THE mini-tasker

DECUS
RT-11 SIG NEWSLETTER

MARCH 1981 VOL. 7 NO. 2

Contributions to the newsletter should be sent to:

Ken Demers
MS-44
United Technologies Research Center
East Hartford, Conn. 06108
203 727-7527

Other communications can be sent to:

John T. Rasted
JTR Associates
58 Rasted Lane
Meriden, Conn. 06450
203 634-1632

or

RT-11 SIG
C/O DECUS
One Iron Way
MR2-3/E55
Marlboro, Mass. 01752
617 481-9511 Ext. 4141

USER INPUT

The Fourty Seven Test

This program performs a (perhaps infinite) test of memory. Although written in MACRO assembler (or just straight binary) it also manages to satisfy most of the stringent and rigorous criteria developed by the proponents of strictured programming. This is because the program consists (initially and at any point during its (perhaps infinite) execution, of a single instruction.

Further, the program is fully portable between all PDP-11's (and perhaps the VAX in compatibility mode). Apart from testing memory it also tests the program counter to the limit (literally) by running it backwards (Thus, it is not only written Top Down but also executes Top Down). The program does not require an operating system of any kind (and will quickly do away with any such if properly run.) The program is completely position independent.

TERMINOLOGY - The algorithm describing the Fourty Seven Test will be described in a new conceptual programming language called Ida (named after Charles Babbages dog). Ida is very much like 'Programmers Vernacular' - that is - poorly spelled English with a lot of gestures, aah's and um's.

PHILOSOPHY - Before beginning with the introduction to the Forty Seven Test I would like to present a bit of background behind its philosophy - but space does not permit. Since The Fourty Seven Test does not involve any data the discussion of Data Types can be elided. In a like vein, the 47 test does not involve any arithmetic or (explicit) transfer instructions. Therefore, Ida not only forbids the use of the GOTO instruction, it also disallows the CALL. In fact Ida only permits a single command: the Fourty Seven command (See footnote 4.7 of section 4.1.7.3.3.5.)

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.

THE PROGRAM - A last minute hitch in ironing out some ambiguities in Ida forces us to revert to MACRO in presenting this program - but, as will be shown in the forthcoming seven volume Rationale, it is possible, using a context frozen grammar, to proove a unique one-to-one mapping between MACRO and Ida.

```
.title   the fourty seven test
.enabl   lc

;ASSEMBLY INSTRUCTIONS :
;
;EXECUTE/LIST/CROSSREFERENCE FOURTY
;
;DATA DEFINITION
;The Fourty Seven test requires only one piece of DATA, and that is
;the START_ADDRESS. This is defined in the following definition and
;is the default setting for a 28k (word) machine.
STARTADDRESS=160000-2 ;Change this to your liking

;The comment line below (;—) must begin with a semicolon. Otherwise
;MACRO will interpret it as a sequence of 66. unary minus signs.
;Since MACRO pushes a couple of words on the stack per unary operator,
;it very quickly runs out of stack space and crashs.

;——————————————————————————————————————————
;        MAIN PROGRAM
;——————————————————————————————————————————
;+
;PROGRAM ENTRY POINT
;
;The program is automatically started here by the concluding .END START
;sequence. RT-11 arrives here with the registers in a mess and no idea
;about what we're about to do.
;-

.=.-1000+STARTADDRESS            ;define the start address

;+
;CODE SECTION
;
;The Code for the Fourty Seven Test is impure and should not be run in
;read-only memory (ROM, PROM or EPROM). Further, its use on machines
;with separate Instruction and Data spaces is unpredictable. To ehance
;portability prospects with future PDP-11 Assembler's we have chosen not
;to use mnemonics (since the infc   tion in the MACRO manual is all
;subject to change without notice), but rather, to return to direct
;octal-coded binary. Here follows the Fourty Seven Instruction :

.word   014747           ;The Fourty Seven Instruction

;PROGRAM CONCLUSION
;The rest of program has been left to your imagination - firstly, since
;it would take up valuable space in the MINI-tasker, and secondly since
;this program hasn't got a chance of running under RT-11 anyway.

.end    startaddress
```

As the concluding comment in the program points out, this program wont run. RT-11 produces a 'Not enough memory' message after an attempt to run this 112 block program. Therefore, we will have to dump Ida and do the job by hand. Thus:

(1) Stop the computer
(2) Load the number 014747 in the highest memory address
(3) Start the computer at the same address as above

This can be accomplished on an 28k LSI-11 with :

```
<break>
@157776/ xxxxxx 014747 [return]
@157776G
```

THE OUTCOME - The question is what will happen? Make a guess now. If you're a novice - then just try to guess what memory will look like after the test. If you know that, try to work out the terminating conditions of the program. Then come back to the next paragraph.

THE FOURTY SEVEN TEST consists of the instruction 'MOV -(PC),-(PC)'. This instruction effectively replicates itself in memory backwards. Thus it fills memory with the pattern 014747. It terminates by trapping when it goes below location zero. What happens then depends on the PDP-11 model involved. (Most LSI processors end up with alternating halts at 000000 and 177777 (which doesn't really exist). This instruction drives a PDP-11 backwards! But it also accomplishs a basic memory test.

THE FUTURE - I will consider submitting the 47 test (and Ida) to the DECUS library after it has been fully field tested (this usually takes around five years). If enough user support emerges it may be worthwhile starting a 47-SIG. If you want a copy of the Fourty Seven Test then send a stamped addressed RL02 Disk to the following address:

Ian Hammond - HAMMOND-software - Am Felborn 22 - D-34 Goettingen - West Germany.

Rozenberg Samuel

Societe INFI
3, Rue des pres-Aubry
92370 Chaville
FRANCE

19-Feb-81

Subject: SUBSTI.TEC , an extension of SEARCH.TEC

The RT11 V4 distributes a set of TECO programs, among which the useful SEARCH.TEC .

SEARCH.TEC allows for one or more character strings to be located in one or more files, and have the result output to a peripheral device or to a file. The use of SEARCH.TEC is very simple since it is interactive. It is called by executing the following command (supposing SET EDIT TECO) :

```
        EDIT/EXEC SEARCH
```

which sets up the dialog with the consol terminal.

SUBSTI.TEC is an extension of SEARCH.TEC.

It allows the substitution of character strings by other character strings in one or more files. You specify to SUBSTI the same arguments as for SEARCH, but each search string is followed by a substitution string. You can specify any number of input strings and any number of input files (see example). Each input string is searched in the input files. If found it is output to the output file, along with its environment. The environment of the string is defined by the line to which the string belongs and n lines around this line (the value of n is supplied by the user). Then those files for which there is a match are Edit Backup-ed and the substitution takes place.

The SUBSTI.TEC program is listed below , followed by an example of use.

The example shows two files upon which SUBSTI.TEC will act. Then the SUBSTI dialog is shown. The output of the searched strings is done to TT: (default). The number of lines to view (n) is set to 3 (the default). Each string found is followed by a ^ . Then the files are typed to show the result of the substitution.

```
^D 4#2ED -1^X 128#4#ETET @EI%%
@^U1/.U0 J Q3-^^-"N ^^*U3 72< Q3@I%% > @I%
% G* @I%
% ' Q3@I%% Q3@I%% @I%Page % Q6\ 65+^S< Q3@I%% > ^^-U3 @I%
% 0,.X7 0,.K Q0J -Q1L .,Q0:X7 J @I%^% 0,.:X7 0,.K Q0J Q1+1L Q0,.:X7
G7 ^YPW ^YK Q0J/
@^U2/HX0 HK !MAIN! EUU0 -1EU :G0 HT Q0EU !CHAR! ^TU0
Q0-27"E 13^T 10^T @O%DONE% '
Q0-13"E ^T^[ @O%DONE% '
Q0-21"E HK
 !LINE! 13^T ET&2"E 10^T | 155^T 0:W-4"E ^^[^T ' ^^K^T 13^T ' @O%MAIN% '
Q0-127"E Z"N -1AU0 -D Q0-32"L @O!LINE! ' ET&2"E @O!LINE! '
 8^T 32^T 8^T ' @O%CHAR% '
ZJ Q0@I%% @O%CHAR%
!DONE!/
HK @I%Output <TT:.SRH>: % M2 Z"E @I%TT:% ' J :@S%.%"U ZJ @I%.SRH% ' HX9
!INP! HK @I%    Input <.MAC>: % M2 Z"N J :@S%.%"U ZJ @I%.MAC% '
^^*U5 J :@S%/ALL%"S @FR%% -1U5 ' HX5 [5 @O%INP% '
HK < :J5; J Q5\ 10@I%% G5 10@I%% > < -Z; ZJ -D 0L ZX5 -L \U5 0L ZK [5 >
HK @I%        Lines <3>: % M2 3U1 Z"N J \U1 '
@^U3/0U4 0,0X4 < :J5; @EN%^EQ5% < :@EN%%; HK Q5U3 G* HX0 @ER%^EQ0% 1U6
< Y/
@^US/0,0X8 <:JB; @EB%^EQB% Y</
!SRH! HK @I%            Search: % M2 Z"N J HXF G3
@I/ J < S/ ZJ 27@I// @I/; M1 >/
HX3 HK @I%      Substitute: % M2 J GS @I/ J < FS/ GF 27@I// ZJ 27@I//
 @I/; >/ HXS @O%SRH% '
HK G3 ZJ @I/ -^E%6^[ ^N^_; > HK G* 13@I%% 10@I%% J Q3-^^*"E G4 HX4 %4^[
| ZJ .U8 G8 HX8  Q8,ZK'
Q3+1"E 72< @I%*% > 13@I%% 10@I%% HPW ' HK > >
HK 72< @I%*% > 13@I%% 10@I%% Q4"N @I%
File% Q4-1"N @I%s% ' @I% with no matches:
% G4 ' HPW EF HK/ HX3
HK GS ZJ @I/ ^N^_; P> EC > / HXS HK
 @I%     Detach <No>: % M2 @EW%TT:% EF @EW%^EQ9%
Z"N J 0A&(32^_)-89"E 64#ETET ' '
HK HX4 M3 EC
HK G8 <J Z-1:; .U8 L Q8,.-2X8 [8 Q8,.K> MS
```

^C ^C $$

```
.TYPE EXAMPL.COM
SET TT NOQUIET
!
! Type the two sample original files.
TYPE (DECLAR,EXAMPL).MAC
!
! Perform substitutions
!
EDIT/EXEC SUBSTI
!
! Type files after substitution.
TYPE (DECLAR,EXAMPL).MAC
!


.@EXAMPL

.SET TT NOQUIET

.TYPE (DECLAR,EXAMPL).MAC
        .TITLE  DECLAR

LINEFEED=12
CR      =15
ESC     =33

        .END


        .TITLE  EXAMPL

        .GLOBL  LINEFEED,CR

MESSAGE:
        .BYTE   CR,LINEFEED
        .ASCIZ  /ABCDEF/
        .EVEN

        .END


.EDIT/EXEC SUBSTI
Output <TT:.SRH>:
    Input <.MAC>: DECLAR.MAC
    Input <.MAC>: EXAMPL.MAC
    Input <.MAC>:
       Lines <3>:
          Search: =
      Substitute: ==
          Search: LINEFEED
      Substitute: LF
          Search: CR
      Substitute: CRRTN
          Search:
     Detach <No>:
*************************************
DK:DECLAR.MAC
**Page 1*****************************
```

```
        .TITLE  DECLAR

LINEFEED=^12
CR      =15
ESC     =33




--Page 1-----------------------
        .TITLE  DECLAR

LINEFEED=12
CR      =^15
ESC     =33

        .END
--Page 1-----------------------

LINEFEED=12
CR      =15
ESC     =^33

        .END

--Page 1-----------------------
        .TITLE  DECLAR

LINEFEED^=12
CR      =.T
~::     =33

--Page 1-------------------------
        .TITLE  DECLAR

LINEFEED=12
CR^     =15
ESC     =33

        .END
*********************************
DK:EXAMPL.MAC
**Page 1*************************
        .TITLE  EXAMPL

        .GLOBL  LINEFEED^,CR

MESSAGE:
        .BYTE   CR,LINEFEED
--Page 1------------------------
        .GLOBL  LINEFEED,CR

MESSAGE:
        .BYTE   CR,LINEFEED^
        .ASCIZ  /ABCDEF/
        .EVEN

--Page 1-----------------------
```

```
        .TITLE  EXAMPL

        .GLOBL  LINEFEED,CR^

MESSAGE:
        .BYTE   CR,LINEFEED
--Page 1-----------------------
        .GLOBL  LINEFEED,CR

MESSAGE:
        .BYTE   CR^,LINEFEED
        .ASCIZ  /ABCDEF/
        .EVEN

********************************
```

```
.TYPE (DECLAR,EXAMPL).MAC
        .TITLE  DECLAR

LF      ==12
CRRTN   ==15
ESC     ==33

        .END


        .TITLE  EXAMPL

        .GLOBL  LF      ,CRRTN

MESSAGE:
        .BYTE   CRRTN,LF
        .ASCIZ  /ABCDEF/
        .EVEN

        .END
```

RECLAIMING PDP-11 I/O PAGE ADDRESS SPACE UNDER RT-11

Richard Krasnow
Harvard University Biolabs
16 Divinity Ave., Cambridge, MA 02138
(617)-495-3716

February 3, 1981.

I.Introduction

On a 16-bit machine, like DEC's PDP-11's, the largest possible integer is 2^{16} = 64K, unsigned, or 32K signed. Since addresses are themselves integers, and bytes are also addressable, the largest even byte address (the last of the 32K words) on a PDP-11 is 64K-2 = "177776, unless memory management is used, in which case address space is divided into 32K word regions. As with many other computers, PDP-11's make use of the top 4K of address space for device I/O. On a PDP-11/10, which cannot have memory management added, the highest reachable word (the 32Kth) is thus at byte address "177776. Hence, even if the machine has physical memory capable of responding to the address range 28K-32K (e.g., two 16K banks is a common configuration), this bank of memory will remain unused, because the processor itself converts a program request for an address in the 28K-32K range, into a bus request for the corresponding address in the 124K-128K range. This paper discusses two means of reclaiming some of that address space for the user, making use of that already present "quiet" memory, and thus incrementing user memory by 2-3K words.

The processor does the low-range to high-range conversion by asserting two internal bits, 16 and 17. There is a circuit on the CPU (on the M7621 board, for the PDP11/10) that asks whether the requested address was greater than or equal to 28K = "160000, by anding bits 13-15. If yes, it asserts internal bits 16,17 thus converting "16xxxx into "76xxxx. This is done with a 7410 3-input nand gate.

The device interfaces themselves respond to this "high" range, not to the "low" range. Thus, although we commonly think of a DL11 being at "176500 (in the 28K-32K) range, it really responds to "776500 (in the 124K-128K range). Now, although DEC has reserved virtually the entire 4K for a variety of devices, and assigned standard addresses for each device, any one machine uses no more than a few addresses, for control/status, i/o buffer and other registers. Thus the 4K are mostly wasted.

What I propose here, and have implemented on a couple of PDP11/10's, is to move all the device addresses up to the top 1K, and then make one of two modifications to access the remaining 3K as user memory. These procedures are predicated on there being unused memory on the machine. The total cost is zero, and the effort is probably one day of down time. The procedure is far simpler than the explanation behind it, so have no fear! Two drawbacks are (i) some device addresses will differ from standard ones, so programs like diagnostics will not run without patching, or resetting the address of the device and disabling the "extra" memory (see below) and (ii) the SHOW/CONFIGURATION command will tell you certain devices exist when in fact they do not. As far as I can determine, the SHOW/CON actually addresses memory in addition to checking the configuration bits, so that it is more work to patch out.

II. Moving the devices to the top 1K

First make a list of the devices on your machine, starting at the top. The PDP-11 processor handbook will show which addresses are part of the processor itself (e.g., the PSW = 777776, R0-R6, the switch console, line clock, etc.). These are the highest addresses, so do not concern us here. Keep checking until you find devices with addresses below 31K = "174000. Noteworthy individuals are bootstrap loader proms (the DSD-440 floppy drive defaults at "171000), the VT11-VS60 graphics display processor (defaulting at "172000, the AR11 ("170400), perhaps a DL11, and the DR11C ("167770). Most devices have addresses jumperable or switchable within a range. Consult the individual device installation manual for the specific jumpers or dip-switch settings. Next, draw up the final address assignment, to make sure no address conflicts will occur. If you have determined that none of your critical devices needed for booting and running are below "174000 (most likely), then you can start moving the devices to the top 1K. Do it one by one, inserting them on the unibus and ascertaining that the devices respond to the new addresses.

2.1 The VT-11 Display Processor

This example shows the kind of bind one can encounter in this process. I will describe it as a general illustration, and for the information of users who have this device.

The VT11 is intended to default at "172000-"172006. Its address is in fact jumperable within "172xxx, although the manual does not appear to encourage it. How to get it to higher than "174000? The address decoder does not compare address bit 11 to a jumpered value; it uses it as is, after passing the decoded and inverted bit 11 through another 7404 inverter gate. The output is anded with the common output of the jumpered address bits, to yield the address enable signal. The solution to the problem turned out to be simple: bypassing one inverter made the 0 into a 1, i.e., "172000 into "176000. Specifically, on the M7014-YA, cut the trace from pin 2 of E18 that leads to pin 1 of E13, and solder the latter to the trace leading the pin 1 of E18, which conveniently emerges on the board at a hole 3 mm next to pin 1 of

E13, requiring only a solder bead. (See drawing DCS-M7014-YA-1, Bus Control and Logic: Address Selection Logic (ASL)).

III. Patching the software

If you have sources, it is elementary to edit in a new address and regenerate the device driver, or user program. Object files, if part of a distribution kit, usually have an installation procedure that asks you the configuration of the system. (E.g, the Fortran extensions for the AR-11, DR11). The difficulty comes with .SAV files. However, with judicious guessing, it is possible to use PATCH, preferably SIPP, to find locations that contain the addresses, and patch them accordingly. Thus, the monitor itself, TECO, and RESORC were quickly patched in a half dozen places, to account for the new position of the VT11. Don't forget that if the devise csr was originally at "172000 (to pick the VT11 example), you must search in SIPP for "172000 and "172002, possibly "172004 and "172006 as well. Not all references to "172000 were related to the VT11. I altered all, then had to undo a couple until everything worked.

3.1 Monitor VT11 Support

When RT11 boots up, it looks at "172000 to see if there is a VT11 on the system. If it gets a response, it sets bit 2 of the configuration word at offset "300 into the beginning of RMON (@"54), and enables the GT ON/OFF commands. If you have a VT11, you must patch the system so that it will look at "176000, instead of "172000. If you do not have a VT11, you still should patch the monitor, because when the memory is enabled above 28K, it will respond to the monitor's "172000 inquiry and think it has a VT11, possibly getting very confused down the line.

Also at boot time, location "172540 is checked for a KW11P programmable clock, with bit 14 of 300(@54) configuration word set if positive. RESORC.SAV checks this bit. If you have a KW11P, you will have moved it elsewhere; if not, this patch is also in order to avoid a SHOW/CON reporting a non-existing KW11P. The patch simply moves the KW11P address test to a higher location.

```
Patches to RT11 V3B-00C DYMNSJ.SYS were:
    "001102  sets configuration bit 2 for VT11
    "035650
    "074020
    "074444
    "075414
    "076110
    "076132 from "172000 to "176000;
    "074024
    "074450
    "076116 from "172002 to "176002
    "1124 from "172032 to "176032

      "1110 from "172540 to "176540 the KW11P patch



  Patches to RT11 V3B-00F DYMNFB.SYS were:
      "1114
      "107034
      "107472
      "110340
      "111034 from "172000 to "176000;
```

"107040
"107476
"111042 from "172002 to "176002
"1136 from "172032 to "176032
"1122 from "172540 to "176540

Patches to RT11 V3B-00C DXMNSJ.SYS were:
"1110 from "172000 to "176000
"1132 from "172032 to "176032
"1116 from "172540 to "176540

Patches to RT11 V3B-00F DXMNFB.SYS were:
"1122 from "172000 to "176000
"1144 from "172032 to "176032
"1130 from "172540 to "176540

3.2 TECO VT11 Support

When TECO starts up, it decides whether there is a VT11 with support on the system. It does so by looking at offset "300 from the beginning of RMON, not by addressing "172000. If it finds support, it activates code that will address "172000.

Locations patched in TECO V28 were
"5376
"5550
"5606
"6254
"15532 from "172000 to "176000;
"15536
"15546 from "172002 to "176002;
("11750 contains "172004; leave as is).

3.3 RESORC.SAV

The following patch to RESORC.SAV prevents the SHOW/CON from telling you a VS48 exists, when actually the VT11 is on the system:
"1000 from "172000 to "176000

"1002 from "172032 to "176032

IV. Enabling the Extra Memory

There are two approaches to this problem. In one, we make use of the fact that some memories can be jumpered to respond to the 28K-31K range in the high range. That is, the top 16K of memory is normally strapped to respond to 16K-32K, but never gets addressed in the 28K-32K range because of the processor mapping. Memories like the Plessey PM1116, PM1116B, PM1105, PM1105B, can be jumpered to respond to 16K-28K, and also from 124K-127K. Thus, although the processor is still mapping a request in the 28K-31K range to the high 124K-127K range, to the user it appears as if his memory goes up to 31K. This is the method I have used, as it takes only inserting one jumper (W) However, for the sake of flexibility, I have pulled the jumper terminals to a toggle switch on the front panel. This is good practice: one ought to be able as much as possible to quickly revert to standard conditions, so as to run unpatched programs like diagnostics. Consult your memory installation instructions to see if such a feature exists. (Incidentally, it is worth investigating at this point whether your memory can be interleaved, to increase speed).

V. Modifying CPU Memory Mapping

The other approach modifies the CPU itself so that it does not map requests in the 28K-30K range to the 124K-126K range. As discussed in the Introduction, the memory mapping circuit tests bits 13-15 in a nand gate, mapping anything above "160000. Forcing it to test bit 12 as well means that it will only map addresses above "170000, or 30K. One can extend this method to nand some specific configuration of bits for any desired cutoff address. The 30K boundary seems a good gain for minimal work. The existing nand gate on a 7410 chip (E61 on the M7261) is a three input (pins 9, 10, 11) gate, outputing on pin 8. There is no way it can be modified to accept another line, and I have not found any exisiting gates not in use on the board. One has to place another chip on the board (glue it on its back on the side) that has a 4-input nand gate, namely a 7420, and pull +5V and ground to it. Looking at the chip pinouts, it becomes obvious how to pull leads from the three inputs to the existing gate, to the comparable inputs to the new four-input gate. The additional input line is the CONA BA12 (1) H (pin 7 of E 51). Make sure the trace from pin 8 of E61 is cut, and the new gate's output fed to pin 9 of E63. (See drawing DCS-M7261-0-1-S, Control Logic and Microprogram, Drivers and Receivers).

V. Patching the Monitor to Boot Higher than 28K.

The RT11 V3B monitor automatically boots to 30K if available (why??). Thus the base address of DYMNSJ is 147566 at 28K, 157566 (up to 30K) if 31K enabled. The following patch is only necessary for those who have enabled 31K and want that extra 1K. The DYMNSJ base then becomes 163566, and the DYMNFB base goes from 136760 to 152760. DXMNSJ goes from 150154 to 164154; DXMNFB from 137346 to 153346.

Using SIPP or PATCH, modify

BHALT	from	407	to	240	BHALT=774 for DYMNxx
BHALT+2		13702		12702	=1002 for DXMNxx
BHALT+4		177570		174000	see SysRel Notes for
BHALT+6		42702		42702	your monitor's BHALT,
BHALT+10		3777		0	table 4-2.
BHALT+30		170000		174000	

See RT11 V2C Software Support Manual, p. 4-5, RT11 V3B SysGen Manual p. 2-36, and my article "Software Hardboot Emulator for RT-11", Minitasker 5(2), pp. 8-9, March 1979, for explications. Do not alter BHALT+34, +36 as indicated in the manual; that would force the boot to 31K, and would not boot when the memory is switched back to 28K.

VI. Conclusion

In summary, for the effort of changing very few jumpers, and entering a few patches, your system may be able to have an extra 3K added for free. Of course, those who have sources for V3B (and understand them...) can do the editing at that level. I may have missed some appropriate patches by going the binary route, but everything seems to work anyway. I would like to hear from individuals who can point to the V3B source level, in case a SYSGEN is warranted in the future, and from others who have found different methods of arriving at the same goal. I would also like to suggest to DEC and other firms that they design systems with less waste. It is entirely possible to design a software/hardware scheme that will use a machine whose addresses are not standard (and thus wasteful of address space), but contiguous from the top down, and yet known to the system at boot time or at configuration time.

I would like to acknowledge the help of John Shriver and Mike Patton, of the M.I.T. Electronics Research Society, for discussions leading up to the implementation of these procedures.

CENTRE NATIONAL
DE LA RECHERCHE SCIENTIFIQUE

LABORATOIRE DE PHYSIOLOGIE
COMPAREE DES REGULATIONS
23, RUE DU LOESS
STRASBOURG-CRONENBOURG
TEL. (88) 3XXXXXX 29.90.33

B. P. 20 CR
67037 STRASBOURG - CEDEX
FRANCE

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$ DECUSCOPE : DIGITAL EQUIPMENT COMPUTER USERS SOCIETY$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

APPLICATION NOTE : RT 11 (FORTRAN IV SOURCES.)
****

CLEDYN : A FORTRAN IV SUBROUTINE TO PROTECT THE ACCESS TO PROGRAMS OR CONFIDENTIAL FILES BY GENERATION OF DYNAMIC PASSWORDS.

****

BY DANIEL GUINIER

LABORATOIRE DE PHYSIOLOGIE COMPAREE DES REGULATIONS
GROUPE DE LABORATOIRES DU CNRS DE STASBOURG-CRONENBOURG
23 RUE DU LOESS
B.P.20 CR
67037 STRASBOURG CEDEX FRANCE

INTRODUCTION :
****

IT IS SOMETIMES DESIRABLE TO PROTECT THE ACCESS TO PROGRAMS OR CONFIDENTIAL FILES. IN SOME CASES, A STATIC PASSWORD IS SUFFICIENT, BUT IN OTHER CASES, A DYNAMIC PASSWORD GIVES MORE PROTECTION (FOR MEDICAL OR PERSONNAL BANKS OF DATA FOR EXAMPLE)

THE SUBROUTINE CLEDYN GENERATES A DYNAMIC PASSWORD WHICH IS THE PRODUCT OF TWO PRIME NUMBERS, THE FIRST IS STATIC AND CAN BE CHANGED BY THE SYSTEM USER IN THE CALLING PROGRAM, TROUGH THE INDEX NOCLE AND THE SECOND IS DYNAMIC, IT IS GENERATE BY THE MEAN OF THE INTERNAL CLOCK OF THE COMPUTER

DESCRIPTION OF THE SYSTEM :
****

CLEDYN : SUBROUTINE TO GENERATE THE PASSWORD AND PERFORM
         A CORRECT OR INCORRECT RETURN

PREM . : SUBROUTINE TO GENERATE THE I TH. PRIME NUMBER P

A PROGRAM IS ASSOCIATE TO GIVE THE CODING TABLE OF THE 3000H TH FIRST PRIME NUMBERS.

LISTING OF THE SOURCE PROGRAMS :
****

```
   IV       V01C-03A

         SUBROUTINE CLEDYN(LEC,IMP,NOCLE,I)
      C
      C  DYNAMIC PASSWORD GENERATOR.
      C
      C  THE PASSWORD IS THE PRODUCT OF TWO PRIME NUMBERS
      C  NOCLE CAN BE CHANGED BY SYSTEM USER.
      C  IF I IS NOT 0 :   CORRECT RETURN.
      C IF I IS     0 : INCORRECT RETURN.
      C
0002        INTEGER*4 N
      C THE INTERNAL CLOCK GIVES ITS CONTENT AS A TWO WORDS INTEGER N
0003        CALL GTIM(N)
      C THIS CONTENT IS TRANSFORMED IN NS SECONDS AND NT TICKS.
0004        CALL CVTTIM(N,NH,NM,NS,NT)
      C NS AND NT INITIALIZE THE PSEUDO-RANDOM GENERATOR TO GIVE I
      C I IS THE INDEX OF THE I TH. PRIME NUMBER CLE
      C WHICH IS THE DYNAMIC KEYWORD.
0005        I=1.0E+06*RAN(NS,NT)
      C NOCLE IS THE INDEX OF THE NOCLE TH. PRIME NUMBER
      C WHICH IS THE STATIC KEYWORD.
0006        CALL PREM(P,I)
0007        CALL PREM(CLE,NOCLE)
      C THE DYNAMIC PASSWORD IS THE PRODUCT OF THE TWO NUMBERS P AND CLE
0008        WRITE(IMP,100)I
0009  100   FORMAT('$DYNAMIC PASSWORD IN ANSWER TO ',I5,' = ')
0010        READ(LEC,200)N
0011  200   FORMAT(I10)
      C THE CORRECT RETURN IS WHEN I IS NOT EQUAL TO 0.
0012        IF(N.NE.P*CLE)I=0
      C ACTIVATION OF THE "BUZZER" ON THE COMMAND TERMINAL TT:
      C IF I=0.
0014        IF(I.EQ.0)N=ITTOUR("007)
0016        RETURN
0017        END

***** COMPILATION STATISTICS *****
*                                *
*                                *
*-------- COMPILER  TABLES -------*
* SYMBOLS:          00139 WORDS *
* PROGRAM:          00103 WORDS *
**********************************
```

```
      SUBROUTINE PREM(P,I)
C
C  GENERATION OF THE I TH. PRIME NUMBER P  (EXCLUDING P=1.)
C
      INTEGER RESTE(8),NP(10)
      DATA RESTE/1,7,11,13,17,19,23,29/
      DATA NP/2,3,5,7,11,13,17,19,23,29/
C
      IF(I.GT.10)GO TO 1
      P=NP(I)
      RETURN
1     J=(I-3)/8
      K=1+AMOD(I-3.,8.*J)
      P=J*30.+RESTE(K)
      RETURN
      END


***** COMPILATION STATISTICS *****
*                                *
*                                *
*-------- COMPILER  TABLES -------*
* SYMBOLS:             00111 WORDS *
* PROGRAM:             00064 WORDS *
**********************************
```

```
     PROGRAM TO GENERATE THE CODING TABLE OF THE FIRST 30000
     PRIME NUMBERS.
     ********************************************************

FORTRAN IV      V01C-03A

      REAL*4 P(5)
      INTEGER*2 L(5)
      K=0
      DO 2 I=1,6000
      DO 1 J=1,5
      K=K+1
      L(J)=K
1     CALL PREM(P(J),K)
2     WRITE(7,100)(L(J),P(J),J=1,5)
100   FORMAT(5(I7,F9.0))
      STOP
      END


***** COMPILATION STATISTICS *****
*                                *
*                                *
*-------- COMPILER  TABLES -------*
* SYMBOLS:             00097 WORDS *
* PROGRAM:             00083 WORDS *
**********************************

     EXAMPLE OF CALL AND USE :
     *************************

     CALLING SEQUENCE
     ------------------
```

```
C  INDEX OF THE STATIC KEYWORD.
      NOCLE=5
C  THE 5 TH. PRIME NUMBER IS P=11.
      CALL CLEDYN(5,7,NOCLE,I)
C  IF I IS EQUAL TO 0 : INCORRECT RETURN
      IF(I.EQ.0)STOP
                .
                .
                .
USING SEQUENCE EXAMPLE :
------------------------

                .
                .
                .

DYNAMIC PASSWORD IN ANSWER TO 1206 = (PASSWORD)

THE CORRECT PASSWORD MUST BE 49643 (INTEGER) WHICH IS THE PRODUCT
OF THE 5 TH. PRIME NUMBER (11) AND THE 1206 TH. PRIME NUMBER
(4513). THE 1206 IS GENERATED BY THE COMPUTER AND THE
CORRESPONDANCE TO 11 AND 1206 IS GIVEN BY THE CODING TABLE.
(EXCLUDING 1).
```

THE UNIVERSITY
OF ASTON
IN BIRMINGHAM

The Sumpner Building, 19 Coleshill Street, Birmingham B4 7PB
Tel: 021.359 3611 Ex 559/284

The Department of Electrical and Electronic Engineering

Head of Department: Professor J E Flood DSc, CEng, FIEE
Professors of Electrical Engineering
E J Davies PhD, CEng, FIEE XXXXXXXXXXXXXXXXXX

Last year you kindly published my note on the PDP-11 Fortran-callable pseudo-random number generator routines RAN and RANDU provided by DEC with the RT-11 Fortran system.

Since I sent the note, several references (1-3) have come to light pointing out that the algorithm used by RAN and RANDU gives pseudo random numbers with rather poor statistical properties. (The algorithm used is $Y_{n+1} = (1+2+2^{16})\ y_n (\text{mod } 2^{31})$ where y_n is a 31-bit integer held in I1, I2, the integer arguments of the routines.) In addition to its poor statistical properties, there is a difficulty in generating independent runs of numbers, since the only acceptable starting pairs of values for I1 and I2 are zeros or pairs generated by previous calls to the routines.

An algorithm which has passed a large number of statistical tests (4), and seems to be the best known at present, is

$$Y_{n+1} = 7^5 \; y_n \pmod{2^{31} - 1}$$

where again, y_n is a 31-bit integer. It may be of interest to readers to know that it can be programmed very simply in PDP-11 Fortran by making use of the fact that large integers (of less than 56 bits) are processed exactly when held as DOUBLE PRECISION variables.

```
      DOUBLE PRECISION SEED
            :
      SEED = 1.0D0 ! Set SEED to an initial value 0 < SEED < 2147483647
            :
    1 SEED = DMOD(16807.0D0 * SEED, 2147483647.0D0) ! Generate new number
```

Successive executions of line 1 generate new pseudo-random values for SEED. These are integers, uniformly distributed in the range 0 < SEED < 2147483647. *Any* initial value in this range may be used to start the sequence. G.S. Fishman (3) (Table A.2) gives 400 numbers, spaced 100000 apart, generated by the algorithm. This is useful as a source of starting numbers to give non-overlapping sequences and also as a check on program execution.

The Fortran code alone above has been tested on a PDP-11/03 with FIS under RT-11 Fortran VOIC. Its only disadvantage, relative to a version of the algorithm coded in Macro, is that it was about 20 times as slow.

Acknowledgements

I am indebted to Mike Tydenham, NBSB, London for correcting and testing the Macro version of the algorithm. I am grateful to him and to Chuck Watson, Pacific Northwest Laboratory, Richland, for bringing references to my attention.

References

1. J.C. Simpson, R.S. Harkins, C.R. Watson, "Evaluation of the multiplier in the multiplicative congruential pseudo-random number generator", Proceedings of the Digital Equipment Users Society, San Diego, pp. 705-710, Dec. 1979.

2. G.W. Hill, "Cyclic properties of pseudo-random sequences of Mersenne prime residues", the Computer J., vol. 22, pp. 80-85.

3. G.S. Fishman, *Principles of Discrete Event Simulation*, Wiley, New York, 1978 (Gives many other references).

4. P.A.W. Lewis, A.S. Goodman, J.M. Miller, "A pseudo-random number generator for the system/360", IBM Syst. vol. 8, pp. 136-146, 1969.

Yours sincerely,

Martin Ackroyd

Martin H Ackroyd

Ray Kaplan
University of Arizona
112 n. 3rd. Ave.
Tucson, Arizona 85705

December 6, 1980

I would like to call attention to the fact that there is a small error in the automated patch kit included on the San Diego 1980 DECUS FALL tape which makes it impossible to install subsequent patches in the MTTINT source file (multi terminal support). The error will be found in the tape file MTTINT.001, which is the MTTINT portion of patch sequence 1.1.3 . Documentation for this can be found on page 11 of the July 1980 Software Dispatch under number 5. The second line of SLP patch file on the tape reads "ELMTIM<tab>== 3". It should read "ELMTIN<tab>== 3", as the Software Dispatch indicates.

This can be corrected either by correcting the line in the patch file on the tape or by changing your patching procedure for the November patch for the MTTIN.MAC file (Software Dispatch page 17, seq 1.1.11 M.). The change to the November patching procedure would be to change the first line from "-/ELMTIN<TAB>== 3/,.,/;002/" to "-/ELMTIM<TAB>== 3/,.,/;002/". This instructs the SLP program to look for the "M" which was put into the source file due to the typo in the patch kit on the San Diego tape.

In the two years that I have had dealings with Nick, I think this is the first error that I have found!

By way of introduction, I am a young fellow who has been around DECUS, the RT-11 SIG, and PDP-11's for only a short time. When I think back over my recent past, I can clearly see that alot of my growth has occured as a result of my contact with the gurus of the RT world. I am now a store house of hard fought RT-11 experience which continues to benefit those who I can manage to help.

The problem is that alot of the little "tricks of the trade" are stored in our collective heads, and are not accessible to people except through alot of reading and/or alot of personal contact and/or the good 'ol school of hard knocks. I don't object to paying my dues, but it seems that there might be a way to better allow those that come after us to stand on our shoulders.

I propose that we start a collection of information which we might call "The Collected Secrets of the RT world"! At first it could be nothing but a loose collection of things from symposia and MINI-TASKERs, but could easily grow into a small handbook which might be indexed in a way to permit quick access to the information in a "crisis".

I would volunteer to be whatever it takes to get this sort of thing going, and stand ready to get it going! What do you think?

Enjoy your computing!

Ray Kaplan

UNIVERSITY OF CALIFORNIA, DAVIS

BERKELEY • DAVIS • IRVINE • LOS ANGELES • RIVERSIDE • SAN DIEGO • SAN FRANCISCO SANTA BARBARA • SANTA CRUZ

COLLEGE OF ENGINEERING
DEPARTMENT OF APPLIED SCIENCE DAVIS-LIVERMORE

DAVIS, CALIFORNIA 95616

I have enclosed listings for two simple RT-11 FORTRAN routines that we find useful. The function IOFILE allows the user to open a file for input or output in a relatively crashproof fashion. Perhaps it would be better to use the FORTRAN OPEN command but we have found that the OPEN command has to many bugs in it for easy use and it has a tendency to crash your program if anything goes wrong (VERY undesirable if you happen to be doing live-time control at the time). The subroutine DIREC allows you to get a directory listing from a user (equivalent to DIR/FU).

Yours Sincerely,

Bob Walraven

Bob Walraven
Sr. Development Engineer
Department of Applied Science

```
*FORTRAN IV      V02.04     Tue 30-Sep-80 16:20:35              PAGE 001

       C     IOFILE.FOR
       C
0001         FUNCTION IOFILE (LUN, TYPE, NAME, ISIZE)
       C
       C-------------------------------------------------------------------
       C
       C     'IOFILE' IS AN RT-11 FORTRAN FUNCTION THAT OPENS A FILE ON THE
       C     SPECIFIED LOGICAL UNIT.  THE USER MAY BE OPTIONALLY QUERIED ON
       C     THE CONSOLE FOR THE FILE NAME.
       C
       C     LUN    IS THE DESIRED LOGICAL UNIT
       C
       C     TYPE   IS AN ASCII STRING SPECIFYING THE TYPE OF FILE TO
       C            BE OPENED:
```

```
C             'OLD' MEANS AN OLD FILE.  THE FILE IS OPENED FOR READING
C                   ONLY.  AN ERROR WILL OCCUR AND THE FILE WILL NOT BE
C                   OPENED IF THE FILE DOES NOT EXIST.
C             'NEW' MEANS A NEW FILE.  THE FILE IS OPENED FOR WRITING
C                   AND READING.  AN ERROR WILL OCCUR AND THE FILE WILL
C                   NOT BE OPENED IF THE FILE ALREADY EXISTS.
C             'UNK' MEANS AN UNKNOWN FILE.  IF THE FILE DOES NOT EXIST
C                   A NEW FILE IS OPENED.  IF THE FILE EXISTS, IT IS
C                   REOPENED (CAUTION:  YOU CAN LOOSE AN OLD FILE
C                   THIS WAY).
C
C     NAME   IS A STRING ARRAY CONTAINING THE FILE NAME OF THE
C            FILE TO BE OPENED.  AN EXAMPLE OF THE CORRECT FORM OF
C            THE STRING IS 'DX2TEST  DAT'.  IF YOU WISH TO BE QUERIED
C            ON THE CONSOLE FOR A FILE NAME, SET NAME EQUAL TO 0.
C
C     ISIZE  IS THE NUMBER OF BLOCKS DESIRED FOR A NEW FILE, OR
C            =  0 GETS HALF OF BIGGEST CONTIGUOUS SPACE ON DISK
C            = -1 GETS BIGGEST CONTIGUOUS SPACE ON DISK
C
C     IOFILE WILL BE RETURNED WITH A VALUE AS FOLLOWS:
C             >0    NUMBER OF BLOCKS IN THE FILE (NORMAL RETURN FOR 'OL')
C             =0    NORMAL RETURN FOR 'NE'
C             -1    NO RT-11 CHANNELS ARE AVAILABLE
C             -2    ILLEGAL DEVICE SPECIFICATION IN FILENAME STRING
C             -3    ILLEGAL FILENAME
C             -4    DEVICE IN USE
C             -5    LOGICAL UNIT IN USE OR NO LU SPACE AVAILABLE
C             -6    LOOKUP ERROR
C             -7    ILLEGAL TYPE
C             -8    FILE NOT FOUND ON 'OLD'
C             -9    FILE ALREADY EXISTS ON 'NEW'
C             -10   DEVICE HANDLER DOES NOT EXIST
C             -11   NOT ENOUGH ROOM FOR HANDLER
C             -12   DEVICE DOES NOT EXIST
C             -13   DEVICE HARDWARE ERROR
C
C     PROGRAMMER:  ROBERT WALRAVEN, UCD  VERSION 3.0   3 JUL 1980
C
C-------------------------------------------------------------------

FORTRAN IV      V02.04     Tue 30-Sep-80 16:20:35              PAGE 002

       C
       C
0002         COMMON /IOFILE/ SPEC,OUT,DBLK,BUF,EXT
0003         INTEGER SPEC(39), OUT(6) ,TYPE, NAME(1), DBLK(4), BUF(65), DEVNAM
0004         REAL EXT(2)
0005         EQUIVALENCE (DEVNAM,SPEC(16))
0006         DATA DATDAT/6RDATDAT/
       C
0007         EXT(1) = DATDAT
0008         EXT(2) = DATDAT
0009         DO 10 I=1,4
0010   10    DBLK(I) = 0
0011         ICHAN = IGETC()                          !GET A FREE CHANNEL
0012         IF (ICHAN.LT.0) GO TO 1000
0014         IF (NAME(1) .EQ. 0) GO TO 20             !BRANCH IF QUERY WANTED
0016         CALL IRAD50 (12, NAME, DEVNAM)           !CONVERT NAME TO RAD50
0017         GO TO 110
```

```
20      WRITE(5,100)                            !QUERY FOR FILENAME
100     FORMAT(1X'FILENAME'/)
        I = -ICSI(SPEC,EXT,,,0)
        IF (I.GT.0) GO TO 1010
110     I = IFETCH(DEVNAM)                      !FETCH THE HANDLER
        IF (I.NE.0) GO TO 1160
        DBLK(1) = DEVNAM                        !GET DEVICE NAME
        I = LOOKUP (ICHAN,DBLK)                 !NON-FILE LOOKUP
        I = ISPFNW ('377, ICHAN, 0, BUF, 1)     !TRY READING
        IF (I.EQ.2) GO TO 1230
        CALL PURGE(ICHAN)
        I = LOOKUP(ICHAN,DEVNAM)                !CHECK STATUS OF FILE
        IF (I.EQ.-3) GO TO 1020                 !ERROR IF DEVICE IN USE
        IOFILE = I
        IF (TYPE .EQ. 'NE') GO TO 200           !'NEW' FILE?
        IF (TYPE .EQ. 'UN') GO TO 300           !'UNKNOWN' FILE?
        IF (TYPE .NE. 'OL') GO TO 1100          !ERROR IF NOT 'OLD' FILE
        IF (I .EQ. -2) GO TO 1120               !OLD FILE MUST EXIST
        IF (I .LT. 0) GO TO 1040                !AND LOOKUP BE O.K.
        ISIZE = 0
        ICODE = 32                              !READ ONLY
        GO TO 400
200     IF (I .GT. 0) GO TO 1140                !NEW FILE MUST NOT EXIST
        IF (I .EQ. 0) GO TO 300                 !FILE-STRUCTURED DEVICE?
        IF (I .NE. -2) GO TO 1040               !LOOKUP MUST BE O.K.
300     ICODE = 1                               !DOUBLE BUFFERED
400     I=IASIGN(LUN,DEVNAM,SPEC(17),ISIZE,ICODE) !ASSIGN THE FILE
        IF (I.NE.0) GO TO 1050
        IF (IOFILE .LT. 0) IOFILE=0
        GO TO 2000
C
C--------------------ERRORS PROCESSED HERE-----------------------------
C
1000    IOFILE = -1
        GO TO 1060
1010    IOFILE = IERR-3
        GO TO 1060

FORTRAN IV      V02.04     Tue 30-Sep-80 16:20:35

1020    IOFILE = -4
        GO TO 1060
1040    IOFILE = -6
        GO TO 1060
1050    IOFILE = -5
1060    WRITE (5,1070) IOFILE
1070    FORMAT(1X'IOFILE ERROR = 'I3)
        GO TO 2000
1100    WRITE (5,1110)
1110    FORMAT(1X'ILLEGAL TYPE')
        IOFILE = -7
        GO TO 2000
1120    WRITE(5,1130)
1130    FORMAT(1X'FILE NOT FOUND')
        IOFILE = -8
        GO TO 2000
1140    WRITE (5,1150)
1150    FORMAT(1X'FILE ALREADY EXISTS')
        IOFILE = -9
        GO TO 2000
1160    IOFILE = I-13
```

```
        GO TO (1170,1190,1210),I
1170    WRITE (5,1180)
1180    FORMAT(1X'DEVICE HANDLER DOES NOT EXIST')
        GO TO 2000
1190    WRITE (5,1200)
1200    FORMAT(1X'NOT ENOUGH ROOM FOR HANDLER')
        GO TO 2000
1210    WRITE (5,1220)
1220    FORMAT(1X'DEVICE DOES NOT EXIST')
        GO TO 2000
1230    WRITE (5,1240)
1240    FORMAT(1X'DEVICE HARDWARE ERROR')
        IOFILE = -13
C
C--------------------PURGE CHANNEL AND RETURN----
C
2000    CALL PURGE (ICHAN)
        CALL IFREEC (ICHAN)
        RETURN
        END
IOFILE
*

C       DIREC.FOR
C
        SUBROUTINE DIREC(LUN,INPUT)
C
C------------------------------------------------------------------------
C
C       'DIREC' OUTPUTS A DIRECTORY OF THE DEVICE NAMED BY THE STRING
C       'INPUT' ON THE LOGICAL UNIT 'LUN'.  FOR EXAMPLE, TO GET A
C       DIRECTORY OF DK: ON THE LINE PRINTER,  CALL DIREC (6,'DK:')
C
C       WRITTEN BY ROBERT WALRAVEN, UCD - APPLIED SCIENCE
C       VERSION 2.0, 6 FEB 80
C
C------------------------------------------------------------------------
C
        INTEGER BUF(512),INPUT(1),OUT(7),LINE(32)
        REAL DATE(12)
        LOGICAL*1 NAME(10)
        DATA DATE/'JAN-','FEB-','MAR-','APR-','MAY-','JUN-','JUL-',
       1          'AUG-','SEP-','OCT-','NOV-','DEC-'/
C
C------------------- INITIALIZATION ----------------------------------
C
        IF (LUN .NE. 5  .AND.  LUN .NE. 6) RETURN
        DO 10 I=1,7
10      OUT(I)=0
        CALL IRAD50 (4,INPUT,OUT)
        ICHAN = IGETC()
        IF (ICHAN .LT. 0) RETURN
        I = IFETCH(OUT)
        IF (I .NE. 0) GO TO 400
        I = LOOKUP(ICHAN,OUT)
        IF (I .LT. 0) GO TO 400
        NEXT = 1
        NPTR = 1
        NBUF = 512
        NFILES = 0
        NFBLKS = 0
        NFREE  = 0
```

```
C
C--------------------- GET NEXT SEGMENT --------------------------------
C
  100 IF (NEXT .EQ. 0) GO TO 300
      NBLK = 2*NEXT + 4
      J = IREADW(NBUF,BUF,NBLK,ICHAN)
      IF (J .LT. 0) GO TO 400
      IF (NEXT .EQ. 1) NN = BUF(4)
      NEXT = BUF(2)
      IWORD = 6
C
C--------------------- PROCESS ENTRIES ---------------------------------
C
  200 DO 205 I = 1,7
      OUT(I) = BUF(IWORD)

  205 IWORD = IWORD + 1
      IWORD = IWORD + NN
      LEN = OUT(5)
      IF (OUT(1) .EQ. "4000) GO TO 100
      IF (OUT(1) .NE. "1000) GO TO 220
      NFREE = NFREE + OUT(5)
      ENCODE (34,210,LINE(NPTR)) LEN
  210 FORMAT ('< UNUSED >',I5,15X'  ')
      GO TO 260
  220 NFILES = NFILES + 1
      NFBLKS = NFBLKS + OUT(5)
      CALL R50ASC (6,OUT(2),NAME)
      NAME(7) = '.'
      CALL R50ASC (3,OUT(4),NAME(8))
      IDATE = OUT(7)
      MO = IDATE/1024
      IDATE = IDATE-MO*1024
      IDAY = IDATE/32
      IYR = IDATE-IDAY*32 + 72
      IF (OUT(7) .EQ. 0) MO = 1
      IF (OUT(1) .EQ.  "400) GO TO 240
      ENCODE (34,230,LINE(NPTR)) NAME,LEN,IDAY,DATE(MO),IYR
  230 FORMAT (10A1,I5,I4,'-',A4,I2,'      ')
      GO TO 260
  240 JOB = OUT(6)/256
      NCHAN = OUT(6)-JOB*256
      ENCODE (34,250,LINE(NPTR)) NAME,LEN,IDAY,DATE(MO),IYR,NCHAN
  250 FORMAT(10A1,I5,I4,'-'A4,I2' T'I3' ')
  260 NSTART = NSTART + LEN
      IF (NPTR .EQ. 1) GO TO 280
      WRITE (LUN,270) LINE
  270 FORMAT (1X,32A2)
      NPTR = 1
      GO TO 200
  280 NPTR = 17
      GO TO 200
C
C--------------------- ALL DONE ----------------------------------------
C
  300 IF (NPTR .EQ. 1) GO TO 320
      WRITE (LUN,310) (LINE(I),I=1,16)
  310 FORMAT (1X,16A2)
  320 WRITE (LUN,330) NFILES, NFBLKS,NFREE
  330 FORMAT(1X,I6' FILES,'I6' BLOCKS'/1X,I6' FREE BLOCKS')
  400 CALL PURGE (ICHAN)
      CALL IFREEC (ICHAN)
      RETURN
      END
```

New Self-Paced RT-11, V4, Training Available

Educational Services announces the availability of new RT-11, V4, Self-Paced Instruction (SPI) courses for users and programmers.

Course Descriptions:

The emphasis of the **RT-11, V4, User** course is on system concepts and use of the system utilities and language processors. Upon completion of this course, students will be able to carry out program development, system installation, and system maintenance.

The **RT-11 Programmer** course emphasizes the use of the RT-11 monitor and device handler services from within user-written programs in FORTRAN and/or MACRO. It is recommended for anyone who intends to program in MACRO under RT-11 and for FORTRAN programmers whose need for functionality and/or efficiency exceeds that provided by the FORTRAN language itself.

Major areas covered in this course include I/O to the console terminal and to other peripherals; inter-task communication; scheduling and timing; and calling conventions for FORTRAN and MACRO routines.

Upon completion of the **Programmer** course, students will be able to:

- Describe how programmed requests are implemented in user programs and how they are executed by the operating system.
- Write code that will transfer information between memory and the console terminal and any non file-structured device or any file on a file-structured device.
- Describe the ways in which system services can be used to minimize program size and maximize program throughput.
- Write code that will allow foreground and background programs to communicate and synchronize with each other.

Benefits of Self-Paced Instruction:

Self-paced instruction is a cost-effective alternative to lecture classes. Students can study at their workplace, so the costs of training associated with travel and time off work are eliminated. Also, the results of training can be seen immediately; students can apply what they learn to their current job assignments. Self-Paced Instruction can be scheduled when it is needed, and students can always use the course for reference and for review.

Resources:

Students will need the following resources and references to benefit fully from either course:

- A course administrator should be appointed to assist students in the mechanics of completing each course successfully.
- Students should have access to a knowledgeable RT-11, V4, user and programmer who acts as a technical expert.
- Students will need access to both a PDP-11 system running RT-11, V4, and a terminal.
- The Digital documentation required to support either course is listed in each module.

RT-11 TRAINING PROGRAM

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

	Course #	Title	Length	Start Day	Tuition	Training Credit	January				February				March					April				May				June				
							5	12	19	26	2	9	16	23	2	9	16	23	30	6	13	20	27	4	11	18	25	1	8	15	22	29
P	EY-J2114-AO	Introduction to Minicomputers A/V	≈5		$ 445	1									CONTINUOUSLY AVAILABLE AT FACILITY																	
P	EY-J2016-AO	Introduction to Minicomputers	5	MO	575	1			●					●						●				●				●				
P	EY-J2116-AO	Introduction to the PDP-11 A/V	≈5		495	1									CONTINUOUSLY AVAILABLE AT FACILITY																	
P	EY-J2024-AO	PDP-11 Assembly Language Programming	5	MO	700	1	●		●		●		●		●		●			●		●		●		●		●		●		
P	EY-J2050-AO	Programming in MACRO-11	5	MO	700	1		●		●		●		●				●			●		●		●				●		●	
P	EY-J2048-AO	PDP-11 Concepts	3	MO	575	1				●						●						●							●			
	EY-J2000-AO	RT-11 User	5	MO	700	1	●													●												
	EY-J0016-AO	Programming in FORTRAN IV	5	MO	700	1		●							●						●					●						
	EY-J2002-AO	RT-11 Programmer ·	5	MO	700	1																										
	EY-J2004-AO	RT-11 System Programmer	3		700	1																										

CHICAGO (312) 640-5520

							5	12	19	26	2	9	16	23	2	9	16	23	30	6	13	20	27	4	11	18	25	1	8	15	22	29
P	EY-J2114-AO	Introduction to Minicomputers A/V	≈5		$ 445	1									CONTINUOUSLY AVAILABLE AT FACILITY																	
P	EY-J2016-AO	Introduction to Minicomputers	5	MO	575	1	●				●						●			●					●					●		
P	EY-J2116-AO	Introduction to the PDP-11 A/V	≈5		495	1									CONTINUOUSLY AVAILABLE AT FACILITY																	
P	EY-J2024-AO	PDP-11 Assembly Language Programming	5	MO	700	1	●			●			●			●			●		●			●		●				●		
P	EY-J2050-AO	Programming in MACRO-11	5	MO	700	1		●			●	●		●			●		●			●			●			●			●	
P	EY-J2048-AO	PDP-11 Concepts	3		575	1																										
	EY-J2000-AO	RT-11 User	5	MO	700	1																	●									
	EY-J0016-AO	Programming in FORTRAN IV	5	MO	700	1			●							●					●									●		
	EY-J2002-AO	RT-11 Programmer	5	MO	700	1																		●								
	EY-J2004-AO	RT-11 System Programmer	3		700	1																										

BOSTON (617) 275-5000 Ext. 2380

							5	12	19	26	2	9	16	23	2	9	16	23	30	6	13	20	27	4	11	18	25	1	8	15	22	29
P	EY-J2114-AO	Introduction to Minicomputers A/V	≈5		$ 445	1									CONTINUOUSLY AVAILABLE AT FACILITY																	
P	EY-J2016-AO	Introduction to Minicomputers	5	MO	575	1	●		●		●				●		●			●			●			●			●		●	
P	EY-J2116-AO	Introduction to the PDP-11 A/V	≈5		495	1									CONTINUOUSLY AVAILABLE AT FACILITY																	
P	EY-J2024-AO	PDP-11 Assembly Language Programming	5	MO	700	1		●		●		●		●		●		●		●		●		●		●		●		●		
P	EY-J2050-AO	Programming in MACRO-11	5	MO	700	1	●		●		●		●		●		●		●	●		●		●		●		●	●		●	
P	EY-J2048-AO	PDP-11 Concepts	3	MO	575	1				●							●						●							●		
	EY-J2000-AO	RT-11 User	5	MO	700	1	●							●		●					●			●					●			
	EY-J0016-AO	Programming in FORTRAN IV	5	MO	700	1				●			●			●			●			●							●			
	EY-J2002-AO	RT-11 Programmer	5	MO	700	1		●									●					●			●					●		
	EY-J2004-AO	RT-11 System Programmer	3	WE	700	1			●														●								●	

U.S. Domestic Prices

USER REQUESTS

Some weeks ago I got my upgrade kit for RT-11 VØ4.ØØ and most of it worked fine. The new TECO is fantastic (up to now I had only V28) and the supplied macros VTEDIT, TYPE etc would be beautiful if adequate documentation were included. It seems to me that the TECO manual is incomplete because it doesn't describe the startup procedure with TECO.INI and TECO.TEC nor the :EGcommand$ under RT-11. Probably you will agree that it is very hard to find out how the VTEDIT macro works without additional documentation. And it means just too great an effort and too much time to me to understand VTEDIT well enough to install the additional features I had implemented in the VT52 macro.
For these reasons I'd like to ask you the following questions:

- how exactly is the initialization done?

- what does the EGmd$ command under RT-11 do?

- where is th Ø:W flag set? Sometimes it is Ø and sometimes 4 (which is correct).
  If I use the switch/LC the message "Lower case not available" is printed and with/SC it is "your terminal doesn't support scrolling".

- Most important of all:
  The header in the VTEDIT macro mentions a formatter macro (q-register M), but no information is given on what it should do and how it should work. Do you have a macro that I could use as a model? On a scope this is really a most important feature.
  I cannot use my old formatter anymore! (I need formatter macros for structured languages and for text).

- Do you plan to write a macro that takes full advantage of the VT1ØØ keypad and that doesn't have the VT52 - overhead? Or a macro for the VT132?

May be some of these questions have been answered in a SIG news letter. But although I applied for a membership to the RT-11 and the TECO-SIG I don't get the 'Minitasker' nor the 'Moby Mungo'? only 'Euroscope'.
I would be very glad if you could take care of my applications this time (my DECUS membership number is 134478) and I am also very grateful for any back issues of the 'Moby Mungo'.
I'm sure you will understand that we depend strongly on a working TECO and therefore rely on a prompt answer.

My address:

Dr. U. Büchler
c/o CIBA-GEIGY AG
R-1055.4.78
Laborautomation

4002 Basel
Switzerland

The University of Michigan

THE HARRISON M. RANDALL LABORATORY
OF PHYSICS
•
ANN ARBOR, MICHIGAN 48109
(313) 764-4437

In response to the user request (Vol. 7, No. 1 page 22) I offer a solution. We also must transport files via tape from RT-11 to VAX and back. We have a solution. Enclosed are listings of programs to read RT-11 tapes on a VAX and make FILES-11 output.

In the other direction we have programs to read DOS format tapes on the RT-11 system and output RT-11 disk files.

These programs are available on tape if required.

Sincerely,

Jonn LoSecco

```
                                        .TITLE  DOS READ PROGRAM  READS DOS FORMAT MAG TAPES
                                        .MCALL  .PRINT,.EXIT
                                        .GLOBL  DENS,R50ASC,BUFRIN,UNLOAD,TRSET
                                        .GLOBL  REWIND
000000                          START:
000000  004567  000000G                 JSR     R5,DENS
000004  000204'                         .WORD   UNI
000006                          NXT:
000006  004567  000000G                 JSR     R5,BUFRIN
000012  000204'                         .WORD   UNI
000014  000206'                         .WORD   BYT
000016  000212'                         .WORD   BUFR
000020  000210'                         .WORD   ERR
000022  005767  000162                  TST     ERR
000026  001775                          BEQ     .-4
000030  100031                          BPL     ERW
000032  005767  000132                  TST     EOFC    ;DID WE JUST PASS AN EOF?
000036  100763                          BMI     NXT     ;NO KEEP LOOKING
000040  012705  000172'                 MOV     #ARG,R5 ;ARG LIST
000044  004767  000000G                 JSR     PC,R50ASC       ;CONVERT TO ASCII
000050  116767  001154  001153          MOVB    OPT+8.,OPT+9.   ;MOVE IT OVER
000056  116767  001145  001144          MOVB    OPT+7.,OPT+8.   ;MOV IT TO
000064  116767  001136  001135          MOVB    OPT+6.,OPT+7.   ;LAST ONE
000072  112767  000056  001126          MOVB    #'.,OPT+6       ;PUT IN THE DOT!
000100                                  .PRINT  #OPT
000106  005367  000056                  DEC     EOFC    ;RESET INDICATOR
000112  000735                          BR NXT
```

```
000114                          ERW:
000114  022767  000002  000066          CMP     #2,ERR  ;IS IT AN EOF?
000122  001412                          BEQ     WRIT    ;THAN FLAG TO WRITE NEXT
000124  116767  000060  001064          MOVB    ERR,ERP+4       ;PUT OUT THE DIGIT
000132  062767  000060  001056          ADD     #60,ERP+4       ;MAKE IT ASCII
000140                                  .PRINT  #ERP
000146  000717                          BR      NXT
000150                          WRIT:
000150  005267  000014                  INC     EOFC    ;LOG IT
000154  003714                          BLE     NXT     ;KEEP GOING IF ONLY ONE
                                ;       JSR     R5,TRSET        ;KICK IT HARD
000156  004567  000000G                 JSR     R5,REWIND
                                ;       JSR     R5,UNLOAD
000162  000204'                         .WORD   UNI
000164  000210'                         .WORD   ERR
000166                                  .EXIT   ;LEAVE IF TWO IN A ROW
000170  177777                  EOFC:   .WORD   -1      ;FLAG FOR EOFS
000172  000003                  ARG:    .WORD   3       ;NUM OF ARGS
000174  000202'                         .WORD   CNT
000176  000212'                         .WORD   BUFR
000200  001220'                         .WORD   OPT
000202  000011                  CNT:    .WORD   9.
000204  000000                  UNI:    .WORD   0
000206  001000                  BYT:    .WORD   512.
000210  000000                  ERR:    .WORD   0
000212                          BUFR:   .BLKW   256.
001212     105     122     122  ERP:    .ASCIZ  /ERR  /
001215     040     040     000
001220                          OPT:    .BLKB   10.     ;UP TO 9 CHARACTERS
001232     000                          .BYTE   0       ;TO END PRINT
```

```
                                        .TITLE  DOS READ PROGRAM  READS DOS FUKMAT MAG TAPES
                                ;               TRANSFERS FILES TO THE DISK
                                        .MCALL  .PRINT,.EXIT,.ENTER,.WRITW,.CLOSE
                                        .GLOBL  DENS,R50ASC,BUFRIN,UNLOAD,TRSET
000000                          START:
000000  004567  000000G                 JSR     R5,DENS
000004  000334'                         .WORD   UNI
000006                          NXT:
000006  004567  000000G                 JSR     R5,BUFRIN
000012  000334'                         .WORD   UNI
000014  000336'                         .WORD   BYT
000016  000360'                         .WORD   BUFR
000020  000340'                         .WORD   ERR
000022  005767  000312                  TST     ERR
000026  001775                          BEQ     .-4
000030  100100                          BPL     ERW
000032  005767  000262                  TST     EOFC    ;DID WE JUST PASS AN EOF?
000036  002033                          BGE     WNAM    ;GET NAME
000040  016767  000274  000274          MOV     ERR,CNTW        ;GET COUNT
000046  062767  001001  000266          ADD     #513.,CNTW      ;GET NUM READ
000054  006267  000262                  ASR     CNTW    ;GET WORD NUM
000060                                  .WRITW  #AREA,#1,#BUFR,CNTW,BLK
000120  005267  000222                  INC     BLK     ;BOUNCE POINTER
000124  000730                          BR      NXT     ;GET MORE
000126                          WNAM:
000126  012705  000322'                 MOV     #ARG,R5 ;ARG LIST
000132  004767  000000G                 JSR     PC,R50ASC       ;CONVERT TO ASCII
000136  116767  001234  001233          MOVB    OPT+8.,OPT+9.   ;MOVE IT OVER
000144  116767  001225  001224          MOVB    OPT+7.,OPT+8.   ;MOV IT TO
000152  116767  001216  001215          MOVB    OPT+6.,OPT+7.   ;LAST ONE
000160  112767  000056  001206          MOVB    #'.,OPT+6       ;PUT IN THE DOT!
```

```
32 000166                                  .PRINT  #OPT
33 000174  005367  000120                  DEC     EOFC    ;RESET INDICATOR
34 000200                                  .ENTER  #AREA,#1,#NAME,#0       ;OPEN FILE
35 000224  005067  000116                  CLR     BLK     ;SART AT FIRST BLOCK
36 000230  000666                          BR NXT
37 000232                          ERW:
38 000232  022767  000002  000100          CMP     #2,ERR  ;IS IT AN EOF?
39 000240  001412                          BEQ     WRIT    ;THAN FLAG TO WRITE NEXT
40 000242  116767  000072  001114          MOVB    ERR,ERP+4       ;PUT OUT THE DIGIT
41 000250  062767  000060  001106          ADD     #60,ERP+4       ;MAKE IT ASCII
42 000256                                  .PRINT  #ERP
43 000264  000650                          BR      NXT
44 000266                          WRIT:
45 000266                                  .CLOSE  #1      ;CLOSE THE FILE
46                                 ;               HOPEFULLY IT WILL IGNORE THE CLOSE
47 000274  005267  000020                  INC     EOFC    ;LOG IT
48 000300  003642                          BLE     NXT     ;KEEP GOING IF ONLY ONE
49 000302  004567  000000G                 JSR     R5,TRSET        ;KICK IT HARD
50 000306  004567  000000G                 JSR     R5,UNLOAD
51 000312  000334'                         .WORD   UNI
52 000314  000340'                         .WORD   ERR
53 000316                                  .EXIT   ;LEAVE IF TWO IN A ROW
54 000320  177777                  EOFC:   .WORD   -1      ;FLAG FOR EOFS
55 000322  000003                  ARG:    .WORD   3       ;NUM OF ARGS
56 000324  000332'                         .WORD   CNT
57 000326  000360'                         .WORD   BUFR

    58 000330  001366'                         .WORD   OPT
    59 000332  000011                  CNT:    .WORD   9.
    60 000334  000000                  UNI:    .WORD   0
    61 000336  001000                  BYT:    .WORD   512.
    62 000340  000000                  ERR:    .WORD   0
    63 000342  000000                  CNTW:   .WORD   0       ;WORD COUNT FOR DISK
    64 000344  000000                  AREA:   .WORD   0       ;AREA FOR I/O
    65 000346  000000  000000  000000  BLK:    .WORD   0,0,0,0 ;RECORD COUNT
       000354  000000
    66 000356  015270                  NAME:   .RAD50  /DK/    ;PUT IT ON THE DISK
    67 000360                          BUFR:   .BLKW   256.
    68 001360     105     122     122  ERP:    .ASCIZ  /ERR  /
       001363     040     040     000
    69 001366                          OPT:    .BLKB   10.     ;UP TO 9 CHARACTERS
    70 001400     000                          .BYTE   0       ;TO END PRINT
    71         000000'                         .END    START

 1                                     .TITLE  TAPE ROUTINES
 2                              ;               MACRO CALLABLE - A VERSION EXISTS FOR FORTRAN USERS
 3        172520                MTS=172520
 4        172522                MTC=172522
 5        172524                MTBRC=172524
 6        172526                MTCMA=172526
 7        172530                MTD=172530
 8        172532                MTRD=172532
 9                              ;       CALL  BUFRIN(UNIT,BYTES,ADDRESS,ERROR)
10                              ;       CALL  BUFOUT(UNIT,BYTES,ADDRESS,ERROR)
11                              ;       CALL  DENS(IDENSE)    IDENSE=0 MEANS LOW DENSITY ; IDENSE NONZERO MEANS 6250BPI
12                              ;       CALL  TWAIT                 -LOOPS UTIL DRIVE READY
13                              ;       CALL  REWIND(UNIT,ERROR)
14                              ;       CALL  WEOF(UNIT,ERROR)
15                              ;       CALL  UNLOAD(UNIT,ERROR)
16                              ;       CALL  TRSET                 DOES A POWER CLEAR ON THE DRIVES
17                                      .GLOBL  BUFRIN,BUFOUT,DENS,WEOF,UNLOAD,TWAIT,REWIND,TRSET
```

```
18 000000                          BUFRIN:
19 000000  012746                          MOV     (PC)+,-(SP)     ;SET READ
20 000002  040103                  DENR:   .WORD   040103  ;READ HIGH DENSITY INTERUPT ON
21 000004  004567  000126                  JSR     R5,TWAIT        ;WAIT FOR UNIT
22 000010  113537  172523                  MOVB    @(R5)+,@#MTC+1  ;GET UNIT #
23 000014  013537  172524                  MOV     @(R5)+,@#MTBRC  ;GET BYTE COUNT
24 000020  005437  172524                  NEG     @#MTBRC         ;USE NEGATIVE COUNT
25 000024  012537  172526                  MOV     (R5)+,@#MTCMA   ;TRANSFER ADDRESS
26 000030                          GOOP:
27 000030  012567  000112                  MOV     (R5)+,ERRA      ;GET ERROR ADDRESS
28 000034  042737  060177  172522          BIC     #60177,@#MTC    ;CLEAR PREVIOUS STUFF
29 000042  052637  172522                  BIS     (SP)+,@#MTC     ;SET DENSITY,READ AND GO BITS
30 000046  032737  000100  172520          BIT     #100,@#MTS      ;DOES UNIT EXIT?
31 000054  001403                          BEQ     NXST            ;NOPE NO UNIT
32 000056  005077  000064                  CLR     @ERRA           ;ZERO MEANS BUSY
33 000062  000205                          RTS     R5              ;LEAVE
34 000064                          NXST:
35 000064  012777  000001  000054          MOV     #1,@ERRA        ;1 CODE FOR NON EXISTANT DRIVE
36 000072  000205                          RTS     R5              ;QUIT
37 000074                          BUFOUT:
38 000074  012746                          MOV     (PC)+,-(SP)     ;SET WRITE
39 000076  040105                  DENW:   .WORD   040105  ;WRITE HIGH DENSITY INTERUPT ON
40 000100  000741                          BR      BUFRIN+4                ;SAME THING
41 000102                          UNLOAD:
42 000102  012746  000101                  MOV     #00101,-(SP)            ;REWIND AND UNLOAD
43 000106  000405                          BR      REWIND+4                ;SAME AS REWIND
44 000110                          WEOF:
45 000110  012746                          MOV     (PC)+,-(SP)     ;GET WEOF AT RIGHT DENSITY
46 000112  040107                  DENE:   .WORD   040107  ;WRITE EOF HIGH DENSITY INTERUPT ON
47 000114  000402                          BR      REWIND+4                ;SAME AS REWIND
48 000116                          REWIND:
49 000116  012746  000117                  MOV     #00117,-(SP)    ;CODE FOR REWIND
50 000122  004567  000010                  JSR     R5,TWAIT        ;WAIT FOR THE DRIVE
51 000126  113537  172523                  MOVB    @(R5)+,@#MTC+1          ;GET UNIT
52 000132  000736                          BR      GOOP            ;GO FINISH OPERATION
53 000134  000001                          WAIT            ;SINCE INTERUPT IS ENABLED SHOULD WORK UNLESS PSW PRI=7
54 000136                          TWAIT:
55 000136  105737  172522                  TSTB    @#MTC           ;CHECK TAPE USIT
56 000142  100374                          BPL     .-6             ;LOOP UNTIL READY
57 000144  000205                          RTS     R5              ;RETURN

  58 000146  000000                  ERRA:   .WORD   0       ;ERROR WORD ADDRESS
  59 000150                          TRSET:
  60 000150  004567  177762                  JSR     R5,TWAIT        ;WAIT FOR THE CONTROLLER
  61 000154  052737  010000  172522          BIS     #10000,@#MTC    ;RESET THE TAPE DRIVES
  62 000162  000205                          RTS     R5      ;RETURN
  63 000164                          DENS:
  64 000164  005735                          TST     @(R5)+  ;GET DENSITY
  65 000166  001412                          BEQ     11$             ;ZERO MEANS LOW DENSITY
  66 000170  052767  040000  177604          BIS     #40000,DENR     ;SET READ DNESITY
  67 000176  052767  040000  177672          BIS     #40000,DENW     ;SET WRITE DENSITY
  68 000204  052767  040000  177700          BIS     #40000,DENE     ;SET EOF DENSITY
  69 000212  000205                          RTS     R5      ;SCRAM
  70 000214                          11$:
  71 000214  042767  060000  177560          BIC     #60000,DENR     ;LOW READ
  72 000222  042767  060000  177646          BIC     #60000,DENW     ;LOW WRITE
  73 000230  042767  060000  177654          BIC     #60000,DENE     ;LOW DENSITY EOF
  74 000236  000205                          RTS     R5      ;SCRAM
  75 000240                          TAPINT:
  76 000240  005737  172522                  TST     @#MTC           ;IS ERROR FLAG ON?
  77 000244  002411                          BLT     FIN     ;OK END
  78 000246  013777  172524  177672          MOV     @#MTBRC,@ERRA   ;SHOULD BE ZERO IF ALL OK
```

```
000254  005377  177666                  DEC     @ERRA           ;-1 MEANS TRANSFER OK
                                                                ;ERRA=WORDS TRANSFERED-BYTES REQUESTED-1
                                ;       MOV     #177777,@ERRA   ;-1 MEANS OP FINISHED OK
000260  042737  000100  172522          BIC     #100,@#MTC      ;DISABLE INTERUPT
000266  000002                          RTI
000270                          FIN:
000270  032727  040000  172520          BIT     #40000,#MTS     ;IS IT EOF?
000276  001403                          BEQ     7$
000300  012777  000002  177640          MOV     #2,@ERRA        ;CODE 2 FOR EOF
000306                          7$:
000306  032737  001000  172520          BIT     #1000,@#MTS     ;WORD COUNT TOO SHORT?
000314  001403                          BEQ     8$      ;NO
000316  012777  000004  177622          MOV     #4,@ERRA        ;CODE 4 FOR LENGTH ERROR
000324                          8$:
000324  032737  002000  172520          BIT     #2000,@#MTS     ;END OF TAPE?
000332  001403                          BEQ     9$      ;NO
000334  012777  000003  177604          MOV     #3,@ERRA        ;CODE 3 FOR EOT
000342                          9$:
000342  032737  134600  172520          BIT     #134600,@#MTS   ;ALL OTHERS
000350  001403                          BEQ     10$     ;NONE
000352  012777  000005  177566          MOV     #5,@ERRA        ;5 OTHER PROBLEMS
000360                          10$:
000360  042737  000100  172522          BIC     #100,@#MTC      ;DISABLE INTERUPT
000366  000002                          RTI             ;CLEAR INTERUPT
000000                                  .ASECT
        000224                  .=224
000224  000240'                         .WORD   TAPINT
000226  000240                          .WORD   240     ;LEVEL 5?
        000001                          .END
```

GETRT.FOR

```
        BYTE INP(512)
        CHARACTER*17 FILID
        EQUIVALENCE (FILID,INP(5))
        DATA NFIL/0/
        PARAMETER IU=1
        CALL BUFFERIN(IU,1,INP,128,IER)
5       CALL BUFFERIN(IU,1,INP,128,IER)
        IF(IER.NE.2)GO TO 7
        NFIL=NFIL+1
        PRINT 3,FILID
        CALL SKIP(IU,1)
C       SKIP OVER HEADER-- JUST AN EOF
        OPEN(UNIT=10,TYPE='NEW',NAME=FILID,FORM='UNFORMATTED')
18      CALL BUFFERIN(IU,1,INP,128,IER)
        IF(IER.NE.2) GO TO 17
        WRITE(10)INP
        GO TO 18
17      CLOSE(UNIT=10,DISPOSE='KEEP')
        CALL SKIP(IU,1)
C       SKIP TRAILER RECORD
C       CALL SKIP(IU,3)
        GO TO 5
7       PRINT 4,NFIL
        STOP
3       FORMAT(' ',A17)
4       FORMAT(' NUMBER OF FILES READ=',I7)
        END
```

GETRTM.FOR

```
C       PROGRAM TO READ RT11 FORMAT TEXT FILES
C       WILL MANGLE .SAV .OBJ ETC.
C       FILES ARE CONVERTED FROM RT11 TAPE FORMAT
C       AND STORED ON DISK IN RSX (FILES11) FORMAT
C
        BYTE INP(512)
        CHARACTER*17 FILID
        CHARACTER*512 STRG
        CHARACTER*2 MATC
        CHARACTER*1 FF
        CHARACTER*140 LINE
        CHARACTER*1024 BUFA
        EQUIVALENCE (FILID,INP(5))
        EQUIVALENCE (STRG,INP(1))
        DATA NFIL/0/
        PARAMETER IU=1
        MATC(1:1)=CHAR(13)          !CR
        MATC(2:2)=CHAR(10)          !LF
        FF=CHAR(12)                 !FF
        CALL BUFFERIN(IU,1,INP,128,IER)
5       CALL BUFFERIN(IU,1,INP,128,IER)
        IF(IER.NE.2)GO TO 7
        NFIL=NFIL+1
        PRINT 3,FILID
        CALL SKIP(IU,1)
C       SKIP OVER HEADER-- JUST AN EOF
        OPEN(UNIT=10,TYPE='NEW',NAME=FILID,FORM='FORMATTED',
        1RECL=140,RECORDTYPE='VARIABLE',CARRIAGECONTROL='LIST')
        NCH=1
18      CALL BUFFERIN(IU,1,INP,128,IER)
        IF(IER.NE.2) GO TO 17
        IF(NCH.GT.513)NCH=1         !GIVE UP IF NO CR LF
        BUFA(NCH:NCH+511)=STRG
        NCH=NCH+512
19      IH=INDEX(BUFA,MATC)
        IF(IH.EQ.0)GO TO 18         !TEMP FOR NOW
        LINE=BUFA(1:IH-1)
        NCH=NCH-IH-1
C       NCH=NCH-IH
        K=INDEX(LINE,FF)            !SEARCH FOR FORM FEED
        IF(K.NE.0)THEN
                IF(K.LT.141)WRITE(10,11)LINE(1:K)
                IF(IH-K.LT.142)WRITE(10,11)LINE(K+1:IH-1)
                GO TO 14
        ENDIF
C       TYPE 111,IH,LINE(1:IH-1)
C111    FORMAT(' ',I3,A)
        IF(IH.LT.142)WRITE(10,11)LINE(1:IH-1)
11      FORMAT(A)
14      BUFA=BUFA(IH+2:)
        GO TO 19
17      CLOSE(UNIT=10,DISPOSE='KEEP')
        CALL SKIP(IU,1)
C       SKIP TRAILER RECORD
C       CALL SKIP(IU,3)
        GO TO 5
7       PRINT 4,NFIL
```

```
        STOP
3       FORMAT(' ',A17)
4       FORMAT(' NUMBER OF FILES READ=',I7)
        END
C       PROGRAM TO READ A SPECIFIC FILE OFF AN RT11 TAPE
C       CONVERTS FROM RT11 FORMAT TO RSX (FILES11) FORMAT
C       WILL MANGLE BINARY FILES SUCH AS .SAV, .OBJ, .LDA
C
```

GETRTX.FOR

```
        BYTE INP(512)
        CHARACTER*17 FILID
        CHARACTER*512 STRG
        CHARACTER*2 MATC
        CHARACTER*1 FF
        CHARACTER*10 NAME,NAMX
        CHARACTER*140 LINE
        CHARACTER*1024 BUFA
        EQUIVALENCE (FILID,INP(5))
        EQUIVALENCE (STRG,INP(1))
        DATA NFIL/0/
        PARAMETER IU=1
        MATC(1:1)=CHAR(13)        !CR
        MATC(2:2)=CHAR(10)        !LF
        FF=CHAR(12)               !FF
        NAMX=' '
        ACCEPT 11,NAME
        IF(NAME.EQ.' ')STOP
        J=INDEX(NAME,'.')
        IF(J.EQ.0)THEN
                J=7
                NAME(7:7)='.'
        ENDIF
        NAMX(1:J-1)=NAME(1:J-1)
        NAMX(7:10)=NAME(J:J+3)
        CALL BUFFERIN(IU,1,INP,128,IER)
5       CALL BUFFERIN(IU,1,INP,128,IER)
        IF(IER.NE.2)GO TO 7
        NFIL=NFIL+1
        IF(FILID.NE.NAMX)GO TO 13
        PRINT 3,FILID
        CALL SKIP(IU,1)
C       SKIP OVER HEADER-- JUST AN EOF
        OPEN(UNIT=10,TYPE='NEW',NAME=FILID,FORM='FORMATTED',
        1RECL=140,RECORDTYPE='VARIABLE',CARRIAGECONTROL='LIST')
        NCH=1
18      CALL BUFFERIN(IU,1,INP,128,IER)
        IF(IER.NE.2) GO TO 17
        IF(NCH.GT.513)NCH=1       !GIVE UP IF NO CR LF
        BUFA(NCH:NCH+511)=STRG
        NCH=NCH+512
19      IH=INDEX(BUFA,MATC)
        IF(IH.EQ.0)GO TO 18       !TEMP FOR NOW
        LINE=BUFA(1:IH-1)
        NCH=NCH-IH-1
C       NCH=NCH-IH
        K=INDEX(LINE,FF)          !SEARCH FOR FORM FEED
        IF(K.NE.0)THEN
                IF(K.LT.141)WRITE(10,11)LINE(1:K)
                IF(IH-K.LT.142)WRITE(10,11)LINE(K+1:IH-1)
                GO TO 14
        ENDIF
C       TYPE 111,IH,LINE(1:IH-1)
C111    FORMAT(' ',I3,A)
        IF(IH.LT.142)WRITE(10,11)LINE(1:IH-1)
11      FORMAT(A)
14      BUFA=BUFA(IH+2:)
        GO TO 19
13      CALL SKIP(IU,3)
        GO TO 5
17      CLOSE(UNIT=10,DISPOSE='KEEP')
        CALL SKIP(IU,1)
C       SKIP TRAILER RECORD
7       PRINT 4,NFIL
        STOP
3       FORMAT(' ',A17)
4       FORMAT(' NUMBER OF FILES READ=',I7)
        END
```

```
          .TITLE BUFFIO
          .IDENT /01/
;
;BINARY MAG TAPE I/O ROUTINES.
;
;THESE EMULATE BUFFERIN-OUT AND BUFFIN-OUT ROUTINES THAT
;EXISTED ON THE SIGMA 7 AND INCLUDE EXTENSIONS IN COMMON
;USE AT HEPL.
;
         $SSDEF
         $IODEF

;        .LIBRARY /DRAO:[UTIL.TAPE]MACLIB.MLB/
          MACRO  DESCRIPTOR Z
          LONG   1$-2$
         .LONG   2$
2$        ASCII  /Z/
1$
          ENDM   DESCRIPTOR

         .PSECT BUFFIO,RD,WRT,EXE,NOSHR,QUAD

IOSB:    .LONG   0[32]


MTCHAN:  .WORD   0[16]


ARGLST:  $QIO
         NAMELEN=80
NAMEDESC:
         .LONG   NAMELEN
         .LONG   NAMEBUFF
TRANDESC:
         .LONG   NAMELEN
         .LONG   NAMEBUFF
NAMEBUFF:
         .BLKB   NAMELEN
FORONN:  DESCRIPTOR      <FORO!2ZL>
MESSAGE:  DESCRIPTOR     <HANG NEW TAPE THEN TYPE "CONTINUE">
```

```

;       FORTRAN CALLABLE PROCEDURE FOR SYNCHRONOUS INPUT FROM TAPE.
;
;               CALL BUFFERIN(UNIT,MODE,IBUF,NWANT,IERR [,NGOT])
;
;       UNIT = FORTRAN LOGICAL UNIT NUMBER BETWEEN 0 AND 15 INCLUSIVE.
;       MODE = (PRESENTLY NOT USED)
;       IBUF = DATA STORAGE BUFFER
;       NWANT = NUMBER OF 32 BIT WORDS REQUESTED.
;       IERR = 1        OPERATION INCOMPLETE
;              2        OPERATION SUCESSFUL
;              3        EOF OR EOT
;              4        RECOVERABLE ERROR (PARITY OR DATACHECK)
;       NGOT = OPTIONAL INTEGER CONTAINING THE ACTUAL NUMBER OF 32 BIT
;              WORDS TRANSFERRED.
;
        .ENTRY  BUFFERIN, ^M<R2,R3,R4>
        MOVAL   ARGLST,R4
        MOVW    #IO$_READLBLK,QIO$_FUNC(R4)
        BRB     SYNC

;       FORTRAN CALLABLE PROCEDURE FOR SYNCHRONOUS OUTPUT TO TAPE.
;
;               CALL BUFFEROUT(UNIT,MODE,IBUF,NWANT,IERR [,NGOT])
;
        .ENTRY  BUFFEROUT,^M<R2,R3,R4>
        MOVAL   ARGLST,R4
        MOVW    #IO$_WRITELBLK,QIO$_FUNC(R4)

SYNC:   MOVL    12(AP),QIO$_P1(R4)              ;DATA BUFFER ADDRESS
        MULL3   #4,@16(AP),QIO$_P2(R4)          ;NUMBER OF BYTES TO XFR
        JSB     GET_CHAN                        ;GET DEVICE CHANNEL NUMBER
        CLRL    QIO$_EFN(R4)                    ;USE EVENT FLAG ZERO
        MOVW    R2,QIO$_CHAN(R4)                ;CHANNEL
        MOVL    R3,QIO$_IOSB(R4)                ;IOSB ADDRESS
        $QIOW_G (R4)
        JSB     ERROR                           ;Q-ING ERROR?

        JSB     GET_STATUS                      ;GET OPERATION COMPLETION STATU
        MOVL    R0,@20(AP)                      ;RETURN IERR

        CMPB    (AP),#6                         ;CHECK IF BYTES XFRED IS WANTEI
        BNEQ    SYNC_RET
        MOVZWL  2(R3),R2                        ;GET BYTES XFRD COUNT FROM IOSI
        DIVL3   #4,R2,@24(AP)
SYNC_RET:
        RET


;       FORTRAN CALLABLE PROCEDURE FOR ASYNCHRONOUS INPUT FROM TAPE.
;
;               CALL BUFFIN(UNIT,MODE,IBUF,NWANT)
;
        .ENTRY  BUFFIN, ^M<R2,R3,R4>
        MOVAL   ARGLST,R4
        MOVW    #IO$_READLBLK,QIO$_FUNC(R4)
        BRB     ASYNC

;       FORTRAN CALLABLE PROCEDURE FOR ASYNCHRONOUL OUTPUT TO TAPE.
;
;               CALL BUFFOUT(UNIT,MODE,IBUF,NWANT)
;
        .ENTRY  BUFFOUT,^M<R2,R3,R4>
        MOVAL   ARGLST,R4
        MOVW    #IO$_WRITELBLK,QIO$_FUNC(R4)

ASYNC:  MOVL    12(AP),QIO$_P1(R4)              ;DATA BUFFER ADDRESS
        MULL3   #4,@16(AP),QIO$_P2(R4)          ;NUMBER OF BYTES TO XFR
        JSB     GET_CHAN                        ;GET DEVICE CHANNEL NUMBER
        MOVL    #23,QIO$_EFN(R4)                ;USE EVENT FLAG 23
        MOVW    R2,QIO$_CHAN(R4)                ;CHANNEL
        MOVL    R3,QIO$_IOSB(R4)                ;IOSB ADDRESS
        $QIO_G  (R4)
        JSB     ERROR                   ;Q-ING ERROR?
        RET


;       FORTRAN CALLABLE PROCEDURE USED IN CONJUNCTION WITH ASYNCHRONOUS
;       TAPE INPUT AND OUTPUT TO CHECK ON THE STATUS OF THE REQUESTED
;       I/O OPERATION
;
;               CALL ICHECK(UNIT [,IERR [,NGOT]])
;       OR
;               INTEGER = ICHECK(UNIT [,IERR [,NGOT]])
;
;       FUR THE FUNCTION CALL ICHECK IS EQUAL TO THE VALUE OF IERR.
;
        .ENTRY  ICHECK,^M<R2,R3>
        MOVL    @4(AP),R3                       ;UNIT NUMBER
        MOVAQ   IOSB[R3],R3                     ;UNIT'S IOSB ADDRESS
        JSB     GET_STATUS                      ;GET ERROR CODE
        CMPB    #1,(AP)
        BEQL    ICHECK_RET                      ;BR IF ONE ARG. CALL
        MOVL    R0,@8(AP)                       ;RETURN IERR AND ICHECK VALUE
        CMPB    #2,(AP)
        BEQL    ICHECK_RET                      ;BR IF TWO ARG. CALL
        MOVZWL  2(R3),R1                        ;RETURN WORDS XFRED FROM IOSB
        DIVL3   #4,R1,@12(AP)
ICHECK_RET:
        RET


;       INTERNAL PROCEDURE TO CHECK THE SUCESS OF SYSTEM SERVICE CALLS
;       ON ENTRY R0 MUST CONTAIN THE SS STATUS VALUE.
;
;               JSB     ERROR
;
;       UNSUCESS RESULTS IN IMAGE TERMINATION WITH DCL REPORTING THE
;       ERROR CONDITION CODE.
;
ERROR:
        BLBC    R0,ERROR_EXIT                   ;IF QIO OK, RETURN
        RSB
ERROR_EXIT:
        $EXIT_S R0                              ;FATAL.  LET DCL REPORT CAUSE.


;       INTERNAL PROCEDURE FOR ASSIGNING AND RETURNING AN I/O CHANNEL
;       NUMBER TO A FORTRAN LOGICAL UNIT NUMBER.
```

```
;
;               JSB     GET_CHAN
;
;       R2 IS RETURNED WITH THE CHANNEL NUMBER.
;       R3 IS RETURNED POINTING TO THE UNIT'S IOSB QUAD WORD.
;       ONCE A CHANNEL IS ASSIGNED, THE CHANNEL IS FETCHED FROM TABLE
;       MTCHAN.

GET_CHAN:
        MOVL    @4(AP),R3                       ;UNIT NUMBER
        MOVZWL  MTCHAN[R3],R2                   ;CHANNEL NUMBER IF ASSIGNED
        BEQL    FAO
        BRW     CHAN_RET

FAO:    MOVL   #80,NAMEDESC
        $FAO_S  CTRSTR=FORONN,-
                OUTBUF=NAMEDESC,-
                OUTLEN=NAMEDESC,-
                P1=R3

TRANSLOG:
        $TRNLOG_S       LOGNAM=NAMEDESC,RSLBUF=TRANDESC,RSLLEN=TRANDESC
        JSB     ERROR

        MOVZWL  TRANDESC,NAMEDESC
        MOVW    #NAMELEN,TRANDESC
        CMPW    #SS$_NOTRAN,R0
        BNEQ    TRANSLOG

        $ASSIGN_S DEVNAM=NAMEDESC,CHAN=MTCHAN[R3]
        JSB     ERROR                           ;ASSIGN ERROR?

        MOVZWL  MTCHAN[R3],R2
CHAN_RET:
        MOVAQ   IOSB[R3],R3                     ;UNIT'S IOSB ADDR.
        MOVW    #SS$_NORMAL,(R3)                ;SET COMPLETION STATUS TO OK
        RSB


;       INTERNAL PROCEDURE FOR CHECKING QIO COMPLETION STATUS.
;       CALLED WITH R3 POINTING TO UNIT'S IOSB QUAD WORD.
;
;               JSB     GET_CHAN
;
;       R0 IS RETURNED WITH THE VALUE OF IERR.
;
GET_STATUS:
        MOVZWL  (R3),R0                 ;GET COMPLETION STATUS
        BEQL    BUSY                            ;IF ZERO, I/O NOT YET COMPLET
        CMPW    R0,#SS$_NORMAL                  ;SUCESSFUL COMPLETION
        BEQL    OKAY
        CMPW    R0,#SS$_DATAOVERUN              ;SUCESSFUL COMPLETION
        BEQL    OKAY
        CMPW    R0,#SS$_ENDOFFILE               ;END OF FILE SENSED
        BEQL    EOF
        CMPW    R0,#SS$_ENDOFTAPE               ;END OF TAPE SENSED
        BEQL    EOF
        CMPW    R0,#SS$_PARITY                  ;VERTICAL PARITY ERROR
        BEQL    RECOV_ERR
        CMPW    R0,#SS$_DATACHECK               ;READ-AFTER-WRITE ERROR
        BEQL    RECOV_ERR
        $EXIT_S R0                              ;IF HERE, NON-RECOVERABLE ERR
                                                ;DCL WILL REPORT CAUSE
```

```

BUSY:   MOVL    #1,R0                                   ;IERR=1
        RSB
OKAY:   MOVL    #2,R0                                   ;IERR=2
        RSB
EOF:    MOVL    #3,R0                                   ;IERR=3
        RSB
RECOV_ERR:
        MOVL    #4,R0                                   ;IERR=4
        RSB


;       FORTRAN CALLABLE PROCEDURE FOR WRITING TWO END OF FILE MARKS
;       AND BACKSPACING OVER ONE.
;
;               CALL MARK(UNIT)
;
         ENTRY  MARK,^M<R2,R3,R4>
        MOVAL   ARGLST,R4                       ;POINT TO QIO PARAMETER BLOCK
        JSB     GET_CHAN                        ;GET ASSIGNED CHANNEL NUMBER
        CLRL    QIO$_EFN(R4)                    ;USE EVENT FLAG ZERO
        MOVW    R2,QIO$_CHAN(R4)                ;PUT CHANNEL INTO PARAM. BLOCK
        MOVW    #IO$_WRITEOF,QIO$_FUNC(R4)      ;FUNCTION CODE INTO PARAM. BLK
        MOVL    R3,QIO$_IOSB(R4)                ;IOSB RETURN ADDRESS

        $QIOW_G (R4)                            ;Q EOF AND WAIT
        JSB     ERROR                           ;Q-ING SUCESSFUL?
        JSB     GET_STATUS                      ;OPERATION SUCESSFUL?
        $QIOW_G (R4)                            ;Q SECOND EOF AND WAIT
        JSB     ERROR                           ;Q-ING SUCESSFUL?
        JSB     GET_STATUS                      ;OPERATION SUCESSFUL?
        MOVW    #IO$_SKIPFILE,QIO$_FUNC(R4)     ;LOAD FILE SKIP FUNC. PARAM.
        MOVL    #-1,QIO$_P1(R4)                 ;BACKSPACE 1 FILE
        $QIOW_G (R4)                            ;Q THE BACKSPACE
        JSB     ERROR                           ;Q-ING SUCESSFUL?
        JSB     GET_STATUS                      ;OPERATION SUCESSFUL?
        RET


;       FORTRAN CALLABLE PROCEDURE TO SPACE TAPE FORWARDS AND BACKWARDS
;       A SPECIFIED NUMBER OF FILES.
;
;               CALL SKIP(UNIT,NFILES)
;
;       UNIT = FORTRAN LOGICAL UNIT NUMBER BETWEEN 0 AND 15 INCLUSIVE.
;       NFILES = -,0,+ INTEGER DETERMING THE NUMBER AND DIRECTION OF
;                FILES TO BE SKIPPED.
;
         ENTRY  SKIP,^M<R2,R3,R4>
        MOVAL   ARGLST,R4                       ;POINT TO QIO PARAMETER BLOCK
        JSB     GET_CHAN                        ;GET ASSIGNED CHANNEL
        CLRL    QIO$_EFN(R4)                    ;USE EVENT FLAG ZERO
        MOVW    R2,QIO$_CHAN(R4)                ;PUT CHANNEL INTO PARAM. BLK
        MOVL    R3,QIO$_IOSB(R4)                ;IOSB RETURN ADDRESS PARAM.
        MOVW    #IO$_SKIPFILE,QIO$_FUNC(R4)     ;FILE SKIP FUNC. CODE
        MOVL    @8(AP),QIO$_P1(R4)              ;PUT NFILES INTO PARAM. BLK
        $QIOW_G (R4)                            ;Q THE FILE SKIP
        JSB     ERROR                           ;Q-ING SUCESSFUL?
        JSB     GET_STATUS                      ;OPERATION SUCESSFUL?
        RET


;       FORTRAN CALLABLE PROCEDURE TO SKIP FORWARDS AND BACKWARDS A
;       SPECIFIED NUMBER OF PHYSICAL RECORDS ON A MAGNETIC TAPE.
;       SKIPPING IS TERMINATED IF EOF OR BOT IS ENCOUNTERED.
;
;               CALL SPACEN(UNIT,NRECS)
;
;       UNIT = FORTRAN LOGICAL UNIT NUMBER BETWEEN 0 AND 15 INCLUSIVE.
;       NRECS = -,0,+ INTEGER NUMBER OF RECORDS AND DIRECTION TO BE SKIPPED.
;
         .ENTRY SPACEN,^M<R2,R3,R4>
        MOVAL   ARGLST,R4                       ;POINT TO QIO PARAM. BLK
        JSB     GET_CHAN                        ;GET ASSIGNED CHANNEL
        CLRL    QIO$_EFN(R4)                    ;USE EVENT FLAG ZERO
        MOVW    R2,QIO$_CHAN(R4)                ;PUT CHANNEL INTO PARAM. BLK
        MOVL    R3,QIO$_IOSB(R4)                ;IOSB RETURN ADDR. IN PARAM.
        MOVW    #IO$_SKIPRECORD,QIO$_FUNC(R4)   ;LOAD RECORD SKIP FUNC. CODE
        MOVL    @8(AP),QIO$_P1(R4)              ;PUT NRECS INTP PARAM. BLK
        $QIOW_G (R4)                            ;Q THE RECORD SKIPPING
        JSB     ERROR                           ;Q-ING SUCESSFUL?
        JSB     GET_STATUS                      ;OPERATION SUCESSFUL?
        RET
;       FORTRAN CALLABLE PROCEDURE TO REWIND AND UNLOAD TAPE WITHOUT
;       A DISMOUNT BEING DONE, SO MULTIREEL VOLUMES CAN BE HANDLED.
;
;               CALL NEWTAPE(UNIT)
;
;       UNIT = FORTRAN LOGICAL UNIT NUMBER BETWEEN 0 AND 15 INCLUSIVE.
;
;
LIS:    $SCHDWK DAYTIM=TIMR
TIMR:    .LONG  -10*1000*1000*10,-1
        $OPCDEF
LOAD:   $SNDOPR MSGBUF=LOADT
LOADT:   .LONG  ZZ-OPMES
         .LONG  OPMES
OPMES:   .LONG  ^X100*OPC$M_NM_CENTRL!OPC$_RQ_RQST
         .LONG  0
         .ASCII "LOAD NEW TAPE ON UNLOADING DRIVE"


ZZ:
         .ENTRY NEWTAPE,^M<R2,R3,R4>
        MOVAL   ARGLST,R4                       ;POINT TO QIO PARAM. BLK
        JSB     GET_CHAN                        ;GET ASSIGNED CHANNEL
        CLRL    QIO$_EFN(R4)                    ;USE EVENT FLAG ZERO
        MOVW    R2,QIO$_CHAN(R4)                ;PUT CHANNEL INTO PARAM. BLK
        MOVL    R3,QIO$_IOSB(R4)        ;IOSB RETURN ADDR. IN PARAM. BLK
        MOVW    #IO$_REWINDOFF,QIO$_FUNC(R4)    ;LOAD RECORD SKIP FUNC. CODE
        $QIOW_G (R4)                            ;Q THE RECORD SKIPPING
        JSB     ERROR                           ;Q-ING SUCESSFUL?
DEBG:   $SNDOPR_G       LOAD            ;NOTIFY COMPUTER OPERATOR
        MOVW    #IO$_SENSEMODE,QIO$_FUNC(R4)    ;LOAD SENSE MODE FUNC. CODE
```

```
QIO     $QIOW_G (R4)                            ;Q THE RECORD SKIPPING
        JSB     ERROR                           ;Q-ING SUCESSFUL?
        BLBS    6(R3),FIN                       ;IF TAPE LOADED EXIT
        $SCHDWK_G       LIS                     ;SCHED. WAIT
        $HIBER_S                                ;HIBERNATE
        BRB     QIO
FIN     RET
         END
```

Ken Bell
8334 Avenida Leon
Cucamonga, California 91730
(714) 989-6461
(Yes, Virginia, there is a Cucamonga.)

Ian Hammond's mention of the "USR collect call" facility in the Jan-81 Mini-tasker sent me scrambling for my RT-11 sources. When I re-emerged I had a good understanding of the facility, and a FORTRAN callable subroutine to (ab)use this feature. The routine does a [NO]PROTECT on the specified file. Sources follow. In breif, the monitor requests .ENTER, .LOOKUP, and .RENAME allow the user to initiate a "collect call" to the USR by the following mechanism:

```
        MOV     #MYSUB,AREA+8.
        .RENAME #AREA,#CHAN,#DBLK!1
```

At some point in its work, the USR will do a JSR PC,@#MYSUB with R1 pointing to the DATE word (word 7) in the directory entry it is working with. I have not explored what registers are available for use and so assume that none are! The user subroutine must return with an RTS PC.

!!!!CAVEAT!!!! The subroutine MYSUB MUST NOT be swapped over by the USR (for reasons that should be obvious)!!!!

This facility can also be (ab)used to create and maintain other data attached to a directory entry. (Extra words can be allocated in the directory entry at INIT time by the use of the /Z:n switch in DUP.) For example:

1) User Password

2) Date last accessed

Etc.

```
        .TITLE  PROTECT/UNPROTECT ROUTINE
        .PSECT  SYS$O,RW,I,LCL,REL,CON
        .MCALL  .RENAME
;
;       PROTECT AND UNPROTECT ARE FORTRAN CALLABLE SUBROUTINES TO SET
;       OR CLEAR THE PROTECT BIT IN THE STATUS WORD OF THE FILE DIRECTORY
;       ENTRY. THE FORM OF THE CALL IS:
;
```

```
,               /  PROTECT  \
;       ISTAT  =  <              > (CHAN,DBLK)
;                \ UNPROTECT /
;
;       WHERE:  CHAN    = CHANNEL NUMBER TO USE
;               DBLK    = 4 WORD RAD50 FILE DESCRIPTOR BLOCK
;
;               ISTAT   = 0 - NORMAL RETURN
;                       = 1 - SPECIFIED CHANNEL ALREADY OPEN
;                       = 2 - SPECIFIED FILE NOT FOUND
;                       = 3 - ILLEGAL OPERATION
;
PROTECT::
        MOV     #1,PFLG                 ; SET FLAG TO PROTECT
        BR      COMMON

UNPROTECT::
        CLR     PFLG                    ; SET FLAG TO UNPROTECT

COMMON:
        MOV     4(R5),R0                ; GET FILE SPEC
        MOV     #DBLK,R1                ; POINT TO DBLK USED BY RENAME
        MOV     #DBLK+10,R2             ; WE NEED TWO COPIES
        MOV     #4,R3                   ; MOVE 4 WORDS
1$:     MOV     (R0),(R1)+
        MOV     (R0)+,(R2)+
        DEC     R3
        BNE     1$
        MOV     #MYSUB,AREA+10  ; GIVE ADDRESS OF COLLECT CALL ROUTINE
        .RENAME #AREA,@2(R5),#DBLK!1 ; RENAME FILE TO ITSELF AND CALL "MYSUB"
                                     ; BEFORE WE WRITE IT BACK TO DIRECTORY
        BCS     ERROR                   ; ANY MISTAKES
        CLR     R0                      ; NO, SET ISTAT TO NORMAL
        BR      EXIT                    ; AND RETURN

ERROR:
        MOVB    @#52,R0                 ; ERROR, GET STATUS BYTE
        BIC     #^C177400,R0
        INC     R0                      ; AND ADD ONE TO GIVE ERROR CODE

EXIT:
        RETURN                          ; RETURN TO SENDER

   ;       THIS ROUTINE IS CALLED BY THE USR WITH R1 POINTING TO THE "DATE"
   ;       WORD IN THE DIRECTORY ENTRY.
   ;
   PBIT    = 100000         ; FILE PROTECT BIT IN DIRECTORY ENTRY STATUS WORD

   MYSUB:
           TST     PFLG                 ; PROT/UNPROT
           BEQ     1$
           BIS     #PBIT,-14(R1)        ; SET FILE PROTECT BIT
           BR      2$
   1$:     BIC     #PBIT,-14(R1)        ; CLEAR FILE PROTECT BIT
   2$:     RETURN                       ; RETURN TO USR

   AREA:   .BLKW   5                    ; PARAMETER BLOCK FOR RENAME
   DBLK:   .BLKW   4                    ; OLD FILE NAME
           .BLKW   4                    ; NEW FILE NAME
   PFLG:   .WORD   0                    ; PROTECT/UNPROTECT FLAG
```

UPCOMING SYMPOSIUM INFORMATION

I will be hosting a new RT-11 session for the Spring Symposia in May and I will need the help of some of the attendees. The session is called the RT-11 SIG Swap Tape Treasure Hunt.

For a long time the RT-11 SIG swap tape has been one of the best features of DECUS conventions. The number of submissions has continued to grow and it has become increasingly hard to keep track of all the programs and updates and revisions. So, in this session we will be looking for the lost 'treasures' buried on the past RT-11 SIG swap tapes.

This session will be run by the users. Each treasure hunter will have two minutes to tell other users the program that he discovered buried in the swap tapes. We are interested in such things as which swap tape the program was buried, why it is a treasure, how you modified it, where the modifications may be found and how you use it. If time permits, we will allow users to identify programs that should remain buried, that is programs that do not work or ones that crash systems. The idea is to keep it short yet give other users a flavor for the really good programs on the Swap tapes.

This session will be scheduled for the third day of the conference and users who are interested in presenting a program to the group are asked to call me at 202-227-1592 before the Symposia or sign-up in the RT-11 campground in Miami. In this way we will not have duplicate presentations. I will be happy to answer questions before the Symposia and discuss your particular program. If you have found a buried treasure and have modified it, please bring it with you so that we can place it on the Spring Swap tape.

Because of the short time available (1 hour), we will limit the discussion to programs buried on the RT-11 swap tapes only. We will make a list of all the buried treasures mentioned at this session so that we can compile a 'Best of' tape for the next DECUS.

Sincerely,

Ned W. Rhodes

Ned W. Rhodes

WISH LIST

DECUS, SAN DIEGO 1980
BY MARILYN L. RUNYON

1. How about a utility to read RSX files? I'd like an RT mini- reference manual, on the order of the RSX material in the bookstore. Tabbed sections would be an advantage over the current reference card. I'd like to see FDT come with the Fortran compiler rather than with the extensions.

2. The directory file protection feature is nice. How about putting a read only bit into the directory entry, also.

Modify DIR.SAV to give largest free block in addition to the total free block count.

The software Support Manual states that the job channel words use is reserved for future use by Digital for permanent files. How about using it to store the last accessed (.LOOKUP) date?

Some way to get up-arrow C to cause entry to a completion routine to do realtime processing. Also a way to set the terminal support to totally ignore &C and &O, etc.

Have the program function keys do something meaningful with KMON such as (a) user defined text insertion into input buffer, (b) insert/delete mode to modify (edit) then execute previous command, (c) a way to repeat prior command.

A way to reboot past a linefeed to U or multiple reboots will get you back to the beginning of the current line but there is no way to go back prior to that when using type ahead.

A way to (re)queue a caracter so that it will be the next character retrieved by a .TTYIN request.

A way to save and restore default system settings and switches for use with Keyboard Interactive Commands, so that by specirying a default of '/TERMINAL' for 'DUMP', for example, one would not always have to add this to the command line.

3. How about a KED and/or TECO that uses virtual memory and/or is a virtual program, a maintenance release.

4. DIR: (a) allow more than 1 device name to be in the filespec list; (b) change sorts to not remove usused - would allow DIF/FULL/BLO/SORT:POS to not lose info.

DA: When given by itself, output day of the week.

PIP: Additional date options like DIR so could copy since some date, etc.

Devices with variable length volumes should be known by a DCW bit rather than a patch to DUP. Could also do by defining a zero length, directory-structured device as variable length (.SPFUN 373 to get size)?

EOF call to non-file-structured devices: >CLOSE should call the handler with an DOF function.' Could be used by LP handlers to output a trailing FF.

For BASIC 11:

Provide 'ON ERROR' statement in SU Basic. Provide 'ON INTERRUPT' statement in MU and SU Basic. Also any other special statements required for interrupt handling.

For RT-11 system:

Make RT-11 system utilities really hardware independent. DUP and RESORC both have hardware dependent tables (RESORC has been fixed in V4.0). The table of multi-density devices in DUP is not really necessary. The device handler header (status word) could contain a multi-density flag bit. Also, disk formatting should be performed through handler special function calls.

6. LP handler should never do a form feed on file open, and always on file close (like UNIX LP handler). MT operations should complete before rewinding the tape, i.e., DIR/PRINT MT: should not wait until the tape is fully rewound before finishing the printing.

7. Printer spooler which is application program transparent. An extension of queue. Wild card capability in queue. DCL command editing on error. A 'REDO' command. User defined commands.

FORTRAN RT-11:

Integer *4 support. Character type - ability to define a long string in a data statement.

8. Want PROTECT command.

9. Subdirectories (please!) - at least one level; preferably tree-structured hierarchy of arbitrary depth.

10. KED patches to permanently change defaults, eg., SET QUIET. Also, after SET QUIET print a small error message in one corner of screen, rather than light to dark.

11. There ought to be some way for one job under XM to share a region with a second job.

12. Need larger directory!

13. User addable commands. Maybe at the expense of COMPILE or EXECUTE. - Remove those and insert our own.

Utility to show wtatus of all SETs on device handlers.

A real TT handler for multiunits instead of adding a KB or LS handler for each terminal. I need to write via FORTRAN to terminals and want to use its formatter instead of doing it and using multiterminal calls.

Inline comments in indirect command files.

14. Possibility to share an area of memory among two or more jobs, in low or high memory. This would allos sending data from one job to another without going through the message queue facility (similar to Global Common in RSX).

More options on the copy command: (a) change the date of the file to insert date, (2) copy files for a given date, or better for a range of dates, (3) option allowing confirm copies if target file of same name already exists.

Why copies to tape do not preserve file date?

File transfer driver allowing the transfer of source files from onr RT system to another. An XON/XOFF protocol would be used. This would avoid having to use DECNET-RT for very simple short data transfers.

With the cost of memory going down and the LSI-11/23, the XM monitor could grow and become more a true multi-tasking system.

15. Please consider having all DEC distribution floppies for RT-11 and layered products write-protect-notched for those of us whose drives are clever enough to sense the notches.

16. A new EMT to get as many characters from the input buffer as possible, rather than a single character, i.e., equivalent to .TTINR but return multiple characters. Would reduce EMT overhead.

17. RT-11 on the VT 103 with TU 58:

Environmental conditions (field data collection) dictate use of TU 58, but overlay structure of operating system results in a lot of tape grinding. We use RT-11 SJ, with PIP, DUP, DIR and KED with 64 K RPM. Possibly a special version of RT would be made available (system option?) that could reduce the amount of overlaying, but obviously using more memory, for typical file handling operations (directory, copy, print). Optimizing the SY tape format helps much, and we can live with it, but it would be nice if.....

ODT or a video terminal:

An efficient machine code debugger would be a very worthwhile addition for those jobs that still need to be done in MACRO. Suggested options: On breakpoint: display (1) register contents (effect by a XXX register) and ASCII equivalent, (2) currently set breakpoints (effect), (3) several user selected XXX of memory (wordXXX or effect, byte, or ASCII). In tape mode in addition to above, display code-mnemonics for last 4-5 instructions, and upcoming 3-4 instructions, so program position can be consistently observed in operation.

18. "Undefined global" output of LINK - would be nice to know what module (out of +-=20 or ?) made the undefined call. Would save much searching of listings of trying to remember!.

19. Banners including date, time and file name on print withous sppl.

Banners including date time and file name on top of screen.

Provision to display message on TT from indirect command files.

Provision to set time and date from indirect command file using DATEC or similar command.

Show sets, via resource program, for instance, "C" language under RT-11, PASCAL language for RT-11.

Suggest parallel development of a cleaned up RT-11.

20. Feature to allow parameter substitution in indirect command files.

PIP modification to allow copying multiple files floppy-to-floppy with system device (floppy) removed.

Distinct file type for overlayed files such as .SVO automatically supplied by linker and recognized by .RUN. Needed by users of floppy systems to distinguish what files must be on-line, or even on the system device (.SYO)!

I would like to implement a system consisting of a host with disk, printer and console supporting several remote LSI-11s such as VT 103. An operator at a remote terminal would seem to be running a RT-11 SJ monitor with some restrictions. After bringing up the system, an operator at the remote could type .EDIT FILESPEC which would be passed to the host to download edit and then the first page of the file in response to *N$$ .R PROG would download PROG. >PRINT FILE would pass command to transfer file to printer or spooler, etc. Each remote user would be running programs in his own dedicated CPU and memory vs. timesharing. Suggest implementation using DL11 link at 19K band +, system jobs on the host and same features of MRRT. 21. 1) Implement SHOW/OUTPUT:DEV:FILNAM.EXT.

2) IMPLEMENT: HELP/OUTPUT:DEV:FILNAM.EXT.

3) Implement: external MACROs for KED and K52 as in TECO "EI" and TECO.INI.

4) Put logical carriage control in LP and LS drivers (or in queue).

5) Add DELETE status to SHOW QUE output.

6) Provide indirect file processor as in RSX-11 or Autopatch.

7) Add (optional) DOS mag tape support for MT, MM, etc, FILEX.

8) Add explaination of SET LP CSR, etc., with description of relationship of LP, CS and the various possible hardware interfaces..p 9) LIST command gives numbered listing equivalent to R SLP, OUTfile=in file.

10) Allow HELP in system job so one can get help during editing, etc.

11) Include subdevice support techniques in Software Support Manual.

12) KED overstrike insertion mode as in FRED to overlay information on QAR forms, etc.

13) Keep up the good dialog between users, especially test sites, and the developers and documenters.

14) SYSGEN to output BATCH command files instead of indirect command files (I want a log!).

15) Stand alone purchase of AUTOPATCH for BASIC-11. We believe DEC owes users this.

16) User command processing if KMON can't find command (sysgen option?).

17) Narrow banner page from QUEUE.

18) Better MT support (asynchronous).

19) 'Watchdog' timer support in RMON. We need the hardware also.

20) How about an UNDERGROUND so that compute-bound programs may be fun concurrently with BG programs. Alternatively, why not allow system jobs to run at a level lower than the BG.

21) /DATE option in LINK to put system date in block 0 of save image, maybe time, also.

22) Keep doing all this good stuff!

23) FORTRAN should check for illegal stack pointer on errors 61 and 63 to help 11V03 users who overvlow stack.

RT-11 MARKETPLACE

The following guidelines apply to all RT-11 Marketplace submissions:
1. typewritten
2. maximum of 1 paragraph with no white space
3. do not include company letterheads
4. do not include the price
5. your Decus membership number must accompany your submission

SPSS-11 is a unique software applications program for tabulation, statistical analysis and general purpose data management developed specifically for DEC PDP-11 computers. Because it was designed with the user in mind, SPSS-11 combines statistical sophistication with an easy-to-use command structure. The SPSS Report Writer, included with the next release, links the analysis process with data presentation in providing the ability to generate custom formatted reports. Supporting documentation, published by McGraw-Hill, is tailored to each operating system. SPSS-11's versatility makes it compatible with any PDP environment -- 11/03 to 11/70 using RSTS, RT-11, RSX-11M or IAS. For more information contact Susan Phelan, SPSS Inc., 444 N. Michigan, Suite 3300, Chicago, IL 60611, 312/329-2400.

The following is information for inclusion in the RT-11 MINITASKER, when possible:

The RTFILE relational data base management system for DEC PDP-11 and LSI-11 computers running under the RT-11 operating system is now being marketed and supported by INTERPROJECT, Inc. RTFILE is currently installed in approximately thirty diverse operating environments ranging from national research institutions, through computer peripheral manufacturers, to local commercial firms.

Two major enhancements to RTFILE are now included in version 2.4:

1. The COMMAND FILE INTERFACE utility allows authorized users to execute any RT-11 keyboard monitor commands without exiting RTFILE. Furthermore, the utility enables the System Manager, or DBA, to assign an access level to each specific RT-11 command; only users who meet or exceed the access level for a particular command are allowed to execute it. This means that, for the first time, RT-11 will not simply "obey" an INIT or SQUEEZE or DELETE command--or any other--without first ascertaining the validity of the request. Control automatically returns to RTFILE after execution of all RT-11 command sequences. Privileged access status is required to exit RTFILE to native RT-11.

Additional information is available from:

Robert C. Natale
INTERPROJECT
Computer Software Management
Post Office Box Thirteen
Brentwood, Maryland 20722
(301) 864-3257

SYMPOSIUM TAPE DISTRIBUTION

We are still looking for a SIG tape copy facility in Miami for the upcoming symposium. If you can help us, please contact:
Nick Bourgeois
505 844-8088

RT-11 Tape Copy Centers

The following shops have offered to copy RT-11 SIG DECUS/US Symposia tapes including the Fall 80 RT-11 tape. Some are willing to copy to media other than mag tape. However, before requesting copies on any media other than mag tape you should contact the copy center for confirmation.

The rules are still quite simple. A mag tape or other media in a reusable mailer along with return label and postage (not cash or check) is required. Include a note stating which tape you want. Any media arriving without the reusable mailer, return label and postage will be treated as a gift to the copy center.

Not all centers have all tapes. Most will have the combined tape and the latest Fall 80 tape. The RT-11 SIG tapes are listed below:

Spring 78	Chicago
Fall 78	San Francisco
Spring 79	New Orleans
Fall 79	San Diego
Spring 80	Chicago
Combination of the above	
Fall 80	San Diego

CENTRAL U.S.

Gary Siftar
Cisco, Inc.
4135 S. 100th E. Ave.
Tulsa, OK 74145
(918) 665-2110
Media: RX01/2, MT

EASTERN U.S.

Ned W. Rhodes
Naval Ship R & D Center
Bethesda, MD 20084
(202) 227-1592
Media: RK05, RL01/2, RX01/2, MT

MIDWESTERN U.S.

Joseph Lachman
Lachman Associates, Inc.
825 North Cass
Westmont, IL 60559
(312) 986-8840
Media: RK05, RL01/2, RX01/2, MT

NORTHEASTERN U.S.

Alfred H. Scholldorf
Phisics Dept.
SUSB
Stony Brook, NY 11794
(516) 246-7110
Media: RL01, RX02, MT

NORTHWESTERN U.S.

Rand Dow
Oregon State University
School of Oceanography
Corvallis, OR 97331
(503) 754-3504
Media: MT

SOUTHEASTERN U.S.

Mary Williams
Science Applications, Inc.
2109 W. Clinton Ave.
Suite 800
Huntsville, AL 35805
(205) 533-5900
Media: MT

SOUTHWESTERN U.S.

Ray Kaplan
Electrical Engineering
Building 20
University of Arizona
Tucson, AZ 85721
(602) 626-4462
Media: RK05, RX01

Carl Lowenstein
University of California
Marine Physical Laboratory
Scripps Institution of Oceanography
San Diego, CA 92152
(714) 294-3678
Media: MT

Nick Bourgeois / 1738
Sandia National Laboratories
P. O. Box 5800
Albuquerque, NM 87185
(505) 844-8088
Media: MT

CANADA

Greg L. Adams
Dept. of National Defence
National Defence Headquarters
Attn: DACS 2-2-4
Ottowa, Canada K1A0K4
(613) 993-9624
Media: MT

GREAT BRITAIN

J. R. Lishman
University of Aberdeen
Department of Psychology
King's College
Aberdeen
AB9 2UB
Scotland
0224-40241
Media: RK05, RX01, MT

Since recently in Holland a real RT11-SIG has been established, I want to inform you that I have handed over the tape-copy-operations for Holland to the chairman of the RT11-SIG.
His name is mr. Ronald Beetz, voorzitter RT11-SIG
Akzo-Pharma
Postbus 20
Oss

DEPARTMENT OF RADIATION THERAPY

Computer Facility
Robert F. Curley, Director
Robert E. Wallace

Hospital of the University of Pennsylvania
3400 Spruce Street
Philadelphia, Pennsylvania 19104
(215) 662-3083

Ken Demers
MS-48
United Technologies Research Center
East Hartford, Connecticut 06108

Dear Mr. Demers:

I would appreciate any aid that the RT SIG members could give me to solve a problem which I pose for RT VT11 scroll support. In short, I would like to be able to GTON/GTOFF the VT11 scroll function from a user MACRO or FORTARN program.

Sincerely,
Robert E. Wallace

Robert E. Wallace

To: RT-11 Decus Newsletter Editor
RSX-11M Decus Newsletter Editor

Date: March 4, 1981
From: Paul W. Shahood
Dept: SDC Mfg. Engineering
DTN: 234-4204
Loc: NR2-2/E34

Subject: 6502 Cross Assembler for PDP-11

I have been attempting to locate a 6502 cross assembler for the PDP-11 for either RT-11 or RSX-11M for quite some time now. Rumor has it that there is 'more than one version out there' which is probably a safe assumption.

If you know of anyone who has such an animal would you PLEASE either give them my name and extension or send theirs to me and I will contact them. If you do not have such information, would you please solicit this information via The Newsletter or any other mechanism available to us.

Thank you for your cooperation and help in this matter.

Paul W. Shahood
DECUS # 125798

DECUS

DIGITAL EQUIPMENT COMPUTER USERS SOCIETY
ONE IRON WAY, MR2-3/E55
MARLBORO, MASSACHUSETTS 01752

MOVING OR REPLACING A DELEGATE?

Please notify us immediately to guarantee continuing receipt of DECUS literature. Allow up to six weeks for change to take effect.

( ) Change of Address
( ) Delegate Replacement

DECUS Membership No.: ____________

Name: ____________

Company: ____________

Address: ____________

____

State/Country: ____________

Zip/Postal Code: ____________

Mail to: DECUS - ATT: Membership
129 Parker Street
Maynard, Massachusetts 01754 USA

Affix mailing label here. If label is not available, print old address here. Include name of installation, company, university, etc.