F 7

```
BBBBBBBBBBBB        AAAAAAAAA        CCCCCCCCCCC   KKK         KKK  UUU         UUU  PPPPPPPPPPPP
BBBBBBBBBBBB        AAAAAAAAA        CCCCCCCCCCC   KKK         KKK  UUU         UUU  PPPPPPPPPPPP
BBBBBBBBBBBB        AAAAAAAAA        CCCCCCCCCCC   KKK         KKK  UUU         UUU  PPPPPPPPPPPP
BBB         BBB  AAA         AAA  CCC              KKK         KKK  UUU         UUU  PPP         PPP
BBB         BBB  AAA         AAA  CCC              KKK         KKK  UUU         UUU  PPP         PPP
BBB         BBB  AAA         AAA  CCC              KKK         KKK  UUU         UUU  PPP         PPP
BBB         BBB  AAA         AAA  CCC              KKK      KKK     UUU         UUU  PPP         PPP
BBB         BBB  AAA         AAA  CCC              KKK      KKK     UUU         UUU  PPP         PPP
BBB         BBB  AAA         AAA  CCC              KKK      KKK     UUU         UUU  PPP         PPP
BBBBBBBBBBBB     AAA         AAA  CCC              KKKKKKKKK        UUU         UUU  PPPPPPPPPPPP
BBBBBBBBBBBB     AAA         AAA  CCC              KKKKKKKKK        UUU         UUU  PPPPPPPPPPPP
BBBBBBBBBBBB     AAA         AAA  CCC              KKKKKKKKK        UUU         UUU  PPPPPPPPPPPP
BBB         BBB  AAAAAAAAAAAAAAA  CCC              KKK      KKK     UUU         UUU  PPP
BBB         BBB  AAAAAAAAAAAAAAA  CCC              KKK      KKK     UUU         UUU  PPP
BBB         BBB  AAAAAAAAAAAAAAA  CCC              KKK      KKK     UUU         UUU  PPP
BBB         BBB  AAA         AAA  CCC              KKK         KKK  UUU         UUU  PPP
BBB         BBB  AAA         AAA  CCC              KKK         KKK  UUU         UUU  PPP
BBB         BBB  AAA         AAA  CCC              KKK         KKK  UUU         UUU  PPP
BBBBBBBBBBBB     AAA         AAA     CCCCCCCCCCC   KKK         KKK  UUUUUUUUUUUUUUU  PPP
BBBBBBBBBBBB     AAA         AAA     CCCCCCCCCCC   KKK         KKK  UUUUUUUUUUUUUUU  PPP
BBBBBBBBBBBB     AAA         AAA     CCCCCCCCCCC   KKK         KKK  UUUUUUUUUUUUUUU  PPP
```

```
**FILE**ID**RESTARTM                                I 3
```

```
RRRRRRRR    EEEEEEEEEE    SSSSSSSS  TTTTTTTTTT    AAAAAA    RRRRRRRR    TTTTTTTTTT  MM      MM
RRRRRRRR    EEEEEEEEEE    SSSSSSSS  TTTTTTTTTT    AAAAAA    RRRRRRRR    TTTTTTTTTT  MM      MM
RR      RR  EE          SS              TT      AA      AA  RR      RR      TT      MMMM  MMMM
RR      RR  EE          SS              TT      AA      AA  RR      RR      TT      MMMM  MMMM
RR      RR  EE          SS              TT      AA      AA  RR      RR      TT      MM  MM  MM
RR      RR  EE          SS              TT      AA      AA  RR      RR      TT      MM  MM  MM
RRRRRRRR    EEEEEEEE      SSSSSS        TT      AA      AA  RRRRRRRR        TT      MM      MM
RRRRRRRR    EEEEEEEE      SSSSSS        TT      AA      AA  RRRRRRRR        TT      MM      MM
RR   RR     EE                  SS      TT      AAAAAAAAAA  RR   RR         TT      MM      MM
RR   RR     EE                  SS      TT      AAAAAAAAAA  RR   RR         TT      MM      MM
RR    RR    EE                  SS      TT      AA      AA  RR    RR        TT      MM      MM      ....
RR    RR    EE                  SS      TT      AA      AA  RR    RR        TT      MM      MM      ....
RR      RR  EEEEEEEEEE  SSSSSSSS        TT      AA      AA  RR      RR      TT      MM      MM      ....
RR      RR  EEEEEEEEEE  SSSSSSSS        TT      AA      AA  RR      RR      TT      MM      MM      ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

Table of contents

```
    0000          .TITLE  RESTARTM        Reel checkpoint and restart
    0000          .IDENT  'V04-000'
    0000  ;---
    0000  ;
    0000  ;****************************************************************************
    0000  ;*                                                                          *
    0000  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
    0000  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
    0000  ;*  ALL RIGHTS RESERVED.                                                    *
    0000  ;*                                                                          *
    0000  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
    0000  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
    0000  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
    0000  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
    0000  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
    0000  ;*  TRANSFERRED.                                                            *
    0000  ;*                                                                          *
    0000  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
    0000  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
    0000  ;*  CORPORATION.                                                            *
    0000  ;*                                                                          *
    0000  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
    0000  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
    0000  ;*                                                                          *
    0000  ;*                                                                          *
    0000  ;****************************************************************************
    0000
    0000  ;++
    0000  ; FACILITY:
    0000  ;       Backup/Restore
    0000  ;
    0000  ; ABSTRACT:
    0000  ;       This module contains routines to save and restore the low-level machine
    0000  ;       state for reel checkpoint and restart.
    0000  ;
    0000  ; ENVIRONMENT:
    0000  ;       VAX/VMS user mode.
    0000  ;
    0000  ;
    0000  ;--
    0000  ;
    0000  ; AUTHOR: M. Jack, CREATION DATE: 30-May-1981
    0000  ;
    0000  ;
    0000  ; MODIFIED BY:
    0000  ;
    0000  ;       V03-001 MLJ0090         Martin L. Jack, 7-May-1982  13:26
    0000  ;               Repair reel restart.
    0000  ;
    0000  ;       V02-001 MLJ0054         Martin L. Jack, 22-Nov-1981  22:24
    0000  ;               Integrate GET_VM and FREE_VM jacket routines.
    0000  ;
    0000  ;
    0000  ;**
    0000
    0000
00000000          .PSECT  CODE,EXE,NOWRT
```

```
                  0000              .SBTTL  CHECKPOINT_M            Checkpoint machine state
                  0000
                  0000      ;++
                  0000      ;
                  0000      ; Functional Description:
                  0000      ;       This routine checkpoints the low-level machine state so that we can
                  0000      ;       later restart at the same point.
                  0000      ;
                  0000      ; Calling Sequence:
                  0000      ;       CALLS/CALLG
                  0000      ;
                  0000      ; Input Parameters:
                  0000      ;       04(AP) = CHKPT_HIGH_SP
                  0000      ;       08(AP) = Address of CHKPT_STACK
                  0000      ;       12(AP) = Address of CHKPT_LOW_SP
                  0000      ;
                  0000      ; Implicit Inputs:
                  0000      ;       none
                  0000      ;
                  0000      ; Output Parameters:
                  0000      ;       none
                  0000      ;
                  0000      ; Implicit Outputs:
                  0000      ;       none
                  0000      ;
                  0000      ; Routines Called:
                  0000      ;       GET_VM
                  0000      ;
                  0000      ; Routine Value:
                  0000      ;       none
                  0000      ;
                  0000      ; Signals:
                  0000      ;       none
                  0000      ;
                  0000      ; Side Effects:
                  0000      ;       none
                  0000      ;
                  0000      ;--
                  0000
            0FFC  0000              .ENTRY  CHECKPOINT_M,^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                  0002      ;
                  0002      ;       The entry mask specifies all registers so that when the RET in
                  0002      ;       RESTART_M executes, all registers are restored.
                  0002      ;
                  0002      ;       Save the frame pointer in the global variable CHKPT_LOW_SP.
                  0002      ;
 0C BC    5D    D0  0002              MOVL    FP,@12(AP)              ; Save stack base pointer
                  0006      ;
                  0006      ;       Compute the difference between the saved frame pointer in the global
                  0006      ;       variable CHKPT_HIGH_SP and CHKPT_LOW_SP.  This value is the length
                  0006      ;       of the section of stack that will be saved.  It includes the stack
                  0006      ;       frame for this routine.
                  0006      ;
 56    04 AC    7D  0006              MOVQ    4(AP),R6                ; R6 = CHKPT_HIGH_SP
                  000A                                              ; R7 = Address of CHKPT_STACK
7E    56    5D    C3  000A              SUBL3   FP,R6,-(SP)             ; Compute size of stack to save
                  000E      ;
```

```
                                 000E   ;       Allocate the dynamic memory, placing the address of the allocated
                                 000E   ;       space in the global variable CHKPT_STACK.
                                 000E   ;
                      6E    DD   000E           PUSHL   (SP)                    ; Copy size of stack to save
          00000000'GF 01    FB   0010           CALLS   #1,G^GET_VM             ; Allocate space to hold saved stack
                   67 50    D0   0017           MOVL    R0,(R7)                 ; Store address in CHKPT_STACK
                                 001A   ;
                                 001A   ;       Copy the stack area to the dynamic memory, and return.
                                 001A   ;
          00 B7    6D 6E    28   001A           MOVC3   (SP),(FP),@(R7)         ; Save the stack area
                            04   001F           RET
```

```
                          0020          .SBTTL  RESTART_M               Restart given saved machine state
                          0020
                          0020  ;++
                          0020  ;
                          0020  ; Functional Description:
                          0020  ;       This routine restarts using the low-level machine state that was saved
                          0020  ;       by CHECKPOINT_M.
                          0020  ;
                          0020  ; Calling Sequence:
                          0020  ;       CALLS/CALLG
                          0020  ;
                          0020  ; Input Parameters:
                          0020  ;       04(AP) - CHKPT_LOW_SP
                          0020  ;       08(AP) = CHKPT_HIGH_SP
                          0020  ;       12(AP) = Address of CHKPT_STACK
                          0020  ;
                          0020  ; Implicit Inputs:
                          0020  ;       none
                          0020  ;
                          0020  ; Output Parameters:
                          0020  ;       none
                          0020  ;
                          0020  ; Implicit Outputs:
                          0020  ;       none
                          0020  ;
                          0020  ; Routines Called:
                          0020  ;       none
                          0020  ;
                          0020  ; Routine Value:
                          0020  ;       none
                          0020  ;
                          0020  ; Signals:
                          0020  ;       none
                          0020  ;
                          0020  ; Side Effects:
                          0020  ;       none
                          0020  ;
                          0020  ;--
                          0020
                    0000  0020          .ENTRY  RESTART_M,^M<>          ; Register save irrelevant
                          0022  ;
                          0022  ;       Compute the difference between the saved frame pointer in the global
                          0022  ;       variable CHKPT_HIGH_SP and CHKPT_LOW_SP  This value is the length
                          0022  ;       of the section of stack that will be restored from the saved copy.
                          0022  ;
       56   08 AC    7D   0022          MOVQ    8(AP),R6                ; R6 = CHKPT_HIGH_SP
                          0026                                          ; R7 = Address of CHKPT_STACK
       56   04 AC    C2   0026          SUBL2   4(AP),R6                ; Compute length of saved area
                          002A  ;
                          002A  ;       Now restore the FP and SP registers to the values they had on entry to
                          002A  ;       routine CHECKPOINT_M.  Then, copy the saved section of stack to the
                          002A  ;       stack.  This restores the stack to the exact state that it was
                          002A  ;       in on entry to routine CHECKPOINT_M.  It is important not to cause
                          002A  ;       any signals or other stack activity during these three instructions.
                          002A  ;
       5D   04 AC    D0   002A          MOVL    4(AP),FP                ; Restore stack pointers
          5E   5D    D0   002E          MOVL    FP,SP                   ;
```

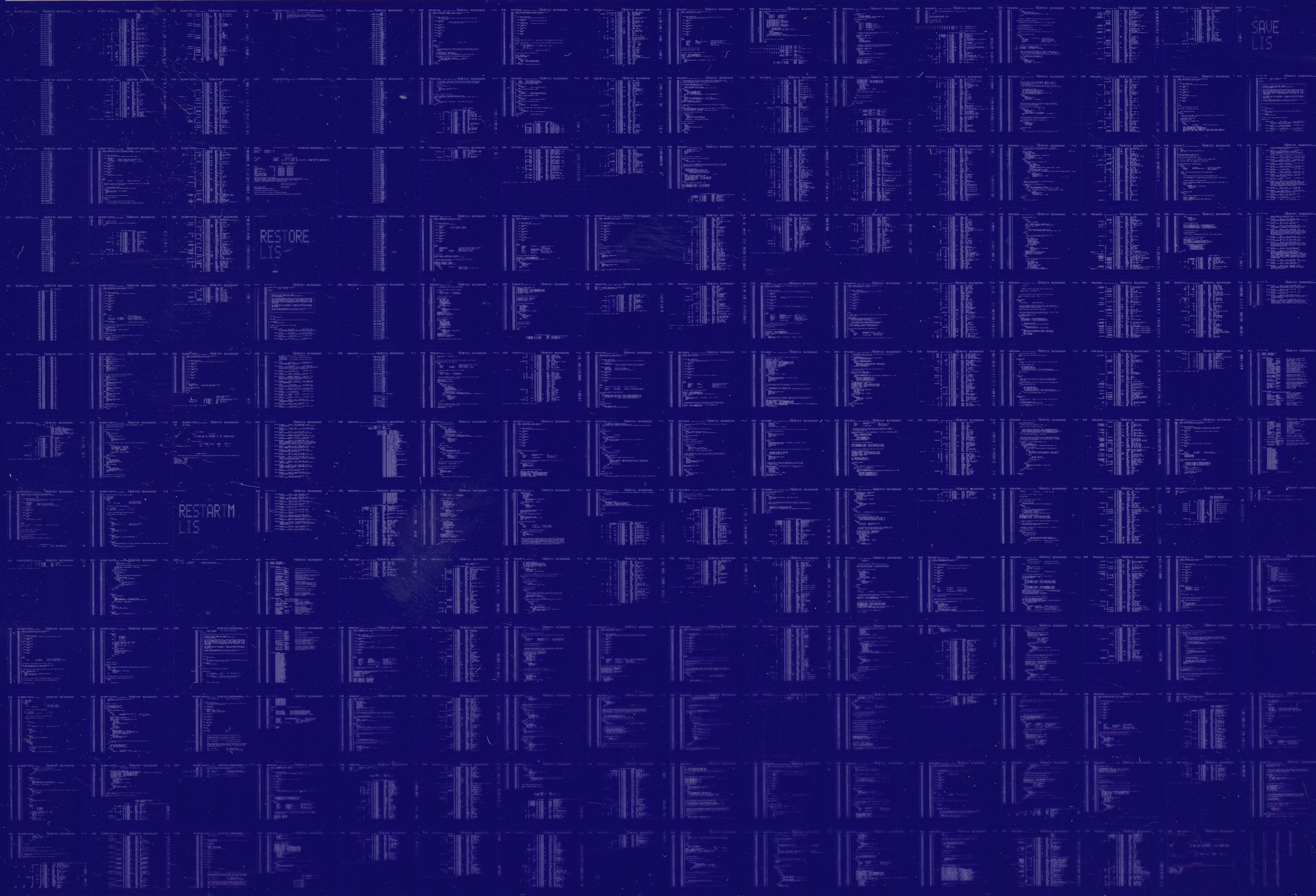
```
               6D   00 B7   56   28  0031          MOVC3   R6,@(R7),(FP)             ; Restore stack
                                     0036      ;
                                     0036      ;       The following RET instruction executes in the context of routine
                                     0036      ;       CHECKPOINT_M, because the stack has been restored, and returns to
                                     0036      ;       the caller of routine CHECKPOINT_M.  The previous execution flow has
                                     0036      ;       been entirely lost.
                                     0036      ;
                                 04  0036              RET                               ; Return to caller of CHECKPOINT_M
```

```
    0037   194         .END
```

```
CHECKPOINT_M     00000000 RG     01
GET_VM           ********    X   01
RESTART_M        00000020 RG     01

                                          +----------------+
                                          ! Psect synopsis !
                                          +----------------+

PSECT name                       Allocation          PSECT No.  Attributes
----------                       ----------          ---------  ----------
.  ABS  .                        00000000  (     0.)  00 (  0.)  NOPIC    USR    CON    ABS    LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
CODE                             00000037  (    55.)  01 (  1.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD   NOWRT NOVEC BYTE

                                        +------------------------+
                                        ! Performance indicators !
                                        +------------------------+

Phase                     Page faults    CPU Time       Elapsed Time
-----                     -----------    --------       ------------
Initialization                     13    00:00:00.08    00:00:00.86
Command processing                 88    00:00:00.62    00:00:03.16
Pass 1                             72    00:00:00.44    00:00:02.14
Symbol table sort                   0    00:00:00.00    00:00:00.00
Pass 2                             52    00:00:00.33    00:00:01.34
Symbol table output                 2    00:00:00.01    00:00:00.01
Psect synopsis output               1    00:00:00.02    00:00:00.09
Cross-reference output              0    00:00:00.00    00:00:00.00
Assembler run totals              230    00:00:01.50    00:00:07.60

The working set limit was 750 pages.
2036 bytes (4 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 3 non-local and 0 local symbols.
194 source lines were read in Pass 1, producing 17 object records in Pass 2.
0 pages of virtual memory were used to define 0 macros.

                                       +--------------------------+
                                       ! Macro library statistics !
                                       +--------------------------+

Macro library name                       Macros defined
------------------                       --------------
_$255$DUA28:[SYSLIB]STARLET.MLB;2                     0

0 GETS were required to define 0 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:RESTARTM/OBJ=OBJ$:RESTARTM MSRC$:RESTARTM/UPDATE=(ENH$:RESTARTM)
```

0013 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



SAVE
LIS

RESTORE
LIS

RESTARTM
LIS