, con l'unica differenza della maggior complessita' dell'algoritmo di scansione derivante dalla possibile presenza di sequenze di parametri separati da virgole (",") anziche' un parametro in posizione fissa come nel caso precedente.

La generazione automatica del parametro con parola chiave TIME del JOB statement e' realizzata dalla EXIT 2. Come gia' detto, la exit routine scandisce il JOB statement e, dopo aver controllato il primo parametro di accounting, verifica la presenza del secondo che rappresenta il tempo: in tal caso, il suo valore viene salvato nel campo JCTUSER1 della JCT; altrimenti viene salvato in JCTUSER1 il valore di default che e' presente nel campo JCTETIME della JCT.
Quindi scandisce il resto della scheda alla ricerca della parola chiave TIME: se non la trova, utilizza il valore presente in JCTUSER1 per generare il parametro TIME. La generazione consiste nell'aggiungere una virgola al termine dei parametri nella scheda JOB e nel creare la scheda continuazione nel campo JCTXWRK della JCT, segnalando al JES2, tramite un flag opportuno (RXCCBDSW), la presenza della scheda continuazione.
La maggior difficolta' incontrata nell'implementazione e' derivata dalla possibilta' che il JOB statement sia suddiviso su piu' schede e che quindi la EXIT 2 sia richiamata piu' volte per lo stesso statement. Per effettuare la scansione alla ricerca del parametro TIME occorre quindi salvare in JCTUSER4 un indicatore che attesti il punto in cui si e' arrestata la scansione relativamente alla scheda correntemente analizzata.

## 3.2 IMPLEMENTAZIONE DELLO SCHEDULING DEI JOB.

Attraverso le exit routine si realizza la parte di scheduling che consiste nell'associare ad un job la classe.

Le informazioni necessarie ad effettuare questa funzione sono raccolte dalle seguenti exit:

EXIT 2 salva il tempo stimato per l'esecuzione individuato dal parametro con parola chiave TIME in JCTUSER0;

EXIT 4 il numero di nastri TPH9 richiesti dal job viene salvato nel primo byte del campo JCTUSER1, il numero di nastri TPE9/TPV9 nel secondo byte di JCTUSER1, il numero di TPE7 nel terzo byte, e il quarto byte viene usato come flag che attesta la richiesta del plotter.

Le varie informazioni sono raccolte durante la fase di scansione degli statement JOB e SETUP.
Per quanto riguarda il plotter, la sua richiesta viene raccolta dalla EXIT 4 che scandisce le schede /*MESSAGE del job alla ricerca della specifica $PLOT o $CALC per un TPE9 o TPV9.

Le informazioni cosi' raccolte sono poi utilizzate dalla EXIT 20, la quale provvede ad assegnare il giusto valore al campo della JCT che rappresenta la classe (JCTCLASS) e a settare opportunamente il flag di hold (JCTJBOPT) sulla base dei criteri precedentemente indicati.

## 3.3 IMPLEMENTAZIONE COMANDI.

L'implementazione dei comandi $Y e $X e' identica per quanto concerne il reperimento delle informazioni e differisce unicamente nel tipo di informazioni mostrate a consolle.

In fase di raccolta delle informazioni, la EXIT 4 provvede a copiare il testo del'eventuale statement JES2 MESSAGE nella JCTXWRK e a settare il flag di presenza messaggio (primo byte JCTUSER2). Questa operazione e' necessaria alla esecuzione di $X.

Il reperimento delle informazioni contenute nella JCT del job desiderato, avviene nel seguente modo: dapprima si converte il numero rappresentante l'identificatore del job in una forma interna opportuna utilizzando per la conversione del numero in binario la macro $CFCVB; indi si reperisce, tramite la macro $QLOC, il Job Queue Element (JQE), un blocco di sistema associato ad ogni job in cui e' presente l'indirizzo di disco della JCT; infine, dopo aver ottenuto un buffer via macro $GETBUF, si legge la JCT dal disco nel buffer (macro JCTIO).

A questo punto entra in gioco la parte di codice specifica del comando desiderato.

$Y utilizza solo i campi JCTLINES e JCTPUNCH.

$X utilizza: JCTROOMN (codice), JCTUSER0 (tempo), JCTESTLN (linee stampa), JCTESTPU (scede punch), JCTCPYCT (copie), JCTUSER1 (nastri) e, se settato flag presenza MESSAGE, JCTXWRK.

L'intera funzione sopra descritta e' inserita in un loop che provvede a generare gli identificatori dei job nel caso in cui il comando indichi un job range anziche' un singolo job.

## 4. AMBIENTE DI SVILUPPO.

Le facility messe a disposizione dall'ambiente JES2/MVS/VM si sono rivelate particolarmente efficaci per il testing delle funzioni JES2 implementate.

Oltre alle possibilita' contemplate dalla implementazione del JES2 (exit routine e campi in tabelle di sistema riservate ai sistemisti), significative per il testing sono risultate le seguenti facility:

- attivazione/disattivazione exit routine (JES2);
- tracing delle chiamate di exit routine (JES2);
- possibilita' di avere piu' sistemi JES2 funzionanti contemporaneamente (MVS);
- creazione di Channel-To-Channel (CTC) virtuali fra macchine virtuali (VM/CP).

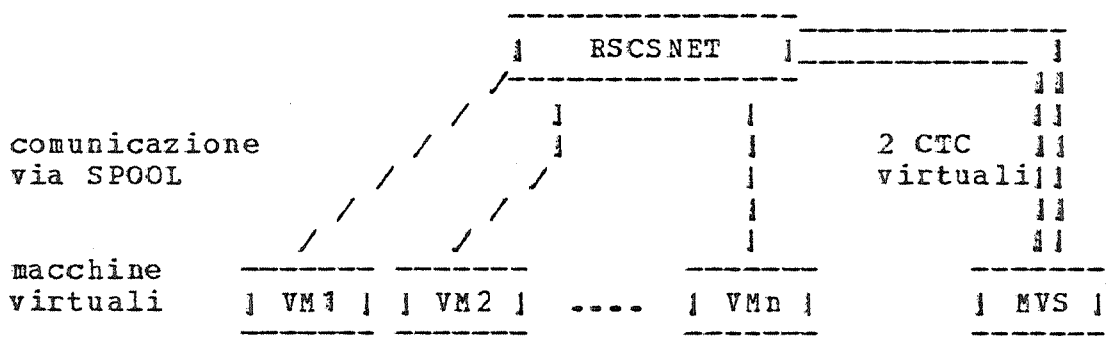
La prima facility permette di individuare errori nel codice procedendo in modo da tentare di "isclare" (disattivare) quella/e exit routine in cui si trova l'errore. La dis/attivazione e' realizzata tramite un comando JES2 cui e' abilitato solo l'operatore di consolle.

La seconda permette di stampare due tipi di informazioni derivanti dal tracing delle exit routine: una e' gestita da un comando JES2 e permette di produrre informazioni limitate quali l'identificazione dell'exit point e il contenuto dei registri al momento della invocazione della exit routine; l'altra permette di avere informazioni complete sulla elaborazione della exit routine e viene gestita tramite la macro JES2 $TRACE.

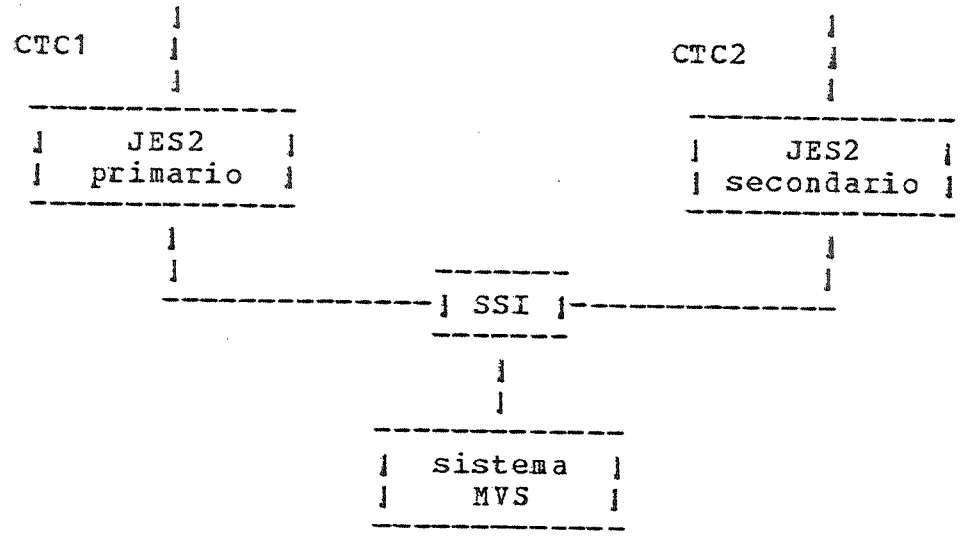
Attraverso la terza facility, quella che permette di avere piu' di un sottosistema operante contemporaneamente, e' stato possibile svolgere il testing delle exit routine senza interrompere l'elaborazione del JES2 primario. Il JES2 primario e quello secondario (di testing) comunicano con il sistema MVS attraverso una interfaccia comune detta Subsystem Interface (SSI).

Infine, potendo definire CTC virtuali, si e' provveduto a collegare la macchina virtuale RSCS alla macchina virtuale MVS con un ulteriore CTC cosi' da avere complessivamente due canali di comunicazione, ognuno dei quali gestito da un JES2.

La configurazione a livello di macchine virtuali usata per il testing puo' essere cosi' schematizzata:



Per quanto riguarda la macchina virtuale MVS, la configurazione dei componenti software e' stata la seguente:




Con questa configurazione software/hardware (virtuale), e' stato possibile sviluppare e fare il testing del nuovo JES2 interferendo solo minimamente con il normale funzionamento del JES2 primario.
In un solo caso si e' verificata una caduta del JES2 secondario che ha determinato una situazione inconsistente nella SSI tale comunque da non danneggiare le attivita' del JES2 primario. All'inconveniente si e' ovviato con una ripartenza a caldo del JES2 primario.

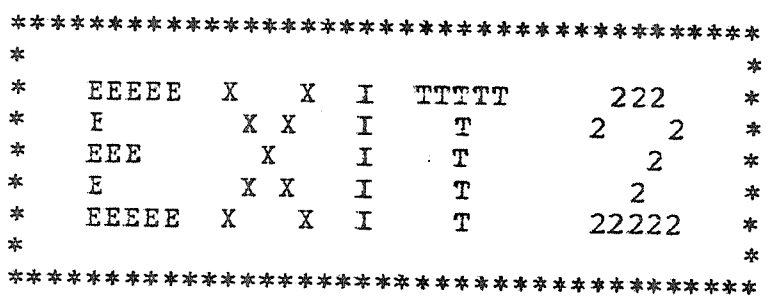
Il linguaggio di implementazione utilizzato e' stato l'ASSEMBLER VS/VM, un linguaggio di livello non eccessivamente alto. Purtroppo, in questi tipi di lavori, occorre fare di necessita' virtu' ed apprezzare gli strumenti utilizzabili.

RINGRAZIAMENTI.

Il lavoro svolto e' stato possibile grazie ai consigli di P. Lazzareschi cui si deve la realizzazione della versione precedente delle funzionalita' specifiche del JES2 installato al CNUCE.

APPENDICE.

In appendice viene riportato il sorgente delle exit routine. Il testo di ogni exit routine e' preceduto da un commento in cui si descrivono la funzione implementata, i valori dei registri in ingresso ed in uscita, i campi della JCT che vengono usati e contiene una indicazione dell'uso fatto nella exit routine dei registri generali. Ogni statement assembler e' commentato.


EEEEE X X I TTTTT 222
E X X I T 2 2
EEE X I T 2
E X X I T 2
EEEEE X X I T 22222

```
         TITLE 'HASP USER EXIT 2 -- PROLOG (MODULE COMMENT BLOCK)'
**** JES2 EXIT2 ************************************************************
*                                                                           *
*     SCOPO:                                                                *
*     SCANDIRE SCHEDA/E COMPONENTE/I JOB STATEMENT PER DETERMINARE IL       *
*     TEMPO RICHIESTO DAL JOB.                                              *
*     DAPPRIMA CERCA IL TEMPO SPECIFICATO NELLE INFORMAZIONI DI             *
*     ACCOUNTING E LO SALVA IN JCTUSER1 (QUALORA NON SIA PRESENTE           *
*     VIENE SALVATO IL VALORE DI DEFAUIT CONTENUTO IN JCTETIME).            *
*     INDI, VIENE CERCATO IL PARAMETRO TIME=.... :                          *
*     SE NON C'E', VIENE GENERATA UNA SCHEDA DI CONTINUAZIONE DEL JOB       *
*     STATEMENT CHE SPECIFICA COME TEMPO RICHIESTO QUELLO SALVATO           *
*     IN JCTUSER1 E TALE VALORE VIENE COPIATO IN JCTUSER0 ;                 *
*     ALTRIMENTI IL VALORE INDICATO DAL PARAMETRO TIME=.... VIENE           *
*     SALVATO NEL CAMPO JCTUSER0.                                           *
*     AL TERMINE DELLA EXIT, IN JCTUSER0 SI TROVA IL VALORE IN              *
*     BINARIO DEL TEMPO RICHIESTO DAL JOB.                                  *
*     INOLTRE FA SI' CHE IL JES2 CANCELLI TUTTI I JOB CHE NON HANNO         *
*     IL PRIMO PARAMETRO DI ACCOUNTING DEL JOB STATEMENT ESATTAMENTE        *
*     COMPOSTO DA QUATTRO CARATTERI.                                        *
*                                                                           *
*     ENTRY POINT = UEXIT2                                                  *
*                                                                           *
*     REGISTRI IN INPUT:                                                    *
*        R0     =0 INDICA LA SK INIZIALE DEL JOB STATEMENT                 *
*               =4 INDICA UNA SK CONTINUAZIONE DEL JOB STATEMENT           *
*        R1     INDIRIZZO DI UN'AREA COMPOSTA DA 3 WORD:                   *
*               WORD1 (+0) PUNTA AL JOB STATEMENT IMAGE BUFFER             *
*               WORD2 (+4) PUNTA ALL'EXIT FLAG BYTE, RXITFLAG, NEL PCE     *
*               WORD3 (+8) PUNTA AL CAMPO JCTXWRK DELLA JCT                *
*        R2-9   N/A                                                        *
*        R10    INDIRIZZO DELLA JCT DEL JOB                                *
*        R11    INDIRIZZO DELLA HCT DEL JES2                               *
*        R12    N/A                                                        *
*        R13    INDIRIZZO DEL PCE                                          *
*        R14    INDIRIZZO DI RITORNO                                       *
*        R15    ENTRY ADDRESS                                              *
*                                                                           *
*     REGISTRI IN OUTPUT:                                                   *
*        R0-14  DOVREBBERO CONTENERE I VALORI CHE AVEVANO IN INPUT         *
*        R15    ZERO O OTTO                                                *
*                                                                           *
*     CODICE DI RITORNO = 0 O 8 (IN R15)                                    *
*                                                                           *
*     CAMPI JCT RISERVATI UTENTE UTILIZZATI :                               *
*        JCTUSER0    TEMPO STIMATO DI ESECUZIONE DEL JOB                    *
*        JCTUSER1    TEMPO DI ACCOUNTING DEI JOB                 (TEMP)     *
*        JCTUSER2    AREA DI LAVORO PER CONVERSIONE BIN<->CHAR    (TEMP)     *
*        JCTUSER4    NUMERO DI PARM SCANDITI                     (TEMP)     *
*        JCTXWRK     PER TESTO EVENTUALE SK GENERATA             (TEMP)     *
*      AL TERMINE DELLA EXIT, I CAMPI TEMPORANEI HANNO QUESTI VALORI:       *
*        JCTUSER1    ZERO BINARIO                                           *
*        JCTUSER2    ZERO BINARIO                                           *
*        JCTUSER4    GARBAGE                                                *
*        JCTXWRK     GARBAGE                                                *
*                                                                           *
*****************************************************************************
         EJECT
```

```
*****************************************************************************
*                                                                           *
*     CONTENUTO REGISTRI DURANTE ESECUZIONE EXIT:                           *
*        R2    PASSO SCANSIONE SK IN INPUT (R2=1)                           *
*        R3    PUNTA ALL'ULTIMO CHAR DELLA SK IN INPUT                      *
*        R4    PUNTA AL CHAR CORRENTE DELLA SK                              *
*        R5    CONTIENE LUNGHEZZA CAMPO IN USCITA DA SUBROUTINE             *
*        R6    INTERO INDICANTE PUNTO SCANSIONE CUI SI E' GIUNTI            *
*        R7    TEMPORANEO IN ALCUNE SUBROUTINE                              *
*        R8    PUNTA ALL'ULTIMA VIRGOLA INCONTRATA DURANTE SCANSIONE        *
*        R9    INDIRIZZO DI RITORNO PER LINK DELLE SUBROUTINE               *
*        R12   REGISTRO BASE                                                *
*                                                                           *
*****************************************************************************
         TITLE 'HASP USER EXIT 2 -- PROLOG (HASP GLOBALS)'
         COPY  $HASPGBL            COPY HASP GLOBALS
         TITLE 'HASP USER EXIT 2 -- PROLOG ($MODULE)'
HASPU2   $MODULE NOTICE=SP133,                                              C
               SYSP=(NOGEN,NOGEN,NODATA,NOGEN,NOGEN),                       C
               $BUFFER,            REQUIRED BY MACRO MODULE                 C
               $CAT,               REQUIRED BY HCT                          C
               $DCT,               REQUIRED BY HCT                          C
               $HASPEQU,           GENERATE HASP EQUATES                    C
               $HCT,               REQUIRED BY MACROS SAVE AND RETURN       C
               $JCT,               JOB CONTROL TABLE DSECT                  C
               $JQE,               REQUIRED BY HCT                          C
               $MIT,               REQUIRED BY MACRO MODEND                 C
               $PCE,               REQUIRED BY HCT                          C
               $PDDB,              REQUIRED BY HCT                          C
               $RDRWORK            REQUIRED FOR RXCCRDSW
         TITLE 'HASP USER EXIT 2 -- PROLOG (EQUATES)'
*****************************************************************************
*        DEFINIZIONE CAMPI JCT UTILIZZATI DALLA EXIT:                       *
*        JCTUSER0   TEMPO STIMATO DI ESECUZIONE DEL JOB                     *
*        JCTUSER1   TEMPO DI ACCOUNTING DEL JOB                  (TEMP)     *
*        JCTUSER2   AREA DI LAVORO PER CONVERSIONE BIN<->CHAR    (TEMP)     *
*        JCTUSER4   NUMERO DI PARM SCANDITI                      (TEMP)     *
*        JCTXWRK    PER TESTO EVENTUALE SK GENERATA              (TEMP)     *
*****************************************************************************
ESTMTIME EQU   JCTUSER0            WORD PER TEMPO ESECUZIONE JOB
ACCTTIME EQU   JCTUSER1            WORD PER TEMPO ACCOUNTING JOB
CONVTEMP EQU   JCTUSER2            DOUBLEWORD PER CONVERSIONE BIN->CHAR
NEXTPARM EQU   JCTUSER4            WORD PER INDICATORE PARM SCANDITI
```

```
         TITLE 'HASP USER EXIT 2 -- ENTRY POINT'
UEXIT2   $ENTRY CSECT=YES,BASE=R12
         $SAVE
         LR      R12,R15           CARICA REGISTRO BASE
         SR      R15,R15           ZERO COME CODICE DI RITORNO
         CLI     JCTJOBID,C'J'     JOB BATCH?
         BNE     EXIT2END          NO, NIENTE SCANSIONE
         L       R4,0(,R1)         R4 PUNTA
         LA      R4,1(,R4)            ALLA SECONDA BARRA DELLA SK
         LA      R3,69(,R4)        R3 ALL'ULTIMO CARATTERE DELLA SK
         LA      R2,1              R2 CONTIENE IL PASSO DI SCANSIONE SK
         LTR     R0,R0             PRIMA SK DEL JOB STATEMENT?
         BNZ     CCARD             NO, SCANDISCI SCHEDA CONTINUAZIONE
         TITLE 'HASP USER EXIT 2 -- JOB STATEMENT SCANNING'
*********************************************************************
*        PRIMA SK JOB STMT: INIZIALIZZAZIONE VARIABILI              *
*********************************************************************
         SR      R6,R6               INDICATORE PARAMETRO SCANDITO
         ST      R6,ESTMTIME         ZERO IN TEMPO DI TIME=XXXX
         ST      R6,ACCTTIME         ZERO IN TEMPO DI ACCOUNTING
         SPACE 3
*********************************************************************
*        PRIMA SK JOB STMT: JOBNAME                                 *
*********************************************************************
         BAL     R9,PICKNBLN    PRENDI IL PRIMO CAMPO (JOBNAME)
         LTR     R5,R5          ERRORE
         BZ      EXIT2ERR             SE NON C'E'
         AR      R4,R5          PUNTATORE SCANSIONE SU ULTIMO CHAR JOBNAME
         S       R5,EIGHT       ERRORE
         BP      EXIT2ERR             SE JOBNAME E' PIU' LUNGO DI 8
         BAL     R9,SKIPBLAN    SKIPPA BLANKS E AGGIORNA PUNT SCANSIONE
         SPACE 3
*********************************************************************
*        PRIMA SK JOB STMT: JOB KEYWORD                             *
*********************************************************************
         CLC     1(4,R4),JOBKEY       C'E' "JOB "?
         BNE     EXIT2ERR       ERRORE SE NON C'E'
         LA      R4,3(,R4)      PUNTATORE SCANSIONE SU 'B' DI "JOB"
         BAL     R9,SKIPBLAN    SKIPPA BLANKS E AGGIORNA PUNT SCANSIONE
         CLI     1(R4),C'('     C'E' UNA PARENTESI TONDA APERTA?
         BNE     EXIT2ERR       ERRORE SE NON C'E'
         LA      R4,1(,R4)      PUNT. SCANSIONE SU APERTA TONDA
```

```
         TITLE 'HASP USER EXIT 2 -- JOB STMT SCANNING (ACCOUNTING PARM)'
*********************************************************************
*        PRIMA SK JOB STMT: PRIMO PARAMETRO DI ACCOUNTING           *
*********************************************************************
         BAL   R9,PICKACC1   PRENDI PRIMO PARM DI ACCOUNTING
         BAL   R9,ACCTTEST   SE PARAMETRO NON CORRETTO, ERRORE
         S     R5,FOUR       LUNGHEZZA DEL CAMPO
         LTR   R5,R5                          UGUALE A 4?
         BNZ   EXIT2ERR      NO, ERRORE
         LA    R4,5(R4)      PUNT. SCANS. SU CHAR SUCC PRIMO PARM ACCT
         BAL   R9,SAVEINDX   AGGIORNA INDICATORE PARAMETRO SCANDITO
         BAL   R9,CONTTEST   SE C'E' ", " VAI A TRATTAMENTO CONTINUAZ
         SPACE 3
*********************************************************************
*        SCANSIONE JOB STMT: SECONDO PARAMETRO DI ACCOUNTING        *
*********************************************************************
ACCT2    BAL   R9,PICKACCT   PRENDI SECONDO PARM DI ACCOUNTING
         BAL   R9,ACCTTEST   SE PARAMETRO NON CORRETTO, ERRORE
         LR    R8,R4         SALVA PUNTATORE PARAMETRO CORRENTE
         AR    R4,R5         PUNTATORE SCANSIONE
         LA    R4,1(,R4)        SU CARATTERE SUCCESSIVO SECONDO PARM
         BAL   R9,SAVEINDX   AGGIORNA INDICATORE PARAMETRO SCANDITO
         BAL   R9,CONTTEST   SE C'E' ", " VAI A TRATTAMENTO CONTINUAZ
         CLI   0(R4),C','    SE C'E' UNA VIRGOLA
         BE    TIMEACCT         TRATTA PARAMETRO TIME DI ACCOUNT,
         MVC   ACCTTIME,JCTETIME  ALTRIMENTI DEFAULT IN TEMPO DI ACCT
         BAL   R9,SAVEINDX   AGGIORNA INDICATORE A TEMPO DI ACCT FATTO
         B     ACCTSTOP      E VAI A FINE PARAMETRI DI ACCOUNT
         EJECT
```

```
*********************************************************************
*        SCANSIONE JOB STMT: TERZO PARAMETRO DI  ACCOUNTING         *
*********************************************************************
TIMEACCT BAL   R9,PICKACCT   SELEZIONA TERZO PARM DI ACCOUNTING
         BAL   R9,SAVEINDX   AGGIORNA INDICATORE A TEMPO DI ACCT FATTO
         LTR   R5,R5         TEST SULLA LUNGHEZZA DEL CAMPO
         BZ    DEFTIME       SE E' ZERO VAI A METTERE VALORE DI DEFAULT
         BAL   R9,ACCTTEST   SE PARAMETRO NON CORRETTO, ESCI
         BAL   R9,NUMTEST    SE IL PARAMETRO E' NUMERICO
         BAL   R9,NUMCONV    ALLORA CONVERTILO IN BINARIO
         LTR   R0,R0         QUANTO E' IL TEMPO DI ACCOUNT?
         BZ    DEFTIME       SE VALE ZERO, ALLORA METTI DEFAULT
         ST    R0,ACCTTIME   ALTRIMENTI METTILO NEL TEMPO DI ACCOUNT
         B     NEXTACCT      E VAI A TRATTARE GLI ULTIMI PARM DI ACCT
DEFTIME  MVC   ACCTTIME,JCTETIME    METTI DEFAULT IN TEMPO DI ACCOUNT
         SPACE 3
*********************************************************************
*        SCANSIONE JOB STMT: ULTIMI PARAMETRI DI ACCOUNTING         *
*********************************************************************
NEXTACCT LR    R8,R4         SALVA PUNTATORE PARAMETRO CORRENTE
         AR    R4,R5         PUNTATORE SCANSIONE
         LA    R4,1(,R4)        SU CARATTERE SUCCESSIVO PARM DI ACCOUNT
         BAL   R9,CONTTEST   SE C'E' UNA CONTINUAZIONE, ALLORA ESCI
         CLI   0(R4),C')'    SE C'E' UNA CHIUSA TONDA,
         BE    ACCTSTOP      ALLORA VAI A FINE PARAMETRI DI ACCOUNTING
ACCTLAST BAL   R9,PICKACCT   ALTRIMENTI, SELEZIONA PROSSIMO PARAMETRO
         B     NEXTACCT                  E RICOMINCIA IL LOOP
ACCTSTOP BAL   R9,SAVEINDX   INDICATORE A FINE PARAMETRI DI ACCOUNT
         LA    R4,1(R4)      PUNTATORE SCANSIONE SU CHAR SUCCESSIVO )
         BAL   R9,ENDTEST    SE E' FINITO LO STATEMENT, ALLORA SK CONT
         BAL   R9,CONTTEST   SE C'E' UNA CONTINUAZIONE ESCI
         CLI   0(R4),C','    SE NON C'E' UNA VIRGOLA
         BNE   EXIT2ERR         ERRORE
```

```
      TITLE 'HASP USER EXIT 2 -- JOB STATEMENT SCANNING(KEYWORD PARM)'
*********************************************************************
*        SCANSIONE JOB STMT: PROGRAMMER'S NAME                      *
*********************************************************************
PGMNCONT BAL   R9,SAVEINDX     AGGIORNA INDICATORE A PRGR'S NAME FATTO
         CLI   1(R4),C','      SE C'E' UNA VIRGOLA
         BE    NEXTKEYW        ALLORA NIENTE PROGRAMMER'S NAME
         CLI   1(R4),C''''     SE NON C'E' UN APICE
         BNE   KEYWCONT        ALLORA NIENTE PROGRAMMER'S NAME
         LR    R8,R4           ALTRIMENTI SALVA PUNTATORE SCANSIONE
         LA    R4,1(,R4)       PUNTATORE SCANSIONE SU APICE
         BAL   R9,SKIPPGMN     E SCANSIONE PROGRAMMER'S NAME
         SPACE 3
*********************************************************************
*        SCANSIONE JOB STMT: KEYWORD PARAMETERS                     *
*********************************************************************
NEXTKEYW LA    R4,1(,R4)       INCREMENTA PUNTATORE SCANSIONE
         BAL   R9,ENDTEST      SE FINITO JOB STATEMENT,AGGIUNGI SK CONT
         BAL   R9,CONTTEST     SE C'E' UNA CONTINUAZIONE ESCI
KEYWCONT LR    R8,R4           SALVA PUNTATORE SCANSIONE
         BAL   R9,PICKKEYW     SELEZIONA KEYWORD
         LTR   R5,R5           TEST SULLA LUNGHEZZA DELLA KEYWORD
         BZ    EXIT2ERR        ERRORE SE E' ZERO
         LA    R4,1(,R4)       PUNTATORE SCANSIONE SU PRIMO CHAR KEYWORD
         CLC   0(4,R4),TIMEKEYW  E' TIME LA KEYWORD?
         BE    TIMEFOUN        SI, VAI A TRATTARE IL PARAMETRO TIME=....
         AR    R4,R5           NO, PUNTATORE SCANSIONE SU =
         BAL   R9,SKIPPARM     SALTA PARAMETRO
         B     NEXTKEYW        E VAI AL SUCCESSIVO
```

```
       TITLE 'HASP USER EXIT 2 -- JOB STATEMENT SCANNING(TIME= PARM)'
*********************************************************************
*        SCANSIONE JOB STMT: TROVATO PARAMETRO TIME=XXXX            *
*********************************************************************
TIMEFOUN LA    R4,4(,R4)      PUNTATORE SU =
         CLI   1(R4),C'('     C'E' UNA APERTA PARENTESI TONDA?
         BNE   MINUTES        NO, ALLORA SOLO MINUTI
         LA    R4,1(,R4)      SI, PUNTATORE SU APERTA TONDA
         BAL   R9,PICKACC1    SELEZIONA MINUTI
         LTR   R5,R5          TESTA NUMERO CIFRE COMPONENTI I MINUTI
         BNZ   TIMECONV       SE NON ZERO,ALLORA VAI A FARE LA CONVERS
         B     TIME1MIN       ALTRIMENTI, METTI 1 MINUTO
MINUTES  BAL   R9,PICKTIME    SELEZIONA MINUTI
TIMECONV BAL   R9,ACCTTEST    SE 0 < LUNGHEZZA <= 4
         BAL   R9,NUMTEST     E SE NUMERICO
         BAL   R9,NUMCONV     ALLORA CONVERTI
         LTR   R0,R0          QUANTI SONO I MINUTI SPECIFICATI?
         BNZ   TIMESAVE
TIME1MIN LA    R0,1           SE ZERO, METTI 1
TIMESAVE ST    R0,ESTMTIME    SALVA TEMPO STIMATO PER ESECUZIONE JOB
TIMESTOP BAL   R9,SAVEINDX  AGGIORNA INDICATORE A TEMPO DI ESECUZ FATTO
         ST    R15,ACCTTIME   AZZERA
         ST    R15,CONVTEMP         CAMPI
         ST    R15,CONVTEMP+4             TEMPORANEI
         B     EXIT2END       E TERMINA
         SPACE 3
*********************************************************************
*        SCANSIONE JOB STMT: GENERAZIONE PARAMETRO TIME=XXXX        *
*********************************************************************
MOVECONT SR    R4,R4          R4 FLAG PER MUOVERE ULTIMO CAMPO
INSRCONT MVC   ESTMTIME,ACCTTIME   TEMPO STIMATO = TEMPO DI ACCOUNTING
         BAL   R9,CREACONT    INIZIA GENERAZIONE SCHEDA CONTINUAZIONE
         LTR   R4,R4          C'E' DA SPOSTARE ULTIMO CAMPO?
         BNZ   INSRTIME       SE NO, ALLORA VAI A METTERE TIME=XXXX
         BAL   R9,LASTCONT    COPIA ULTIMO PARAMETRO JOB STMT SU SK CONT
         LR    R4,R8          R4 PUNTA PRIMO CHAR ULTIMO CAMPO
INSRTIME BAL   R9,TIMECONT    TIME=XXXX IN SK CONT E ", " IN SK CORRENTE
         B     TIMESTOP       TERMINATA INSERZIONE SCHEDA CONTINUAZIONE
```

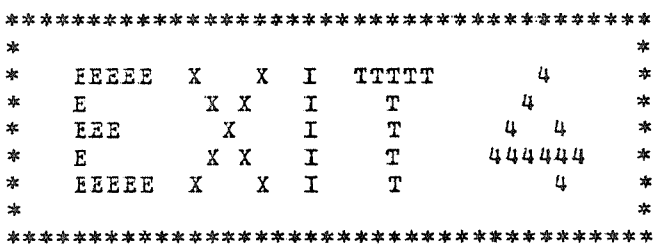
```
       TITLE 'HASP USER EXIT 2 -- JOB STMT SCANNING(CONTINUATION CARD)'
*********************************************************************
*        TRATTAMENTO SCHEDA CONTINUAZIONE JOB STATEMENT             *
*********************************************************************
CCARD    BAL    R9,SKIPBLAN    SKIPPA BLANKS E AGGIORNA PUNTATORE SCANS
         L      R6,NEXTPARM    IN REG6 INDICATORE PARAMETRO DA SCANDIRE
         B      CONTTABL(R6)   CONTINUA ELABORAZIONE
CONTTABL B      EXIT2ERR       +0  ERRORE
         B      ACCT2          +4  SCANDITO PRIMO PARM DI ACCOUNTING
         B      TIMEACCT       +8  SCANDITO SECON PARM DI ACCOUNTING
         B      ACCTLAST       +12 SCANDITO TERZO PARM DI ACCOUNT (TIME)
         B      PGMNCONT       +16 SCANDITI PARM DI ACCOUNTING
         B      INITCONT       +20 SCANDITO PRGM'S NAME, MANCA TIME=
         B      EXIT2END       +24 SCANDITO TIME= PARM
         SPACE 3
*********************************************************************
*        CONTINUAZIONE JOB STMT: POSIZIONAMENTO PUNTATORI           *
*********************************************************************
INITCONT LR     R5,R4
CONTLOOP BXH    R5,R2,EXIT2ERR ERRORE SE HO FINITO SK
         CLI    0(R5),C'='     E' UN UGUALE?
         BE     KEYWCONT       SI, VAI A TRATTARE PAROLA CHIAVE
         CLI    0(R5),C','     E' UNA VIRGOLA?
         BNE    LASTTEST       NO, FAI ULTIMO TEST
         LR     R4,R5          R4 PUNTA ALLA VIRGOLA
         BAL    R9,CONTTEST    SE C'E' UNA CONTINUAZIONE ESCI
         B      CONTLOOP       ALTRIMENTI CONTINUA LOOP
LASTTEST CLI    0(R5),C' '     E' UN BLANK?
         BNE    CONTLOOP       NO, CONTINUA LOOP
         LR     R4,R5          R4 PUNTA AL BLANK
         B      INSRCONT       E SI INSERISCE SK CONT
       TITLE 'HASP USER EXIT 2 -- COSTANTS,ERROR AND NORMAL RETURN'
*********************************************************************
*        COSTANTI, SEGNALAZIONE ERRORE, RITORNO                     *
*********************************************************************
         DS     0F
FOUR     DC     F'4'
EIGHT    DC     F'8'
JOBKEY   DC     CL4'JOB '
TIMEKEYW DC     CL4'TIME'
EXIT2ERR ST     R15,NEXTPARM   0->NEXTPARM PER ERRORE IN CASO DI SK CONT
         LA     R15,8          8 COME CODICE DI RITORNO
EXIT2END $RETURN RC=(R15)
```

```
       TITLE 'HASP USER EXIT 2 -- SUBROUTINES (PICK----)'
*********************************************************************
*         PICKNBLN ROUTINE                                          *
*********************************************************************
PICKNBLN LR     R5,R4
NBLNLOOP BXH    R5,R2,EXIT2ERR ERRORE SE HO FINITO SK
         CLI    0(R5),C' '     E' UN BLANK?
         BNE    NBLNLOOP       NO, CONTINUA
         SR     R5,R4          IN R5
         BCTR   R5,0             LUNGHEZZA CAMPO
         BR     R9             E RITORNO
         SPACE 3
*********************************************************************
*         PICKACC1 ROUTINE                                          *
*********************************************************************
PICKACC1 LR     R5,R4
ACC1LOOP BXH    R5,R2,EXIT2ERR ERRORE SE HO FINITO SK
         CLI    0(R5),C','     E' UNA VIRGOLA?
         BNE    ACC1LOOP       NO, CONTINUA
         SR     R5,R4          SI, ALLORA
         BCTR   R5,0           IN R5 LUNGHEZZA CAMPO
         BR     R9             E RITORNO
         SPACE 3
*********************************************************************
*         PICKACCT ROUTINE                                          *
*********************************************************************
PICKACCT LR     R5,R4
ACCTLOOP BXH    R5,R2,EXIT2ERR ERRORE SE HO FINITO SK
         CLI    0(R5),C','     E' UNA VIRGOLA?
         BE     ACCTEND        SI, TERMINA
         CLI    0(R5),C')'     E' UNA PARENTESI TONDA CHIUSA?
         BNE    ACCTLOOP       NO, CONTINUA
ACCTEND  SR     R5,R4          IN R5
         BCTR   R5,0             LUNGHEZZA CAMPO
         BR     R9             ERRORE
         SPACE 3
*********************************************************************
*         PICKKEYW ROUTINE                                          *
*********************************************************************
PICKKEYW LR     R5,R4
KEYWLOOP BXH    R5,R2,EXIT2ERR ERRORE SE HO FINITO SK
         CLI    0(R5),C'='     E' UN UGUALE?
         BNE    KEYWLOOP       NO, CONTINUA
         SR     R5,R4          IN R5
         BCTR   R5,0             LUNGHEZZA CAMPO
         BR     R9             E RITORNO
         SPACE 4
*********************************************************************
*         PICKTIME ROUTINE                                          *
*********************************************************************
PICKTIME LR     R5,R4
TIMELOOP BXH    R5,R2,TIMEEND  SALTA SE HO FINITO SK
         CLI    0(R5),C','     E' UNA VIRGOLA?
         BE     TIMEEND        SI, TERMINA
         CLI    0(R5),C' '     E' UN BLANK?
         BNE    TIMELOOP       NO, CONTINUA
TIMEEND  SR     R5,R4          IN R5
         BCTR   R5,0             LUNGHEZZA CAMPO
```

```
         BR    R9
        TITLE 'HASP USER EXIT 2 -- SUBROUTINES (TEST....)'
*********************************************************************
*        ACCTTEST ROUTINE                                           *
*********************************************************************
ACCTTEST LTR   R5,R5          TEST SULLA LUNGHEZZA DEL CAMPO
         BZ    EXIT2ERR       ERRORE SE E'ZERO
         S     R5,FOUR        SOTTRAI 4 ALLA LUNGHEZZA DEL CAMPO
         BP    EXIT2ERR       ERRORE SE IL CAMPO E' PIU' LUNGO DI 4
         A     R5,FOUR        RIPRISTINA CONTENUTO REGISTRO 5
         BR    R9             ALTRIMENTI RITORNA
         SPACE 3
*********************************************************************
*        NUMTEST  ROUTINE                                           *
*********************************************************************
NUMTEST  LR    R7,R5          IN R7 LUNGHEZZA CAMPO
NUMLOOP  CLI   1(R4),C'0'     E' UN CHAR MINORE DI 0?
         BM    EXIT2ERR       SE SI, ERRORE
         CLI   1(R4),C'9'     E' UN CHAR MAGGIORE DI 9?
         BH    EXIT2ERR       SE SI, ERRORE
         LA    R4,1(,R4)      INCREMENTA PUNTATORE
         BCT   R7,NUMLOOP     DECREMENTA CONTATORE E CONTINUA
         SR    R4,R5          SE TERMINATO, RIPRISTINA PUNTATORE
         BR    R9             E RITORNA
         SPACE 3
*********************************************************************
*        CONTTEST ROUTINE                                           *
*********************************************************************
CONTTEST CLI   0(R4),C','     C'E' ","?
         BNE   CONTEND        SE NO, RITORNA
         CR    R4,R3          SIAMO A COLONNA 71?
         BE    EXIT2END       SE SI, VAI A TRATTAMENTO CONTINUAZIONE
         CLI   1(R4),C' '     C'E' ", "?
         BE    EXIT2END       SE SI, VAI A TRATTAMENTO CONTINUAZIONE
CONTEND  BR    R9             ALTRIMENTI RITORNA
         SPACE 3
*********************************************************************
*        ENDTEST  ROUTINE                                           *
*********************************************************************
ENDTEST  CR    R4,R3           R4 > R3?
         BH    MOVECONT        SI, SPOSTA ULTIMO PARM E SCHEDA CONTIN.
         CLI   0(R4),C' '      SE C'E' UN BLANK
         BE    INSRCONT           ALLORA SCHEDA CONTINUAZIONE
         BR    R9              ALTRIMENTI RITORNA
         TITLE 'HASP USER EXIT 2 -- SUBROUTINES (----SKIP)'
*********************************************************************
*        SKIPBLAN ROUTINE                                           *
*********************************************************************
SKIPBLAN BXH   R4,R2,EXIT2ERR  ERRORE SE HO FINITO SK
         CLI   0(R4),C' '      E' UN BLANK?
         BE    SKIPBLAN        SI, CONTINUA
         BCTR  R4,0            NO, PUNTATORE SU ULTIMO CHAR BLK
         BR    R9              E RITORNO
         SPACE 3
*********************************************************************
*        SKIPPGMN ROUTINE                                           *
*********************************************************************
SKIPPGMN BXH   R4,R2,EXIT2ERR  SALTA SE HO FINITO SK
```

```
         CLI   0(R4),C''''      E' UN APICE?
         BNE   SKIPPGMN         NO, VAI A VEDERE IL PRCSSIMC CARATTERE
         CLI   1(R4),C''''      SI, CE NE E' ANCORA UNO DOPO?
         BNE   PGMEND           NO, TERMINA
         LA    R4,1(R4)         ALTRIMENTI SONO DUE GLI APICI
         B     SKIPPGMN         E QUINDI SI DEVE CONTINUARE LA RICERCA
PGMEND   BR    R9               RITORNC
         SPACE   3
*********************************************************************
*        SKIPPARM ROUTINE                                           *
*********************************************************************
SKIPPARM LR    R5,R4            INIZIALIZZA R5 A PUNTARE ALL'UGUALE
PARMLOOP BXH   R4,R2,PARMEND SALTA SE HO FINITO SK
         CLI   0(R4),C' '       E' UN BLANK?
         BE    PARMEND          SI, HO FINITO SK
         CLI   0(R4),C','       E' UNA VIRGOLA?
         BE    COMMFOUN         SI, SALTA
         CLI   0(R4),C'='       E' UN UGUALE?
         BNE   PARMLOOP         NO, CONTINUA CICLO
         LR    R4,R5            R4 PUNTA ALL'ULTIMA VIRGOLA TROVATA
PARMEND  BCTR  R4,0             ANZI, PRIMA
         BR    R9               E RITORNO
COMMFOUN CR    R4,R3            SONO ARRIVATO AL 71-SIMO CHAR DELLA SK?
         BE    PARMEND          SE SI, TERMINA
         CLI   1(R4),C' '       C'E' UN BLANK DOPO LA VIRGOLA?
         BE    PARMEND          SI, HO FINITO SK
         LR    R5,R4            NO, SALVA PUNTATORE LOCALE A VIRGOLA
         LR    R8,R4            E PUNTATORE GLOBALE
         B     PARMLOOP         INDI CONTINUA RICERCA
         TITLE 'HASP USER EXIT 2 -- SUBROUTINES (----CONT)'
*********************************************************************
*        CREACONT ROUTINE                                           *
*********************************************************************
CREACONT L     R5,4(R1)         R5 PUNTA AL BYTE RXITFLAG DI PCE
         OI    0(R5),RXCCRDSW   INDICA CHE LA EXIT HA CREATO SK CONT
         L     R5,8(R1)         R5 PUNTA AREA IN CUI INSERIRE SK CONT
         MVI   0(R5),C'/'       SLASH NEL PRIMO CHAR
         MVI   1(R5),C'/'       SLASH NEL SECONDO
         MVI   2(R5),C' '       BLANK NEL TERZO CHAR
         MVC   3(77,R5),2(R5)   BLANK IN TUTTO IL RESTC DELLA SK
         LA    R5,3(R5)         R5 PUNTA PRIMO CHAR UTILE DI SK CONT
         BR    R9               RITORNO
         SPACE 3
*********************************************************************
*        LASTCONT ROUTINE                                           *
*********************************************************************
LASTCONT SR    R3,R8            IN R3 LUNGHEZZA CAMPO DA MUOVERE
         BCTR  R3,0             MENO UNO PER EXECUTE
         EX    R3,MOVELAST      MUOVI CAMPO IN SK CONTINUAZIONE
         AR    R5,R3            AGGIORNA R5
         MVI   1(R5),C','       INSERISCI ',' DOPO CAMPO MOSSO
         LA    R5,2(R5)         R5 PUNTA PRIMO CHAR UTILE SK CONT
         MVI   1(R8),C' '       BLANK SU PRIMO CHAR DEL CAMPO MOSSO
         EX    R3,MOVEBLNK      BLANK SU TUTTO IL CAMPO
         BR    R9               E RITORNO
MOVELAST MVC   0(1,R5),1(R8)  R5 PUNTA SK CONT E R8 ULTIMC CAMPO SK CCRR
MOVEBLNK MVC   2(1,R8),1(R8)  R8 PUNTA PRIMO CHAR ULTIMO CAMPO
         SPACE 3
```

```
*********************************************************************
*        TIMECONT ROUTINE                                           *
*********************************************************************
TIMECONT MVI   0(R4),C','
         MVI   1(R4),C' '          ', ' IN SK CORRENTE
         MVC   0(24,R5),TIMETEXT   TIME=XXXX IN SK CONTINUAZIONE
         L     R7,ESTMTIME         IN R7 TEMPO STIMATO FORMATO BINARIO
         CVD   R7,CONVTEMP         IN CONVTEMP TEMPO STIMATO IN PACKED
         UNPK  5(4,R5),CONVTEMP(8) IN XXXX TEMPO STIMATO FORMATO ZONED
         OI    8(R5),X'F0'         IN XXXX TEMPO STIMATO FORMATO CHAR
         BR    R9                  E RITORNO
TIMETEXT DC    CL24'TIME=XXXX GENERATED CARD'
         TITLE 'HASP USER EXIT 2 -- SUBROUTINES (........)'
*********************************************************************
*        NUMCONV  ROUTINE                                           *
*********************************************************************
NUMCONV  BCTR  R5,0               DECREMENTA R5 PER EXECUTE
         EX    R5,PACKTIME       CONVERTI DA ZONED IN PACKED
         CVB   R0,CONVTEMP       CONVERTI DA PACKED IN BINARIO
         LA    R5,1(R5)          RIPRISTINA R5
         BR    R9                RITORNA: IN R0 IL VALORE IN BINARIO
PACKTIME PACK  CONVTEMP(8),1(,R4)  R4 PUNTA CHAR PRIMA CAMPO NUMERICO
         SPACE 3
*********************************************************************
*        SAVEINDX ROUTINE                                           *
*********************************************************************
SAVEINDX LA    R6,4(R6)          INCREMENTA INDICATORE PARAMETRI SCANDITI
         ST    R6,NEXTPARM       SALVA INDICATORE PARAMETRI SCANDITI
         BR    R9                E RITORNA
         SPACE 5
         DROP  R12
         $MODEND
         END
```


EEEEE X X I TTTTT 4
E X X I T 4
EEE X I T 4 4
E X X I T 444444
EEEEE X X I T 4

```
         TITLE 'HASP USER EXIT 4 -- MODULE COMMENT BLOCK'
**** JES2 EXIT4 ************************************************************
*                                                                           *
*     SCOPO:                                                                *
*     SCANDIRE STMT DI CONTROLLO DEL JES2 SIA PER RACCOGLIERE INFORM.  *
*     UTILI ALLO SCHEDULING CHE PER INIBIRE L'USO DI CERTI PARAMETRI.  *
*     IN PARTICOLARE, VENGONO CONTROLLATI I SEGUENTI JES2 STATEMENT:   *
*     JOBPARM - SI INIBISCE L'USO DEI PARAMETRI COPIES, LINES, ROOM E *
*               TIME                                                        *
*     MESSAGE - VIENE SALVATO IL TESTO DEL MESSAGGIO NEL CAMPO          *
*               JCTXWRK DELLA JCT E SI CONTROLLA LA PRESENZA DI          *
*               RICHIESTE DI NASTRI PER PLOTTER                           *
*     SETUP   - SI CONTROLLANO LE RICHIESTE DI NASTRI:                  *
*               TPE9,TPV9,TPH9,TPE7                                       *
*                                                                         *
*     ENTRY POINT = UEXIT4                                                *
*                                                                         *
*     REGISTRI IN INPUT:                                                  *
*        R0     =0 INDICA UNO STATEMENT DI CONTROLLO DEL JES2            *
*               =4 INDICA UNO STATEMENT DEL JCL                           *
*        R1     INDIRIZZO DI UN'AREA COMPOSTA DA 3 WORD:                 *
*               WORD1 (+0) PUNTA AL CONTROL STATEMENT IMAGE BUFFER       *
*               WORD2 (+4) PUNTA ALL'EXIT FLAG BYTE, RXITFLAG, NEL PCE   *
*               WORD3 (+8) PUNTA AL CAMPO JCTXWRK DELLA JCT              *
*        R2-9   N/A                                                       *
*        R10    INDIRIZZO DELLA JCT DEL JOB                               *
*        R11    INDIRIZZO DELLA HCT DEL JES2                              *
*        R12    N/A                                                       *
*        R13    INDIRIZZO DEL PCE                                         *
*        R14    INDIRIZZO DI RITCRNO                                      *
*        R15    ENTRY ADDRESS                                             *
*                                                                         *
*     REGISTRI IN OUTPUT:                                                 *
*        R0-14 DOVREBBERO CONTENERE I VALORI CHE AVEVANO IN INPUT         *
*        R15  = 0 OPPURE 12                                               *
*                                                                         *
*     CODICE DI RITORNO = 0 OPPURE 12 (IN R15)                            *
*                                                                         *
*     CONTENUTO REGISTRI:                                                 *
*        R2     PASSO SCANSIONE SK IN INPUT (R2=1)                        *
*        R3     PUNTA ALL'ULTIMO CHAR DELLA SK IN INPUT                   *
*        R4     PUNTA AL CHAR CORRENTE DELLA SK                           *
*        R5     TEMPORANEO                                                *
*        R6     INTERO INDICANTE PUNTO SCANSIONE CUI SI E' GIUNTI         *
*        R7     TEMPORANEO IN ALCUNE SUBROUTINE                           *
*        R8     PUNTA ALL'ULTIMA VIRGOLA INCONTRATA DURANTE SCANSIONE     *
*        R9     INDIRIZZO DI RITORNO PER LINK DELLE SUBROUTINE            *
*        R12    REGISTRO BASE                                             *
*                                                                         *
*     CAMPI JCT RISERVATI UTENTE UTILIZZATI :                             *
*        JCTUSER1    BYTE1 (+0) NUMERO TPH9 RICHIESTI                     *
*                    BYTE2 (+1) NUMERO TPE9 RICHIESTI                     *
*                    BYTE3 (+2) NUMERO TPE7 RICHIESTI                     *
*                    BYTE4 (+3) FLAG PER LA RICHIESTA DEL PLOTTER         *
*        JCTUSER2    BYTE1 (+0) FLAG PRESENZA  /*MESSAGE IN JOB          *
*****************************************************************************
```

```
         TITLE 'HASP USER EXIT 4 -- PROLOG (HASP GLOBALS)'
         COPY  $HASPGBL              COPY HASP GLOBALS
         TITLE 'HASP USER EXIT 4 -- PROLOG ($MODULE)'
HASPU4   $MODULE NOTICE=SP133,                                          C
               SYSP=(NOGEN,NOGEN,NODATA,NOGEN,NOGEN),                   C
               $BUFFER,              REQUIRED BY MACRO MCDULE           C
               $CAT,                 REQUIRED BY HCT                    C
               $DCT,                 REQUIRED BY HCT                    C
               $HASPEQU,             GENERATE HASP EQUATES              C
               $HCT,                 REQUIRED BY MACRO SAVE AND RETURN  C
               $JCT,                 JOB CONTROL TABLE DSECT            C
               $JQE,                 REQUIRED BY HCT                    C
               $MIT,                 REQUIRED BY MACRO MODEND           C
               $PCE,                 REQUIRED BY HCT                    C
               $PDDB                 REQUIRED BY HCT
         TITLE 'HASP USER EXIT 4 -- PROLOG (EQUATES)'
*************************************************************************
*                                                                       *
*        CAMPI JCT RISERVATI UTENTE UTILIZZATI :                        *
*        JCTUSER1   PER CONTENERE LE VARIE RICHIESTE DEL JOB            *
*                                                                       *
*************************************************************************
TPH9NUMB EQU    JCTUSER1,1            BYTE PER NUMERO TPH9 RICHIESTI
TPE9NUMB EQU    JCTUSER1+1,1          BYTE PER NUMERO TPE9 RICHIESTI
TPE7NUMB EQU    JCTUSER1+2,1          BYTE PER NUMERO TPE7 RICHIESTI
PLOTFLAG EQU    JCTUSER1+3,1          BYTE PER RICHIESTA PLOTTER
MSSGFLAG EQU    JCTUSER2,1            FLAG PRESENZA /*MESSAGE
         TITLE 'HASP USER EXIT 4 -- ENTRY POINT'
*************************************************************************
*                                                                       *
*        INIZIO EXIT4 DEL JES2:                                         *
*        VERIFICA SE STMT JES2 JOBPARM,MESSAGE O SETUP                  *
*                                                                       *
*************************************************************************
UEXIT4   $ENTRY CSECT=YES,BASE=R12
         $SAVE
         LR     R12,R15          CARICA REGISTRO BASE
         SR     R15,R15          ZERO COME CODICE DI RITORNO
         LTR    R0,R0            STATEMENT DI CONTROLLO DEL JES2?
         BNZ    EXIT4END         NO, ALLORA TERMINA
         L      R4,0(,R1)        R4 PUNTA ALLA / DEL JES2 STMT
         CLC    2(7,R4),JOBPARM  E' UN JOBPARM STMT?
         BE     JOBPMNGM         SI, VAI A TRATTARLO
         CLC    2(7,R4),MESSAGE  E' UN MESSAGE STMT?
         BE     MSSGMNGM         SI, VAI A TRATTARLO
         CLC    2(5,R4),SETUP    E' UN SETUP STMT?
         BNE    EXIT4END         NO, ALLORA TERMINA
```

```
         TITLE 'HASP USER EXIT 4 -- SETUP STATEMENT PROCESSING'
*********************************************************************
*                                                                   *
*        JES2 SETUP STATEMENT                                       *
*                                                                   *
*                                                                   *
*********************************************************************
         LA    R4,7(R4)          R4 PUNTA AL CHAR DOPO /*SETUP
         CLI   0(R4),C' '        C'E' UN BLANK?
         BNE   EXIT4ERR          NO, ERRORE
         LA    R3,64(,R4)        R3 ALL'ULTIMO CARATTERE DELLA SK
         LA    R2,1              R2 CONTIENE IL PASSO DI SCANSIONE SK
         BAL   R9,SKIPBLAN       SKIP BLANKS E AGGIORNA PUNT SCANSIONE
         LA    R7,4              PREPARA R7 PER TEST LUNGH KEYWORD
NEXTPARM BAL   R9,NEXTKEYW       SELEZIONA KEYWORD
         LTR   R5,R5             TEST SULLA LUNGHEZZA DELLA KEYWORD
         BZ    EXIT4ERR          ERRORE SE E' ZERO
         SR    R5,R7             SECONDO TEST SULLA LUNGHEZZA
         LTR   R5,R5                                 DELLA KEYWORD
         BNZ   EXIT4ERR          ERRORE SE LUNGHEZZA DIVERSA DA 4
         LA    R4,1(,R4)         PUNT SCANSIONE SU PRIMO CHAR KEYWORD
         CLC   0(4,R4),TPV9      C'E' LA SPECIFICA DI UN NASTRO 1600?
         BE    NUMB1600          SI, VERIFICA QUANTI NE SONO RICHIESTI
         CLC   0(4,R4),TPE9      C'E' LA SPECIFICA DI UN NASTRO 1600?
         BE    NUMB1600          SI, VERIFICA QUANTI NE SONO RICHIESTI
         CLC   0(4,R4),TPH9      C'E' LA SPECIFICA DI UN NASTRO 6250?
         BE    NUMB6250          SI, VERIFICA QUANTI NE SONO RICHIESTI
         CLC   0(4,R4),TPE7      C'E' LA SPECIFICA DI UN NASTRO 7-TRK?
         BNE   EXIT4ERR          NO, ERRORE
NUMB7TRK LA    R4,4(,R4)         PUNT SCANSIONE SU =
         BAL   R9,TESTNUMB       RICHIESTA CORRETTA?
         IC    R5,TPE7NUMB       R5 <- NUM RICHIESTE GIA' SPECIFICATE
         BAL   R9,GLOBPERF       R5 <- NUM RICHIESTE TOTALI
         STC   R5,TPE7NUMB       TPE7NUMB <- R5
         B     NEXTPARM          CONTINUA
NUMB1600 LA    R4,4(,R4)         PUNT SCANSIONE SU =
         BAL   R9,TESTNUMB       RICHIESTA CORRETTA?
         IC    R5,TPE9NUMB       R5 <- NUM RICHIESTE GIA' SPECIFICATE
         BAL   R9,GLOBPERF       R5 <- NUM RICHIESTE TOTALI
         STC   R5,TPE9NUMB       TPE9NUMB <- R5
         B     NEXTPARM          CONTINUA
NUMB6250 LA    R4,4(,R4)         PUNT SCANSIONE SU =
         BAL   R9,TESTNUMB       RICHIESTA CORRETTA?
         IC    R5,TPH9NUMB       R5 <- NUM RICHIESTE GIA' SPECIFICATE
         BAL   R9,GLOBPERF       R5 <- NUM RICHIESTE TOTALI
         STC   R5,TPH9NUMB       TPH9NUMB <- R5
         B     NEXTPARM          CONTINUA
TPE7     DC    CL4'TPE7'
TPE9     DC    CL4'TPE9'
TPV9     DC    CL4'TPV9'
TPH9     DC    CL4'TPH9'
SETUP    DC    CL5'SETUP'
         DS    0H
         EJECT
```

```
*********************************************************************
*        TESTNUMB ROUTINE                                           *
*********************************************************************
TESTNUMB CLI     1(R4),C'1'        E' UNA CIFRA >= 1?
         BL      EXIT4ERR          NO,ERRORE
         CLI     1(R4),C'9'        E' UNA CIFRA <= 9?
         BH      EXIT4ERR          NO, ERRORE
         BR      R9                RITORNO
         SPACE   3
*********************************************************************
*        GLOBPERF ROUTINE                                           *
*********************************************************************
GLOBPERF SR      R8,R8             AZZERA R8
         IC      R8,1(R4)          R8 <- NUM RICHIESTE ORA  SPECIFICATE
         SLL     R8,28
         SRL     R8,28             IN FORMATO BINARIO
         AR      R5,R8             R5 = R5 + R8
         BR      R9                RITORNO
```

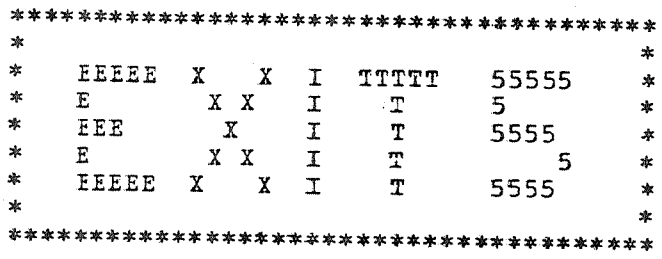
```
         TITLE 'HASP USER EXIT 4 -- MESSAGE STATEMENT PROCESSING'
*********************************************************************
*                                                                   *
*        JES2 MESSAGE STATEMENT                                     *
*                                                                   *
*********************************************************************
MSSGMNGM L       R5,8(R1)          R5 PUNTA AL CAMPO JCTXWRK DELLA JCT
         LA      R4,9(R4)          R4 PUNTA AL CHAR DOPO /*MESSAGE
         CLI     0(R4),C' '        E' UN BLANK?
         BNE     EXIT4ERR          NO, ERRORE
         MVI     0(R5),C' '        BLANK
         MVC     1(79,R5),0(R5)          NELLA JCTXWRK
         MVC     0(5,R5),MSSGINIT /*MSG ALL'INIZIO
         MVI     MSSGFLAG,X'01'    SETTA FLAG PRESENZA MESSAGGIO
         LA      R3,62(,R4)        R3 PUNTA ULTIMO CARATTERE DELLA SK
         LR      R7,R3             R7 ANCHE (SI PREPARA PER EXECUTE)
         LA      R2,1              R2 CONTIENE IL PASSO DI SCANSIONE SK
         BAL     R9,SKIPBLAN       SKIP BLANKS E AGGIORNA PUNT SCANSIONE
         SR      R7,R4             IN R7 LUNGHEZZA MESSAGGIO
         BCTR    R7,0              MENO 1 PER EXECUTE
         EX      R7,MOVEMSSG       MUOVI MESSAGGIO IN JCTXWRK
TEST1600 CLC     1(4,R4),TPE9      C'E' LA SPECIFICA DI NASTRI TPE9?
         BE      PLOTVERI          SI, VERIFICA SE E' PER PLOTTER
         CLC     1(4,R4),TPV9      C'E' LA SPECIFICA DI NASTRI TPV9?
         BE      PLOTVERI          SI, VERIFICA SE E' PER PLOTTER
         CLC     1(4,R4),TPH9      C'E' LA SPECIFICA DI NASTRI TPH9?
         BE      NEXTTAPE          SI, VAI ALLA PROSSIMA SPECIFICA
         CLC     1(4,R4),TPE7      C'E' LA SPECIFICA DI NASTRI TPE7?
         BNE     EXIT4END          NO, TERMINA
NEXTTAPE BAL     R9,SKIPTAPE       SKIPPA RICHIESTA NASTRO TPH9 O TPE7
TESTNEXT LA      R4,1(R4)          R4 SU CHAR SUCCESSIVO A )
         CLI     0(R4),C','        C'E' UNA VIRGOLA?
         BE      TEST1600          SI,VERIFICA SE SPECIFICATI TPE9
         B       EXIT4END          NO, TERMINA
PLOTVERI CLI     5(R4),C'='        C'E' UN UGUALE?
         BNE     EXIT4ERR          ERRORE SE MANCA
         CLI     6(R4),C'('        C'E' UNA APERTA TONDA?
         BNE     EXIT4ERR          ERRORE SE MANCA
         LA      R4,6(R4)          R4 PUNTA AL CHAR PRECEDENTE
TESTPLOT CLC     1(5,R4),PLOT      RICHIESTA DI NASTRO PER PLOTTER?
         BE      PLOTFOUN          EUREKA!
         CLC     1(5,R4),CALC      RICHIESTA DI NASTRO PER PLOTTER?
         BE      PLOTFOUN          EUREKA!!
         BAL     R9,NEXTFILD       SKIPPA CAMPO
         CLI     0(R4),C','        TERMINA CON UNA VIRGOLA?
         BE      TESTPLOT          SI, VERIFICA RICHIESTA PLOTTER
         B       TESTNEXT          VERIFICA SE SPECIFICATI ALTRI NASTRI
PLOTFOUN MVI     PLOTFLAG,X'01'    SETTA FLAG DEL PLOTTER
         B       EXIT4END          E TERMINA
MOVEMSSG MVC     6(1,R5),1(R4)     R4 PUNTA CHAR MSSG MENO 1, R5 JCTXWRK
PLOT     DC      CL5'$PLOT'
CALC     DC      CL5'$CALC'
MSSGINIT DC      CL5'/*MSG'
MESSAGE  DC      CL7'MESSAGE'
         DS      0H
```

```
         EJECT
*****************************************************************
*        SKIPTAPE ROUTINE                                       *
*        INPUT :  R4 PUNTA AL CHAR  PRECEDENTE UN PARM DELLA MESSAGE  *
*        OUTPUT:  R4 PUNTA ALL'ULTIMO CHAR DEL PARAMETRO        *
*        ESCAPE:  TERMINA SE NON TROVA )                        *
*****************************************************************
SKIPTAPE BXH   R4,R2,EXIT4END   TERMINA SE HC FINITO SK
         CLI   0(R4),C')'       E' UNA TONDA CHIUSA?
         BNE   SKIPTAPE         NO, CONTINUA
         BR    R9               RITORNO
         SPACE 3
*****************************************************************
*        NEXTFILD ROUTINE                                       *
*        INPUT :  R4 PUNTA AL CHAB  PRECEDENTE UN PARM DELLA MESSAGE  *
*        OUTPUT:  R4 PUNTA AL CHAR DOPO IL CAMPO E CIOE' A , O A )    *
*        ESCAPE:  TERMINA SE NON TROVA NE' ) NE' ,              *
*****************************************************************
NEXTFILD BXH   R4,R2,EXIT4END   TERMINA SE HO FINITO SK
         CLI   0(R4),C')'       E' UNA TONDA CHIUSA?
         BE    NEXTEND          SI, TERMINA
         CLI   0(R4),C','       E' UNA VIRGOLA?
         BNE   NEXTFILD         NO, CONTINUA
NEXTEND  BR    R9               RITORNO
```

```
         TITLE 'HASP USER EXIT 4 -- JOBPARM STATEMENT PROCESSING'
*****************************************************************
*                                                               *
*        JES2 JOBPARM STATEMENT                                 *
*                                                               *
*****************************************************************
JOBPMNGM LA      R4,8(,R4)          R4 PUNTA ALLA M DI JOBPARM
         CLI     1(R4),C' '         C'E' UN BLANK DOPO JOBPARM?
         BNE     EXIT4ERR           NO, ERRORE
         LA      R3,62(,R4)         R3 ALL'ULTIMO CARATTERE DELLA SK
         LA      R2,1               R2 CONTIENE IL PASSO DI SCANSIONE SK
         BAL     R9,SKIPBLAN        SKIP BLANKS E AGGIORNA PUNT SCANSIONE
         LA      R7,3               PREPARA R7 PER TEST LUNGH KEYWORD
CONTINUE BAL     R9,NEXTKEYW        SELEZIONA KEYWORD
         LTR     R5,R5              TEST SULLA LUNGHEZZA DELLA KEYWORD
         BZ      EXIT4ERR           ERRORE SE E' ZERO
         LA      R4,1(,R4)          PUNT SCANSIONE SU PRIMO CHAR KEYWORD
         LR      R6,R5              SALVA LUNGHEZZA KEYWORD
         BCTR    R5,0               PRIMO TEST SULLA LUNGHEZZA
         LTR     R5,R5                                 DELLA KEYWORD
         BZ      KEYWONE            VAI A FARE TEST PER LUNGHEZZA 1
         SR      R5,R7              SECONDO TEST SULLA LUNGHEZZA
         LTR     R5,R5                                 DELLA KEYWORD
         BZ      KEYWFOUR           VAI A FARE TEST PER LUNGHEZZA 4
         BCTR    R5,0               TERZO TEST SULLA LUNGHEZZA
         LTR     R5,R5                                 DELLA KEYWORD
         BZ      KEYWFIVE           VAI A FARE TEST PER LUNGHEZZA 5
         BCTR    R5,0               QUARTO TEST SULLA LUNGHEZZA
         LTR     R5,R5                                 DELLA KEYWORD
         BZ      KEYWSIX            VAI A FARE TEST PER LUNGHEZZA 6
         AR      R4,R6              ALTRIMENTI PUNTATORE SCANSIONE SU =
         B       CONTINUE           E VAI AL SUCCESSIVO PARAMETRO
         EJECT
*****************************************************************
*        TEST SU PARAMETRI JOBPARM                              *
*****************************************************************
KEYWONE  CLI     0(R4),C'L'         E' L (LINES) LA KEYWORD?
         BE      EXIT4ERR           SI: ERRORE
         CLI     0(R4),C'N'         E' N (COPIES) LA KEYWORD?
         BE      EXIT4ERR           SI: ERRORE
         CLI     0(R4),C'R'         E' R (ROOM) LA KEYWORD?
         BE      EXIT4ERR           SI: ERRORE
         CLI     0(R4),C'T'         E' T (TIME) LA KEYWORD?
         BE      EXIT4ERR           SI: ERRORE
         AR      R4,R6              ALTRIMENTI PUNTATORE SCANSIONE SU =
         B       CONTINUE           E VAI AL SUCCESSIVO PARAMETRO
KEYWFOUR CLC     0(4,R4),TIMEKEYW   E' TIME LA KEYWORD?
         BE      EXIT4ERR           SI: ERRORE
         CLC     0(4,R4),ROOMKEYW   E' ROOM LA KEYWORD?
         BE      EXIT4ERR           SI: ERRORE
         AR      R4,R6              ALTRIMENTI PUNTATORE SCANSIONE SU =
         B       CONTINUE           E VAI AL SUCCESSIVO PARAMETRO
KEYWFIVE CLC     0(5,R4),LINEKEYW   E' LINES LA KEYWORD?
         BE      EXIT4ERR           SI: ERRORE
         AR      R4,R6              ALTRIMENTI PUNTATORE SCANSIONE SU =
         B       CONTINUE           E VAI AL SUCCESSIVO PARAMETRO
KEYWSIX  CLC     0(6,R4),COPIKEYW   E' COPIES LA KEYWORD?
         BE      EXIT4ERR           SI: ERRORE
```

```
         AR      R4,R6             ALTRIMENTI PUNTATORE SCANSIONE SU =
         B       CONTINUE          E VAI AL SUCCESSIVO PARAMETRO
TIMEKEYW DC      CL4'TIME'
ROOMKEYW DC      CL4'ROOM'
COPIKEYW DC      CL6'COPIES'
LINEKEYW DC      CL5'LINES'
JOBPARM  DC      CL7'JOBPARM'
         DS      0H
```

```
         TITLE 'HASP USER EXIT 4 -- SUNBROUTINES'
*****************************************************************
*        NEXTKEYW ROUTINE                                       *
*        INPUT :  R4 PUNTA AL BLANK PRECEDENTE IL PRIMO PARM O A UN = *
*        OUTPUT:  R4 CONTINUA A PUNTARE AL BLANK PRECEDENTE     *
*                    IL PRIMO PARM, OPPURE ALLA VIRGOLA PRECEDENTE *
*                    UN PARAMETRO CON UN =                      *
*                 R5 SPECIFICA LUNGHEZZA PARAMETRO              *
*        ESCAPE:  SE SI INCONTRA UN BLANK (FINE PARAMETRI)      *
*****************************************************************
NEXTKEYW LR    R6,R4            INIZIALIZZA R6 A R4 (PUNTA A BLANK O =)
KEYWLOOP BXH   R4,R2,EXIT4END   SALTA SE HO FINITO STMT
         CLI   0(R4),C'='       C'E' UN UGUALE?
         BE    KEYWEND          SI, TERMINA CICLO
         CLI   0(R4),C','       E' UNA VIRGOLA?
         BNE   LASTTEST         NO, FAI L'ULTIMO TEST
         LR    R6,R4            SI, SALVA PUNTATORE LOCALE A VIRGOLA
         B     KEYWLOOP         E CONTINUA RICERCA
LASTTEST CLI   0(R4),C' '       C'E' UN BLANK?
         BE    EXIT4END         SI, HO FINITO JES2 STMT
         B     KEYWLOOP         NO, CONTINUA CICLO
KEYWEND  LR    R5,R4            R5 PUNTA A =
         LR    R4,R6            R4 PUNTA A VIRGOLA O BLANK
         SR    R5,R6            IN R5
         BCTR  R5,0                  LUNGHEZZA KEYWORD
         BR    R9               E RITORNO
         SPACE 2
*****************************************************************
*        SKIPBLAN ROUTINE                                       *
*        INPUT :  R4 PUNTA AL CHAR  PRECEDENTE UN BLANK         *
*        OUTPUT:  R4 PUNTA ALL'ULTIMO BLANK PRECEDENTE IL PRIMO  CHAR *
*                    NON BLANK                                  *
*        ESCAPE:  ERRORE SE SI TROVANO SOLO BLANK SULLA SCHEDA  *
*****************************************************************
SKIPBLAN BXH   R4,R2,EXIT4ERR  ERRORE SE HO FINITO SK
         CLI   0(R4),C' '      E' UN BLANK?
         BE    SKIPBLAN        SI, CONTINUA
         BCTR  R4,0            NO, PUNTATORE SU ULTIMO CHAR BLK
         BR    R9              E RITORNO
         TITLE 'HASP USER EXIT 4 -- ABEND AND NORMAL RETURN'
*****************************************************************
*                                                               *
*        FINE EXIT4: CORRETTAMENTE O PER ERRORE IN JES2 STMT    *
*                                                               *
*****************************************************************
EXIT4ERR LA    R15,12          CODICE DI RITORNO CHE SEGNALA ERRORE
EXIT4END $RETURN RC=(R15)
         DROP  R12
         $MODEND
         END
```


EEEEE X X I TTTTT 55555
E X X I T 5
EEE X I T 5555
E X X I T 5
EEEEE X X I T 5555

```
         TITLE 'HASP USER EXIT 5 -- PRCLOG (MODULE COMMENT BLOCK)'
**** JES2 EXIT5 ************************************************************
*                                                                           *
*                                                                           *
*     SCOPO:                                                                *
*     ESEGUIRE I COMANDI NON STANDARD $X E $Y.                              *
*                                                                           *
*     SINTASSI COMANDI:                                                     *
*                                                                           *
*         ] J [                   ] J ]                                     *
*     $Y  ] S [  n (-m) [, ([ S ]) n (-m)) -----                            *
*         ] T [                   ] T ]                                     *
*                                                                           *
*     $X  Jn (-m) [, (J) n (-m)) -----                                      *
*                                                                           *
*     SIGNIFICATO SIMBOLI METASINTATTICI:                                   *
*     ] [     E' POSSIBILE SCEGLIERE UNO SOLO FRA I SIMBOLI INDICATI        *
*     ( )     PARAMETRO OPZIONALE CHE PUO' MANCARE                          *
*     ----    IL PARAMETRO OPZIONALE PUO' ESSERE RIPETUTO PIU' VOLTE        *
*      n      SEQUENZA DI CIFRE DECIMALI                                    *
*      m      SEQUENZA DI CIFRE DECIMALI                                    *
*                                                                           *
*     SEMANTICA COMANDI:                                                    *
*     $Y    SI INFORMA L'OPERATORE DELLA QUANTITA' DI STAMPA E PUNCH        *
*           GIA' EFFETTUATA DAL/I JOB IN QUESTIONE                          *
*                                                                           *
*                                                                           *
*     $X    SI INFORMA L'OPERATORE DEL CODICE, TEMPO, LINEE DI STAMPA,      *
*           SCHEDE DA PUNCHARE, FORM, NUMERO DI COPIE, E NASTRI             *
*           RICHIESTI DAL/I JOB. INOLTRE VIENE DISPLAYATA L'ULTIMA          *
*           /*MESSAGE PRESENTE NEL JOB.                                     *
*                                                                           *
*****************************************************************************
         EJECT
```

```
*************************************************************************
*                                                                       *
*     ENTRY POINT = UEXIT5                                              *
*                                                                       *
*     REGISTRI IN INPUT:                                                *
*        R0-4   N/A                                                     *
*        R5     PUNTA AD UN' AREA LA CUI PRIMA PAROLA PUNTA AL PRIMO    *
*               OPERANDO DEL COMANDO                                    *
*        R6     =4 VALORE DI INCREMENTO                                 *
*        R7     PUNTA ALL' ULTIMA PAROLA DELL' AREA LA QUALE PUNTA      *
*               ALL'ULTIMO OPERANDO DEL COMANDO                         *
*        R8-10  N/A                                                     *
*        R11    INDIRIZZO DELLA HCT DEL JES2                            *
*        R12    N/A                                                     *
*        R13    INDIRIZZO DEL PCE                                       *
*        R14    INDIRIZZO DI RITORNO                                    *
*        R15    ENTRY ADDRESS                                           *
*                                                                       *
*     REGISTRI IN OUTPUT:                                               *
*        R0-14  DOVREBBERO CONTENERE I VALORI CHE AVEVANO IN INPUT      *
*        R15    =0 CONTINUA LA NORMALE ELABORAZIONE DEL COMANDO         *
*               =8 TERMINA LA ELABORAZIONE STANDARD DEL COMANDO         *
*                  ED ESEGUE LA MACRO $CRET PER RITORNARE IL CONTROLLO  *
*                  AL MAIN COMMAND PROCESSOR; I COMMAND SUBPROCESSORS   *
*                  VENGONO BYPASSATI.                                   *
*                                                                       *
*     CODICE DI RITORNO = 0 OPPURE 8 (IN R15)                           *
*                                                                       *
*     CONTENUTO REGISTRI DURANTE ESECUZIONE EXIT:                       *
*        R0-1   UTILIZZATI DA VARIE MACRO                               *
*        R3     FLAG PER ASSERIRE COMAND X (R3=0) O Y (R3=4)            *
*        R4     TEMPORANEO PER VARI SCOPI                               *
*        R5-7   VEDI INPUT                                              *
*        R8     PUNTA ALL'AREA IN CUI E' CONTENUTO UN JQE               *
*        R9     REGISTRO DI LINK PER CHIAMATE A SUBROUTINE              *
*        R10    PUNTA ALL'AREA IN CUI E' CONTENUTA LA JCT               *
*        R11    INDIRIZZO DELLA HCT DEL JES2                            *
*        R12    REGISTRO BASE                                           *
*        R13    VEDI INPUT                                              *
*        R14-15 UTILIZZATI DA VARIE MACRO COME LINK E RETURN REGISTER   *
*                                                                       *
*     CAMPI JCT UTILIZZATI :                                            *
*        JCTXWRK    CONTIENE TESTO EVENTUALE ULTIMA /*MESSAGE           *
*        JCTUSER0   TEMPO STIMATO DI ESECUZIONE DEL JOB                 *
*        JCTUSER1   BYTE1 (+0) NUMERO TPH9 RICHIESTI                    *
*                   BYTE2 (+1) NUMERO TPE9 RICHIESTI                    *
*                   BYTE3 (+2) NUMERO TPE7 RICHIESTI                    *
*                   BYTE4 (+3) FLAG PER LA RICHIESTA DEL PLOTTER        *
*        JCTUSER2   BYTE1 (+0) FLAG PRESENZA /*MESSAGE                  *
*                                                                       *
*************************************************************************
```

```
         TITLE 'HASP USER EXIT 5 -- PROLOG (LOCAL MACRO)'
         MACRO -- $CFCVB -- CONVERT TO BINARY
&NAME    $CFCVB &TYPE=CALL,&POINTER=(R1),&NUM=2,&INFO=NO,&NOK=,        C
               &MAX=9999
         LCLC  &R
         LCLA  &CNT
&R       SETC  '&SYSNDX'
         AIF   ('&INFO' EQ '' OR '&INFO' EQ 'NO').Z
         AIF   ('&TYPE' EQ 'RES').NOJECT
         EJECT
.NOJECT  SPACE 2
*********************************************************************
*                                                                   *
*         COFCVB -- CONVERT TO BINARY                               *
*                                                                   *
*         ROUTINE CONVERTS A PAIR OF NUMBERS OF THE FORM            *
*                                                                   *
*         TEXIN1-N2 WHERE                                           *
*         TEXT  = OPTIONAL TEXT IDENTIFIERS  JOB, PRT, RM ETC.      *
*         N1    = FIRST OF A SERIES OF NUMBERS LT &MAX IN VALUE     *
*         N2    = OPTIONAL LAST OF A SERIES OF NUMBERS LT &MAX.     *
*         IN A SERIES OF VALUES, N1-N2-N3-N4, THE LAST TWO VALUES   *
*         ARE CONSIDERED TO BE START-STOP VALUES.                   *
*         THE MEANING OF THE START-STOP VALUES FOR EXAMPLE 1-5 ARE  *
*         THE USER DESIRES AN OPERATION PERFORMED ON JOB OR FACILITY*
*         TYPE INDICATED, NUMBERS 1, 2, 3, 4, AND 5.                *
*                                                                   *
*         NO $WAITS ARE ISSUED.                                     *
*                                                                   *
* REGISTERS USED                                                    *
*         R0    = ACCUMULATOR - STOP VALUE                          *
*         R1    = ADDRESS OF OPERAND POINTER -- START VALUE         *
*         R14   = LINK REGISTER                                     *
*         R15   = WORK REGISTER, UPON RETURN, CONTAINS THE ADDRESS  *
*                    OF THE CHARACTER THAT STOPPED THE CONVERSION   *
*                                                                   *
* EXITS                                                             *
*         R14+2 OPERAND DOES NOT CONTAIN NUMERIC OR NUMERIC TOO LARGE*
*         R14+6 NORMAL EXIT                                         *
*                                                                   *
* NOTES                                                             *
*         R1 VALUE IS EQUAL TO R0 IF USER REQUESTS NUM=2 AND ONLY   *
*         ONE VALUE IS PRESENT. IF NUM=1 IS USED R1 VALUE IS        *
*         UNPREDICTABLE. ( NUM=1 MAY BE IGNORED ).                  *
*                                                                   *
*         MAX IS PASSED VIA AN INLINE PARAMETER LIST GENERATED      *
*         BY THE $CFCVB MACRO.                                      *
*                                                                   *
*********************************************************************
.Z       ANOP
&NAME    DS    0H
         AIF   ('&TYPE' NE 'CALL').INL
         $DECODE &POINTER
.CAL     BAL   R14,COFCVB&NUM     CONVERT NUMBERS TO BINARY
         AIF   ('&NOK' NE '').NOKOK
         MNOTE 4,'NOK KEYWORD MUST BE SUPPLIED'
         AGO   .XIT
.NOKOK   ANOP                                                  01470000
```

```
         DC    AL2(&MAX)                                          @133 01475000
         AIF   ('&NOK'(1,1) EQ '(').REG                              01480000
         B     &NOK                BRANCH IF OPERAND INVALID         01485000
         AGO   .XIT                                                  01490000
.REG     ANOP
&CNT     SETA  K'&NOK-2
&R       SETC  '&NOK'(2,&CNT)
         B     0(,&R)              BRANCH IF INVALID OPERAND
         MEXIT
.INL     L     R15,0(0,R1)         PICK UP FIRST PARAMETER
COF&R.K  CLI   0(R15),C'0'         CHARACTER NUMERIC...
         BNL   COF&R.L             IF NUMERIC, CONVERT
         LA    R15,1(0,R15)        NEXT CHARACTER
         CLM   R15,7,5(R1)         DID WE GO TOO FAR
         BL    COF&R.K             LOOP IF NO
         B     2(,R14)            RETURN INVALID OPERAND
         EJECT
*********************************************************************
*        CONVERT PAIR OF START STOP VALUES                          *
*********************************************************************
         SPACE 1
COF&R.L  CLI   0(15),C'9'          CHECK IF REALLY NUMERIC
         BH    2(,R14)             INVALID OPERAND IF NOT
         SLR   R0,R0               ZERO ACCUMULATOR, STOP VALUE
         ST    R0,$DOUBLE          CLEAR WORK AREA
         AIF   ('&NUM' NE '2').BA
         LNR   R1,R15              SET NEGATIVE FLAGS
.BA      ANOP
COF&R.C  MVN   $DOUBLE+1(1),0(R15) MOVE NUMERIC TO WORK AREA
         MH    R0,=H'10'           MULTIPLY BY TEN FOR TEN'S DIGIT
         AH    R0,$DOUBLE          ADD LOW DIGIT
         CH    R0,0(,R14)          LOOK FOR MAXIMUM IN INLINE LIST
         BH    2(,R14)             RETURN INVALID OPERAND
COF&R.N  LA    R15,1(0,R15)        NEXT CHARACTER
         CLI   0(R15),C'0'         TEST FOR NUMERIC ( FA-FF INCLUDED )
         BNL   COF&R.C             IF NUMERIC, CONVERT THE CHARACTER
         AIF   ('&NUM' NE '2').G
         CLI   0(R15),C'-'         LOOK FOR SEPARATOR
         BE    COF&R.O             BR IF YES TO CONVERT NEXT VALUE
         CLR   R1,R0               TEST FOR BOTH VALUES
         BNH   6(,R14)             RETURN WITH CONVERTED VALUES
         LR    R1,R0               SET TO SAME
         B     6(,R14)             RETURN WITH CONVERTED VALUES
COF&R.O  CLI   1(R15),C'0'         NEXT CHARACTER NUMERIC...
         BL    2(,R14)             RETURN INVALID OPERAND IF NO
         LR    R1,R0               SET START OF STRING
         SLR   R0,R0               PREPARE FOR NEXT
         B     COF&R.N             CONVERT NEXT VALUE OF SET
         AGO   .I
.G       ANOP
         B     6(,R14)             RETURN
.I       ANOP
.XIT     MEND
```

```
         TITLE 'HASP USER EXIT 5 -- PROLOG (HASP GLOBALS)'
         COPY  $HASPGBL              COPY HASP GLOBALS
         TITLE 'HASP USER EXIT 5 -- PROLOG ($MODULE)'
HASPU5   $MODULE NOTICE=(SP120,SP133),                                  C
               SYSP=(NOGEN,NOGEN,NODATA,NOGEN,NOGEN),                   C
               RESPA,              REQUIRED BY COMWORK                  C
               $BUFFER,            REQUIRED BY MACRO MODULE             C
               $CAT,               REQUIRED BY HCT                      C
               $COMWORK,           REQUIRED BY COMMAND PROCESSING       C
               $DCT,               REQUIRED BY HCT                      C
               $HASPEQU,           GENERATE HASP EQUATES                C
               $HCT,               REQUIRED BY MACRO SAVE AND RETURN    C
               $JCT,               JOB CONTROL TABLE DSECT              C
               $JOE,               REQUIRED BY COMWORK                  C
               $JQE,               REQUIRED BY HCT                      C
               $MIT,               REQUIRED BY MACRO MODEND             C
               $PCE,               REQUIRED BY HCT                      C
               $PDDB               REQUIRED BY HCT
         TITLE 'HASP USER EXIT 5 -- PROLOG (EQUATES)'
*********************************************************************
*        CAMPI JCT RISERVATI UTENTE UTILIZZATI :                    *
*        JCTUSER1   PER CONTENERE LE VARIE RICHIESTE DEL JOB        *
*********************************************************************
ESTMTIME EQU    JCTUSER0             WORD PER TEMPO ESECUZIONE JOB
TAPENUMB EQU    JCTUSER1             WORD PER NUMERO TAPE RICHIESTI
MSSGFLAG EQU    JCTUSER2,1           BYTE PER PRESENZA /*MESSAGE
         SPACE 3
*********************************************************************
*        EQUATES USATE PER LA SELEZIONE DEL TIPO DI JOE             *
*********************************************************************
COFJOBE  EQU   X'C0'             LIST BATCH TYPE JOBS WHEN ZERO
COFSTCE  EQU   X'40'             LIST STC TYPE JOBS
COFTSUE  EQU   X'80'             LIST TSU TYPE JOBS
         SPACE 3
*********************************************************************
*        COMMULOP FLAG SETTING                                      *
*********************************************************************
CCJFJOBF EQU   X'80'             JOB FOUND FLAG
         SPACE 3
*********************************************************************
*        DEFINITIONS FOR MESSAGE DEFINED IN COMMAND AREA            *
*********************************************************************
COFJOB   EQU   COMMAND,3           TEXT 'JOB', 'STC', OR 'TSU'
COFJNO   EQU   COFJOB+3,5          JOB NUMBER WITH LEADING BLANK
COFJNAME EQU   COFJNO+6,8          JOB NAME
COFQUE   EQU   COFJNAME+9,8        TEXT 'AWAITING'
COFQX    EQU   COFQUE+9,9          TEXT 'EXECUTION'
```

```
         TITLE 'HASP USER EXIT 5 -- ENTRY POINT'
*********************************************************************
*                                                                   *
*        INIZIO EXIT5: TEST PER SAPERE SE COMANDO X O Y             *
*                                                                   *
*********************************************************************
UEXIT5   $ENTRY CSECT=YES,BASE=R12
         $SAVE
         LR    R12,R15           CARICA REGISTRO BASE
         SLR   R15,R15           ZERO COME CODICE DI RITORNO
         TM    COMINCON,X'80'    SSI FORMATTED COMMAND?
         BNZ   EXIT5END          SI, TERMINA
         CLI   COMVERB,C'X'      COMANDO X?
         BE    SETX              SI, VAI A TRATTARLO
         CLI   COMVERB,C'Y'      COMANDO Y?
         BNE   EXIT5END          NO, TERMINA
         LA    R3,4              SETTA R3 A 4 (FLAG COMANDO Y)
         B     TESTY             VAI A TESTARE FORMATO COMANDO
SETX     SLR   R3,R3             SETTA R3 A 0 (FLAG COMANDO X)
         B     TESTXY            VAI A TESTARE FORMATO COMANDO
```

```
         TITLE 'HASP USER EXIT 5 -- OPERAND CONVERSION'
*********************************************************************
*        INIZIO ELABORAZIONE COMANDI X,Y: TEST SUL FORMATO          *
*********************************************************************
TESTY    CLI   COMOPRND,C'S'     STC?
         BE    CAJLOOP           SI, CONTINUA ELABORAZIONE
         CLI   COMOPRND,C'T'     TSU?
         BE    CAJLOOP           SI, CONTINUA ELABORAZIONE
TESTXY   CLI   COMOPRND,C'J'     JOB?
         BNE   ERR649            NO, ERRORE DI FORMATO DEL COMANDO
         SPACE 3
*********************************************************************
*                                                                   *
*        LOOP DI CONVERSIONE DEI PARAMETRI CON TEST SU FORMATO      *
*                                                                   *
*********************************************************************
CAJLOOP  L     R1,0(0,R5)        R1 PUNTA ALL'OPERANDO
         CLI   0(R1),C'0'        INIZIA CON UN NUMERICO?
         BNL   CAJLOOPC          SI, CONTINUA CONVERSIONE
         L     R4,COFJOBS        SETTA FLAG PER JOB BATCH
         CLI   0(R1),C'J'        C'E' UNA J COME PRIMA LETTERA?
         BE    CAJLOOPC          SI, CONVERTI
         L     R4,COFSTCS        SETTA FLAG PER STC
         CLI   0(R1),C'S'        C'E' UNA S COME PRIMA LETTERA?
         BE    CAJLOOPC          SI, CONVERTI
         L     R4,COFTSUS        SETTA FLAG PER TSU
         CLI   0(R1),C'T'        C'E' UNA T COME PRIMA LETTERA?
         BNE   ERR649            NO, ERRORE FORMATO COMANDO
CAJLOOPC CLI   1(R1),C'='        NON PERMETTERE J= O S= O T=
         BE    ERR650            ERRORE FORMATO OPERANDO
        $CFCVB POINTER=(R5),NOK=ERR650      CONVERTI IN BINARIO
         LTR   R1,R1             ZERO?
         BZ    ERR650            SI, ERRORE OPERANDO
         OR    R1,R4             SETTA TIPO
         STH   R0,2(0,R5)        MEMORIZZA VALORE FINALE
         STH   R1,0(0,R5)        MEMORIZZA VALORE INIZIALE
CAJNXTOP EXLE  R5,R6,CAJLOOP     COMPLETA PER TUTTI GLI OPERANDI
```

```
         TITLE 'HASP USER EXIT 5 -- MAIN LOOP'
*****************************************************************
*                                                               *
*        LOOP PER OTTENERE I JQE DELLA JOB LISI                 *
*                                                               *
*****************************************************************
         LA    R5,COMPNTER       RIPRISTINA R5
CAJLOOPA LH    R4,0(0,R5)        PRENDI PRIMA COPPIA DI JOB
         N     R4,COFCLRTY       PULISCI HI-ORDER FLAG BITS DEL JOBNUM
         SPACE 3
*****************************************************************
*        SCANSIONE JIX  PER AVERE IL SINGOLO JQE                *
*****************************************************************
         USING JQEDSECT,R8
CAJQLOC  $QLOC (R4)              OTTIENI IL JQE PER IL JOB NUM
         BZ    CAJNEXT           SALTA SE QLOC NON HA TROVATO IL JOB
         LR    R8,R1             CARICA IN R8 INDIRIZZO JQE
         TM    JQEFLAG3,QUEJOB   E' UN JOB BATCH...
         BNZ   CAJCKSTC          NO, SALTA
         TM    0(R5),COFJOBE     E' UN JOB BATCH RANGE...
         BZ    HITJOB            SI, SALTA
         B     CAJNEXT           SALTA PER OTTENERE IL PROSSIMO JOB
         SPACE 2
CAJCKSTC TM    JQEFLAG3,QUESTC   E' UN STC...
         BO    CAJCKTSU          NO, SALTA
         TM    0(R5),COFSTCE     E' UN STC JOB RANGE...
         BZ    HITSTC            SI, SALTA
         B     CAJNEXT           SALTA PER OTTENERE IL PROSSIMO JOB
         SPACE 2
CAJCKTSU TM    0(R5),COFTSUE     E' UN TSU JOB RANGE...
         BZ    HITTSU            SI, SALTA
         SPACE 2
CAJNEXT  LA    R4,1(,R4)         INDICE AL PROSSIMO JOE NUM
         CH    R4,2(,R5)         ELABORATI TUTTI I JOB NUM...
         BNH   CAJQLOC           NO, LOOP
         SPACE 2
CAJNEXTP BXLE  R5,R6,CAJLOOPA    VAI ALLA PROSSIMA COPPIA SE PRESENTE
         TM    COMNULOP,CCJFJCBF TEST SU FLAG JOB FOUND
         BO    EXITSTOP          TERMINA SE JOB TROVATO
         B     ERR610            ERRORE SE NON TROVATO NEANCHE UNO
```

```
         TITLE 'HASP USER EXIT 5 -- JCT SEARCH'
*********************************************************************
*        ESECUZIONE COMANDI: INIZIO PREPARAZIONE OUTPUT             *
*********************************************************************
HITSTC   LTR   R3,R3              COMANDO X?
         BNZ   ERR650             SI, ERRORE FORMATO OPERANDI
         MVC   COFJOB,=C'STC'     NO, SETTA STC
         B     HITCONT            CONTINUA
HITTSU   LTR   R3,R3              COMANDO X?
         BNZ   ERR650             SI, ERRORE FORMATO OPERANDI
         MVC   COFJOB,=C'TSU'     NO, SETTA TSU
         B     HITCONT            CONTINUA
HITJOB   MVC   COFJOB,=C'JOB'     SETTA JOB
HITCONT  OI    COMNULOP,CCJFJOBF  SETTA FLAG JOB FOUND
         SLR   R1,R1              AZZERA R1
         LH    R1,JQEJOBNO        IN R1 JOBNO IN BINARIO
         LA    R15,COFJNO+1       IN R15 PUNTATORE A CAMPO IN CUI --
         BAL   R9,CONVMOV         SI CONVERTE JOB NUMBER A EBCDIC
         MVI   COFJNO,C' '        UN BLANK ALL'INIZIO DEL CAMPO
         SPACE 3
*********************************************************************
*                                                                   *
*        ESECUZIONE COMANDI: RITROVAMENTO JCT                       *
*                                                                   *
*********************************************************************
        $GETBUF WAIT=YES          GET BUFFER FOR JCT
         BZ    ERRXXX             ERROR, NO BUFFER AVAILABLE
         ST    R1,PCEBUFAD        STORE BUFFER ADDRESS
         LR    R10,R1             R10 POINTS TO BUFFER ADDRESS
         MVI   PCEDEVTP,PCEDARD   SET READ
         L     R0,JQETRAK         LOAD JCT DASD ADDRESS
         ST    R0,PCESEEK         STORE IN PCE
        $JCTIO                    READ JCT
         BO    MOVEMSG            I/O OK
         LR    R1,R10             I/O ERROR
        $FREEBUF (R1)             FREE BUFFER
         B     ERRYYY             DISPLAY ERROR MESSAGE
```

```
         TITLE 'HASP USER EXIT 5 -- $Y EXECUTION'
*********************************************************************
*        ESECUZIONE COMANDI: BLANK IN AREA DI OUTPUT                *
*********************************************************************
MOVEMSG  MVI   COFJNAME,C' '          BLANK
         MVC   COFJNAME+1(79),COFJNAME   IN MSG AREA
         LTR   R3,R3                  QUALE COMANDO E'?
         BZ    XEXEC                  SE $X, SALTA
         SPACE 3
*********************************************************************
*                                                                   *
*        ESECUZIONE COMANDI: COSTRUZIONE OUTPUT DI $Y               *
*                                                                   *
*********************************************************************
         MVC   COFJNAME,JCTJNAME      MUOVE JOB NAME NEL MESSAGGIO
         MVC   COFQUE(3),=C'PR='      PR= IN MESSAGGIO
         L     R1,JCTLINES            NUMERO DI LINEE IN R1
         CVD   R1,COMDWORK            CONVERTI
         UNPK  COFQUE+3(6),COMDWORK(8)   IN EBCDIC
         OI    COFQUE+8,X'F0'
         MVC   COFQX+1(3),=C'PU='     PU= IN MESSAGGIO
         L     R1,JCTPUNCH            NUMERO DI SCHEDE PERFORATE IN R1
         CVD   R1,COMDWORK            CONVERTI
         UNPK  COFQX+4(6),COMDWORK(8)    IN EBCDIC
         OI    COFQX+9,X'F0'
         LA    R15,COFJOB+36          R15 PUNTA A FINE MESSAGGIO
         B     SENDMSG                INVIA MESSAGGIO
```

```
         TITLE 'HASP USER EXIT 5 -- $X EXECUTION'
*********************************************************************
*                                                                   *
*        ESECUZIONE COMANDI: COSTRUZIONE OUTPUT DI $X               *
*                                                                   *
*********************************************************************
XEXEC    MVC   COFJNAME(4),JCTROOMN    MUOVI ROOM NUMBER
         MVC   COFJNAME+5(2),=C'T='    T= IN MESSAGGIO
         LA    R15,COFJNAME+7          POSIZIONATI SUL PRIMO PARM
         L     R1,ESTMTIME             EXECUTION TIME IN R1
         BAL   R9,CONVMOV              CONVERTI IN EBCDIC E MUOVI
         MVC   0(3,R15),=C'PR='        PR= IN MESSAGGIO
         LA    R15,3(,R15)             POSIZIONATI SUL SECONDO PARM
         L     R1,JCTESTLN             NUMERO DI LINEE IN R1
         XR    R0,R0
         D     R0,=F'1000'             NUMERO DI PAGINE IN R1
         BAL   R9,CONVMOV              CONVERTI IN EBCDIC E MUOVI
         MVC   0(3,R15),=C'PU='        PU= IN MESSAGGIO
         LA    R15,3(,R15)             POSIZIONATI SU PARM SUCCESSIVO
         L     R1,JCTESTPU             NUMERO DI SCHEDE IN R1
         BAL   R9,CONVMOV              CONVERTI IN EBCDIC E MUOVI
         MVC   0(2,R15),=C'F='         F= IN MESSAGGIO
         MVC   2(4,R15),JCTFORMS       MUOVI FORMS IN MESSAGGIO
         MVC   7(4,R15),=C'CPY='       CPY= IN MESSAGGIO
         LA    R15,11(,R15)            POSIZIONATI SU PARM SUCCESSIVO
         XR    R1,R1
         IC    R1,JCTCPYCT             NUMERO DI COPIE IN R1
         BAL   R9,CONVMOV              CONVERTI IN EBCDIC E MUOVI
         MVC   0(4,R15),TAPENUMB       MUOVI RICHIESTE NASTRI
         OC    0(4,R15),=X'F0F0F0F0'   NUMERO STAMPABILE
         LA    R15,4(R15)              AGGIORNA PUNT FINE MSSG
         BAL   R9,OUTMSG               STAMPA RISPOSTA
         CLI   MSSGFLAG,X'00'          C'E' MESSAGGIO?
         BNE   XMSG                    SI, VAI A STAMPARLO
         MVC   COFJNAME(8),=C'NO /*MSG'     NIENTE MESSAGGIO
         LA    R15,COFJNAME+8          INDIRIZ. FINE MESSAGGIO
         B     SENDMSG                 VAI A INVIARE MESSAGGIO
XMSG     MVC   COFJNAME(80),JCTXWRK    COPIA MESSAGGIO DA JCTXWRK
         LA    R15,COFJNAME+79         INDIRIZ. FINE MSG IN R15
XELBL    BCTR  R15,0                   AGGIUSTA
         CLI   0(R15),C' '                INDIRIZ.
         BE    XELBL                         FINE
         LA    R15,1(,R15)                      MESSAGGIO
SENDMSG  BAL   R9,OUTMSG               INVIA MESSAGGIO
         LR    R1,R10                  INDIR. BUFFER IN R1
        $FREEBUF (R1)                  FREE JCT BUFFER
         B     CAJNEXT                 CONTINUA CICLO
```

```
         TITLE 'HASP USER EXIT 5 -- SUBROUTINES'
*********************************************************************
*        CONVMOV ROUTINE: INSERISCE IL CONTENUTO DI R1,CONVERTITO IN  *
*                         EBCIDIC, NELL'AREA PUNTATA DA R15.         *
*                         UTILIZZA COME AREA DI APPOGGIO COMDWORK.   *
*                         R9  LINK REGISTER.                         *
*********************************************************************
CONVMOV  CVD   R1,COMDWORK           CONVERTE IN DECIMALE
         UNPK  0(4,R15),COMDWORK(8)  CONVERTE
         OI    3(R15),X'F0'                IN EBCDIC
         LA    R15,5(,R15)           INCREMENTA R15
         BR    R9                    E RITORNA
         SPACE 3
*********************************************************************
*        OUTMSG  ROUTINE: STAMPA VIA $CWTO IL MESSAGGIO CONTENUTO IN  *
*                         COFJOB, IL CUI ULTIMO CARATTERE E' PUNTATO  *
*                         R15. UTILIZZA R0. R9  LINK REGISTER.       *
*********************************************************************
OUTMSG   LA    R14,COFJOB                R14 INDIRIZZO INIZIO MESSAGGIO
         SLR   R15,R14                   LUNG. MESSAGGIO IN R15
         LR    R0,R15                    LUNG. IN R0
         AR    R15,R14                   RIPRISTINA R15
        $CWTO  MSG=COFJOB,L=(R0),MSGID=608,JOB=YES  SCRIVI MESSAGGIO
         BR    R9                        RITORNO
         TITLE 'HASP USER EXIT 5 -- EPILOG (ERROR,RETURN AND COSTANTS)'
*********************************************************************
*                                                                   *
*        FINE EXIT5: STAMPA MESSAGGIO EVENTUALMENTE DI ERRORE       *
*                                                                   *
*********************************************************************
ERRXXX  $CWTO  MSG='          NO BUFFER AVAILABLE',L=28,                C
               MSGID=69,JOB=YES
         B     EXITSTOP          STAMPA ERRORE E TERMINA
         SPACE 2
ERRYYY  $CWTO  MSG='          I/O ERROR IN JCT SEARCH',L=32,            C
               MSGID=69,JOB=YES
         B     EXITSTOP          STAMPA ERRORE E TERMINA
         SPACE 2
ERR610  $CWTO  MSG='          JOB(S) NOT FOUND',L=25,MSGID=610,JOB=YES
         B     EXITSTOP          STAMPA ERRORE E TERMINA
         SPACE 2
ERR649   LA    R15,0             SEGNALA DI ESEGUIRE COMMAND PROCESSOR
         B     EXIT5END          TERMINA (IL JES RILEVERA' L'ERRORE)
         SPACE 2
ERR650  $CWTO  MSG='          INVALID OPERAND',L=24,MSGID=650,JOB=YES
         B     EXITSTOP          STAMPA ERRORE E TERMINA
         SPACE 2
EXITSTOP LA    R15,8            SEGNALA DI NON ESEGUIRE COMMAND PROCESSOR
EXIT5END $RETURN RC=(R15)       RITORNO
         SPACE 5
*********************************************************************
*        COSTANTI USATE PER LA SELEZIONE DEL TIPO DI JOB            *
*********************************************************************
         DS    0F
COFJOBS  DC    X'00000000'       LIST BATCH TYPE JOBS
COFSTCS  DC    X'00004000'       LIST STC TYPE JOBS
COFTSUS  DC    X'00008000'       LIST TSU TYPE JOBS
COFCLRTY DC    X'00003FFF'       CLEAR LIST TYPE FLAG
```

```
         TITLE 'HASP USER EXIT 5 -- (COFCVB) - CONVERT TO BINARY'
         PRINT GEN
COFCVB2  $CFCVB TYPE=RES,INFO=YES
         PRINT NOGEN
         EJECT
         LTORG
         DROP  R12,R8,R10
         $MODEND
         END
```

```
*********************************************
*                                           *
*   EEEEE  X   X  I  TTTTT     222   000    *
*   E       X X   I    T      2   2 0   0   *
*   EEE      X    I    T        2  0   0   *
*   E       X X   I    T       2   0   0   *
*   EEEEE  X   X  I    T      22222  000    *
*                                           *
*********************************************
```

```
         TITLE 'HASP USER EXIT 20 -- PROLOG (MODULE COMMENT BLOCK)'
**** JES2 EXIT20 ************************************************************
*                                                                             *
*     SCOPO:                                                                  *
*     ASSEGNARE AD UN JOB LA CLASSE.                                          *
*     SE UN JOB NON HA SPECIFICATO IL PARAMETRO CLASS= O SE LA CLASSE         *
*     SPECIFICATA E' UNA LETTERA <T, ALLORA LA CLASSE VIENE ASSEGNATA         *
*     SECONDO I SEGUENTI CRITERI:                                             *
*                                                                             *
*          ---------------------------------------------                      *
*          |PLOTTER| NASTRI | MINUTI CPU |CLASSE| HOLD |                      *
*          |       |        |  RICHIESTI |      |      |                      *
*          |-------|--------|------------|------|------|                      *
*          |       |        |    <=3     |   A  |  NO  |                      *
*          |       |   NO   | >3  E <=10 |   B  |  NO  |                      *
*          |       |        |    >10     |   C  |  NO  |                      *
*          |       |--------|------------|------|------|                      *
*          |  NO   | 1 O 2  |   <=10     |   D  |  NO  |                      *
*          |       | TPV9   |   > 10     |   E  |  NO  |                      *
*          |       |--------|------------|------|------|                      *
*          |       | 1 TPH9 |   <=10     |   F  |  NO  |                      *
*          |       |        |   > 10     |   H  |  SI  |                      *
*          |       |--------|------------|------|------|                      *
*          |       | VARI(*)|   <=10     |   G  |  SI  |                      *
*          |       |        |   > 10     |   H  |  SI  |                      *
*          |-------|--------|------------|------|------|                      *
*          |  SI   |   -    |     -      |   K  |  SI  |                      *
*          |-------|--------|------------|------|------|                      *
*                                                                             *
*                                                                             *
*     SE UN JOB HA SPECIFICATO LA CLASSE T O W, ALLORA SI DECIDE SOLO         *
*     SE ESSO DEBBA ESSERE MESSO IN HOLD:                                     *
*                                                                             *
*          ---------------------------------------                            *
*          | PLOTTER | NUMERO NASTRI | CLASSE | HOLD |                       *
*          |---------|---------------|--------|------|                       *
*          |         |      <3       |  T/W   |  NO  |                       *
*          |   NO    |---------------|--------|------|                       *
*          |         |      >=3      |  T/W   |  SI  |                       *
*          |---------|---------------|--------|------|                       *
*          |   SI    |       -       |  T/W   |  SI  |                       *
*          |---------|---------------|--------|------|                       *
*                                                                             *
*                                                                             *
*          (*) TPE7  OPPURE  TPH9 > 1  OPPURE  TPE9 > 2                       *
*                    OPPURE  1 TPH9 E 1 <= TPE9 <=2                           *
*          (-) NON  PRESO IN CONSIDERAZIONE                                   *
*                                                                             *
*******************************************************************************
         EJECT
```

```
*************************************************************************
*     ENTRY POINT = UEXIT20                                             *
*                                                                       *
*     REGISTRI IN INPUT:                                                *
*        R0    =0                                                       *
*        R1-9  N/A                                                      *
*        R10   INDIRIZZO DELLA JCT DEL JOB                              *
*        R11   INDIRIZZO DELLA HCT DEL JES2                             *
*        R12   N/A                                                      *
*        R13   INDIRIZZO DEL HASPRDR PCE                                *
*        R14   INDIRIZZO DI RITORNO                                     *
*        R15   ENTRY ADDRESS                                            *
*                                                                       *
*     REGISTRI IN OUTPUT:                                               *
*        R0-14 DOVREBBERO CONTENERE I VALORI CHE AVEVANO IN INPUT       *
*        R15   =4                                                       *
*                                                                       *
*     CODICE DI RITORNO = 0 IL JES2 DEVE CONTINUARE LA SUA ELABORA_     *
*                           ZIONE PASSANDO IL CONTROLLO AD EVENTUALI    *
*                           ALTRE ROUTINES ASSOCIATE A QUESTA EXIT      *
*                                                                       *
*     CONTENUTO REGISTRI DURANTE ESECUZIONE EXIT:                       *
*        R0      N/A                                                    *
*        R1      ACCUMULATORE PER CONTARE NUMERO NASTRI RICHIESTI       *
*        R2      TEMPORANEO                                             *
*        R3-9    N/A                                                    *
*        R10-11  VEDI INPUT                                             *
*        R12     REGISTRO BASE                                          *
*        R13     VEDI INPUT                                             *
*        R14-15  N/A                                                    *
*                                                                       *
*     CAMPI JCT RISERVATI UTENTE UTILIZZATI :                           *
*        JCTUSER0   WORD  CONTENENTE MINUTI CPU RICHIESTI               *
*        JCTUSER1   BYTE1 (+0) NUMERO TPH9 RICHIESTI                    *
*                   BYTE2 (+1) NUMERO TPE9 RICHIESTI                    *
*                   BYTE3 (+2) NUMERO TPE7 RICHIESTI                    *
*                   BYTE4 (+3) FLAG PER LA RICHIESTA DEL PLOTTER        *
*                                                                       *
*************************************************************************
```

```
         TITLE 'HASP USER EXIT 20 -- PROLOG (HASP GLOBALS)'
         COPY  $HASPGBL            COPY HASP GLOBALS
         TITLE 'HASP USER EXIT 20 -- PROLOG($MODULE)'
HASPU20  $MODULE NOTICE=SP133,                                          C
               SYSP=(NOGEN,NOGEN,NODATA,NOGEN,NOGEN),                   C
               $BUFFER,            REQUIRED BY MACRO MODULE             C
               $CAT,               REQUIRED BY HCT                      C
               $DCT,               REQUIRED BY HCT                      C
               $HASPEQU,           GENERATE HASP EQUATES                C
               $HCT,               REQUIRED BY MACROS SAVE AND RETURN   C
               $JCT,               JOB CONTROL TABLE DSECT              C
               $JQE,               REQUIRED BY HCT                      C
               $MIT,               REQUIRED BY MACRO MODEND             C
               $PCE,               REQUIRED BY HCT                      C
               $PDDB               REQUIRED BY HCT
         TITLE 'HASP USER EXIT 20 -- PROLOG (EQUATES)'
*************************************************************************
*                                                                       *
*        CAMPI JCT RISERVATI UTENTE UTILIZZATI :                        *
*        JCTUSER0   CONTIENE TEMPO STIMATO DI ESECUZIONE DEL JOB        *
*        JCTUSER1   CONTIENE VARIE RICHIESTE DEL JOB(NASTRI,PLOTTER)    *
*                                                                       *
*************************************************************************
ESTMTIME EQU    JCTUSER0            WORD PER TEMPO ESECUZICNE JOB
TPH9NUMB EQU    JCTUSER1,1          BYTE PER NUMERO TPH9 RICHIESTI
TPE9NUMB EQU    JCTUSER1+1,1        BYTE PER NUMERO TPE9 RICHIESTI
TPE7NUMB EQU    JCTUSER1+2,1        BYTE PER NUMERO TPE7 RICHIESTI
PLOTFLAG EQU    JCTUSER1+3,1        BYTE PER RICHIESTA PLOTTER
```

```
         TITLE 'HASP USER EXIT 20 -- ENTRY POINT'
*****************************************************************************
*                                                                           *
*        INIZIO EXIT20 DEL JES2                                             *
*                                                                           *
*****************************************************************************
UEXIT20  $ENTRY CSECT=YES,BASE=R12
         $SAVE
         LR    R12,R15                 CARICA REGISTRO BASE
         SR    R15,R15                 ZERO COME CODICE DI RITORNO
         CLI   JCTJOBID,C'J'           JOB BATCH?
         BNE   XIT20END                NO, NON FARE SCHEDULAZIONE
         CLI   JCTJCLAS,C'T'           CLASSE MINORE DI T?
         BL    RCLMS                   SI, CONTROLLA RICHIESTE
         TITLE 'HASP USER EXIT 20 -- JOB CLASS DEFINITION (T/W CLASS)'
*****************************************************************************
*                                                                           *
*        CLASSE MAGGIORE OD UGUALE A T: DECIDE SE LASCIARE IN HOLD          *
*                                                                           *
*****************************************************************************
         CLI   PLOTFLAG,X'00'          RICHIESTO PLOTTER?
         BH    XIT20END                SI, LASCIA IN HOLD E TERMINA
         SLR   R1,R1                   AZZERA R1
         SLR   R2,R2                   AZZERA R2
         IC    R1,TPH9NUMB             NUMERO DI TPH9 IN R1
         IC    R2,TPE9NUMB             NUMERO DI TPE9/TPV9 IN R2
         AR    R1,R2                   IN R1 NUMERO DI TPE9 + TPH9
         IC    R2,TPE7NUMB             NUMERO DI TPE7 IN R2
         AR    R1,R2                   IN R1 NUM. TOTALE NASTRI RICHIESTI
         CL    R1,=F'3'                RICHIESTI MENO DI 3 NASTRI?
         BNL   XIT20END                NO, LASCIA IN HOLD E TERMINA
NOHOLD   NI    JCTJBOPT,255-JCTSETUP   SI, TOGLI HOLD
         B     XIT20END                E TERMINA
```

```
         TITLE 'HASP USER EXIT 20 -- JOB CLASS DEFINITION (K/F/H CLASS)'
*********************************************************************
*                                                                   *
*        CLASSE MINORE DI T                                         *
*                                                                   *
*********************************************************************
RCLMS    CLI   PLOTFLAG,X'00'        CHIESTI NASTRI PLOTTER?
         BH    RCLK                  SI, CLASSE K
         CLI   TPE7NUMB,X'00'        CHIESTI NASTRI A 7 PISTE?
         BH    RCLGH                 SI, CLASSE G,H
         CLI   TPH9NUMB,X'01'        CHIESTO PIU' DI UN TPH9?
         BH    RCLGH                 SI, CLASSE G,H
         CLI   TPE9NUMB,X'02'        CHIESTI PIU' DI 2 TPE/TPV?
         BH    RCLGH                 SI, CLASSE G,H
         CLI   TPE9NUMB,X'00'        1 O 2 TPE9/TPV9?
         BH    RCLTAP                SI, CLASSE D,E,G,H
         CLI   TPH9NUMB,X'01'        1 TPH9?
         BM    RCLNOTAP              NO, VAI A SOLO CPU
         SPACE 2
*********************************************************************
*                                                                   *
*        SOLO UN TPH9  (CLASSE F OD H)                              *
*                                                                   *
*********************************************************************
         MVI   JCTJCLAS,C'H'         CIASSE H
         MVI   JCTCLASS,C'H'         CLASSE H
         CLC   ESTMTIME,=F'10'       PIU' DI 10 MINUTI?
         BH    XIT20END              SI, LASCIA IN HOLD E TERMINA
         MVI   JCTJCLAS,C'F'         AITRIMENTI CLASSE F
         MVI   JCTCLASS,C'F'         ALTRIMENTI CLASSE F
         B     NOHOLD                TOGLI DA HOLD E TERMINA
         SPACE 2
*********************************************************************
*                                                                   *
*        RICHIESTA DI PLOTTER  (CLASSE K)                           *
*                                                                   *
*********************************************************************
RCLK     MVI   JCTJCLAS,C'K'         METTI CLASSE K
         MVI   JCTCLASS,C'K'         METTI CLASSE K
         B     XIT20END              LASCIA IN HOLD E TERMINA
```

```
        TITLE 'HASP USER EXIT 20 -- JOB CLASS DEFINITION (OTHER CLASS)'
*********************************************************************
*                                                                   *
*         NIENTE NASTRI (CLASSE A O B O C)                          *
*                                                                   *
*********************************************************************
RCLNOTAP MVI   JCTJCLAS,C'C'          ASSUMI CLASSE C
         MVI   JCTCLASS,C'C'          ASSUMI CLASSE C
         CLC   ESTMTIME,=F'10'        CPU PIU' DI 10 MINUTI?
         BH    XIT20END               SI, FINITO
         MVI   JCTJCLAS,C'B'          ASSUMI CLASSE B
         MVI   JCTCLASS,C'B'          ASSUMI CLASSE B
         CLC   ESTMTIME,=F'3'         PIU' DI 3 MINUTI?
         BH    XIT20END               SI, FINITO
         MVI   JCTJCLAS,C'A'          ALTRIMENTI CLASSE A
         MVI   JCTCLASS,C'A'          ALTRIMENTI CLASSE A
         B     XIT20END               TERMINA (NON E' IN HOLD)
         SPACE 2
*********************************************************************
*                                                                   *
*         UNO O DUE TPE9/TPV9  (CLASSE E OD F O G O H)              *
*                                                                   *
*********************************************************************
RCLTAP   CLI   TPH9NUMB,X'00'         CHIESTI ANCHE TPH9?
         BH    RCLGH                  SI VAI A IMPOSTARE CLASSE G,H
         MVI   JCTJCLAS,C'E'          ASSUMI CLASSE E
         MVI   JCTCLASS,C'E'          ASSUMI CLASSE E
         CLC   ESTMTIME,=F'10'        CPU PIU' DI 10 MINUTI?
         BH    NOHOLD                 TOGLI DA HOLD E TERMINA
         MVI   JCTJCLAS,C'D'          ALTRIMENTI CLASSE D
         MVI   JCTCLASS,C'D'          ALTRIMENTI CLASSE D
         B     NOHOLD                 TOGLI DA HOLD E TERMINA
         SPACE 2
*********************************************************************
*                                                                   *
*         TPE7 OPPURE TPH9>1 OPPURE TPE9>2 OPPURE TPH9=1 E 1<=TPE9<=2  *
*                    (CLASSE G O H)                                 *
*                                                                   *
*********************************************************************
RCLGH    MVI   JCTJCLAS,C'H'          ASSUMI CLASSE H
         MVI   JCTCLASS,C'H'          ASSUMI CLASSE H
         CLC   ESTMTIME,=F'10'        PIU' DI 10 MINUTI?
         BH    XIT20END               SI, FINITO (LASCIA IN HOLD)
         MVI   JCTJCLAS,C'G'          ALTRIMENTI CLASSE G
         MVI   JCTCLASS,C'G'          ALTRIMENTI CLASSE G
         TITLE 'HASP USER EXIT 20 -- RETURN'
*********************************************************************
*                                                                   *
*         FINE EXIT 20                                              *
*                                                                   *
*********************************************************************
XIT20END $RETURN RC=(R15)
         LTORG
         DROP  R12
         $MODEND
         END
```