```
                        ;***COPYRIGHT 1969, DIGITAL EQUIPMENT CORP., MAYNARD, MASS.***


                        ;THIS SUB-PROGRAM ASSEMBLED WITH SYSTEM PARAMETER FILE - S.MAC(V414)
                                XLIST
                                LIST
                        ;THIS SUB-PROGRAM ASSEMBLED WITH CONFIGURATION DEPENDENT FEATURE SWITCHES - FT50SB,MAC(
                        V003)
                                XLIST
                                LIST
                        TITLE CORE1 - LOGICAL AND PHYSICAL CORE ALLOCATION ROUTINES - V414
                        SUBTTL T. HASTINGS/TH/RCC  TS  04 JUN 69
                        XP VCORE1,414+
                                                ;THIS MACRO PUTS VERSION NO. IN STORAGE MAP AND GLOB

                                ENTRY CORE1     ;ALWAYS LOAD CORE1(FOR LIB SEARCH)


                        ;CORE ALLOCATION IS DONE ON A 1K BLOCK BASIS

                        ;USING A USE BIT TABLE(CORTAB) WHICH HAS A 1
                        ;FOR EVERY BLOCK WHICH IS NOT AVAILABLE BECAUSE:
                        ; 1.IN USE BY MONITOR
                        ; 2.IN USE BY USER
                        ; 3.NON-EXISTANT MEMORY
                        ;0 MEANS BLOCK IS NOT IN USE

                        ;WHEN THE SYSTEM IS STARTED, SYSINI SETS THE CORTAB TABLE
                        ;IT ALSO SETS A BYTE POINTER(CORLST) WHICH POINTS TO THE
                        ;LOWEST NON-EXISTANT BLOCK IMMEDIATELY ABOVE THE HIGHEST
                        ;EXISTANT BLOCK. IT ALSO SETS CORTAL TO THE NO. OF
                        ;FREE BLOCKS AVAILABLE.
                        ;THE CORE1 ROUTINE ASSIGNS CORE IF POSSIBLE, SETS THE USE BITS,
                        ;AND MOVES THE JOB IF NEW ASSIGNMENT IS A DIFFERENT PLACE THAN OLD
                        ;THE JBTADR TABLE IS ALSO UPDATED BY THE CORE ROUTINES
                        ;LH=PROTECTION,RH=RELOCATION
                        ;JOBADR IS MODIFIED IF CORE FOR CURRENT JOB
                        ;HARDWARE RELOC. AND PROTEC. ARE RESET IF CURRENT JOB
                        ;FINALLY JOBREL(PROTECTION) IN JOB DATA AREA IS ALWAYS UPDATED

                        ;LIST OF GLOBALS AFFECTED:
                        ;JBTADR,CORTAL,CORTAB,HOLEF,SHFWAT,JOBADR

                        ;ACS USED(BESIDES TAC,TAC1,JDAT,IOS,DEVDAT,AND PDP)
               000012   BLK=BUFPNT       ;HIGHEST REL. ADR. IN USER AREA
               000013   LOC=BUFWRD       ;ABS. LOC. OF FIRST BLOCK IN USER AREA
               000015   T=AC1    ;TEMPORARY
               000016   T1=AC2   ;"
```

```
                        ;CORE UUO
                        ;CALL:  MOVEI AC,HIGHEST REL. ADR. DESIRED IN LOW SEG
                        ;       HRLI AC,HIGHEST REL. ADR. DESIRED IN HIGH SEG
                        ;       CALL AC,[SIXBIT /CORE/]
                        ;       ERROR RETURN
                        ;       OK RETURN TO USER, JOB MOVED IF NECESSARY
                        ;RETURN NO. OF FREE 1K BLOCKS IN AC(OR MAX. NO. BLOCKS ALLOWED IF SWAPPING SYS)
                        ;BOTH HALVES Ø MEANS ERROR RETURN NO. OF FREE 1K BLOCKS(OR MAX. NO. OF BLOCKS
                        ; ALLOWED IF SWAPPING SYSTEM) IMMEDIATELY WITHOUT AFFECTING CORE
                        ; OR WAITING FOR IO DEVICES
                        ;LH=Ø MEANS DO NOT CHANGE HIGH SEG ASSIGNMENT
                        ;RH=Ø MEANS DO NOT CHANGE LOW SEG ASSIGNMENT

                        INTERNAL CORUUO
                        EXTERNAL USRREL,JOB,CORTAL
                        EXTERNAL ESTOP1,IOWAIT,OERROR,SETREL,STOTAC,WSCHED

000000  322040  000016' CORUUO: JUMPE TAC,ŻERCOR        ;IS HE ASKING FOR ZERO CORE?
000001  261140  000001          PUSH PDP,TAC            ;NO, SAVE HIGHEST DESIRED ADDRESS IN BOTH SEGS
000002  260140  000000          PUSHJ PDP,IOWAIT        ;WAIT FOR ALL DEVICE INACTIVE
000003  550043  000000          HRRŻ TAC,(PDP)          ;HIGHEST REL. LOC. DESIRED FOR LOW SEG(RH)
000004  322040  000010'        JUMPE TAC,CORU1         ;IS RH Ø(IF YES DO NOT CHANGE LOW SEG)?
000005  435040  001777          IORI TAC,1777           ;NO, MAKE IT AN EVEN MULTIPLE OF 1K-1
000006  260140  000126'        PUSHJ PDP,CORE1         ;TRY TO ASSIGN CORE
000007  254000  000014'        JRST CORERR             ;NOT AVAILABLE, ERROR RETURN
                                                        ; SUM OF NEW LOW AND OLD HIGH SEG TOO BIG
000010  554043  000000  CORU1:  HLRŻ TAC,(PDP)         ;CHECK TO SEE IF USER IS REQUESTING HIGH CORE
                        IFE FT2REL,<
                                CAMG TAC,USRREL         ;Ø MEANS NO CHANGE, 1 THRU TOP OF LOW SEG MEANS
                                                        ; RETURN HIGH CORE (IF ANY), ALWAYS LEGAL
                                                        ; EVEN IF NO HIGH SEG SHARP WARE OR SOFTWARE
                        >
                        IFN FT2REL,<
                                EXTERN UCORHI
000011  260140  000000          PUSHJ PDP,UCORHI        ;TRY TO ASSIGN CORE FOR HIGH SEG,
                                                        ; UCORHI EXPECTS ARG ON PD LIST
000012  254000  000014'        JRST CORERR             ;ERROR-ACTIVE IO(SAVE IN PROGRESS FOR SOME USER)
                                                        ; OR SUM OF NEW LOW SEG AND NEW HIGH SEG TOO BIG
                        >
000013  350003  777777          AOS -1(PDP)             ;SET FOR OK(SKIP) RETURN
000014  262140  000001  CORERR: POP PDP,TAC            ;REMOVE ARG FROM LIST
                        IFN FTSWAP,<
000015  260140  000000          PUSHJ PDP,WSCHED        ;CALL SCHEDULER TO STOP JOB
                                                        ; IN CASE LOW SEG MUST BE SWAPPED OUT TO EXPAND
                                                        ; OR HIGH SEG LOGICAL CORE ASSIGNED ON DISK
                                                        ; SAVE ALL ACS EXCEPT AC1,AC2,AC3,
                        >
000016                  ŻERCOR:
                        IFE FTSWAP,<
                                MOVE TAC,CORTAL         ;RETURN NO. OF FREE 1K BLOCKS (COUNTING
                                                        ; DORMANT AND IDLE SEGMENTS AS FREE)
                        >
                        IFN FTSWAP,<
```

```
101  000016  200040  000000          MOVE TAC,CORMAX          ;RETURN MAX. NO. 1K BLOCKS ALLOWED FOR 1 USER TO HAVE
102  000017  242040  777766          LSH TAC,-12
103                             >
104  000020  254000  000000          JRST STOTAC              ;STORE IN USER AC AND RETURN TO USER
```

```
                        ;ROUTINE TO CHECK JOBS TO SEE IF ANY JOB CAN BE SHUFFLED
                        ;IT IS CALLED EVERY 60TH OF A SECOND BY CLOCK ROUTINE
                        ;PROVIDING CURRENT JOB IS IN USER MODE OR JUST ENTERING
                        ;IO WAIT OR SHARABLE DEVICE WAIT OR RETURNING ON UUO CALLS
                        ;IE CURRENT JOB AND ALL OTHER JOB ARE SHUFFABLE WITH RESPECT
                        ;TO RELOCATION INFO IN MONITOR
                        ;SINCE RESCHEDULING IS NEVER DONE WHEN CURRENT JOB
                        ;IS NOT SHUFFABLE, ALL JOBS ARE SHUFFABLE WHEN
                        ;CHKSHF IS CALLED.
                        ;THE NOSHUFFLE MACRO SHOULD STILL BE USED WHEN EVER
                        ;THE MONITOR MOVES RELOCATION OUT OF ACS PDP,PROG, OR JDAT
                        ;IN CASE IT SHOULD PROVE DESIRABLE IN THE FUTURE TO SHUFFLE
                        ;MORE FREQUENTLY THAN AT THE ABOVE TIMES
                        ;FINALLY A JOB MUST HAVE ALL DEVICES INACTIVE(SINCE SOME
                        ;OF THEM USE ABSOLUTE ADDRESSES)BEFORE IT CAN BE MOVED
                        ;SO CORE CANNOT BE REASSIGNED WHILE DEVICES ARE ACTIVE
                        ;IF DEVICES ARE ACTIVE, JOB WILL BE STOPPED SO THAT IO WILL
                        ;CEASE SOON SO JOB CAN BE SHUFFLED
                        ;ALL DEVICES LOOK AT SHF BIT IN JBTSTS(ADVBFF OR ADVBFE)
                        ;TO SEE IF MONITOR IS WAITING TO SHUFFLE JOB
                        ;THE NSHF BIT IN JBTSTS WHOULD BE SET FOR JOBS USING DISPLAYS
                        ;SINCE DISSER CONTINUALLY REFERENCES USER AREA EVEN THOUGH
                        ;IOACT IS OFF.
```

```
                        ;THIS VERSION OF THE CORE SHUFFLER WORKS AS FOLLOWS:
                        ;EVERY CLOCK TICK FOR WHICH ALL JOBS ARE SHUFFABLE(NOT COUNTING ACTIVE
                        ;IO DEVICES), THE JOB IMMEDIATELY ABOVE THE LOWEST HOLE
                        ;(IF ANY) WILL BE MOVED DOWN INTO HOLE.  THE HOLEF IS SET NON-ZERO
                        ;TO THE ADDRESS OF JOB IMMEDIATELY ABOVE THE LOWEST
                        ;HOLE(0 IF NONE), EVERY TIME CORE IS REASSIGNED.
                        ;CANNOT BE CALLED WHILE SWAPPING IN PROGRESS FOR THIS JOB(BECAUSE IT CALLS CORE0)

                        INTERNAL CHKSHF,FTSWAP
                        EXTERNAL SHFWAT,HOLEF,CLKCHL,JBTADR,JBTSTS,JBTMAX

000021  332200  000000  CHKSHF: SKIPE ITEM,SHFWAT        ;HAS CHKSHF STOPPED A JOB AND IS WAITING FOR
                                                         ; DEVICES TO BECOME INACTIVE?
000022  254000  000034'         JRST SHFLOP              ;YES, SEE IF IO HAS STOPPED YET
000023  336040  000000          SKIPN TAC,HOLEF          ;NO, DOES CORE HAVE A HOLE IN IT?
000024  263140  000000          POPJ PDP,                ;NO
000025  201200  000000          MOVEI ITEM,JBTMAX        ;SEARCH FOR JOB OR HIGH SEG JUST ABOVE HOLE
000026  550104  000000  HOLLOP: HRRZ TAC1,JBTADR(ITEM)   ;ADR. OF JOB

000027  312100  000001          CAME TAC1,TAC
000030  367200  000026'         SOJG ITEM,HOLLOP         ;NOT JOB ABOVE HOLE KEEP LOOKG
000031  327200  000034'         JUMPG ITEM,SHFLOP        ;FOUND ONE?
000032  402000  000023'         SETZM HOLEF              ;NO, CLEAR HOLEF OR ELSE ERROR WILL BE PRINTED EVERY
                                                         ; CLOCK TICK AND ERROR MESSAGE WILL NEVER PRINT
000033  265240  000000          JSP DAT,ERROR            ;SYSTEM ERROR(TELL OPERATOR)

000034  332344  000026' SHFLOP: SKIPE PROG,JBTADR(ITEM) ;JOB ABOVE HOLE STILL IN CORE ?
000035  260140  000057'         PUSHJ PDP,ANYACT         ;NO ALL DEVICES FINISHED ?
000036  254000  000045'         JRST NOTSHF              ;YES, GO SEE IF HOLE STILL THERE
                        IFN JDAT-PROG,<
                                MOVE JDAT,JBTDAT(ITEM)   ;JOB DATA AREA
                        >
000037  554040  000007          HLRZ TAC,PROG            ;YES, REASSIGN SAME AMOUNT OF CORE,
000040  260140  000111'         PUSHJ PDP,SCORE1         ;IN A LOWER POSITION IN CORE

000041  402000  000021' NOTSH1: SETZM SHFWAT            ;JOB SHUFFLED, CLEAR FLAG
000042  205040  004000          MOVSI TAC,SHF            ;CLEAR SHUFFLE WAIT BIT IN CASE IT WAS ON
000043  412044  000000          ANDCAM TAC,JBTSTS(ITEM)
000044  263140  000000          POPJ PDP,

                        ;JOB CANNOT BE MOVED BECAUSE IT HAS ACTIVE DEVICES OR NSHF BIT SET(DISPLAY,REALTIME)
                        ; SO JOB CANNOT BY SHUFFLED AT THIS TIME


000045  336000  000032' NOTSHF: SKIPN HOLEF             ;IS HOLE STILL THERE?
000046  254000  000041'         JRST NOTSH1              ;NO
                        IFN FTSWAP,<
                                EXTERNAL FIT,FORCE
000047  214040  000000          MOVM TAC,FORCE
000050  312200  000000          CAME ITEM,FIT
000051  316200  000001          CAMN ITEM,TAC
000052  254000  000041'         JRST NOTSH1
                        >
```

```
182  000053  202200  000041'          MOVEM ITEM,SHFWAT          ;SET SHUFFLE WAIT FLAG WITH JOB NO.
183  000054  205040  004000            MOVSI TAC,SHF              ;SET SHF WAIT BIT IN JOB STATUS WORD
184  000055  436044  000043'          IORM TAC,JBTSTS(ITEM)      ;SO JOB WILL NOT BE RUN
185  000056  263140  000000            POPJ POP,                  ;AND IO WILL STOP SOON
```

```
                        ;ROUTINE TO TEST FOR ANY ACTIVE DEVICES

                        ;CALL:  MOVE ITEM,JOB NUMBER OR HIGH SEG NUMBER
                        ;       MOVE JDAT,ADDRESS OF JOB DATA AREA
                        ;       PUSHJ PDP,ANYACT
                        ;       DEVICES ACTIVE
                        ;       DEVICES NOT ACTIVE EXCEPT POSSIBLY TTY
                        ;IN SWAPPING SYSTEMS ANYACT IS BROKEN INTO 2 CALLS, TRYSWP AND ANYDEV
                        ;TRYSWP TESTS WHTETHER SWAPPER SHOULD EVEN CONSIDER SWAPPING JOB

                                INTERN ANYACT,ANYDEV
                                EXTERN JOBMAX

000057                  ANYACT:
                        IFN FTSWAP,<
                                INTERN ANYDEV
000057  260140  000104'          PUSHJ PDP,TRYSWP        ;SHOULD SWAPPER MAKE JOB UNRUNNABLE
                                                        ; IN ORDER TO SWAP OUT?
000060  263140  000000          POPJ PDP,               ;NO, ACTIVE SAVE IN PROGRESS OR NSHF,NSWP SET
                                                        ; (REAL TIME OR DISPLAY)
                        >
000061  200644  000055' ANYDEV: MOVE T,JRTSTS(ITEM)     ;IS JOB(OR HIGH SEG) NOT SHUFFABLE?
000062  603640  001000          TLNE T,NSHF             ;DISPLAY AND REAL TIME SET NSHF
000063  263140  000000          POPJ PDP,               ;CANNOT BE SHUFFLED
                        IFN FT2REL,<
000064  303200  000000          CAILE ITEM,JOBMAX       ;YES, IT THIS A HIGH SEG?
000065  254000  000000          JRST CPOPJ1             ;YES, OK TO SHUFFLE OR SWAP SINCE NSHF,NSWP NOT SET
                                                        ; AND NO SAVE IN PROGRESS
                        >
000066  201647  000000          MOVEI T,JOBJDA(JDAT)    ;ASSUME JOB IS NOT CURRENT JOB
000067  316200  000000          CAMN ITEM,JOB           ;IS IT?
000070  201640  000000          MOVEI T,USRJDA          ;IT IS CURRENT JOB
                                                        ; DEVICE ASSIGNMENT TABLE IN MONITOR
000071  201000  010000          MOVEI IOS,IOACT         ;IO DEVICE ACTIVE BIT
000072  550715  000000          HRRZ T1,JOBJMH(T)       ;GET NO. OF USER IO CHANNELS IN USE
                                                        ; FOR JOB(EITHER FROM JOB DATA AREA
                                                        ; OR FROM MONITOR (IGNORE LH WHICH MAY BE-1
                                                        ; IF SAVGET IN PROGRESS)
000073  661640  000016          TLO T,T1                ;SET TO ADD T TO T1
000074  205740  000010          MOVSI AC3,DVTTY ;DEVICE IS A TTY BIT
000075  332320  000015  ANY:    SKIPE DEVDAT,@T         ;IS A DEVICE ASSIGNED TO THIS CHANNEL?
000076  616006  000002          TDNN IOS,DEVIOS(DEVDAT) ;YES, IS IT ACTIVE?
000077  365700  000075' ANY2:   SOJGE T1,ANY            ;NO, KEEP LOOKING
000100  321700  000065'         JUMPL T1,CPOPJ1         ;YES, FINISHED YET?
000101  616746  000004          TDNN AC3,DEVMOD(DEVDAT) ;NOT FINISHED, IS DEVICE TTY?
000102  263140  000000          POPJ PDP,               ;NO, ERROR RETURN, CANNOT ASSIGN CORE
000103  254000  000077'         JRST ANY2               ;YES, KEEP LOOKING FOR AN ACTIVE DEVICE
```

```
                        ;ROUTINE TO TEST TO SEE IF JOB OR HIGH SEG CAN BE SWAPPED
                        ; OR WHETHER IT SHOULD BE ALLOWED TO CONTINUE RUNNING
                        ; UNTIL A MORE FAVORABLE TIME
                        ;CALL:  MOVE ITEM,HIGH RO LOW SEG NUMBER
                        ;       PUSHJ PDP,TRYSWP
                        ;       RETURN1 - JOB MUST REMAIN RUNABLE(NSHF,NSWP SET OR SAVE,GET IN PROGRESS)
                        ;       RETURN2 - OK TO SWAP HIGH OR LOW SEG

                        IFN FTSWAP,<
                                INTERN TRYSWP
                                EXTERN JBTSTS,CPOPJ1

000104  200644  000061' TRYSWP: MOVE T,JBTSTS(ITEM)      ;IS JOB OF HIGH SEG NOT SWAPPABLE?
000105  603640  011000          TLNE T,NSHF!NSWP          ;OR SHUFFABLE(DISPLAY,REAL TIME)?
000106  263140  000000          POPJ PDP,                 ;YES, ERROR RETURN
                        IFN FT2REL,<
                                EXTERN ANYSAV
000107  254000  000000          JRST ANYSAV               ;NO, SEE IF THIS JOB IS INVLOVED IN A SAV,GET
                                                          ; WHICH IS STILL ACTIVE
                        >
                        IFE FT2REL,<
                                JRST CPOPJ1               ;NO, GIVE OK RETURN
                        >
                        >
```

```
                        ;ROUTINE TO FLUSH PHYSICAL CORE ASSIGNED IN MEMORY
                        ;NOTE: THIS ROUTINE DIFERS SIGNIFICANTLY FROM CORE0 AND CORE1 IN THAT
                        ;IT IS ONLY A PHYSICAL  REMOVAL OF CORE(VIRTAL IS NOT AFFECTED)
                        ;SEE COMMENTS FOR CORE1
                        ;CALL:  MOVE ITEM,HIGH OR LOW SEGMENT NUMBER
                        ;       PUSHJ PDP,KCORE1
                        ;       ALWAYS RETURN HERE
                        ;SCORE1 IS CALLED FROM SHUFFLER WITH TAC SET TO SEG SIZE

                                INTERN KCORE1

000110  201040  000000  KCORE1: MOVEI TAC,0              ;SETUP DESIRED HIGHEST ADR
000111  370003  000000  SCORE1: SOS (PDP)               ;CORE1 WILL ALWAYS SKIP RETURN
000112  254000  000141'        JRST CORE1A              ;BYPASS LOGICAL CORE ASSIGNMENT PART
                                                         ; AND FLUSH PHYSICAL CORE(LOGICAL CORE UNEFFECTED)



                        ;CORE0 IS CALLED BY THE CORE MONITOR COMMAND AND THE CORE SHUFFLER

                        ;AND RUN COMMAND
                        ;BOTH LOGICAL AND PHYSICAL CORE ASSIGNMENT ARE AFFECTED

                        ;CALL:  MOVE TAC,HIGHEST LEGAL ASSRESSABLE LOC. DESIRED
                        ;       MOVE ITEM,JOB NUMBER
                        ;       MOVE PROG,[XWD PROT.,RELOC.]=JBTADR(ITEM)
                        ;       PUSHJ PDP,CORE0
                        ;       ERROR   ;EITHER JOB HAS ACTIVE IO
                        ;                ; OR NOT ENOUGH CORE
                        ;       OK RETURN
                        ;JOB IS MOVED IF NECESSARY TO SATISFY REQUEST
                        ;PROG AND JDAT ARE SET TO NEW CORE ASSIGNMENT ON EITHER RETURN
                        ;0 MEANS NONE ASSIGNED IN MEMORY, ASSIGNED ON DISK
                        INTERNAL CORE0
                        INTERNAL CORE1,FTTIME,FTTRPSET,CORGET,FTSWAP
                        EXTERNAL JOBJDA,JOB,USRJDA,JOBADR,JBTADR
                        EXTERNAL JOBREL,JOBADR,JOBDAC,JOBPC,JOBDAT,JBTDAT
                        EXTERNAL CORTAL,CORLST,CORTAB,HOLEF,CLRWRD
                        EXTERNAL USRREL,CPOPJ1,JOBJMH,JOBENB,JOBDPD,JOBDPG
                        EXTERNAL JOBPR1,CPOPJ1,JOBPRT,USRPC,CORMAX
                        ;ENTER HERE FROM CORE CONSOLE COMMAND OR INITIAL CORE
                        ;ASSIGNMENT OF JUST A JOB DATA AREA FOR RUN COMMAND
                        ;IE ENTER WHEN DEVICES MAY BE ACTIVE OR JOB MAY HAVE NO PREVIOUS CORE
                        ;JOB CAN HAVE CORE IN MEMORY, CORE ON DISK, OR NONE EITHER PLACE
                        ;JOB CANNOT BE IN PROCESS OF SWAP OUT OR SWAP IN(CALLER'S RESPONSIBILITY)
                        ;CORE0 NO LONGER REASSIGN CORE ON DISK IF OLD CORE ON DISK
                        ;BECAUSE OF FRAGMENTED SWAPPING(TOO HARD) UNLESS 0 BEING ASKED FOR
                        ;THEREFORE THE CORE COMMAND CAUSES JOB TO BE SWAPPED INTO CORE FIRST(INCORE=1)
                        ;HOWEVER, THE R,RUN,GET,KJOB COMMANDS DO NOT REQUIRE THE PREVIOUS CORE IMAGE TO
                        ;BE SWAPPED IN(AS THIS IS SLOW), THEY ASK FOR 140 WORDS, AND LARGER DISK SPACE IS RELIN
                        QUISHED
                        ;UPON SWAPIN BY THE SWAPPER, VIRTAL IS INCREASED THEN RATHER THAN
                        ;ON THE CALL TO CORE0.
                        ;IT WILL TRY TO REASSIGN CORE IN MEMORY IF OLD CORE IN MEMORY
```

```
310                     ;IF THIS FAILS, IT WILL REASSIGN NEW CORE ON DISK AND ASK SWAPPER TO EXPAND
311                     ;IF JOB DID NOT HAVE OLD CORE, AN ATTEMPT WILL BE MADE TO ASSIGN CORE IN MEMORY
312                     ;IF THIS FAILS, TI WILL BE ASSIGNED ON THE DISK AND ASK SWAPPER TO EXPAND
313                     ;THE OTHER PLACES IN THE MONITOR WHERE THE IN-CORE COUNT IS TOUCHED IS
314                     ;IN GET WHERE IT INCREMENTS TO SHARE COPY ALREADY IN CORE.
315                     ;AND END OF SWAPIN OF LOW SEG AND HIGH SEG IS ALREADY IN CORE FOR OTHER USER
316                     ;THE CORE ROUTINES DO NOT ALTER THE HIGH SEG IN CORE COUNT.  IT IS UP TO THE  CALLER
317                     ;(IN SEGCON) TO CALL THE CORE ROUTINES ONLY IF IN CORE COUNT IS 0,
318                     ;AND END OF SWAPIN OF LOW SEG AND HIGH SEG IS ALREADY IN CORE FOR OTHER USER
```

```
000113                  CORE0:
                        IFE FTSWAP,<
                                JUMPE PROG,CORGET       ;IS JOB WITHOUT CORE IN MEMORY?

                        >
                        IFN FTSWAP,<
                                EXTERN IMGOUT,CPOPJ1,CHGSWP
000113  326340  000123'          JUMPN PROG,CORE0A       ;DOES JOB HAVE CORE IN MEMORY?
                                                        ; (ALWAYS TRUE BOTH SEGS IF CORE UUO)
                                IFE FT2REL,<
                                EXTERN CORMAX
                                CAML TAC,CORMAX         ;NO, WILL REQUEST FIT IN PHYSICAL CORE?
                                                        ; COMPARE WITH LARGEST PERMITTED ADR+1(BUILD AND
                                                        ; ONCE CAN RESTART CORMAX)


                                >
                                IFN FT2REL,<
                                EXTERN SUMSEG
000114  260140  000000          PUSHJ PDP,SUMSEG        ;NO, WILL SUM OF BOTH SEGMENTS FIT IN PHYSICAL CORE?
                                                        ; LARGEST PERMITTED CORE, COMPARE SUM WITH CORMAX

                                >
000115  263140  000000          POPJ PDP,               ;NO, GIVE ERROR RETURN
000116  205100  002000          MOVSI TAC1,SWP          ;IS JOB SWAPPED OUT?
000117  616104  000104'          TDNN TAC1,JBTSTS(ITEM) ;(MAY HAVE 0 DISK SPACE ALTHOUGH SWAPPED OUT)
000120  254000  000126'          JRST CORE1              ;NO, TRY TO ASSIGN CORE IN MEMORY
000121  260140  000000          PUSHJ PDP,CHGSWP        ;YES, CHANGE ASSIGNMENT OF SWAPPING SPACE ON DISK
                                                        ; INCREASE VIRTAL(COUNT OF FREE 1K BLOCKS OF SWAPPING
                                                        ; (SHOULD NEVER NEED TO DECREASE VIRTAL SINCE
                                                        ; CORE COMMAND ALWAYS SWAPS JOB IN FIRST),
                                                        ; (FRAGMENTATION POSTPONES RETURNING SPACE AND
                                                        ; INCREASING VIRTAL UNTIL SWAP IN IF NOT ASKING
                                                        ; FOR 0 VIRTUAL CORE)
000122  254000  000100'          JRST CPOPJ1             ;GIVE OK RETURN TO CALLER

000123                  CORE0A:
                        EXTERNAL JBTSWP
                        >
                        IFN FTTRPSET,<
                                EXTERNAL STOPTS
000123  336000  000000          SKIPN STOPTS            ;NO,IS TIME SHARING STOPPED BY
                                                        ; TRPSET UUO DONE FOR JOB 1?

                        >

000124  260140  000057'          PUSHJ PDP,ANYACT        ;NO,ANY ACTIVE DEVICE?
000125  263140  000000          POPJ PDP,               ;YES, CANNOT ASSIGN CORE
                                                        ; NO, FALL INTO CORE1
```

```
                        ;ROUTINE TO TRY TO ASSIGN CORE IN CORE
                        ;LOW OR HIGH SEG MUST NOT BE SWAPPED OUT(CALLER'S RESPONSIBILITY)
                        ;AND MUST NOT HAVE ANY ACTIVE DEVICES(IT MAY HAVE 0 CORE IN CORE THOUGH)
                        ;IN OTHER WORDS HIGH OR LOW SEG MAY OR MAY NOT HAVE VIRTUAL CORE
                        ;BUT IF IT HAS VIRTUAL CORE IT MUST BE IN PHYSICAL CORE
                        ;THIS IS BOTH A LOGICAL AND A PHYSICAL CORE ASSIGNMENT
                        ;FIRST OLD CORE IS RETURNED TO SYSTEM
                        ;THEN NEW REQUEST IS ATTEMPTED TO BE SATISFIED IN LOWEST
                        ;POSITION POSSIBLE.  THUS CORE TENDS TO BE PACKED
                        ;IF NEW REQUEST CANNOT BE GRANTED, OLD AMOUNT IS RETAINED, IF NON-SWAPPING SYS
                        ;OTHERWISE SWAPPER IS CALLED(XPAND) TO EXPAND CORE BY SWAPPING OUT


                                EXTERN SEGSIZ

000126                  CORE1:  NOSCHEDULE      **      ;PREVENT SCHEDULING

                        IFN FTSWAP,<
                                EXTERN VIRTAL
000126  260140  000000          PUSHJ PDP,SEGSIZ        ;TAC1=OLD SEG SIZE
000127  336540  000001          SKIPN LOC,TAC           ;IS 0 BEING REQUESTED?
000130  211540  000001          MOVNI LOC,1             ;YES, PRETEND -1(DEPEND ON ASH BUG WHICH KEEPS -1
                                                        ; ON RT. SHIFT)
000131  240540  777766          ASH LOC,-12             ;CONVERT TO NO. OF K-1(01,0,1,2,...)
000132  274540  000002          SUB LOC,TAC1            ;NO. OF K-1 INCREASE=NEW-OLD-1
000133  315540  000000          CAMGE LOC,VIRTAL        ;IS THERE ENOUGH FREE VIRTUAL CORE IN SYSTEM?
                        >
                        IFE FT2REL,<
                                CAML TAC,CORMAX         ; YES, IS REQUEST LESS THAN MAX, ALLOWED COR+1?
                        >
                        IFN FT2REL,<
                                EXTERN SUMSEG
000134  260140  000114'          PUSHJ PDP,SUMSEG        ;YES, IS SUM OF SEGS LESS THAN MAX, ALLOWED CORE+1?
                        >
000135  263140  000000          POPJ PDP,               ;NO, ERROR RETURN
                        IFN FTSWAP,<
000136  271540  000001          ADDI LOC,1              ;YES, GET NO. OF K OF INCREASE
000137  213000  000013          MOVNS LOC               ;MAKE MINUS FOR UPDATE
000140  272540  000133'          ADDM LOC,VIRTAL         ;AND UPDATE TOTAL VIRTUAL CORE IN SYSTEM
                                                        ; SINCE THIS REQUEST CAN BE SATISFIED
                        >
000141                  CORE1A: NOSCHEDULE      **      ;PREVENT JOB SCHEDULING
000141  322340  000146'          JUMPE PROG,CORGET       ;OLD ASSIGNMENT 0?
                                                        ; IF YES, DO NOT ATTEMPT TO RETURN OLD CORE
000142  550540  000007          HRRZ LOC,PROG           ;NO, ABS. LOC. OF OLD CORE
000143  554500  000007          HLRZ BLK,PROG           ;HIGHEST LEGAL REL. ADR.
000144  201640  000000          MOVEI T,0               ;CLEAR FOR CORSTG CALL
000145  260140  000272'          PUSHJ PDP,CORSTG        ;RETURN OLD CORE TO FREE STORAGE
```

```
                        ;CORGET IS CALLED BY SWAPPER WHEN JOB IS ON DISC AND IS
                        ;WANTED IN CORE.

000146  403540  000007  CORGET: SETZB LOC,PROG          ;SET NEW ASSIGNMENT TO 0 AND DIST. MOVED
000147  322040  000227'          JUMPE TAC,DIDLE1       ;IS ZERO CORE BEING REQUESTED?
000150  260140  000252'          PUSHJ PDP,HOLSRC       ;NO, SEARCH FOR HOLE BIG ENOUGH
000151  254000  000165'          JRST BAKOLD            ;NONE, GIVE BACK OLD AMOUNT
                        INTERN FTTRACK
                        IFN FTTRACK,<
                                EXTERN LASCOR
                                MOVEM   ITEM,LASCOR    ;LEAVE TRACKS FOR LAST JOB USING
                                                       ; PHYSICAL CORE ALLOCATION
                                                       ; (FOR DEBUGGING ONLY)
                        >
000152  202540  000007          MOVEM LOC,PROG         ;SETUP NEW RELOC
000153  506040  000007          HRLM TAC,PROG          ;AND NEW PROTECT.
000154  201501  000000          MOVEI BLK,(TAC)        ;HIGHEST REL ADR. BEING REQUESTED
000155  201640  000001          MOVEI T,1              ;SET USE BITS IN CORE TABLE
000156  260140  000272'          PUSHJ PDP,CORSTG
000157  200504  000034'          MOVE BLK,JRTADR(ITEM)  ;OLD CORE ASSIGNMENT
000160  326500  000170'          JUMPN BLK,MOVCOR       ;WAS THERE OLD MEMORY ASSIGNED?
                                                       ; NO,
                        IFN FTSWAP,<
000161  135100  000000          LDB TAC1,IMGOUT        ;SIZE(IN K) OF SEG ON DISK TO BE SWAPPED IN
000162  242100  000012          LSH TAC1,12            ;CONVERT TO WORDS
                        >
                        IFE FTSWAP,<
                                MOVEI TAC1,0           ;JOB HAS NO PREVIOUS VIRT. CORE(NON-SWAP SYS)
                        >
000163  200500  000007          MOVE BLK,PROG          ;MAKE OLD ASSIGNMENT(BLK) APPEAR TO START
                                                       ; AT SAME PLACE AS NEW ASSIGNMENT(FOR CLEARING)
000164  364100  000204'          SOJA TAC1,CLRCR1               ;IF NEW CORE SIZE IS BIGGER THAN
                                                       ;OLD, CLEAR OUT INCREASE SO SECURITY WILL
                                                       ; BE MAINTAINED, TAC1 IS SIZE-1 OF OLD
                                                       ; ASSIGNMENT, -1 OF NO OLD ASSIGNMENT
```

```
                                ;HERE WHEN FREE CORE TABLE DOES NOT HAVE ENOUGH ROOM FOR REQUEST

000165                          BAKOLD:
                                IFN FTSWAP,<
                                        EXTERN XPAND
000165  260140  000000                  PUSHJ PDP,XPAND         ;TELL SWAPPER TO SWAP OUT
                                >
                                IFE FTSWAP,<
                                        IFE FT2REL,<
                                        SOS (PDP)               ;SET FOR ERROR RETURN AND GET BACK LD CORE
                                        >
                                        IFN FT2REL,<
                                        EXTERN FRECOR
                                        PUSHJ PDP,FRECOR        ;TRY TO DELETE 1 DORMANT HIGH SEGMENT,
                                                                ; IF REQUEST DOES NOT EXCEED FREE CORE+DORMANT SEGS
                                        SOSA (PDP)              ;ERROR RETURN-NOT ENOUGH CORE, GET OLD AMOUNT BACK
                                        JRST CORGET             ;1 DORMANT SEG DELETED, TRY REQUEST AGAIN
                                        >
                                >
000166  554044  000157'  BKOLD1: HLRZ TAC,JBTADR(ITEM)    ;GIVE BACK OLD CORE.
000167  254000  000146'          JRST CORGET
```

```
                                ;MOVE OLD CORE TO NEW AREA

000170  306552  000000  MOVCOR: CAIN LOC,(BLK)          ;IS NEW CORE IN SAME PLACE AS OLD?
000171  254070  000203'         JRST CLRCOR             ;YES, DO NOT MOVE IT,CLEAR IF INCREASE
000172  554100  000012         HLRZ TAC1,BLK           ;LENGTH OF OLD CORE
000173  303101  000000         CAILE TAC1,(TAC)        ;IS OLD CORE LESS THAN NEW?
000174  550100  000001         HRRZ TAC1,TAC           ;NO, MOVE THE SHORTENED NEW CORE
                                IFN FTTIME,<
                                EXTERNAL SHFWRD
000175  272100  000000         ADDM TAC1,SHFWRD        ;INCREMENT TOTAL NO. WORDS SHUFFLED
                                >
000176  270100  000013         ADD TAC1,LOC            ;ADD IN NEW RELOC.
000177  200640  000013         MOVE AC1,LOC            ;DEST.=NEW RELOC.
000200  504640  000012         HRL AC1,BLK             ;SOURCE=OLD RELOC.
000201  402004  000166'         SETZM JBTADR(ITEM)      ;FLAG THAT CORE IS IN TRANSIT(TTY ROUTINES)
000202  251642  000000         BLT AC1,(TAC1)          ;MOVE CORE TO NEW ASSIGNMENT

                                ;CLEAR INCREASE IF NEW CORE IS BIGGER THAN OLD

000203  554100  000012  CLRCOR: HLRZ TAC1,BLK           ;OLD CORE SIZE-1
000204  317040  000002  CLRCR1: CAMG TAC,TAC1           ;IS NEW CORE SIZE-1 VREATER THAN OLD CORE SIZE-1
000205  254070  000216'         JRST DIDLE              ;NO, DO NOT CLEAT ANY CORE
                                IFN FTTIME,<
000206  200640  000071         MOVE AC1,TAC            ;NEW CORE SIZE
000207  274640  000002         SUB AC1,TAC1            ;LESS OLD CORE SIZE
000210  272640  000000         ADDM AC1,CLRWRD         ;ACCUMULATE NO. OF WORDS CLEARED
                                >
000211  271177  000002         ADDI TAC1,2(PROG)       ;YES, OLD SIZE-1+NEW RELOC+2=2ND LOC TO CLEAR
000212  271047  000000         ADDI TAC,(PROG)         ;NEW SIZE-1+NEW RELOC=LAST LOC TO CLEAR
000213  402002  777777         SETZM -1(TAC1)          ;CLEAR FIRST WORD
000214  505172  777777         HRLI TAC1,-1(TAC1)      ;SET LH TO FIRST ADR. TO CLEAR
000215  251171  000000         BLT TAC1,(TAC)          ;CLEAR THE INCREASE PORTION
```

```
520
521                             ;IF THE SHUFFLED JOB IS IN EXEC MODE, ITS DUMP ACS
522                             ;(PDP,PROG,JDAT SAVED IN JOB DATA AREA) MUST BE
523                             ;ALTERED BY DISTANCE CORE WAS MOVED
524
525                             ;IF THE SHUFFLED JOB IS CURRENT JOB, THE SOFTWARE STATE OF
526                             ;THE MONITOR(IE SOFTWARE INFO OF JOB) MUST BE ALTERED BY AMOUNT
527                             ;CORE WAS MOVED
528
529                             EXTERNAL SYSSIZ
510
511  000216  275552  000000  DIDLE:  SUBI LOC,(BLK)            ;DISTANCE JOB WAS MOVED(DEST.-SOURCE)
512  000217  312200  000067'         CAME ITEM,JOB            ;IS THIS CURRENT JOB?
513  000220  334047  000000          SKIPA TAC,JOBPC(JDAT)    ;NO, GET PC IN JOB DATA AREA
514  000221  200040  000000          MOVE TAC,USRPC           ;YES, PC IN PROTECTED SYSTEM AREA
515                             IFN FT2REL,<
516                                     EXTERN JOBMAX
517  000222  307200  000064'         CAIG ITEM,JOBMAX         ;IS THIS A HIGH SEGMENT?
518                             >
519  000223  603040  010000          TLNE TAC,USRMOD          ;NO, IS JOB IN USER MODE?
520  000224  254000  000227'         JRST DIDLE1              ;YES, DO NOT ALTER DUMP ACS
521                                                          ; BECAUSE THEY ARE THE USERS(OR HIGH SEG)
522  000225  272547  000000          ADDM LOC,JOBDPD(JDAT)    ;NO, ALTER DUMP PDP BY DIST, OF MOVE
523  000226  272547  000000          ADDM LOC,JOBDPG(JDAT)    ;AND ALTER PROG BY DIST, MOVED
524  000227  202344  000201' DIDLE1: MOVEM PROG,JRTADR(ITEM) ;STORE NEW CORE ASSIGNMENT(LOW OR HIGH SEG)
525  000230  312200  000217'         CAME ITEM,JOB            ;IS THIS CURRENT JOB?
526  000231  254000  000236'         JRST DIDLE3              ;NO, DO NOT ALTER STATE OF MONITOR
527  000232  200040  000000          MOVE    TAC,SYSSIZ       ;DONT CHANGE PDP IF LIST
528  000233  307043  000000          CAIG    TAC,(PDP)        ;IS IN SYSTEM
529  000234  270140  000013          ADD PDP,LOC              ;YES, ALTER PUSH DOWN POINTER BY AMOUNT OF MOVE
530  000235  260140  000000          PUSHJ PDP,SETREL         ;GO SETUP NEW HARDWARE AND SOFTWARE RELOCATION
531                                                          ; INFORMATION FOR HIGH AND LOW SEGS FOR CURRENT JOB
532  000236                  DIDLE3:
533                             IFN FT2REL,<
534                                     EXTERN CURHGH
535  000236  260140  000000          PUSHJ PDP,CURHGH         ;CHECK TO SEE IF THIS CORE ASSIGNMENT IS FOR
536                                                          ; HIGH SEG WHICH CURRENT USER MAY ALSO BE USING
537                                                          ; IF YES, RESET HARDWARE AND SOFTWARE RELOC INFO,
538                                                          ; RETURN WITH ITEM PRESERVED,PROG SET TO RELOC
539                                                          ; OF SEG WHICH HAS JUST HAD CORE REASSIGNED
540
541                             >
542  000237  403040  000045'         SETZR TAC,HOLEF          ;CLEAR HOLE FLAG
543  000240  260140  000252'         PUSHJ PDP,HOLSRC         ;IS THERE A NON-ZERO HOLE?
544  000241  254000  000245'         JRST COROK               ;NO
545  000242  271552  000001          ADDI LOC,1(BLK)          ;YES, FORM ADR, OF JOB JUST ABOVE HOLE
546  000243  312640  000000          CAME T,CORLST            ;IS HOLE AT TOP OF MEMORY
547  000244  202540  000237'         MOVEM LOC,HOLEF          ;NO, FLAG WITH ADDRESS OF JOB ABOVE HOLE
548  000245                  COROK:
549                             IFN FTSWAP,<
550                                     EXTERNAL BIGHOLE
551  000245  201040  777777          MOVEI TAC,-1             ;FIND BIGGEST HOLE
552  000246  260140  000252'         PUSHJ PDP,HOLSRC         ;ALWAYS GET ERROR RETURN
```

```
  553  000247  240700  777766          ASH T1,-↑D10              ;CONVERT TO 1K BLOCKS
  554  000250  202700  000000          MOVEM T1,BIGHOLE
  555                              >
  556                                  SCHEDULE↑↑
  557  000251  254000  000122'         JRST CPOPJ1               ;SKIP RETURN(UNLESS ERROR)
```

```
                                ;ROUTINE TO FIND HOLE BIG ENOUGH FOR REQUEST
                                ;CALL:  MOVE TAC,HIGHEST REL. ADR. ASKING FOR
                                ;       PUSHJ PDP,HOLSRC
                                ;       RETURN1 ;NO HOLES BIG ENOUGH
                                ;       RETURN2 ;T BYTE SET TO LAST BLOCK+1 IN HOLE
                                ;               ;BLK SET TO HIGHEST REL. LOC. IN THAT HOLE
                                ;               ;LOC SET TO ADDRESS OF FIRST BLOCK IN HOLE
                                ;               ;T1=LARGEST HOLE SEEN
                                ;USES TAC1

000252  200640  000313' HOLSRC: MOVE T,CORE2P           ;BYTE POINTER TO FIRST BIT-1
000253  403540  000016          SETZM LOC,T1            ;START AT BOTTOM OF MEMORY
                                                        ; LARGEST HOLE SO FAR=0
000254  634540  000012  CORHOL: TDZA BLK,BLK            ;START BLK AT 0 AND SKIP

000255  271540  002000  CORH00: ADDI BLK,2000          ;INCREMENT HIGHEST REL LOC.
000256  316640  000243' CORH01: CAMN T,CORLST          ;BYTE POINTER TO 1ST NON-EXISTANT BLOCK
000257  263140  000000          POPJ PDP,               ;NO MORE CORE TO SEARCH
000260  134140  000015          ILDB TAC1,T             ;GET NEXT CORE USE BIT
000261  271540  002000          ADDI LOC,2000           ;INCREMENT ADDRESS OF BLOCK
000262  322140  000255'          JUMPE TAC1,CORH00       ;IS THIS BLOCK IN USE?
000263  322540  000256'          JUMPE BLK,CORH01        ;YES, HAVE ANY FREE BLOCKS BEEN SEEN YET?
                                IFN FISWAP,<
000264  313540  000016          CAMLE BLK,T1            ;YES, BIGGEST SO FAR?
000265  202540  000016          MOVEM BLK,T1            ;YES, SAVE IN T1.
                                >
000266  317540  000071          CAMG BLK,TAC            ;YES, IS THIS HOLE EQUAL OR GREATER THAN REQUEST?
000267  254000  000254'          JRST CORHOL             ;NO, KEEP LOOKING FOR HOLES
000270  275552  002000          SUBI LOC,2000(BLK)      ;YES, SET LOC TO FIRST BLOCK IN HOLE
000271  364540  000251'          SOJA BLK,CPOPJ1         ;SET BLK TO HIGHEST REL. LOC.
                                                        ; AND RETURN
```

```
590                             ;ROUTINE TO SET AND CLEAR CORE USE TABLE
591                             ;CALL:  MOVEI T,1       ;TO SET TABLE
592                             ;       MOVEI T,0       ;TO CLEAR TABLE
593                             ;       MOVE BLK,HIGHEST REL. LOC. IN USER AREA
594                             ;       MOVE LOC,ADDRESS OF FIRST BLOCK TO SET CLEAR
595
596                                     INTERN CORE2P
597                                     EXTERN TPOPJ
598
599  000272  261140  000001  CORSTG: PUSH PDP,TAC            ;SAVE HIGHEST LOC. BEING REQUESTED
600  000273  240500  777766          ASH BLK,-12            ;CONVERT TO NO. OF BLOCKS-1
601  000274  271500  000001          ADDI BLK,1             ;NO. OF BLOCKS
602  000275  332000  000015          SKIPE T                ;UPDATE NO OF FREE BLOCKS
603  000276  213000  000012          MOVNS BLK              ;DECREASE IF SETTING BITS
604  000277  272500  000000          ADDM BLK,CORTAL        ;INCREASE IF CLEARING,DECREASE IF SSETTING BITS
605  000300  200040  000013          MOVE TAC,LOC           ;ADDRESS OF FIRST BLOCK
606  000301  240040  777766          ASH TAC,-12            ;FORM BYTE POINTER TO BIT-1
607  000302  231040  000044          IDIVI TAC,^D36         ;TAC=WORD,TAC1=BIT
608  000303  270040  000313'        ADD TAC,CORE2P         ;FORM BYTE POINTER

609  000304  213000  000002          MOVNS TAC1
610  000305  271100  000044          ADDI TAC1,^D36
611  000306  137100  000325'        DPB TAC1,[POINT 6,TAC,5]
612  000307  217000  000012          MOVMS BLK              ;GET MAG. OF NO. OF BLOCKS INVOLVED
613  000310  136640  000001          IDPB T,TAC             ;SET OR CLEAR EACH USE BIT
614  000311  367500  000310'        SOJG BLK,.-1
615  000312  254000  000000          JRST TPOPJ             ;RESTORE TAC, AND POPJ
616
617  000313  440100  000000  CORE2P: POINT 1,CORTAB          ;BYTE POINTER TO FIRST BIT-1
```

```
                                ;ROUTINE TO CLEAR PART OF JOB DAT AREA(PART PROTECTED FROM USER IO)
                                ;CALLED WHEN NEW CORE ASSIGNED AND AT SYSTEM RESTART(140)
                                ;       MOVE ITEM,JOB NO.
                                ;CALL:  MOVE JDAT,ADR. OF JOB DATA AREA
                                ;       PUSHJ PDP,CLRJOB

                                INTERNAL CLRJOB
                                EXTERNAL JOBPRT,JOBPR1,JOBPFI,JORENB
                                EXTERNAL JOBPD1,JORDDT

000314  402047  000000  CLRJOB: SETZM JOBPRT(JDAT)       ;FIRST LOC. PROTECTED FROM USER
000315  205047  000314'         MOVSI TAC,JOBPRT(JDAT)
000316  541047  000000          HRRI TAC,JOBPR1(JDAT)
000317  200147  000000          MOVE TAC1,JORDDT(JDAT)  ;SAVE DDT STARTING ADDRESS
000320  251047  000000          BLT TAC,JOBPFI(JDAT)
000321  202147  000317'         MOVEM TAC1,JOBDDT(JDAT)
000322  402047  000000          SETZM JORENB(JDAT)      ;ALSO CLEAR APR ENABLE WORD
000323  402047  000000          SETZM JOBPD1(JDAT)      ;AND UUO PC FLAGS(USED WHEN JOB STARTS)
000324  254000  000000          JRST ESTOP1             ;GO SET JOB STATUS, SO CONT WILL
                                                        ; NOT WORK,(DO NOT CLEAR JACCT BIT)

000325                  COREND: END
000325  360670  000271
```

NO ERRORS DETECTED

PROGRAM BREAK IS 000326

SYMBOL TABLE

AC1		000015	INT	AC2		000016	INT	AC3		000017	INT
ANY		000075'		ANY2		000077'		ANYACT		000057'	INT
ANYDEV		000061'	INT	ANYSAV		000107'	EXT	BAKOLD		000165'	
BIGHOL		000250'	EXT	BKOLD1		000166'		BLK		000012	
BUFPNT		000012	INT	BUFWRD		000013	INT	CHGSWP		000121'	EXT
CHKSHF		000021'	INT	CLKCHL		000000	EXT	CLRCOR		000203'	
CLRCR1		000204'		CLRJOB		000314'	INT	CLRWRD		000210'	EXT
CORE0		000113'	INT	CORE0A		000123'		CORE1		000126'	INT
CORE1A		000141'		CORE2P		000313'	INT	COREND		000325'	
CORERR		000014'		CORGET		000146'	INT	CORH00		000255'	
CORH01		000256'		CORHOL		000254'		CORLST		000256'	EXT
CORMAX		000016'	EXT	COROK		000245'		CORSTG		000272'	
CORTAB		000313'	EXT	CORTAL		000277'	EXT	CORU1		000010'	
CORUUO		000000'	INT	CPOPJ1		000271'	EXT	CURHGH		000236'	EXT
DAT		000005	INT	DEVDAT		000006	INT	DEVIOS		000002	INT
DEVMOD		000004	INT	DIDLE		000216'		DIDLE1		000227'	
DIDLE3		000236'		DVTTY		000010	INT	ESTOP1		000324'	EXT
FIT		000050'	EXT	FORCE		000047'	EXT	FT2REL	777777	777777	
FTCCL	777777	777777		FTDISK	777777	777777		FTLOGI	777777	777777	
FTRC10	777777	777777		FTSWAP	777777	777777	INT	FTTIME	777777	777777	INT
FTTRAC		000000	INT	FTTRPS	777777	777777	INT	HOLEF		000244'	EXT
HOLLOP		000026'		HOLSRC		000252'		IMGOUT		000161'	EXT
IOACT		010000	INT	IOS		000000	INT	IOWAIT		000002'	EXT
ITEM		000004	INT	JBTADR		000227'	EXT	JBTDAT		000000	EXT
JBTMAX		000025'	EXT	JBTSTS		000117'	EXT	JBTSWP		000000	EXT
JDAT		000007	INT	JOB		000230'	EXT	JOBADR		000000	EXT
JOBDAC		000000	EXT	JORDAT		000000	EXT	JOBDDT		000321'	EXT
JOBDPD		000225'	EXT	JORDPG		000226'	EXT	JOBENB		000322'	EXT
JORJDA		000066'	EXT	JORJMH		000072'	EXT	JOBMAX		000222'	EXT
JORPC		000220'	EXT	JORPD1		000323'	EXT	JOBPFI		000320'	EXT
JOBPR1		000316'	EXT	JORPRT		000315'	EXT	JOBREL		000000	EXT
KCORE1		000110'	INT	LOC		000013		MOVCOR		000170'	
NOTSH1		000041'		NOTSHF		000045'		NSHF		001000	INT
NSWP		010000	INT	OERROR		000033'	EXT	PDP		000003	INT
PROG		000007	INT	SCORE1		000111'		SEGSIZ		000126'	EXT
SETREL		000235'	EXT	SHF		004000	INT	SHFLOP		000034'	
SHFWAT		000053'	EXT	SHFWRD		000175'	EXT	STOPTS		000123'	EXT
STOTAC		000020'	EXT	SUMSEG		000134'	EXT	SWP		002000	INT
SYSSIZ		000232'	EXT	T		000015		T1		000016	
TAC		000001	INT	TAC1		000002	INT	TPOPJ		000312'	EXT
TRYSWP		000104'	INT	UCORHI		000011'	EXT	USRJDA		000070'	EXT
USRMOD		010000	INT	USRPC		000221'	EXT	USRREL		000000	EXT
VCORE1		000414	INT	VIRTAL		000140'	EXT	WSCHED		000015'	EXT
XPAND		000165'	EXT	ZERCOR		000016'					

```
CURHGH      534      535
D             6#       6
DAT           6#       6      153
DCL           6#       6
DCLI          6#       6
DCLO          6#       6
DCLR          6#       6
DDI           6#       6
DDO           6#       6
DEN           6#       6
DEVADR        6#       6
DEVBUF        6#       6
DEVCHR        6#       6
DEVCTR        6#       6
DEVDAT        6#       6      226      227      230
DEVEXT        6#       6
DEVFIL        6#       6
DEVIAD        6#       6
DEVIOS        6#       6      227
DEVLOG        6#       6
DEVMOD        6#       6      230
DEVNAM        6#       6

DEVOAD        6#       6
DEVPPN        6#       6
DEVPTR        6#       6
DEVSER        6#       6
DGF           6#       6
DHNG          6#       6
DIDLE       489      511#
DIDLF1      416      520      524#
DIDLF3      526      532#
DIN           6#       6
DINI          6#       6
DLK           6#       6
DMT           6#       6
DNAERR        6#       6
DOU           6#       6
DR            6#       6
DRL           6#       6
DRN           6#       6
DSER          6#       6
DSI           6#       6
DSKRLB        6#       6
DSO           6#       6
DVAVAL        6#       6
DVCDR         6#       6
DVDIR         6#       6
DVDIRI        6#       6
DVDIS         6#       6
DVDSK         6#       6
DVDTA         6#       6
DVIN          6#       6
DVLNG         6#       6
```

CODES	6#		
DISABL	6#		
ENABLE	6#		
NOSCHE	6#	379	405
NOSHUF	6#		
QUEUES	6#		
SCHEDU	6#	556	
SHUFFL	6#		
STARTO	6#		
XP	6#	6	13