

**THE MULTI-TASKER**  
The Newsletter of the RSX-11/IAS Special Interest Group

Volume 13, Number 6

December 1980

Contributions should be sent to: Editor, The Multi-Tasker, c/o DECUS, One Iron Way, MR2-3/E55, Marlboro, MA 01752

European members should send contributions to: Colin A. Mercer, Tennant Post, High Street, FAREHAM, PO16 7BQ, Hants, England

Members in Australia or New Zealand should send contributions to: Clive Edington, CSIRO, Computing Research  
314 Albert St., East Melbourne, VIC 3002, Australia

Letters and articles for publication are requested from members of the SIG. They may include helpful hints, inquiries to other users, reports on SIG business, summaries of SPR's submitted to Digital or other information for the members of RSX-11/IAS SIG.

All contributions should be "camera-ready copy" e.g. sharp black type in a 160x240 mm area (8 1/2" x 11" paper with 1" margins) and should not include xerox copies. If you use RUNOFF to prepare your contribution the following parameters have been found to be satisfactory:

.PAPER SIZE 60,80 .LEFT MARGIN 8 .RIGHT MARGIN 72 .SPACING 1

These parameters assume output on a lineprinter with a pitch of 10 char/inch. Adjust the parameters to maintain the same margins if another pitch is used.

**FROM THE EDITOR**

This is my first issue as the new MULTITASKER editor. I am rapidly becoming aware of the enormous time and effort Mike Blake-Knox put into the MULTITASKER in the past three years as editor. Mike deserves the thanks of all RSX/IAS SIG members. It will be several months before I can begin to match the standards he has set.

In the coming months, I will be working with the RSX/IAS Steering Committee and the MULTITASKER working group to strengthen the newsletter. If you have any comments on the MULTITASKER, please send them to me care of the DECUS office. The only immediate change will be in the publication deadlines. In the future, material must be received by me by the first of the month to make that month's edition. I will try to have the issue in the mail to the DECUS office one week later for publication and mailing.

Ralph Stamerjohn

Copyright © 1980, Digital Equipment Corporation  
All Rights Reserved

It is assumed that all articles submitted to the editor of this newsletter are with the authors' permission to publish in any DECUS publication. The articles are the responsibility of the authors and, therefore, DECUS, Digital Equipment Corporation, and the editor assume no responsibility or liability for articles or information appearing in the document. The views herein expressed are those of the authors and do not necessarily express the views of DECUS or Digital Equipment Corporation.



PPG INDUSTRIES, INC./ONE GATEWAY CENTER/PITTSBURGH, PENNSYLVANIA 15222

October 9, 1980

Dear People:

We have three general questions about programs for RSX-11M, and a fourth question about ADA. We hope either you or your readers will be able to help us with answers.

1. We have heard that MODULA is a good programming language for creating small, stand-alone systems for time-critical real-time applications. We know nothing else about the language: what it is, whether a compiler is available for RSX-11M, etc. How can we learn more about MODULA?

2. How can we learn more about DECX11? What difficulties would be involved in modifying DECX11 source to run under RSX-11M? (These questions come from our hardware technician, who has been looking at hardware diagnostics in DECX11 source).

3. Are there any RSX-11M programs for maintaining a multiple-person daily calendar/appointment notebook on a computer? My supervisor, for example, would be able to run this program to find out what I was doing next Thursday at 3 pm., or insert a note in everybody's schedule about a product demo three Tuesdays hence. A related problem is finding a multi-person project log program.

4. We have seen advertisements for a British newsletter informing subscribers on how ADA is developing. Are there any U.S.-based ADA newsletters? Who publishes them?

Please print this letter in The Multi-Tasker. Please send any correspondence to me at the above address, "Attention: P. E. Pfeiffer, 552". You can also call me collect at (412) 434-3316, if you can help out.

Respectfully yours,

*Phillip E. Pfeiffer IV*  
Phillip E. Pfeiffer IV

USING UNIBUS RM02 EMULATOR ON PDP-11/70  
(It can be done)

We ran into problems when we tried to integrate a foreign UNIBUS storage module disk (SMD) controller, which emulates the RM02, into an 11/70 running in 22-bit addressing mode.

First, we found a bus in the RM02 driver, where R2 was being wired out across a call to the 'map UNIBUS to memory' (\$MPUBM) exec routine. We had already looked at the Software Dispatch for all RM02/03 articles, and found none listed. So we SPR'ed the bus. It seemed incredible that no-one had run into this bus, since the driver could not have worked on an 11/44 with a real RM02/RH-11.

There HAD been a patch published, as an 'optional feature' (believe it!) as article 1.1.2.4F, under 'Distribution Kits', with the unlikely title 'PDP-11/44 Support in RSX-11M/S'. I don't think they could have made this any more obscure! Why not publish it as a 'Disk Device Driver' article?

Anyway, that's not the end of it. After fixing the driver, we were able to access the RM02 emulator SMD from our system which was running on a real RK-05. Next thing was to set a bootable system on the SMD, which was to be the main system disk. So we copied over all the files, created a new 'virgin' exec on the SMD with VMR and booted it. It died.

After a lot of hunting, we discovered that the DV.MBC bit in the DR driver's UCB configuration word 1, UC.CW1, was set. When set, all of the UNIBUS mapping services in the driver are disabled. No wonder.

The bit should only be set if the RM controller hooks up to the 11/70 with an RH-70 (MASSBUS) controller, i.e., it's an RM03, not an RM02. But we had specified 'RM02' not 'RM03' during the sysgen. And anyway, things worked fine when running from the RK-05 based system.

We looked at SYSTB.MAC, which contains the DEC supplied driver data bases (loadable or not) from our SYSGEN, and found that DV.MBC was set! Apparently SYSGEN doesn't care whether you say RM02 or RM03. Well, OK except why was the driver working while running from the RK-05 based system? Who shut that bit off?

Deferring that question till later, we ZAPPED the virgin exec on the SMD to turn off the DV.MBC bit, booted it, SAV /WB'ed it and all was well. We had a bootable system on the SMD.

Out of curiosity, we decided to find out how (or at least when) the DV.MBC bit got shut off in our RK-05 based system, when it had been set by SYSGEN in the driver's data base.


Recreating the process of bringing up a system on the RK-05, we created a new virgin system there with VMR. Then we looked at the DR driver's UCB in the virgin RK-05 system with ZAP, and as expected, we found the bit was set. Then we booted the virgin RK-05 system and checked the DR driver's UCB again, this time with OPEN. The bit was still set. Then we SAV/WB'ed the RK-05 system. When it came up the next time, the bit had been cleared.

Apparently, B00 or SAV looks at the current system device, and if it is a UNIBUS device, it clears out the DV.MBC bits in all appropriate UCB's. We didn't so on and read B00 and SAV's code to find out which one did it.

A final note. SGNPER.CMD sets the DV.MBC bit if memory size is specified greater than 124K, and if the processor is NOT an 11/44. It pays no attention to your selection of RM02 vs RM03.

To summarize, if you plan to integrate one of the many UNIBUS mass storage controllers which emulate DEC controllers with RM-11's with an 11/70, do the following:

1. Fix the bus in the DRDRV.MAC source if it's an RM02 emulator.
2. Build and VMR the virgin system on the new disk.
3. ZAP the virgin system to turn off the DV.MBC bit.
4. Boot the virgin system and SAV /WB it.

Robert B. Denny, dba   
Creative System Design  
3452 E. Foothill Blvd. Suite 601  
Pasadena, CA 91107  
(213) 792-9474

## PATCH FOR PROPER BRU OPERATION ON FOREIGN TM-11 EMULATORS

### 1.0 PROBLEM

BRU gives "BRU -- TAPE LABEL ERROR ON MTn" message one or more times when reading BRU tapes on a foreign TM-11 emulator. Worse, on some or all tapes, BRU gives "BRU -- \*WARNING\* -- TAPE OUT OF SEQUENCE. etc." which makes it impossible to read BRU tapes at all.

This problem appeared with the release of RSX-11M V3.2 and the BRU utility. It has been observed only on systems using 'foreign' TM-11 tape controller emulators; thus is not a problem that could have been SPR'ed.

### 2.0 APPROACH

The approach to solving this problem was to study carefully the MT driver code, then insert a BPT instruction at the initiator entry point in the driver and run BRU on an offending tape. This procedure requires that XDT be included in the executive. The BPT instruction causes a trap to XDT. Be sure to read section 3.4.1.2 of Guide to Writing an I/O Driver (for V3.2). You can't just type "BRK" and set the breakpoint from XDT if you're working with a loadable MT: driver.

After setting the trap in the MT driver, I ran BRU. Each time the driver was called, it trapped to XDT and I wrote down the contents of each I/O packet that BRU passed. Finally, I got to the packet that preceded the failure.

Then I dumped the first few blocks of several BRU tapes, noting the contents and size of the records. It is clearly an ANSI-ish tape, with a VOL record a 'boot' block, and HDRn records, making up a header file terminated in a tape mark.

### 3.0 ANALYSIS

With this data (MT driver listings, list of I/O packets passed and tape dumps) an analysis revealed the following:

1. There is a 'double interrupt timing window' in the MT driver interrupt service routine. If a second interrupt occurs during the time the driver is executing the fork level code in this 'window', the second interrupt is disregarded.
2. BRU was reading the 512. byte ANSI 'boot' block on the tape with an 80. byte read request. So it MUST have expected the transfer to terminate in a 'data overrun' error.
3. The "... TAPE OUT OF SEQUENCE ..." error occurred following the read request for the record after the 'boot' block. Apparently, BRU was setting incorrect data or an unexpected error from this read request.

Without pursuing it any further, it was clear that the 'foreign' controller was acting differently than the real TM-11 when it got a data overrun condition (more data in the record than was requested). Perhaps it was interrupting when the request was satisfied, but leaving the tape moving till it reached the real record gap (at which time it interrupted again??). If so, the next read request could have in fact read the tail end of the 512. byte boot block. Perhaps the double interrupt timing window was in the act. I would be interested to hear from anyone who knows. I was still faced with implementing a solution, so I quit analyzing right here.

### 4.0 SOLUTION

The obvious solution was to change BRU so it issued an 'honest' 512. byte request for the boot block in the header. This proved to be moderately difficult. I dumped the module and global names in BRU.OLB, using LBR, and found a module with the likely name of TAPEIO. I then disassembled TAPEIO (with the object disassembler I got off the SIG tape) and located the QIO's which produced the I/O packets I had looked at. Then I broke the code all around that area, writing in comments.

This revealed a real spaghetti plate of code. Apparently, a good deal of code-crunching effort was expended to get rid of a few words of code.

Anyway, I then designed the following patch, which jumps out to do a 'real' 512. byte I/O request for the boot block, then does a request that had to be wired out to make space for the patch jump, then jumps back into the spaghetti. Surprisingly, it worked the first time.

Anyone interested in more info or who really knows the physical cause of the error can contact me at:

Robert B. Denny, dba *RBA*  
Creative System Design  
3452 E. Foothill Blvd. Suite 601  
Pasadena, CA 91107  
(213) 792-9474

**kms  
fusion**

inc. A Subsidiary of KMS Industries, Inc.  
3621 South State Road, P.O. Box 1567, Ann Arbor, Michigan 48106 313/769-8500

10 November 1980

Patch file: TAPEIO.PAT  
See the RSX-11M V3.2 release notes for procedure for patching a driver.  
; TITLE TAPEIO  
; IDENT 'SPCIAL'  
;  
; Patch to TAPEIO as distributed with RSX-11M V3.2  
; Required for BRU to work with certain foreign MT: controllers.  
;

```

START = .
.=START+2442
JMP PATCH
.=START+2470
RESUME:
.PSECT PATCH
PATCH:
MOV #512,,Q.IOPL+2(R0) ;SET FOR 512. BYTE BLOCK
JSR PC,START+3300 ;READ IT
MOV #EFN9,-(SP) ;WAIT
MOV #1051,-(SP)
EMT 377
MOV #80,,Q.IOPL+2(R0) ;RESET BLOCK SIZE TO 80.
TSTB #TPSTA ;OK?
BPL 10$ ;BR IF OK.
MOVB #TPSTA,#I0COD ;OOPS, ERR. GET MSG CODE
TRAP -EC.BOT ;DRAT
10$: JSR PC,START+3300 ;READ THE 80. BYT'ER
MOV #EFN9,-(SP) ;WAIT
MOV #1051,-(SP)
EMT 377
JMP RESUME ;AND PICK BACK UP.
.END

```

228

The Editor  
The MultiTasker  
C/O DECUS  
One Iron Way MR2-3/E55  
Marlboro, MA 01752

Dear Sir:

During the Fall Decus it was brought to my attention that one additional serious bug exists with KMSKIT for systems using DECNET. If a user selects that remote DECNET terminals are to be used, the UCB's for the DECNET remote terminals get generated without the modifications which KMSGEN incorporates into the SYSGEN procedure for the terminal driver UCB's, CO's UCB's, and VT's UCB's. As a consequence, the accounting modifications will crash the system as soon as a DECNET remote terminal is used on the system incorporating accounting. This problem can be fixed simply by modifying the NETGEN files which create the UCB's (in much the same way SGNTT was modified).

Enclosed is a "PATCH" for F4P V3.0 which enables it to work the same way which V2.5 worked. Now one can say:  
F4P F00.OBJ;1=F00  
even if F00.OBJ;1 exists. Without the patch, a fatal error occurs. If user's want this functionality, they should let DEC know. There is no guarantee that I can negotiate the inclusion of this feature into the next F4P release.

Sincerely,

*Jim Downward*  
James G. Downward  
Sr. Research Scientist

JGD/amm  
Enclosure

229

\*\*\*\*\* DIGITAL EQUIPMENT CORP. assumes no responsibility or liability  
 \*\*\*\*\* for any effects of this patch to its software. The customer  
 \*\*\*\*\* releases DIGITAL from all liability by accepting this patch.  
 \*\*\*\*\* It is unsupported, and should not be serviced by Software  
 \*\*\*\*\* field specialists.

#### OPENING THE .LST AND .OBJ FILES

The FORTRAN IV-PLUS V3.0-1 compiler generates a diagnostic for the opening of .LST and .OBJ files with explicit version numbers when a file with the same name and version number already exists. Serious errors in the FORTRAN IV-PLUS V3.0 compiler are corrected using the PAT object module patch utility. Consult the RSX-11 Utilities Procedures Manual (DEC-11-0XMDA-A-0 or AA-55678-TC) for detailed instructions on using PAT.

All FORTRAN IV-PLUS patches are done in the FORTRAN IV-PLUS reserved UFD [11,36] on the system disk. The object modules to be patched are extracted from the compiler object module library ([11,36]F4P.OLB), patched using PAT and replaced in the compiler object module library. A new, corrected, compiler is then task-built from the corrected compiler object module library.

#### NOTE

This compiler object module library update procedure using the PAT utility is applied to either a previous corrected object module library or to the distribution library. All correction files for all previous compiler patches MUST be reapplied to the compiler object library whenever a new compiler is task built from the distribution library. PAT checksum failures will occur if any patches are omitted.

*The compile*

S:  
 .S+1204

One PAT correction file is applied to FORTRAN IV-PLUS compiler object modules to correct this problem. Since this is a "one of a kind patch" version number will not be updated.

1. If correcting the distribution library, apply the previous patches.
2. Create the two following correction files (COMAND.PAT)
3. Make a copy of the compiler object module library file ([11,36]F4P.OLB).
4. Update the compiler object module library file:
 

```

; RSX-11M V3.2 AND RSX-11M PLUS V1.0 UPDATE PROCEDURE
;
; CORRECTIONS FOR THE FORTRAN IV-PLUS V3.0 COMPILER.
;
MAC COMAND.POB=COMAND.PAT
LBR COMAND.OBJ=F4P/EX:COMAND
PAT COMAND.OBJ=COMAND.OBJ,COMAND.POB
LBR F4P/RP=COMAND.OBJ
      
```
5. Rebuild the compiler as described in the FORTRAN IV-PLUS Installation Guide (AA-18730-TC).

Build the following correction files:

```

.TITLE COMAND
; COPYRIGHT (c) 1980
; DIGITAL EQUIPMENT CORPORATION, MAYNARD, MA. 01754
;
; MODIFICATIONS:
; /000310/ -- ALLOW FILE VERSION TO SUPERCEDE ON COMPILER OUTPUT
;
.IDENT /000310/

.MCALL OPENS

.PSECT

OPENS R0,#FO.WRT ; OPEN OUTPUT FILE

.END
      
```



A. S. THOMAS, INC.  
ENGINEERING - RESEARCH - ANALYSIS  
355 PROVIDENCE HIGHWAY  
WESTWOOD, MASS. 02090

AREA CODE 617  
329-9200

Editor, The Multi-Tasker  
c/o DECUS  
One Iron Way  
Mr2-3/E55  
Marlboro, MA 01752

Gentlemen:

Since the release of V3.2 of RSX-11M, we have extensively utilized the ability to spawn the indirect command file processor.

One problem that we have encountered ( of purely a cosmetic nature) is that ...AT. always exits with the display of:

> @<EOF>

When all the operations performed by the indirect command processor are displayed at the terminal, the above behavior causes no problem; but when no commands are displayed ( using .ENABLE QUIET) it is disconcerting to the novice user to see these messages appear for no apparent reason.

The enclosed SLP patches to the modules:

[12,10]INDERR.MAC  
[12,10]INDIMP.MAC  
[12,10]INDOPN.MAC

cause the exit display to be suppressed whenever .ENABLE QUIET is in effect upon exit from ...AT.

Sincerely;

*Robert F. Thomas*  
Robert F. Thomas

```
>;
>; PATCHES TO IND
>; TO ENABLE FULLY QUIET OPERATION
>; I.E. NO @<EOF> MESSAGE WHEN
>; .ENABLE QUIET
>; UPON EXIT
>TYPE INDERR.LST
-36
;      AST001  7-SEP-80 SUPPRESS > @<EOF> WHEN QUIET ENABLED
;
-150,153
; .ENABL  LSB                      ; AST001
EPEOF: MOV  (SP)+,R1                ; GET MESSAGE BUFFER
        MOV  #EMEOF, R2             ; GET LENGTH
        TST  .QUIET                 ; QUIET EXIT?                      ; AST001
        BNE  1$                     ; YES, SUPPRESS PROMPT          ; AST001
        CALL PRMSG1                 ; PRINT AFTER PROMPT
1$:      CLR  ERLBL                  ; NO ERROR PROCESSING
        .DSABL LSB                      ; AST001
/
>TYPE INDIMP.LST
-133
; .QUIET::WORD  0                      ; QUIET EXIT FLAG                      ; AST001
/
>TYPE INDOPN.LST
-283
        CMP  #20$, (SP)              ; ENABLE?                      ; AST001
        BNE  17$                     ; NO, THEN DISABLE          ; AST001
        INC  .QUIET                  ; MARK FOR QUIET EXIT          ; AST001
        BR   16$                     ;
17$:      CLR  .QUIET                  ; NORMAL EXIT                      ; AST001
/
>;
>; EXAMPLE OF USE:
>; ...CA. SPAWNS AT. TO PROCESS COMMANDS
>;
>TYPE SY2:[1,2]TYPE.CMD
        .ENABLE SUBSTITUTION
        .ENABLE QUIET
        .SETS F P1
        .IF F = "" .ASKS F File
        .IF F = "" .STOP
        PIP TI:='F'
>; THE ABOVE APPEARS TO FUNCTION
>; AS ANY MCR COMMAND
>;
>;
>TYPE SY2:[1,2]WHO.CMD
        .ENABLE SUBSTITUTION
        .TESTFILE TI:
; Terminal '<FILSPC>' is at '<SYDISK>' '<SYUNIT>' '<UIC>'
>WHO
>; Terminal TT005: is at DK1:[12,10]
>@ <EOF>
>
```



LABORATORIES, INC.

SOMERTON ROAD • TREVOSE, PENNSYLVANIA 19047 • U.S.A. / TELEPHONE: 215 • 355-3300 • TELEX: 84-5159

17-Nov-80

Dear Sir,

The new enhancements to the print spooler provide many options. for output however, there are no hooks available to allow programs to use these new features. The Fortran callable subroutine listed below will provide an access to the print spooler by creating a command line and SPAWNING it to the task. We have found this routine to be useful for directing spooled output to places other than the default device. It also appears to be the only way to do this. If there are any problems/suggestions I can be reached at 215-355-3300 X436.

Sincerely yours,

*Jay C. Nelson*

Jay C. Nelson  
Systems Programmer

```
.TITLE  FORTRAN CALLABLE SPOOLER
.IDENT  /VER01/
.MCALL  SPWN$,DIR$,WTSE$C,QIOW$$
.GLOBL  SPOLER
.ENABL  LC
```

```
;
;
; THIS ROUTINE WILL BUILD A COMMAND LINE FOR THE PRINT SPOOLER
; USING THE PARAMETERS PASSED FROM THE CALL
; THE FORM OF THE CALL IS :
```

```
;      CALL SPOLER (filename [,out dev][,delete][,pg len]
;                  [,jobnam][, # copies][,form #][,IDS])
```

```
; Where:
```

```
;      filename - an array containing the filename followed by
;                  a zero byte
;      out dev  - the output device (it must be a spoolable
;                  device) followed by a zero byte
;      delete   - .TRUE. to delete
;      pg len   - the implied page length for the spooler
;      jobnam   - a valid job name followed by a zero
;      # copies - the number of copies of the file to
;                  be spooled
;      form #   - the form number to use for printing
;      IDS      - 2 word array to return the status of the
;                  call
```

```
; Defaults :
```

```
;      filename - must be specified
;      out dev  - PRINT queue
;      delete   - /NODELETE
;      pg len   - infinite
;      jobnam   - none (include /NOFL)
;      # copies - 1
;      form #   - 0
```

```
; ***** NOTES
```

```
; EVENT FLAG 24. IS USED BY THIS SUBROUTINE
;
; ALL STRINGS MUST END WITH A ZERO BYTE
```

```
; ***** ERROR RETURNS POSSIBLE
```

```
; FIRST WORD OF THE IDS
```

```
; 1 - SUCCESSFUL
; 0 - ?????
; -20 - LINE TOO LONG
```

235

```

;
; -21 - SPAWN DID NOT SUCCEED (SEE NEXT PARM)
;
;
; SECOND WORD OF THE IDS
;
; IF 1ST WORD WAS -21 THIS CONTAINS THE EXIT STATUS FROM
; THE SPAWN
;
; ***** REGISTER DEFINITIONS
;
; R0 - PARAMETER COUNT (FROM FORTRAN)
; R1 - POINTER TO NEXT CHARACTER IN COMMAND LINE
; R2 - ADR OF THE SOURCE STRING (FOR PARAMETER TRANSFERS)
; or ADR OF SOURCE PARAMETER FOR CVT
; R3 - ERROR 1 REGISTER
; R4 - ERROR 2 REGISTER
; R5 - POINTER TO PARAMETER ADR BLOCK (IN FORTRAN)
;
; SET UP FOR COMMAND
SPOLER: MOV (R5)+,R0 ;LOAD THE PARM COUNT
MOV #COMLIN,R1 ;LOAD THE ADR OF THE LINE INTO
MOV #P,(R1)+ ;R1
MOV #R,(R1)+ ;LOAD PRI
MOV #I,(R1)+ ;USE R1 AS THE POINTER TO THE
MOV #',(R1)+ ;COMMAND LINE
;
; JOB NAME NEXT
CMP #JOBN,R0 ;TEST FOR JOBNAM
BGT 2$ ;DO NEXT PARM
MOV JOBN(R5),R2 ;LOAD ADR OF JOBNAM
CMP #-1,R2 ;IS THERE REALLY A JOB NAME
BEQ 2$ ;NO
1$: MOV (R2)+,(R1)+ ;MOVE THE JOBNAM INTO THE
BNE 1$ ;COMMAND LINE LOOK FOR 0
DEC R1 ;DON'T INCLUDE THE ZERO
BR 3$ ;NEXT COMMAND
2$: MOV #',(R1)+
MOV #N,(R1)+
MOV #O,(R1)+ ;MOVE /NOFL INTO THE
MOV #F,(R1)+ ;BUFFER IF NO JOBNAM
MOV #L,(R1)+
;
; LENGTH OF THE PAGE
3$: CMP #PGL,R0
BGT N1 ;SKIP PAGE LENGTH
MOV PGLA(R5),R2 ;LOAD ADR OF PAG LEN
CMP #-1,R2 ;IS IT REALLY THE ADR
BEQ N1 ;NO - SKIP PAG LEN
MOV #',(R1)+
MOV #L,(R1)+
MOV #E,(R1)+ ;MOVE /LE: INTO BUFFER
MOV #',(R1)+
JSR PC,CVT ;CONVERT DEC TO ASCII
;

```

```

; FORM NUMBER
N1: CMP #FRM,R0
BGT N1A ;SKIP FORM NUMBER
MOV FRMA(R5),R2 ;LOAD ADR OF FORM NUMBER
CMP #-1,R2 ;IS IT REALLY THE ADR
BEQ N1A ;NO - SKIP FORM NUMBER
MOV #',(R1)+
MOV #F,(R1)+ ;MOVE /FO: INTO BUFFER
MOV #O,(R1)+
MOV #',(R1)+
JSR PC,CVT ;CONVERT DEC TO ASCII
;
; MOVE FILENAME
N1A: MOV #',(R1)+ ;INCLUDE EQUALS
;
CMP #FIL,R0 ;IS THERE ONE
BGT ERROR ;NO - ERROR
MOV FILA(R5),R2 ;LOAD ADR OF FILNAME
CMP #-1,R2 ;IS THE NO NAME
BEQ ERROR ;DEFAULT NOT ALLOWED
1$: MOV (R2)+,(R1)+ ;MOVE FILNAME UNTIL ZERO
BNE 1$
DEC R1 ;DON'T INCLUDE ZERO
;
; ADD FILE SWITCHES
;
; COPIES /CO:n
CMP #COP,R0 ;IS THERE A /CO
BGT N2 ;NO - NEXT COMMAND
MOV COPA(R5),R2 ;GET ADR OF COP
CMP #-1,R2 ;NOT REALLY
BEQ N2 ;NO - NEXT COMMAND
MOV #',(R1)+
MOV #C,(R1)+
MOV #O,(R1)+
MOV #',(R1)+
JSR PC,CVT ;CONVERT DEC TO ASCII
;
; DELETE ?
N2: CMP #DEL,R0 ;IS THERE A DELETE
BGT 4$
MOV DELA(R5),R2 ;ARE YOU SURE
CMP #-1,R2 ;NO - OUT
BEQ 4$
TST (R2) ;SHOULD WE ADD /DEL
BEQ 4$ ;NO - OUT
MOV #',(R1)+
MOV #D,(R1)+ ; 0 - FALSE
MOV #E,(R1)+ ; <0 - TRUE
MOV #L,(R1)+
;
; DO THE SPAWN
4$: SUB #COMLIN,R1 ;SUB OFF THE OFFSET
CMP #79.,R1 ;LINE TOO LONG
BLT LIN1 ;YES - ERROR

```



```

MOV R1,EXCMD+S.PWCL ;LOAD LINE LENGTH INTO THE DPB
;DEBUG QIOWSS #IO.WVB,#5,#24.,,<#COMLIN,R1,#40.>
DIR$ #EXCMD,DRERR ;SPAWN THE COMMAND
WTSE$C 24. ;WAIT FOR QMG TO FINISH
CMP #1,EXSTAT ;DID IT COMPLETE SUCCESSFULLY
BNE SPERR ;NO - ERROR
CMP #IDS,R0 ;IS THERE AN IDS
BGT EXIT ;NO - OUT
MOV IDSA(R5),R2 ;SURE ?
CMP #-1,R2
BEQ EXIT ;NO - OUT
MOV #1,(R2)+ ;LOAD SUCCESS IN IDS(1)
CLR (R2) ;CLR THE IDS(2)
EXIT: RTS PC ;RETURN TO HOME
;
; ***** ERROR TRAPS
;
; LINE TOO LONG
LIN1: MOV #-20.,R3
CLR R4
BR ERROR
;
; SPAWN ERROR
SPERR: MOV #-21.,R3
MOV EXSTAT,R4
BR ERROR
;
; DIRECTIVE ERROR (JSR ENTRY)
DRERR: TST (SP)+
MOV $DSW,R3 ;LOAD THE DIREC STAT WORD
CLR R4
BR ERROR
;
; STICK R3 AND R4 IN IDS
ERROR: CMP #IDS,R0
BGT 1$ ;SKIP ERROR IF NOT THERE
MOV IDSA(R5),R2
CMP #-1,R2 ;SURE ?
BEQ 1$ ;NO -EXIT
MOV R3,(R2)+ ;LOAD THE ERR VAL
TST R4
BEQ 1$
MOV R4,(R2)+ ;LOAD THE OTHER ONE IF THERE
1$: RTS PC
;
; ***** SUBROUTINES
;
; CVT - Convert the number pointed to by R2 to an
; ASCII string and store the string in the
; buffer pointed to by R1.
CVT: MOV R0,-(SP) ;SAVE R0
MOV R1,R0 ;MOV THE STR ADR TO R0
MOV (R2),R1 ;LOAD # IN R1
MOV #30,R2 ;LOAD PARM

```

```

SWAB R2 ;SWAP
ADD #10.,R2 ;LOAD RADIX IN LOWER BYTE
CALL $CBTA ;CALL THE SYSTEM CVT ROUTINE
MOV R0,R1 ;RESTORE R1
MOV (SP)+,R0 ; " R0
RTS PC ;RETURN
;
; BUFFERS
;
COMLIN: .BLKB 132.
EXSTAT: .BLKW 8.
;
EXCMD: SPWN$ MCR.....,24.,EXSTAT,COMLIN,0,0
;
; SYMBOL DEFINITION
FIL = 1 ;SYMBOLS FOR THE PARM
OUTD = 2 ;LOCATIONS
DEL = 3
PGL = 4
JOBNA = 5
COP = 6
FRM = 7
IDS = 10
;
FILA = 0 ;ADR OFFSETS FOR PARM
OUTDA = 2
DELA = 4
PGLA = 6
JOBNA = 10
COPA = 12
FRMA = 14
IDSA = 16
;
.END

```

[16 SEP 1980

Editor, Multi-Tasker,  
 c/o Colin A. Mercer,  
 Tennant Post,  
 High St.,  
 FAREHAM, PO16 7BQ

Dear Sir,

We operate a PDP 11/34 installation running under RSX-11M V3.1 with 7 RK05 Disc drives. We have developed the attached FORTRAN program to retrieve Files-11 disc files accidentally deleted in circumstances where the archive copy is either not available or obsolete.

The technique does not attempt to re-build the file structure of the disc but does enable the user to make a file copy from the previously allocated blocks of the deleted file. Obviously no file creation should take place on the disc with the deleted file(s) until an image copy has been made of that disc.

The multi-drive configuration of our installation, with normally one drive dedicated to one user, has enabled this condition to be met relatively easily.

Adopt the following procedure :-

1. Take a PRESRV copy of the disc with the deleted file(s).
2. Log onto a privileged UIC and assign to a volume with that UIC.
3. Mount the disc copy on a spare drive unlocked.  
 e.g. MOU DKn:/DVR/UNL
4. Run UNDELETE.

5. The first part of the program will produce a spooled listing of possible deleted files. Identify the file header of the correct file and press <cr> to continue.

6. When UNDELETE has finished copy the reclaimed file by file ID. e.g.

PIP NewFile = DKn:/FI:x:y

7. Then delete the reclaimed file i.e.

PIP DKn:/FI:x:y/DE

Ignore the 'Bad File Header' message.

A program listing and a sample program run are attached.

Sincerely,

*P. G. Hansell*

P.G. Hansell,  
 Software Engineer,  
 Plessey Assessment Services,  
 15th September 1980.

HEL

## PROCEDURE

DUMP OF VIRTUAL BLOCK :

145

Please enter your Initials : PH  
Please enter your Job No. : 052500  
Please enter your Op No. : 300  
Please enter Disc Drive Unit No. : 0

UIC Set to [10 ,10 ]

15-SEP-80 09:55:38

Hello Again

>SET /UIC=[1,2]  
>DMO DK2:  
>MOU DK2:/OVR/UNL  
>RUN DK1:\$UNDELETE  
Default Device : DK0  
Device No. for Search ? : 2  
Filename ? : TEST  
File Type ? : DAT  
Index Bit Map Size = : 1 Blocks

Pausing while listing checked ...

Start Reclamation ? [Y/N]: Y  
Virtual Block No. ? : 145  
TT5 -- STOP Complete  
>

PIP TEST.NEW/LI  
PIP -- NO SUCH FILE(S)

>PIP TE  
ST.NEW=DK2:/FI:142:5  
>PIP TEST.NEW/LI

DIRECTORY DK0:[1,2]  
15-SEP-80 09:57

TEST.NEW:1 1. 15-SEP-80 09:57

TOTAL OF 1./1. BLOCKS IN 1. FILE

>PIP DK2:/FI:142:5/DE  
PIP -- FAILED TO MARK FILE FOR DELETE  
DK2:. -- BAD FILE HEADER  
>PIP DK2:/FI:142:5/DE  
PIP -- FAILED TO MARK FILE FOR DELETE  
DK2:. -- NO SUCH FILE  
>BYE  
USER PH LOGGED OFF

ON FOR 0 HOURS 3 MINS

15-SEP-80 09:58:44

242

0	27027	142	5	401	40101	160000	0	1002
20	22	0	1	0	1	24	0	0
40	0	0	0	0	0	0	0	76733
60	76400	0	14474	5	1	32461	42523	34120
100	30060	32071	30065	30464	51465	50105	30070	34460
120	32464	32060	0	0	0	0	0	0
140	0	1401	146002	0	10373	0	0	0
160	0	0	0	0	0	0	0	0
200	0	0	0	0	0	0	0	0
220	0	0	0	0	0	0	0	0
240	0	0	0	0	0	0	0	0
260	0	0	0	0	0	0	0	0
300	0	0	0	0	0	0	0	0
320	0	0	0	0	0	0	0	0
340	0	0	0	0	0	0	0	0
360	0	0	0	0	0	0	0	0
400	0	0	0	0	0	0	0	0
420	0	0	0	0	0	0	0	0
440	0	0	0	0	0	0	0	0
460	0	0	0	0	0	0	0	0
500	0	0	0	0	0	0	0	0
520	0	0	0	0	0	0	0	0
540	0	0	0	0	0	0	0	0
560	0	0	0	0	0	0	0	0
600	0	0	0	0	0	0	0	0
620	0	0	0	0	0	0	0	0
640	0	0	0	0	0	0	0	0
660	0	0	0	0	0	0	0	0
700	0	0	0	0	0	0	0	0
720	0	0	0	0	0	0	0	0
740	0	0	0	0	0	0	0	0
760	0	0	0	0	0	0	0	32361

File No. : 142  
Seq. No. : 5  
UIC = 100,101  
Variable Records  
Carriage Control Characteristic  
Record size = 22  
Filename : TEST  
Extension: DAT  
Version : 5  
Revision : 1  
Revision Date : 15 SEP 80  
Revision Time : 09:45:04  
Creation Date : 15 SEP 80  
Creation Time : 09:45:04

DUMP of RE-CLAIMED

FILE HEADER

243

```

C
C      Source of UNDELETE.FTN - Reclaim deleted files
C
0001      LOGICAL*1 IDAT(12),FILEB(20),IDEV
C
0002      INTEGER SNAME(5),STYPE(2),RNAME(3),RTYPE
0003      INTEGER MAPSZ, IANS, YES, VBN, SEQNO, IMAX
0004      INTEGER FUN, LUN, EFN, PRI, ISB(2), PRL(6), IDS, BUFF(256)
C
0005      COMMON /A/ BUFF
C
0006      DATA FILEB /'D','K','X',' ',' ','L','0',' ',' ','0',' ','J',
+ 'I','N','D','E','X','F',' ',' ','S','Y','S',' ',' '/
0007      DATA YES /'Y' /
C
0008      IMAX = 1 ! Flag for Full search
C
C      Assign List File
C
0009      CALL ASSIGN(2,'LIST.DMP')
0010      CALL FDBSET(2,'N')
C
C      Get Default device
C
0011      CALL GETLUN(2,IDAT,IDS)
0012      TYPE 500,(IDAT(I),I=1,3)
C
C      TYPE 502
0013      ACCEPT 504,IDEV
0014      ACCEPT 504,IDEV
0015      IF (IDEV.NE.IDAT(3)) GO TO 10
0016      TYPE 506
0017      GO TO 5
C
0018      FILEB(3) = IDEV + '60'
C
C      Assign Index File
C
0019      CALL ASSIGN (1,FILEB,19)
0020      CALL FDBSET(1,'R')
C
C      Get Filename and Type
C
0021      TYPE 507
0022      ACCEPT 508,SNAME
0023      TYPE 509
0024      ACCEPT 510,STYPE
C
C      Convert to Radix 50
C
0025      CALL IRAD50(9,SNAME,RNAME)
0026      CALL IRAD50(3,STYPE,RTYPE)
C
0027      IBLK = 2 ! HOME BLOCK
C
C      Kill error message on Fortran read
C
0028      CALL ERRSET (24,.,TRUE,.,FALSE,.,TRUE,.,FALSE,.,31)

```

## SOURCE LISTING

```

0030      FUN = '10400 ! ID,RVB
0031      LUN = 1
0032      EFN = 1
0033      PRI = 50
      C
0034      CALL GETADR(PRL(1),BUFF(1))
0035      PRL(2) = 512
0036      PRL(3) = 0
0037      PRL(4) = 0 ! BLOCK HIGH
0038      PRL(5) = IBLK ! BLOCK LOW
0039      PRL(6) = 0
      C
0040      READ(1,512,ERR=18) ! Force OPEN (with error)
0041  18    CALL QIO(FUN,LUN,EFN,PRI,ISB,PRL,IDS) ! Read Home Block
0042      CALL WAITFR(EFN,JDS)
      C
0043      IF (ISB(1),EQ.1) GO TO 20
0045  19    IF (ISB(1),NE,'366) WRITE(5,534)ISB(1) ! Don't print EOF error
0047      GO TO 60
      C
      C      Calc. No. of Index File Bit Map Blocks
      C
0048  20    MAPSZ = BUFF(1)
0049      TYPE 520,MAPSZ
0050      IBLK = 8 + MAPSZ
      C
      C      Read Index File until error
      C
0051  21    PRL(5) = IBLK
0052      CALL QIO (FUN,LUN,EFN,PRI,ISB,PRL,IDS)
0053      CALL WAITFR(EFN,JDS)
0054      IF (ISB(1),EQ.1) GO TO 22
0056      GO TO 19
      C
      C      Compare Names
      C
0057  22    DO 30 I = 1,3
0058      IF (RNAME(I),NE,BUFF(I+23)) GO TO 32
0060      CONTINUE
      C
      C      Compare File Type
      C
0061      IF (RTYPE,EQ,BUFF(27)) GO TO 35
      C
0063  32    IBLK = IBLK + 1
0064      GO TO 21
      C
      C      Dump File Header (word format)
      C
0065  35    WRITE(2,516)IBLK
0066      J = 1
0067      DO 40 I = '0','760','20
0068      WRITE(2,518)I,(BUFF(K),K=J,J+7)
0069      J = J + 8
0070  40    CONTINUE
      C

```

FORTRAN IV V01B-02A MON 15-SEP-80 10:09:44 PAGE 003  
CORE=14K, UIC=[100,100] ,LP=UNDELETE

```

C      Decode File Header
C
0071      CALL PRINTH
0072      IF (IMAX.EQ.1)GO TO 32
0074 60    CALL SPOOL(2,IERR) ! Spool List File to LP:
0075      IF (IMAX.EQ.0) STOP 'Complete'
C
C      Reclamation Section
C
0077      TYPE 522
0078      ACCEPT 523, IANS
0079      TYPE 524
0080      ACCEPT 523, IANS
0081      IF (IANS.NE.YES) STOP
C
C      Re-open File for write (NB: MOU DKn:/UNL)
C
0083      CALL CLOSE(1)
0084      CALL ASSIGN(1,FILEB,19)
0085      CALL FDBSET (1,'M')
0086      READ(1,512,ERR=65) ! Force Open
C
C      Get Virtual Block No. of Deleted File Header
C
0087 65    TYPE 526
0088      ACCEPT 528,VBN
0089      IF (VBN.EQ.0) GO TO 90 ! Allow Panic exit
0091      IBLK = VBN
0092      PRL(5) = IBLK
0093      CALL QIO (FUN,LUN,EFN,PRI,ISB,PRL,IDS)
0094      CALL WAITFR(EFN,JDS)
0095      IF (ISB(1).EQ.1) GO TO 70
0097      GO TO 19
C
C      Check name against Input
C
0098 70    DO 71 I = 1,3
0099      IF (RNAME(I).NE.BUFF(I+23)) GO TO 73
0101 71    CONTINUE
C
C      Check File Type
C
0102      IF (RTYPE.EQ.BUFF(27)) GO TO 74
C
0104 73    TYPE 530
0105      GO TO 65
C
0106 74    IF (BUFF(7).AND."100000") GO TO 76
0108      TYPE 532
0109      GO TO 65
C
0110 76    BUFF(7) = BUFF(7) .AND. " 200 ! Keep Contiguous
0111      BUFF(2) = VBN - 2 - MAPSZ ! File Sequence No.
0112      BUFF(256) = 0
0113      ICHECK = 0
C

```

FORTRAN IV V01B-02A MON 15-SEP-80 10:09:44 PAGE 004  
CORE=14K, UIC=[100,100] ,LP=UNDELETE

```

C      Re-calculate checksum
C
0114      DO 78 I = 1,255
0115      ICHECK = ICHECK + BUFF(I)
0116 78    CONTINUE
C
C      Re-write Header Block
C
0117      BUFF(256) = ICHECK
0118      FUN = '11000 ! IO.WVB
0119      CALL QIO (FUN,LUN,EFN,PRI,ISB,PRL,IDS)
0120      CALL WAITFR(EFN,JDS)
0121      IF (ISB(1).EQ.1) GO TO 80
0123      GO TO 19
C
C      Re-list Header Block for Verification
C
0124 80    CALL CLOSE(2)
0125      CALL ASSIGN(2,'LIST.DMP')
0126      CALL FDBSET (2,'N')
0127      FUN = '10400 ! IO.RVB
0128      IMAX = 0 ! Only one more time
0129      GO TO 21
0130 90    STOP
0131 500    FORMAT(1X,'Default Device : ',2A1,I1)
0132 502    FORMAT(1X,'Device No. for Search ? : ',%)
0133 504    FORMAT(01)
0134 506    FORMAT(1X,'Error - Cannot examine default device')
0135 507    FORMAT(1X,'Filename ? : ',%)
0136 508    FORMAT(5A2)
0137 509    FORMAT(1X,'File Type ? : ',%)
0138 510    FORMAT(2A2)
0139 512    FORMAT(/)
0140 516    FORMAT('IDUMP OF VIRTUAL BLOCK : ',010//)
0141 518    FORMAT(1X,010,5X,80B)
0142 520    FORMAT(1X,'Index Bit Map Size = : ',05,' Blocks')
0143 522    FORMAT(//1X,'Pausing while listings checked ...')
0144 523    FORMAT(A1)
0145 524    FORMAT(//1X,'Start Reclamation ? [Y/N]: ',%)
0146 526    FORMAT(1X,'Virtual Block No. ? : ',%)
0147 528    FORMAT(010)
0148 530    FORMAT(1X,'Error - Filename/type mismatch')
0149 532    FORMAT(1X,'Error - File not marked for delete')
0150 534    FORMAT(1X,'Error Status Return : ',08)
0151      END

```

FORTRAN IV STORAGE MAP

NAME	OFFSET	ATTRIBUTES
IDAT	000006	LOGICAL*1 ARRAY (12)
FILEB	000022	LOGICAL*1 ARRAY (20)
SNAME	000046	INTEGER*2 ARRAY (5)
STYPE	000060	INTEGER*2 ARRAY (2)
RNAME	000064	INTEGER*2 ARRAY (3)
ISB	000072	INTEGER*2 ARRAY (2)
PRL	000076	INTEGER*2 ARRAY (6)
IDEV	001070	LOGICAL*1 VARIABLE
RTYPE	001072	INTEGER*2 VARIABLE
MAPSZ	001074	INTEGER*2 VARIABLE
IAN5	001076	INTEGER*2 VARIABLE
YES	000112	INTEGER*2 VARIABLE
VBN	001100	INTEGER*2 VARIABLE
SEQNO	001102	INTEGER*2 VARIABLE
IMAX	001104	INTEGER*2 VARIABLE
FUN	001106	INTEGER*2 VARIABLE
LUN	001110	INTEGER*2 VARIABLE
EFN	001112	INTEGER*2 VARIABLE
PRI	001114	INTEGER*2 VARIABLE
IDS	001116	INTEGER*2 VARIABLE
ASSIGN	000000	REAL*4 PROCEDURE
FDGET	000000	REAL*4 PROCEDURE
GETLUN	000000	REAL*4 PROCEDURE
AD50	001120	INTEGER*2 VARIABLE
BLK	001122	INTEGER*2 VARIABLE
ERRSET	000000	REAL*4 PROCEDURE
GETADR	000000	REAL*4 PROCEDURE
QIO	000000	REAL*4 PROCEDURE
WAITFR	000000	REAL*4 PROCEDURE
JDS	001124	INTEGER*2 VARIABLE
J	001126	INTEGER*2 VARIABLE
K	001130	INTEGER*2 VARIABLE
PRINTH	000000	REAL*4 PROCEDURE
SPDOL	000000	REAL*4 PROCEDURE
IERR	001132	INTEGER*2 VARIABLE
CLOSE	000000	REAL*4 PROCEDURE
ICHECK	001134	INTEGER*2 VARIABLE

COMMON BLOCK /A/ LENGTH 001000

BUFF 000000 INTEGER\*2 ARRAY (256)

FORTRAN IV V01B-02A  
CORE=14K, UIC=[100,100]

MON 15-SEP-80 10:09:51

PAGE 001  
LP=UNDELETE

```

C
C
C
0001 SUBROUTINE PRINTH
C
0002 LOGICAL*1 MEM,GRP,USER,SYS,TYPE,ATTR,RDATE(7),CDATE(7)
0003 LOGICAL *1 RTIM(6)
C
0004 INTEGER BUFF(256),GRPI,MEMI
0005 INTEGER OFFSET,FILENO,SEQNO,D1,PROTEC,SIZE,D2(5),D3(9)
0006 INTEGER NAME(3),EXT,VER,REV,CTIM(3),D4(4),DNAME(5)
0007 INTEGER DEXT(2)
C
0008 COMMON /A/ OFFSET,FILENO,SEQNO,D1,MEM,GRP,PROTEC,USER,SYS,
+TYPE,ATTR,SIZE,D2,D3,NAME,EXT,VER,REV,RDATE,RTIM,CDATE,
+CTIM
C
0009 GRPI = GRP
0010 MEMI = MEM
0011 GRPI = IAND (GRPI,"377)
0012 MEMI = IAND (MEMI,"377)
C
0013 WRITE(2,600)FILENO
0014 WRITE(2,601)SEQNO
0015 WRITE(2,602)GRPI,MEMI
0016 IF (USER.EQ."100) WRITE(2,603)
0018 IF (USER.EQ."200) WRITE(2,604)
0020 IF (SYS.AND."100) WRITE(2,620)
0022 IF (SYS.AND."200) WRITE(2,621)
0024 IF (TYPE.EQ.1) WRITE(2,605)
0026 IF (TYPE.EQ.2) WRITE(2,606)
0028 IF (TYPE.EQ.3) WRITE(2,607)
0030 IF (ATTR.AND.1) WRITE(2,608)
0032 IF (ATTR.AND.2) WRITE(2,609)
0034 IF (ATTR.AND.8) WRITE(2,610)
0036 WRITE(2,611)SIZE
C
0037 CALL R50ASC(9,NAME,DNAME)
0038 WRITE(2,612)DNAME
C
0039 CALL R50ASC(3,EXT,DEXT)
0040 WRITE(2,613)DEXT
C
0041 WRITE(2,614)VER
C
0042 WRITE(2,615)REV
C
0043 WRITE(2,616)RDATE
C
0044 WRITE(2,617)RTIM
C
0045 WRITE(2,618)CDATE
C
0046 WRITE(2,619)CTIM
0047 RETURN
0048 600 FORMAT(///' File No. : ',05)

```

FORTRAN IV V01B-02A MON 15-SEP-80 10:09:51 PAGE 002  
CORE=14K, UIC=I100,100J ,LP=UNDELETE

```
0049 601 FORMAT(' Seq. No. : '05)
0050 602 FORMAT(' UIC = ',03,',',03)
0051 603 FORMAT(' File Not Properly Closed')
0052 604 FORMAT(' File Losically Contiguous')
0053 605 FORMAT(' Fixed Records')
0054 606 FORMAT(' Variable Records')
0055 607 FORMAT(' Sequential Records')
0056 608 FORMAT(' Fortran Characteristic')
0057 609 FORMAT(' Carriage Control Characteristic')
0058 610 FORMAT(' Records may not cross Blocks')
0059 611 FORMAT(' Record size = ',05)
0060 612 FORMAT(' Filename : ',5A2)
0061 613 FORMAT(' Extension: ',2A2)
0062 614 FORMAT(' Version : ',03)
0063 615 FORMAT(' Revision : ',03)
0064 616 FORMAT(' Revision Date : ',2A1,',',3A1,',',2A1)
0065 618 FORMAT(' Creation Date : ',2A1,',',3A1,',',2A1)
0066 617 FORMAT(' Revision Time : ',2A1,',',2A1,',',2A1)
0067 619 FORMAT(' Creation Time : ',A2,',',A2,',',A2)
0068 620 FORMAT(' Bad Data in File')
0069 621 FORMAT(' File marked for delete')
0070      END
```

FORTRAN IV STORAGE MAP

NAME	OFFSET	ATTRIBUTES
BUFF	000014	INTEGER*2 ARRAY (256)
D4	001014	INTEGER*2 ARRAY (4)
NAME	001024	INTEGER*2 ARRAY (5)
EXT	001036	INTEGER*2 ARRAY (2)
CRPI	002100	INTEGER*2 VARIABLE
MEMI	002102	INTEGER*2 VARIABLE
IAND	000000	INTEGER*2 PROCEDURE
R50ASC	000000	REAL*4 PROCEDURE

COMMON BLOCK /A/ LENGTH 000124

OFFSET	000000	INTEGER*2 VARIABLE
FILENO	000002	INTEGER*2 VARIABLE
SEQNO	000004	INTEGER*2 VARIABLE
D1	000006	INTEGER*2 VARIABLE
MEM	000010	LOGICAL*1 VARIABLE
GRP	000011	LOGICAL*1 VARIABLE
PROTEC	000012	INTEGER*2 VARIABLE
USER	000014	LOGICAL*1 VARIABLE
SYS	000015	LOGICAL*1 VARIABLE
TYPE	000016	LOGICAL*1 VARIABLE
ATTR	000017	LOGICAL*1 VARIABLE
SIZE	000020	INTEGER*2 VARIABLE
D2	000022	INTEGER*2 ARRAY (5)
D3	000034	INTEGER*2 ARRAY (9)
NAME	000056	INTEGER*2 ARRAY (3)
EXT	000064	INTEGER*2 VARIABLE
VER	000066	INTEGER*2 VARIABLE
REV	000070	INTEGER*2 VARIABLE
RDATE	000072	LOGICAL*1 ARRAY (7)
RTIM	000101	LOGICAL*1 ARRAY (6)
CDATE	000107	LOGICAL*1 ARRAY (7)
CTIM	000116	INTEGER*2 ARRAY (3)

250

The full tuition is payable in advance and includes the cost of course materials. Courses not listed may be offered on a demand basis. Schedule and prices are subject to change without notice. For a description of each course, please refer to Educational Services 1981 Customer Training Catalog or our RSX-11M-M-PLUS Product Brochure. All courses listed on these pages that are prefaced with a P are prerequisites to our RSX-11M-M-PLUS training curricula. Please consult your local Training Center for more information on prerequisite training.

## RSX-11M/M-PLUS TRAINING PROGRAMS SAN FRANCISCO (408) 727-0200 Ext. 2142

Canadian Customers: All registrations for U.S. courses should be made through the Ottawa (Kanata) Registrar, (813) 592-5111.

Course #	Title	Length	Start Day	Tuition	Credit	October	November	December	January	February	March
EY-J2022-AD	Computer Programming Concepts	5	MO	\$ 520	1						
P EY-J2114-AD	Introduction to Minicomputers A/V	~5		400	1						
P EY-J2016-AD	Introduction to Minicomputers	5	MO	520	1						
P EY-J2116-AD	Introduction to the PDP-11 A/V	~5		450	1						
P EY-J2024-AD	PDP-11 Assembly Language Programming	5	MO	580	1						
P EY-J2050-AD	Programming in MACRO-11	5	MO	630	1						
P EY-J2048-AD	PDP-11 Concepts	3		480	1						
EY-J2140-AD	RSX-11M User	5	MO	630	1						
EY-J2232-AD	RSX-11M-PLUS User	5	MO	630	1						
EY-J0002-AD	Programming in PDP-11 COBOL	5	MO	630	1						
EY-J2230-AD	Programming in BASIC-PLUS-2	5	MO	630	1						
EY-J2218-AD	Programming with RMS-11	5	MO	690	1						
EY-J0016-AD	Programming in FORTRAN IV	5	MO	630	1						
EY-J2142-AD	RSX-11M Programmer	5	MO	630	1						
EY-J2144-AD	RSX-11M System Programmer	5	MO	690	1						
EY-J2278-AD	RSX-11M-PLUS Programmer	5	MO	630	1						
EY-J2234-AD	RSX-11M-PLUS System Programmer	5	MO	690	1						
EY-J2166-AD	RSX-11M System Manager	2		350	1						
EY-J2068-AD	RSX-11M/IAS DECnet User	5	MO	750	1						
EY-J2006-AD	RSX-11M Operator	3		520	1						

## LOS ANGELES (213) 937-3870

Course #	Title	Length	Start Day	Tuition	Credit	October	November	December	January	February	March
P EY-J2022-AD	Commercial Programming Concepts	5	MO	\$ 520	1						
P EY-J2114-AD	Introduction to Minicomputers A/V	~5		400	1						
P EY-J2016-AD	Introduction to Minicomputers	5	MO	520	1						
P EY-J2116-AD	Introduction to the PDP-11 A/V	~5		450	1						
P EY-J2024-AD	PDP-11 Assembly Language Programming	5	MO	580	1						
P EY-J2050-AD	Programming in MACRO-11	5	MO	630	1						
P EY-J2048-AD	PDP-11 Concepts	3		480	1						
EY-J2140-AD	RSX-11M User	5	MO	630	1						
EY-J2232-AD	RSX-11M-PLUS User	5	MO	630	1						
EY-J0002-AD	Programming in PDP-11 COBOL	5	MO	630	1						
EY-J2230-AD	Programming in BASIC-PLUS-2	5	MO	630	1						
EY-J2134-AD	BASIC-PLUS-2 Prog. with RMS-11	5	MO	630	1						
EY-J0016-AD	Programming in FORTRAN IV	5	MO	630	1						
EY-J2142-AD	RSX-11M Programmer	5	MO	630	1						
EY-J2144-AD	RSX-11M System Programmer	5	MO	690	1						
EY-J2278-AD	RSX-11M-PLUS Programmer	5	MO	630	1						
EY-J2234-AD	RSX-11M-PLUS System Programmer	5	MO	690	1						
EY-J2166-AD	RSX-11M System Manager	2	M/W	350	1						
EY-J2068-AD	RSX-11M/IAS DECnet User	5	MO	750	1						
EY-J2006-AD	RSX-11M Operator	3	WE	520	1						

## DALLAS (214) 620-2051 Ext. 301, 303

Course #	Title	Length	Start Day	Tuition	Credit	October	November	December	January	February	March
P EY-J2022-AD	Commercial Programming Concepts	5	MO	\$ 520	1						
P EY-J2114-AD	Introduction to Minicomputers A/V	~5		400	1						
P EY-J2016-AD	Introduction to Minicomputers	5	MO	520	1						
P EY-J2116-AD	Introduction to the PDP-11 A/V	~5		450	1						
P EY-J2024-AD	PDP-11 Assembly Language Programming	5	MO	580	1						
P EY-J2050-AD	Programming in MACRO-11	5	MO	630	1						
P EY-J2048-AD	PDP-11 Concepts	3		480	1						
EY-J2140-AD	RSX-11M User	5	MO	630	1						
EY-J2232-AD	RSX-11M-PLUS User	5	MO	630	1						
EY-J0002-AD	Programming in PDP-11 COBOL	5	MO	630	1						
EY-J2230-AD	Programming in BASIC-PLUS-2	5	MO	630	1						
EY-J2134-AD	BASIC-PLUS-2 Prog. with RMS-11	5	MO	630	1						
EY-J0016-AD	Programming in FORTRAN IV	5	MO	630	1						
EY-J2142-AD	RSX-11M Programmer	5	MO	630	1						
EY-J2144-AD	RSX-11M System Programmer	5	MO	690	1						
EY-J2278-AD	RSX-11M-PLUS Programmer	5	MO	630	1						
EY-J2234-AD	RSX-11M-PLUS System Programmer	5	MO	690	1						
EY-J2166-AD	RSX-11M System Manager	2	MO	350	1						
EY-J2068-AD	RSX-11M/IAS DECnet User	5	MO	750	1						
EY-J2006-AD	RSX-11M Operator	3		520	1						

On-Site courses are available on request. When conducted at your job site, these courses should provide you with a more cost-effective training method. For standard software courses, our standard price per course week (\$6500 for RSX-11M/M-PLUS courses — U.S. Domestic Prices) will cover up to 10 students; prices increase with additional students. For more information about On-Site courses, consult your local Training Center.

## RSX-11M/M-PLUS TRAINING PROGRAMS (Continued) WASHINGTON, D.C. (301) 459-7900 Ext. 2580, 2582

Canadian Customers: All registrations for U.S. courses should be made through the Ottawa (Kanata) Registrar, (613) 592-5111.

Course #	Title	Length	Start Day	Training Credit	October	November	December	January	February	March
EY-J2022-AD	Commercial Programming Concepts	5 MO	\$ 520	1						
EY-J2114-AD	Introduction to Minicomputers A/V	~5	400	1						
EY-J2016-AD	Introduction to Minicomputers	5 MO	520	1						
EY-J2116-AD	Introduction to the PDP-11 A/V	~5	450	1						
EY-J2024-AD	PDP-11 Assembly Language Programming	5 MO	580	1						
EY-J2050-AD	Programming in MACRO-11	5 MO	630	1						
EY-J2048-AD	PDP-11 Concepts	3	460	1						
EY-J2140-AD	RSX-11M User	5 MO	630	1						
EY-J2232-AD	RSX-11M-PLUS User	5 MO	630	1						
EY-J0002-AD	Programming in PDP-11 COBOL	5 MO	630	1						
EY-J2230-AD	Programming in BASIC-PLUS-2	5 MO	630	1						
EY-J2134-AD	BASIC-PLUS-2 Prog. with RMS-11	5 MO	630	1						
EY-J0016-AD	Programming in FORTRAN IV	5 MO	630	1						
EY-J2142-AD	RSX-11M Programmer	5 MO	630	1						
EY-J2144-AD	RSX-11M System Programmer	5 MO	680	1						
EY-J2278-AD	RSX-11M-PLUS Programmer	5 MO	630	1						
EY-J2234-AD	RSX-11M-PLUS System Programmer	5 MO	680	1						
EY-J2166-AD	RSX-11M System Manager	2	350	1						
EY-J2068-AD	RSX-11M/IAS DECON User	5 MO	750	1						
EY-J2006-AD	RSX-11M Operator	3	MO	520	1					

## NEW YORK (212) 971-3545

Course #	Title	Length	Start Day	Training Credit	October	November	December	January	February	March
EY-J2022-AD	Commercial Programming Concepts	5 MO	\$ 520	1						
EY-J2114-AD	Introduction to Minicomputers A/V	~5	400	1						
EY-J2016-AD	Introduction to Minicomputers	5 MO	520	1						
EY-J2116-AD	Introduction to the PDP-11 A/V	~5	450	1						
EY-J2024-AD	PDP-11 Assembly Language Programming	5 MO	580	1						
EY-J2050-AD	Programming in MACRO-11	5 MO	630	1						
EY-J2048-AD	PDP-11 Concepts	3	460	1						
EY-J2140-AD	RSX-11M User	5 MO	630	1						
EY-J2232-AD	RSX-11M-PLUS User	5 MO	630	1						
EY-J0002-AD	Programming in PDP-11 COBOL	5 MO	630	1						
EY-J2230-AD	Programming in BASIC-PLUS-2	5 MO	630	1						
EY-J2134-AD	BASIC-PLUS-2 Prog. with RMS-11	5 MO	630	1						
EY-J0016-AD	Programming in FORTRAN IV	5 MO	630	1						
EY-J2142-AD	RSX-11M Programmer	5 MO	630	1						
EY-J2144-AD	RSX-11M System Programmer	5 MO	680	1						
EY-J2278-AD	RSX-11M-PLUS Programmer	5 MO	630	1						
EY-J2234-AD	RSX-11M-PLUS System Programmer	5 MO	680	1						
EY-J2166-AD	RSX-11M System Manager	2	MO	350	1					
EY-J2068-AD	RSX-11M/IAS DECON User	5 MO	750	1						
EY-J2006-AD	RSX-11M Operator	3	M/W	520	1					

## CHICAGO (312) 640-5520

Course #	Title	Length	Start Day	Training Credit	October	November	December	January	February	March
EY-J2022-AD	Commercial Programming Concepts	5 MO	\$ 520	1						
EY-J2114-AD	Introduction to Minicomputers A/V	~5	400	1						
EY-J2016-AD	Introduction to Minicomputers	5 MO	520	1						
EY-J2116-AD	Introduction to the PDP-11 A/V	~5	450	1						
EY-J2024-AD	PDP-11 Assembly Language Programming	5 MO	580	1						
EY-J2050-AD	Programming in MACRO-11	5 MO	630	1						
EY-J2048-AD	PDP-11 Concepts	3	460	1						
EY-J2140-AD	RSX-11M User	5 MO	630	1						
EY-J2232-AD	RSX-11M-PLUS User	5 MO	630	1						
EY-J0002-AD	Programming in PDP-11 COBOL	5 MO	630	1						
EY-J2230-AD	Programming in BASIC-PLUS-2	5 MO	630	1						
EY-J2134-AD	BASIC-PLUS-2 Prog. with RMS-11	5 MO	630	1						
EY-J0016-AD	Programming in FORTRAN IV	5 MO	630	1						
EY-J2142-AD	RSX-11M Programmer	5 MO	630	1						
EY-J2144-AD	RSX-11M System Programmer	5 MO	680	1						
EY-J2278-AD	RSX-11M-PLUS Programmer	5 MO	630	1						
EY-J2234-AD	RSX-11M-PLUS System Programmer	5 MO	680	1						
EY-J2166-AD	RSX-11M System Manager	2	MO	350	1					
EY-J2068-AD	RSX-11M/IAS DECON User	5 MO	750	1						
EY-J2006-AD	RSX-11M Operator	3	MO	520	1					

TURN PAGE FOR BOSTON RSX-11M/M-PLUS SCHEDULES

The full tuition is payable in advance and includes the cost of course materials. Courses not listed may be offered on a demand basis. Schedule and prices are subject to change without notice. For a description of each course, please refer to Educational Services' 1981 Customer Training Catalog or a course-specific Product Brochure. All courses listed on these pages that are prefaced with a P are prerequisite courses to our RSX-11M/M-PLUS, RT-11, IAS, CTS-300 and DSM-11 training curricula. Please consult your local Training Center for more information on prerequisite training.

## RSX-11M/M-PLUS CUSTOMER TRAINING PROGRAMS (Continued) BOSTON (617) 275-5000 Ext. 2380

Canadian Customers: All registrations for U.S. courses should be made through the Ottawa (Kanata) Registrar, (613) 592-5111.

Course #	Title	Length	Start Day	Training Credit	October	November	December	January	February	March
EY-J2022-AD	Commercial Programming Concepts	5 MO	\$ 520	1						
EY-J2114-AD	Introduction to Minicomputers A/V	~5	400	1						
EY-J2016-AD	Introduction to Minicomputers	5 MO	520	1						
EY-J2116-AD	Introduction to the PDP-11 A/V	~5	450	1						
EY-J2024-AD	PDP-11 Assembly Language Programming	5 MO	580	1						
EY-J2050-AD	Programming in MACRO-11	5 MO	630	1						
EY-J2048-AD	PDP-11 Concepts	3	MO	460	1					
EY-J2140-AD	RSX-11M User	5 MO	630	1						
EY-J2232-AD	RSX-11M-PLUS User	5 MO	630	1						
EY-J0002-AD	Programming in PDP-11 COBOL	5 MO	630	1						
EY-J2230-AD	Programming in BASIC-PLUS-2	5 MO	630	1						
EY-J2134-AD	BASIC-PLUS-2 Prog. with RMS-11	5 MO	630	1						
EY-J0016-AD	Programming in FORTRAN IV	5 MO	630	1						
EY-J2142-AD	RSX-11M Programmer	5 MO	630	1						
EY-J2144-AD	RSX-11M System Programmer	5 MO	680	1						
EY-J2278-AD	RSX-11M-PLUS Programmer	5 MO	630	1						
EY-J2234-AD	RSX-11M-PLUS System Programmer	5 MO	680	1						
EY-J2042-AD	RSX-11M Systems Installation and Management	3	MO	480	1					
EY-J2068-AD	RSX-11M/IAS DECON User	5 MO	750	1						
EY-J2006-AD	RSX-11M Operator	3	MO	520	1					
EY-J2404-AD	RSX-11M Executive Internals	3	MO	525	1					

## IAS TRAINING PROGRAM

## WASHINGTON, D.C. (301) 459-7900 Ext. 2580, 2582

Course #	Title	Length	Start Day	Training Credit	October	November	December	January	February	March
EY-J2022-AD	Commercial Programming Concepts	5 MO	\$ 520	1						
EY-J2114-AD	Introduction to Minicomputers A/V	~5	400	1						
EY-J2016-AD	Introduction to Minicomputers	5 MO	520	1						
EY-J2116-AD	Introduction to the PDP-11 A/V	~5	450	1						
EY-J2024-AD	PDP-11 Assembly Language Programming	5 MO	580	1						
EY-J2050-AD	Programming in MACRO-11	5 MO	630	1						
EY-J2048-AD	PDP-11 Concepts	3	MO	460	1					
EY-J2202-AD	IAS User	5 MO	630	1						
EY-J0016-AD	Programming in FORTRAN IV	5 MO	630	1						
EY-J2230-AD	Programming in BASIC-PLUS-2	5 MO	630	1						
EY-J2208-AD	IAS Programmer	5 MO	630	1						
EY-J2206-AD	IAS System Management	5 MO	630	1						
EY-J2204-AD	IAS System Programmer	5 MO	680	1						
EY-J2068-AD	RSX-11M/IAS DECON User	5 MO	750	1						
EY-J2134-AD	BASIC-PLUS-2 Prog. with RMS-11	5 MO	630	1						

**digital**

Educational Services  
12 Crosby Drive  
Bedford, MA 01803

Attn: U.S. Marketing Group

Please add my name to your mailing list to receive DIGEST, Training Update & Schedules. I understand I am under no obligation whatsoever.

Name \_\_\_\_\_  
Title \_\_\_\_\_  
Company \_\_\_\_\_  
Address \_\_\_\_\_  
City \_\_\_\_\_ State \_\_\_\_\_ Zip \_\_\_\_\_