THE mini-tasker

DECUS
RT-11 SIG NEWSLETTER

March 1983 Vol. 9, No. 1

Contributions to the newsletter should be sent to:

Ken Demers
MS-44
United Technologies Research Center
East Hartford, Conn. 06108
(203) 727-7527 or 7240

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) 467-4141

TABLE OF CONTENTS

USER SURVEY

FORTRAN-77 and PASCAL for RT-11 Survey

Ron Trellue
Division 7523
Sandia National Laboratories
Albuquerque, New Mexico 87185
(505) 844-0955

Number of surveys received : 218

1. Can a floating point unit be mandatory on an RT system running a FORTRAN-77 product? (Can EIS be mandatory? Is FIS required?)

FPU	Yes	111
	No	50
	Optional	15
	Total	176
EIS	Yes	144
	No	26
	Total	170
FIS	Yes	58
	No	38
	Total	96
Who Cares ?		4

2. Is it acceptable that compilations using this FORTRAN-77 product require the XM monitor and 128K bytes of memory? In general, would required minimal configurations (e.g. amount of memory and disk space required) still make this a useful product for you?

XM Monitor	Yes	61
	No	121
	Total	182
128 KB Memory	Yes	88
	No	93
	Total	181

3. Is it acceptable that there be no option to generate threaded code with a FORTRAN-77 product on RT? (FORTRAN IV/RT does generate threaded code. In addition to the loss of ability to reduce application task sizes through generation of threaded code, lack of a threaded code capability would mean there would be less checking for overflow than is available with FORTRAN IV/RT.)

OK	102
Not OK	77
Don't care	12
Total	191

4. Do you need a FORTRAN-77 product to conform to the ANSI FORTRAN Standard at the subset level, or do you require just some of the ANSI features?

Yes	104
No	77
Total	181

5. What features of F77/RSX would you find most useful? (CHARACTER variables, IF THEN ELSE, INTEGER * 4 arithmetic, in-line floating point code...) Would this product need to implement only some of the language features in F77/RSX, or would you require all of those features?

All	58
IF/ELSE	93
I*4	91
CHAR	54
INLINE FPU	44

6. Can the compile speed of a FORTRAN-77 product on RT be slower than that of FORTRAN IV?

Yes	175
No	6
Don't Care	4
Total	185

7. What size constraints (on user programs, memory, compiler task image) would you consider acceptable? In particular, in comparison to the FORTRAN IV product, what would an acceptable percentage growth in compiler task size be (10%, 20%, ...)? How about user program growth (10%, 20%, ...)?

Compiler	0%	6
	10%	20
	20%	50
	30%	14
	Don't care	87
	Total	177

Programs	0%	23
	10%	78
	20%	43
	30%	8
	Don't know	23
	Total	175

8. Quote a price range that you believe is reasonable for a FORTRAN-77 product on RT.

?	29
Same as F4	10
10 to 20 % more	5
50 % more	4
$200-300	9
$500-1000	60
$1000-2000	42
$2000-5000	25
$5000 +	1
Do we have a choice	1
Total	186

9. What is an acceptable window of availability for a FORTRAN-77 product on RT (within the next year, next two years, etc.)?

ASAP	42
1 year	80
2 years	48
Don't care	13
Total	183

10. Do you need to be able to run applications 'stand alone' as is now possible with FORTRAN IV/RT and SIMRT?

Yes	39
No	140
Total	179

11. Would you be willing to accept a phase-out of support of the FORTRAN IV product within a year of the release of a FORTRAN-77 product (after phase-out, the FORTRAN IV product would be available in a stable condition on a totally unsupported basis)?

Yes	152
No	22
2 yrs	5
Don't care	4
What support ?	2
Total	185

12. Are you especially interested in a Pascal on RT? Before or instead of a FORTRAN-77? After a FORTRAN-77 or never? Simultaneously?

Before	30	
Simult.	26	119
After	63	
Never	61	
'C'	8	
ADA	4	
Total	192	

USER INPUT

I was particularly interested in Mr. Rozenberg's DATTIM program as I have been using a somewhat different routine for the same purpose. For diskette-based systems and others that require frequent rebooting, I think my program may have some advantages over Mr. Rozenberg's. With the thought that you might think the readers of Mini-tasker would be interested, I am enclosing a listing that you may use as you see fit.

Alfred K. Blackadar
805 West Foster Avenue
State College, PA 16801


```
        .TITLE DATCMD -- STARTUP DATE-TIME COMMAND

;     by Alfred K. Blackadar, 805 W. Foster Ave., State College, PA 16801
;
;
;     This routine is designed to be run by the 'STARTS.COM' indirect file
;as part of the BOOT SY: procedure, and it must reside on the system disk, as
;coded. It first calls up the date when the last boot was made and asks for a
;change. If a carriage return is pressed at this point the displayed date is
;recorded in the file DATCMD.SAV and is passed to the CSI as the system date.
;If a change of any part of the date is made, the new date is displayed and
;and a request for further change is made, etc. Following any plain carriage
;return, a simulated monitor TIME command is displayed, and should be answered
;in the usual manner either to display the system time or to reset it.
;
        .MCALL  .LOOKUP .TTYIN  .WRITW  .PRINT  .EXIT   .CLOSE

        .ENABL  LC
;
        JSW = 44                ;JOB STATUS WORD
        ERRWD = 52
        CR = 15
        LF = 12
                                ;
DATCMD: BIS     #40000,@#JSW    ;ENABLE LOW CASE INPUT
        .LOOKUP #AREA,#2,#FILE
INSPCT: CLR     R5              ;RESET FLAG
        .PRINT  #MSGB           ;INSERT A LINE FEED
        .PRINT  #DAT            ;DISPLAY THE OLD DATE
        .PRINT  #MSGA           ;ASK FOR CHANGE
        MOV     #40,R2          ;PAD IN CASE SINGLE DIGIT DATE
        MOV     #DAT+7,R1       ; -> FIRST ALPHA IN DATE
                                ;
GETCHR: .TTYIN                  ;GET A CHARACTER
        CMPB    R0,#LF          ;IGNORE LINE FEEDS
        BEQ     GETCHR
        CMPB    R0,#CR
        BNE     CHANGE
        TST     R5              ;IS IT A LEADING CR?
        BEQ     SAVE            ;YES. ACCEPT DATE
        BR      INSPCT          ;NO. INSPECT NEW DATE
CHANGE: INC     R5                      ;FLAG CHANGE
        CMPB    R0,#'0
        BMI     ALPHA           ;TREAT DASH AS A LETTER
        CMPB    R0,#'A          ;INSERT LETTERS DIFFERENTLY
        BPL     ALPHA
        CMPB    R2,#75          ;PROVIDE FOR YEAR-DIGITS
        BEQ     ALPHA
        MOVB    R2,-2(R1)       ;REPLACE LEADING DIGIT (SPACE)
        MOVB    R0,-1(R1)       ;REPLACE SECOND DIGIT
        MOV     R0,R2           ;PREPARE FOR ANOTHER DIGIT
        BR      GETCHR          ;GET ANOTHER CHARACTER
ALPHA:  MOVB    R0,(R1)+        ;INSERT NEW LETTER
        MOV     #75,R2          ;TREAT FURTHER DIGITS AS LETTERS
        BR      GETCHR          ;GET ANOTHER CHARACTER
```

```
SAVE:   .WRITW  #AREA,#2,#1000,#400,#1
        .CLOSE  #2              ;RECORD NEW DATE IN 'DATCMD.SAV'
        .PRINT  #MSGC           ;PROMPT FOR TIME
        MOV     #TIM+5,R1
1$:     .TTYIN  (R1)+
        CMPB    R0,#LF
        BNE     2$
        TSTB    -(R1)
2$:     CMPB    R0,#CR
        BNE     1$
        CLRB    -1(R1)          ;REPLACE #CR BY A NUL
        MOV     #DAT,R1         ; -> DATE COMMAND STR IN BUFFER
        MOV     #510,R2         ; -> KMON COMMAND AREA
        MOV     #30.,(R2)+      ;BYTE-LENGTH OF COMMAND STRING
3$:     MOVB    (R1)+,(R2)+     ;TRANSFER DATE COMMAND STRING
        BNE     3$
4$:     MOVB    (R1)+,(R2)+
        BNE     4$
        CLR     R0
        BIS     #4000,@#JSW
        .EXIT                   ;EXECUTE KMON COMMAND STRINGS
DAT:    .ASCIZ  /DATE 30-Nov-80/        ;INITIAL DATE COMMAND STR
TIM:    .ASCIZ  /TIME /
        .BLKB   11              ;TIME INPUT BUFFER
        .EVEN
FILE:   .RAD50  /DK DATCMDSAV/
AREA:   .BLKW   10
MSGC:   .ASCII  /.TIME /<200>
MSGA:   .ASCII  /CHANGE? /
MSGB:   .ASCII  <15><12>
        .END    DATCMD
```

As an alternative to the one published in the October Minitasker, I would like to submit for a future issue the enclosed program. It overcomes the problems of other date programs by:-

a) forcing a date to be entered under all circumstances

b) ensures that a valid date and time is entered by checking the input before exiting to RT11.

This means that our backup programs will operate correctly and overcome simple errors from inexperienced users.

Computer Centre
University of Canterbury Christchurch 1 New Zealand

Yours sincerely

R.M. Harrington
DEVELOPMENT ENGINEER.


```
.TITLE  DATE
.IDENT  /V01.06/
;
; PROGRAM TO GET DATE FROM KEYBOARD
;
; LAST DATE IS KEPT IN DISK FILE DEFINED AT "DFILE"
; AND ALLOWS THIS AS DEFAULT. DATE AND TIME ARE CHECKED
; BEFORE PROGRAM EXITS VIA A CHAIN COMMAND
;
; ALSO INSERTS INTO CHAIN AREA A "@START.COM" TO
; COMPLETE SYSTEM INITIALIZATION.
;
; NOTE: WHILE RUNNING CNTRL/C'S ARE DISABLED
;
.MCALL  .DATE,.EXIT,.PRINT,.TTYIN,.SCCA
.MCALL  .LOOKUP,.ENTER,.READW,.WRITW,.CLOSE

$JSW    =44
J.CHNI  =4000

START:  .SCCA   #AREA,#CCA      ;STOP CTRL/C 'S
        CLR     CCA
;
; PUT COMMAND INTO COMMUNICATION AREA
;
; A COMMAND IS FOLLOWED BY A NULL
; NO CR OR LF CHARACTERS ARE ALLOWED
;
        MOV     #512,R1         ;POINT TO THE COMMAND AREA
        MOV     #CMDSTR,R0
4$:     MOVB    (R0)+,(R1)+     ;MOVE COMMAND "DATE"
        BNE     4$
        MOV     R1,DPOINT       ;SAVE R1
;
; CHECK IF DATE ALREADY SET
;
        .DATE                   ;GET DATE
        TST     R0              ;CK IF ONE
        BEQ     RDFILE
        JMP     TIM
;
; GET DATE FROM FILE
;
RDFILE: .LOOKUP #AREA,#0,#DFILE
        BCC     1$              ;FILE IS THERE
        INC     NFF             ;SET NO FILE FLAG
        BR      DATIN
1$:     .READW  #AREA,#0,#BUFF,#5,#0
        .PRINT  #CURDAT         ;PRINT DATE IN FILE
        BR      DATIN
;
; ERROR RETRY,CHECK FOR CTRL/C 'S
;
RETRY:  TST     CCA
        BEQ     DATIN
        CLR     CCA
        .PRINT  #NOCC
```

```
; GET DATE FROM OPERATOR
;
DATIN:  MOV     DPOINT,R1       ;RESET POINTER
        MOVB    #40,-1(R1)      ;REPLACE NULL BY " "
        .PRINT  #DATQRY         ;PRINT PROMPT
2$:     .TTYIN
        CMPB    #15,R0          ;CK FOR CR
        BEQ     3$
        MOVB    R0,(R1)+        ;PUT IN COMMAND LINE
        BR      2$
3$:     .TTYIN                  ;GET LF
        CLRB    (R1)+           ;PUT IN NULL
        MOV     R1,TEMP         ;SAVE POINTER
;
; CHECK FOR NULL INPUT
;
        CMP     R1,#520
        BNE     CKD             ;NON NULL,CHECK IT
;
; TAKE DATE FROM FILE
;
        TST     NFF             ;WAS THERE A FILE
        BNE     RETRY           ;NO
        MOV     #BUFF,R0
        DEC     R1              ;BACK UP ONE CHAR
5$:     MOVB    (R0)+,(R1)+     ;COPY DATE
        BNE     5$
        MOV     R1,DPOINT       ;SAVE POINTER
        BR      TIM


; CHECK INPUT FOR VALIDITY
;
; FIRST CHECK YEAR IS BETWEEN 80 AND 89
;
CKD:    CMP     R1,#532
        BPL     RETRY           ;TOO MUCH INPUT
        CLR     R3              ;CLEAR ERROR COUNT
        CLR     R4              ;CLEAR CKSUM
        DEC     R1              ;MOVE PAST NULL
        MOV     #34460,R2       ;RANGE 0-9
        JSR     PC,CHECKR
        MOV     #34070,R2       ;RANGE 8-9
        JSR     PC,CHECKR
        TST     R3              ;CK ERROR COUNT
        BNE     RETRY
        MOV     #26455,R2       ;HYPHEN ONLY
        JSR     PC,CHECKR
;
; NOW CHECK MONTH
;
        MOV     #54501,R2       ;RANGE A-Y
        CLR     R4
        JSR     PC,CHECKR
        JSR     PC,CHECKR
        JSR     PC,CHECKR
        MOV     #MTAB,R2        ;POINT TO TABLE
        JSR     PC,CKSUM
        TST     R2
        BNE     RETRY
```



```
; NOW CHECK DAY
;
        MOV     #26455,R2       ;HYPHEN ONLY
        JSR     PC,CHECKR
        CLR     R4              ;NEW CHECKSUM
        CMPB    -2(R1),#40      ;CK POSITION OF BLANK
        BEQ     6$              ;ONLY SINGLE NO
        CMPB    -2(R1),#63      ;CK LEADING DIGIT
        BEQ     7$              ;MUST BE 30,31
        MOV     #34460,R2       ;RANGE 0-9
        JSR     PC,CHECKR
        MOV     #31061,R2       ;RANGE 1-2
        JSR     PC,CHECKR
        BR      8$
;
; ONLY SINGLE NO FOR DAY
;
6$:     MOV     #34461,R2       ;RANGE 1-9
        JSR     PC,CHECKR
        BR      8$
;
; DAY EITHER 30,31
;
7$:     MOV     #30460,R2       ;RANGE 0-1
        JSR     PC,CHECKR
        MOV     #31463,R2       ;ALLOW ONLY 3
        JSR     PC,CHECKR
;
; NOW CHECK ERROR COUNT
;
8$:     TST     R3
        BNE     DATIN
        MOV     #DTAB,R2        ;POINT TO TABLE
        JSR     PC,CKSUM
        TST     R2              ;CK POINTER
        BNE     RETRY
;
; COPY NEW DATE TO FILE
;
        TST     NFF
        BEQ     9$              ;FILE ALREADY EXISTS
        .ENTER  #AREA,#0,#DFILE
9$:     MOV     #BUFF,R0
        MOV     DPOINT,R1
11$:    MOVB    (R1)+,(R0)+
        BNE     11$
        .WRITW  #AREA,#0,#BUFF,#5,#0
        BCC     10$
        .PRINT  #WRTERR
10$:    .CLOSE  #0
        MOV     TEMP,DPOINT     ;UPDATE POINTER


;
; PUT TIME REQUEST IN COMMAND STRING
;
TIM:    MOV     DPOINT,R1       ;RESTORE R1
        MOV     #TIMSTR,R0
5$:     MOVB    (R0)+,(R1)+     ;MOVE COMMAND "TIME"
        BNE     5$
```

```
; CHECK FOR DATE ALREADY SET
;
        .DATE                   ;GET DATE
        TST     R0              ;CK IF ONE
        BNE     EX
;
; GET TIME FROM OPERATOR
;
        MOVB    #40,-1(R1)      ;REPLACE NULL BY " "
        .PRINT  #TIMQRY         ;PRINT PROMPT
6$:     .TTYIN
        CMPB    #15,R0          ;CK FOR CR
        BEQ     7$
        MOVB    R0,(R1)+        ;STORE
        BR      6$
7$:     .TTYIN                  ;GET LF
        CLRB    (R1)+           ;MARK END

  ; CHECK TIME FOR CORRECT FORMAT
  ;
  ; HH:MM:SS OR
  ; HH:MM OR
  ; HH
  ;
  ; MINUTES AND SECONDS MUST BE FROM 00-59
  ; AND HOURS BETWEEN 1 AND 19
  ;
          MOV     R1,-(SP)        ;SAVE R1
          CLR     R3              ;CLEAR ERROR COUNT
          CLR     R4              ;CLEAR CKSUM
          DEC     R1              ;MOVE PAST NULL
          CMPB    -6(R1),#72      ;CK FOR COLON
          BEQ     HMS             ;MUST BE HH:MM:SS
          CMPB    -3(R1),#72
          BEQ     HM              ;MUST BE HH:MM
          CMPB    -2(R1),#40      ;CK FOR SPACE
          BEQ     HH              ;H ONLY
          CMPB    -3(R1),#40
          BEQ     HH              ;HH FORMAT
          BR      TIM             ;ERROR IN FORMAT
  ;
  ; HH:MM:SS FORMAT
  ;
  HMS:    CMP     R1,#547
          BPL     TIM             ;TOO MUCH INPUT
          MOV     #34460,R2       ;RANGE 0-9
          JSR     PC,CHECKR
          MOV     #32460,R2       ;RANGE 0-6
          JSR     PC,CHECKR
          DEC     R1              ;MOVE PAST COLON
  ;
  ; HH:MM FORMAT
  ;
  HM:     CMP     R1,#544
          BPL     TIM             ;TOO MUCH INPUT
          MOV     #34460,R2       ;RANGE 0-9
          JSR     PC,CHECKR
          MOV     #32460,R2       ;RANGE 0-5
          JSR     PC,CHECKR
          DEC     R1              ;MOVE PAST COLON
```



```
  ; HH OR H FORMAT
  ;
  HH:     CMP     R1,#541
          BPL     TIM             ;TOO MUCH INPUT
          MOV     #34460,R2
          JSR     PC,CHECKR       ;RANGE 0-9
          CMPB    -1(R1),#40      ;CK FOR SPACE
          BEQ     CKE
          CMPB    -1(R1),#61      ;CK FOR 1
          BNE     TIM
          DEC     R1              ;ADJUST POINTER
          CMPB    -1(R1),#40
          BNE     TIM
  CKE:    CMPB    -2(R1),#'E      ;CK FOR E OF TIME
          BNE     TIM             ;NOT THERE!

; NOW CHECK FOR ERRORS
;
        TST     R3
        BNE     TIM
        MOV     (SP)+,R1        ;RESTORE R1
;
; FIX UP BYTE COUNT AND EXIT
;
EX:     MOV     #CFILE,R0       ;POINT TO COMMAND FILE
1$:     MOVB    (R0)+,(R1)+     ;MOVE "@ FILENAME"
        BNE     1$
        SUB     #512,R1         ;GET LENGTH
        MOV     R1,@#510        ;SAVE
        BIS     #J.CHNI,@#$JSW  ;SET THE CHAIN INDIRECT BIT
        CLR     R0              ;NEED HARD EXIT
        .EXIT

DPOINT:         .WORD   0
TEMP:   .WORD   0
        .NLIST  BEX
MTAB:   .WORD   12722,11356,13325,13501,14225,13162
        .WORD   12762,12251,13173,13547,14106,11454
        .WORD   0
DTAB:   .WORD   61,62,63,64,65,66
        .WORD   67,70,71,721,722,663
        .WORD   731,732,673,741
        .WORD   742,751,752,761,752,761
        .WORD   762,771,772,1001,1002,1011
        .WORD   1012,1021,1022,1031,1032,0
NOCC:   .ASCIZ  \YOU CANNOT EXIT WITH CTRL/C'S\
WRTERR: .ASCIZ  /WRITE ERROR!/
DATQRY: .ASCII  /ENTER DATE (DD-MMM-YY) ? /<200>
TIMQRY: .ASCII  /ENTER TIME (HH:MM) ? /<200>
CMDSTR: .ASCIZ  /DATE/
TIMSTR: .ASCIZ  /TIME/
CFILE:  .ASCIZ  /@ START/
        .EVEN
CURDAT: .ASCII  /DATE ON DISK IS /      ;MUST BE EVEN NO
BUFF:   .BLKW   5
AREA:   .BLKW   5
DFILE:  .RAD50  /DK /
        .RAD50  /DATE /
        .RAD50  /DAT/
NFF:    .WORD   0
CCA:    .WORD   0
        .LIST   BEX
```

```
; ROUTINES TO CHECK INPUT
;
; FIRST A CHECK ON THE RANGE
;
; THE CHARACTER MOVED TO R0 IS COMPARED
; TO THE CHARACTERS IN R2
; IF OUTSIDE RANGE THEN ERROR COUNT
; IN R3 IS INCREMENTED
;
; ALSO ADDED TO CHECKSUM IN R4
;
CHECKR: MOVB    -(R1),R0
        BIC     #177600,R0
        ROL     R4
        ROL     R4      ;MODIFY CKSUM
        ROL     R4
        ADD     R0,R4
        CMPB    R0,R2
        BPL     1$
        INC     R3
1$:     SWAB    R2
        CMPB    R2,R0
        BPL     2$
        INC     R3
2$:     SWAB    R2
        RTS     PC
;
; NOW A CHECK ON THE CKSUM
;
; COMPARES NO IN R4 WITH LIST
; POINTED TO BY R2
;
; RETURNS R2=0 IF FOUND
;
CKSUM:  TST     (R2)
        BNE     1$
        RTS     PC
1$:     CMP     (R2)+,R4
        BNE     CKSUM
        CLR     R2
        RTS     PC
        .END    START
```

I've found another 4 blocks of user space for the latest release of the Virtual Memory (VM:) handler. The version that was on the Fall 1979 RT SIG swap tape provided 376 blocks of user space on PDP11 systems with 128K words of memory. I received an updated VM with patches to make the handler compatible with version 4 of RT-11 when the Spring 1981 RT SIG swap tape arrived. The new handler booted properly but only gave me 372 blocks for files. How come? A quick calculation showed that the 96K words of memory available for use as virtual disk storage should translate into 384 - (6 boot blocks + 2 blocks for the one directory segment) = 376 blocks of file space. The missing space in the new version seemed like a good excuse for me to find out how the darn thing works. Thank goodness the handler's author provided a commented source file!

The VM handler code that is executed at installation time determines the amount of memory available. A patch made to the 1981 version's high address check limit sized the full 128K word memory 1K low. I got my 4 blocks back by changing this one constant from -

```
        MOV     #7540,R0    ;R0 = max APR value
VMIPAT: BR      3$          ;NOMODE22; Don't check for 11/70
                    to -
        MOV     #7600,R0    ;R0 = max APR value
VMIPAT: BR      3$          ;NOMODE22; Don't check for 11/70
```

Although I'm sure that few people would describe the flaw I found as earth-shattering, the fix is simple. I hope those "mini tasker" readers who use the Virtual Memory handler will find it worthwhile to add this patch.

Sincerely

S. C. Cribbs

Steve Cribbs
Atomic Energy of Canada Ltd.
Pinawa, Manitoba
Canada ROE 1L0

RT-11 BASIC V2.

Ilan Caron
Kibbutz Ga'ash 60950
Israel

Date: 24- 11-82

PROBLEM:

Sometimes it is desirable to be able to choose the DATE format displayed by the BASIC-11 DAT$ function.

At our installation we work with an alternate character set - upper case converts to Hebrew, lower case to upper case. In any case, it's nice doing VAL(SEG$(DAT$,...)) to flush out the month.

SOLUTION:

The following program converts the internal BASIC-11 date format to/from alphanumeric (dd-MON-yy) from/to numeric (dd-mm-yy) format. A la RSTS/E - sort of.

BASIC-11 is a notoriously weak language. It's main redeeming quality is that it is interpretive. Among other things that it doesn't know how to do, is to read binary files of unknown length. There is almost no way to field the file via a reasonable DIM statement.

You end up having to choose between the following:

1) Writing ALR's to simulate RECORD I/O (again a la RSTS/E) - eg: GET , PUT , OPEN routines. But who likes inventing the wheel again anyway (not that I didn't write those routines - hating myself every minute for doing so).

2) Writing a special-purpose MACRO-11 program to perform a small set of functions. Something that can be surprisingly easy . . . (I was programmed in RSTS/E BASIC-PLUS).

In brief, in order to perform the trivial (in a higher-level language) task of replacing a binary string in a binary file, I ended up writing directly in MACRO-11.

UPDATE.SAV finds the month list in the respective compiler (normally this will be SY:BASIC.SAV) and substitutes the corresponding alphanumeric/numeric month list in the .SAV file. I have assumed (with no justification) that the month list does not span more than one block.

WISHLISTS ETC.

By the way, my Hanukah wishlist for RT-11/BASIC-11 contains:

1) EOF treatment for binary files
2) A SYS(Channel%,...) call - returning several goodies about an open channel, like filesize, current block, R/W mode, etc.
3) EXTEND mode (this is 1982 after all).
4) PEEK , POKE , Boolean algebra

One last comment, real-time is possible with a BASIC interpreter, just barely and only if you're willing to flesh it out with a set of ALR's.

```
.MAIN.  MACRO V03.01 26-NOV-82 12:18:20
TABLE OF CONTENTS

    1-   1      UPDATE - MAIN ROUTINE
    2-   1      READ - Sequential I/O of input file routine
    3-   1      MODE - Get user mode routine
    4-   1      SCAN - Block scan for substring routine
    5-   1      DATA - Parameter space and messages

 1                                      .SBTTL  UPDATE - MAIN ROUTINE
 2                                      .TITLE  UPDATE.MAC 24-NOV-82
 3                                      .IDENT  /1.0/
 4                                      .ENABL  LC
 5                      .LIST   TTM
 6                .NLIST  BIN
 7 000000         .PSECT  UPDATE
 8        ;-
```



```
 9        ; ************************************************************************
10        ; *                                                                      *
11        ; * Program Name: UPDATE                                                 *
12        ; *                                                                      *
13        ; * by Ilan Caron (Ga'ash, Israel)                                       *
14        ; *                                                                      *
15        ; * Purpose:                                                             *
16        ; *     switches BASIC-11 date format from                               *
17        ; *     alphanumeric to numeric format and vice versa                    *
18        ; *                                                                      *
19        ; *     22-NOV-82 <--> 22- 11-82                                         *
20        ; *                                                                      *
21        ; *                  Modification history                                *
22        ; *                                                                      *
23        ; * Date    Reason                                          Label        *
24        ; *                                                                      *
25        ; *                                                                      *
26        ; ************************************************************************
27        ;+
28
29
30                .MCALL  .CSIGEN , .CSTAT , .READW , .WRITW
31                .MCALL  .EXIT , .CLOSE , .SRESET
32                .MCALL  .PRINT , .GTLIN
33
34 000000  UPDATE:
35 000000         .PRINT  #BANNER                 ; say hello to the nice people...
36
37 000006  OPEN:
38 000006         CLR     IOBLK                   ; block # to be scanned.
39 000012         MOV     #NAME,R5                ; setup clear name buffer ptr.
40 000016  1$:
41 000016         CLR     (R5)+
42 000020         CMP     R5,#NAME+40.            ; finished
43 000024         BNE     1$                      ; no...
44
45 000026         .PRINT  #PROMPT
46 000034         .CSIGEN #DSPACE , #DEXT , #0 , #NAME
47                                                ; set input from KB:
48                                                ; (that's my RSTS/E blood showing)
49 000054         BCS     ERROR                   ; error in string
50 000056         TSTB    NAME                    ; null string?
51 000062         BEQ     EXIT                    ; yes, so exit...
52 000064         .CSTAT  #EMTLST,#CHNL,#STATUS   ; get channel status info.
53 000104         BCS     NOCHAN                  ; channel not open
54
55
56 000106         JSR     PC,MODE                 ; get FORMAT
57 000112         MOV     #512.,BLEN              ; default buffer length for block

 58 000120          MOV     #BUFFER,BADDR          ; buffer address for file block
 59 000126          MOV     #AREA,R5               ; point to arg list
 60
 61 000132          JSR     PC,READ                ; read the input file...
 62 000136          TST     FOUND                  ; converted?
 63 000142          BEQ     10$                    ; no...
 64 000144          .PRINT  #SUCCES                ; yes, success message
 65 000152          .PRINT  #NAME
 66 000160          BR      DONE
```

```
                10$:
000162                  .PRINT  #FAIL                   ; failure message
000170                  .PRINT  #NAME
000176                  BR      DONE

                NOCHAN:
000200                  .PRINT  #OERR                   ; channel not open error
000206                  BR      DONE

                ERROR:
000210                  .PRINT  #GLBERR                 ; error...

                DONE:
000216                  .CLOSE  #3                      ; close input file
000224                  .SRESET                         ; release handlers from memory
000226                  JMP     OPEN                    ; get another file...

                EXIT:
000232                  .EXIT                           ; back to RT-11


                        .SBTTL  READ - Sequential I/O of input file routine
000000                  .PSECT  READ

                ;+
                ; READ the input file block by block
                ;-

                READ:
000000                  .READW  R5,#CHNL                ; R5 points to list of args.
000014                  BCC     2$                      ; no errors...

000016                  TSTB    @#ERRBYT                ; what was the error? (EOF)
000022                  BEQ     3$                      ; *** FINISHED THE FILE ***
000024                  .PRINT  #RERR                   ; otherwise - read error
000032                  BR      100$                    ; and exit...

                2$:
000034                  JSR     PC,SCAN                 ; scan the block in buffer
000040                  INC     IOBLK                   ; bump block # to scan next
000044                  TST     FOUND                   ; did we find substrings?
000050                  BEQ     READ                    ; no, do another block...

000052                  .PRINT  #FNDSTR                 ; let them know - substring found
000060                  BR      100$                    ; and EXIT

                3$:                                     ; process EOF
000062                  BR      100$

                100$:
000064                  RTS     PC                      ; bye


                        .SBTTL  MODE - Get user mode routine
000000                  .PSECT  MODE

                ;+
                ; MODE sets format to convert to from user
                ;
```



```
                ; OUTPUTS:
                ;       FORMAT =0 --- convert numeric to alphanumeric
                ;       FORMAT =1 --- convert alphanumeric to numeric
                ;-

                MODE:
000000                  CLR     FORMAT
000004                  CLR     FMTTMP
000010                  .GTLIN  #BUF1,#PMODE            ; get user input
000030                  MOVB    BUF1,FMTTMP
000036                  SUB     #60,FMTTMP              ; make it binary

000044                  CMP     #1,FMTTMP
000052                  BNE     5$
000054                  INC     FORMAT                  ; yes, they want numeric format
000060                  BR      10$

000062          5$:     CMP     #2,FMTTMP               ; alphanumeric?
000070                  BNE     MODE                    ; out of range - try again
000072                  BR      10$                     ; alphanumeric...

000074          10$:
000074                  RTS     PC


                        .SBTTL  SCAN - Block scan for substrings routine
000000                  .PSECT  SCAN

                ;+
                ; SCAN scans a block of binary input
                ; looking for BASIC-11 date strings
                ;       1)  ' 01- 02- 03- 04- 05- 06- 07- 08- 09- 10- 11- 12'
                ;       2)  'JAN-FEB-MAR-APR-MAY-JUN-JUL-AUG-SEP-OCT-NOV-DEC'
                ;
                ; According to FORMAT will convert from one format to other.
                ;
                ; OUTPUTS:
                ;       FOUND =0 if substrings not found in block
                ;       FOUND =1 if substrings found in block
                ;-

                SCAN:
000000                  CLR     FOUND                   ; flag =0
000004                  CLR     RCOUNT                  ; counts the bytes in buffer
000010                  MOV     BADDR,R1                ; setup buffer address

000014          1$:
000014                  MOV     R1,SAVE1                ; save buffer pointer
000020                  TST     FORMAT                  ; which conversion do they want?
000024                  BNE     10$                     ;

000026                  MOV     #NUMRIC,R2              ; numeric...
000032                  MOV     #ALPHA,SAVE2            ; pointer to replace string
000040                  BR      20$

000042          10$:
000042                  MOV     #ALPHA,R2               ; alphanumeric...
000046                  MOV     #NUMRIC,SAVE2           ; pointer to replace string
```

```
000054  20$:
000054          INC     BCOUNT
000060          CMP     BCOUNT,BLEN             ; finished the buffer?
000066          BGT     100$                    ; yes...

000070          TSTB    (R2)                    ; reached end of string?
000072          BEQ     30$                     ; therefore found substring!!

000074          CMPB    (R2)+,(R1)+             ; compare the two strings...
000076          BEQ     20$                     ; so far, so good

000100          BR      1$                      ; no, try again

000102  30$:                                    ; found substring
000102          INC     FOUND                   ; set found flag =1
000106          MOV     SAVE1,R1                ; restore pointer to offset in
                                                ; block of substring
000112          MOV     SAVE2,R2                ; restore pointer to replace string

000116  40$:                                    ; REPLACE old with new
000116          TSTB    (R2)                    ; finished replacing old string
                                                ; with new string?
000120          BEQ     50$                     ; yes, so reWRITE the block

000122          MOVB    (R2)+,(R1)+             ; no, replace a character...
000124          BR      40$                     ; get another character to replace

000126  50$:
000126          .WRITW  R5,#CHNL                ; overwrite the block
000142          BCC     100$                    ; ok, if no carry
000144          CLR     FOUND                   ; flag =0
000150          .PRINT  #WERR                   ; write error!

000156  100$:
000156          RTS     PC                      ; and exit...


                .SBTTL  DATA - Parameter space and messages
000000          .PSECT  DATA

        ;+
        ;       D       A       T       A
        ;
        ;       parameter space for .CSIGEN
        ;-

000000  DEXT:   .WORD   0,0,0,0                 ; no default exts.

        ;+
        ;       parameter space for .READW/.WRITW
        ;-

000010  AREA:   .WORD   0                       ; EMT arg. block
000012  IOBLK:  .WORD   0                       ; block #
000014          .WORD   BUFFER                  ; buffer address
000016          .WORD   256.                    ; block length in words.
000020          .WORD   0

        ;+
        ;       parameter space for .CSTAT
        ;-
```



```
000022  EMTLST: .WORD   0,0                     ; 2 wd. emt arg list
000026  STATUS: .BLKW   6                       ; status is returned her

000042  FORMAT: .WORD   0
000044  FMTTMP: .WORD   0

        ERRBYT = 52                             ; error byte address
        CHNL = 3                                ; input file channel #

000046  BUFFER:
                .BLKW   256.                    ; buffer area for I/O
001046  NAME:   .BLKW   41.                     ; filename string will be copied here
001170  BUF1:   .BLKW   41.                     ; mode string will be copied her
001312  BCOUNT: .WORD   0                       ; # of bytes read in block already
001314  FOUND:  .WORD   0                       ; found flag
001316  BLEN:   .WORD   0                       ; buffer length
001320  BADDR:  .WORD   0                       ; buffer address
001322  SAVE1:  .WORD   0
001324  SAVE2:  .WORD   0

        ;+
        ;       M       E       S       S       A       G       E       S
        ;-

001326  RERR:   .ASCIZ  /?Read error/
001342  TERR:   .ASCIZ  /?Scan error/
001356  WERR:   .ASCIZ  /?Write error/
001373  OERR:   .ASCIZ  /?Channel not open/
001415  GLBERR: .ASCIZ  /?Error/
001424  SUCCES: .ASCII  /Successful conversion of: /<200>
001457  FAIL:   .ASCII  /?Failed to convert: /<200>
001504  PROMPT: .ASCIZ  <15><12>/Enter BASIC-11 compiler to convert: /
001553  FMODE:  .ASCII  /Convert date format to: 1=numeric  2=alphanumeric? /<200>

001637  NUMRIC: .ASCII  / 01- 02- 03- 04- 05- 06- 07- 08- 09- 10- 11- 12/<0>
001717  ALPHA:  .ASCII  /JAN-FEB-MAR-APR-MAY-JUN-JUL-AUG-SEP-OCT-NOV-DEC/<0>
001777  FNDSTR: .ASCIZ  /Found substring/
002017  BANNER: .ASCIZ  /UPDATE V1.0/
                .EVEN

        DSPACE = .                              ; load handlers here...

                .END    UPDATE
```

```
!       This example was executed on a
!       RSTS/E system using the RT-11
!       emulator.
!
SWITCH RT11

.RUN UPDATE
UPDATE V1.0

Enter BASIC-11 compiler to convert:
*BASIC.SAV
Convert date format to: 1=numeric  2=alphanumeric? 2
Found substring
Successful conversion of: BASIC.SAV
```

```
Enter BASIC-11 compiler to convert:
*BASIC.SAV
Convert date format to: 1=numeric  2=alphanumeric? 2
?Failed to convert: BASIC.SAV

Enter BASIC-11 compiler to convert:
*BASIC.SAV
Convert date format to: 1=numeric  2=alphanumeric? 1
Found substring
Successful conversion of: BASIC.SAV

Enter BASIC-11 compiler to convert:
*^C
```

At one of the RT-11 sessions at the '82 Fall DECUS Symposium, interest was expressed by several users in a document I have written, entitled Indirect Command File Processing For RT-11 V4.0.

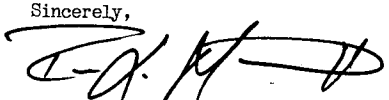
This manual describes in detail the patching procedure for RT-11. In addition, all of the working indirect command file processing directives and their use are described.

In keeping with Plessey Peripheral Systems' commitment to provide high quality software for its computers, we are providing this manual free of charge to users of RT-11 V4.0. In addition, this manual may be reproduced and distributed as desired; all we ask is that the Plessey name remain on the cover of the manual.

Enclosed, please find two copies of the manual: one hardcopy, one on single density diskette. The diskette version is in print-ready format, not runoff. Please feel free to print either the entire manual, or this letter, in the MINI-TASKER. Persons wishing to be sent copies of this manual may contact me at the above address.

Plessey Peripheral Systems
17466 Daimler
Irvine, CA 92714
(714)554-6095

Sincerely,



Russell L. Morrison II
Computer Systems Analyst

TABLE OF CONTENTS

1.0 Introduction

The RT-11 AUTOPATCH function uses a program called IND to emulate the RSX Indirect Command Processor. Some interest was expressed at the U.S. DECUS Symposium in using this program under RT-11 to do fancy indirect command files. However, in trying to do so, users have discovered that IND does not work apart from RT-11 AUTOPATCH.

In the February, 1982 MINI-TASKER (the RT-11 SIG Newsletter from DECUS), John M. Crowell, of CROW4ELL, Ltd., published a set of patches that unlock IND from AUTOPATCH Revision D. This document gives the instructions for applying those patches, and also details the use of IND in the patched system.

The patches detailed in Section 2.0 will create a version of IND that works under RT-11, and a version of RT-11 that will support a CCL IND command. These patches are effective for the SJ (with timer support), FB, and XM RT-11 monitors, and allow the use of all RSX command file directives except those that are RSX specific. The patched IND program does not work under TSX or TSX-Plus.

No guarantee is made by Plessey Peripheral Systems, DECUS, or CROW4ELL, Ltd., as to the functionality of IND or the accuracy of the statements contained in this document. No support of the products described is implied by this document, and no support will be given.

2.0 Patching the System

2.1 Patching IND

IND is locked to AUTOPATCH Revision D in two ways. The first is straightforward and rather obvious: IND automatically looks for the file SY:AUTOP.COM. The second lock is more subtle: IND checks for a double carriage return in the copyright and disclaimer notice that heads up the AUTOP.COM file. This double carriage return would be invisible in any listings of the file, but if this aberration is not present, IND refuses to process the file.

The following command file disables the routine that checks for a double carriage return, and enables the subroutine that asks for input (a command file name and a switch) from the terminal.

```
R SIPP
IND.SAV/A/C
0
14512
45104
41515
53040
30460
33456
^Z
15324
3310
^Z
15346
3266
^Z
21044
207
^Z
```



```
25004
1142
^Z
26250
427
^Z
27266
165354
^Y
14533
^C
```

Once IND has been patched, an operator may enter command files at the terminal in the following form:

```
R IND
filspec[/switch] [par1 par2 par3 . . . parn]

(brackets not part of syntax)
```

where

filspec — is the indirect command file to be processed, with a default extension of .COM.

switch — is one of the valid IND switches described in Section 3.1.

par1... — are parameters to be passed to the indirect file in the manner described in Section 4.5

It is also possible to use IND in a one line command, without parameters, thus:

```
RUN IND filspec/switch
```

If these two command forms are enough, the user may not want to apply the patch to the keyboard monitor which will enable the CCL command IND.

Please Note: This patch is designed for the IND program that is supplied with AUTOPATCH Revision D. It does not work on other versions of IND.

2.2 Patching the Keyboard Monitor

If the user wishes to use IND as a CCL command, this patch for the RT-11 keyboard monitor will enable it. Before applying the patch, RT-11 should be patched by the current revision of AUTOPATCH. The patch consists of a series of commands to the TECO editor which alter KMON.MAC and KMOVLY.MAC so that the IND command is enabled.

CAUTION: This patch will alter KMON.MAC and KMOVLY.MAC in such a way that DEC supplied SLP patches to them will not work. SAVE THE ORIGINAL FILES!!! When DEC supplies a new patch or a new version of AUTOPATCH, use it with the original files and re-apply these commands to the newly patched files.

Edit the following into a file, IND.TXT:

```
@EB/SRC:KMON.MAC/
@N/DEFALT       COPY/@N/BF/
OI@I/DEFALT     IND$$   U$TIL
/@N/PROG        APL/
I@I/.IIF NE     IND$$ PROG      IND
/@N/CMDTBL      FOCAL/
I@I/.IIF NE     IND$$ CMDTBL    IND
/@N/FLGTXT      <FOC AL>/
I@I/.IIF NE     IND$$ FLGTXT    <IND>
/@EC
@EB/SRC:KMOVLY.MAC/
@N/SYNTAX       FORMAT/ZJ12@I//
@I/.IF NE       IND$$
.SBTTL  IND COMMAND
        SYNTAX  IND
        PROMPT  $FILEE
        SCALL   GSWIT,<1>

        REQBLNK
        SCALL   RINSPC,<1>
        EOLSEQ  IND
        END
        SWITS   IND
SWIT    DELETE          1       IND             - D
SWIT    TRACE           1       IND             - T
SWIT    NOEXEC          1       IND             - N
        ENDNO
        NOS
        ENDS
        OVCMD   IND
        ITBLE   1
        OJSR    PC,INITIT
        OJMP    CMDEXE
.ENDC
/@N/BOOT$$/L
@I/.SBTTL       BOOT COMMAND
/@N/SWTDEF/@S/.EVEN/OL
@I/     SWTDEF  <DELETE>
        FLGTXT          <D ELETE>
        SWTDEF  <TRACE>
        FLGTXT          <T RACE>
        SWTDEF  <NOEXEC>
        FLGTXT          <N OEXEC>
/@EX
```

Once this text file has been created, enter the following commands at the console:

```
ASS ddu SRC
R TECO
ERIND.TXT<esc>AJ44XD<esc>HKMD<esc><esc>
```

Where ddu is the name of the device on which the source files KMON.MAC and KMOVLY.MAC reside and <esc> represents an escape (ASCII 27 decimal) character. After this command has executed, a SYSGEN should be performed. Once the SYSGEN has been performed and a new monitor created, RT-11 will recognize an IND command of the form

```
IND filspec/switch
```


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

****

HDAN : A KEY-TO-ADDRESS TRANSFORM "HASH-CODING" LINEAR FUNCTION
ACCEDING DIRECTLY TO ELEVEN COMPRESSED CHARACTERS.

****

BY DANIEL GUINIER

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

****

INTRODUCTION :
****

A DIRECT ACCESS METHOD PROVIDES THE BEST MEANS OF CONSULTING ELEMENTS IN A FILE. IN SOME CASES A RECORD CAN BE REPRESENTED BY AN ALPHA-NUMERIC KEY. THE DIFFICULTY RESIDES IN A GOOD TRANSFORMATION OF THE PRIME INFORMATION WHICH IS A PART OF A RECORD ASSIMILABLE TO A KEY BY AN ALTERATION OR NOT OF A PART OF IT. THIS TECHNIQUE IS CALLED THE "HASH-CODING" TECHNIQUE.

WE HAVE DEVELOPPED A TECHNIQUE WHICH TAKES DIRECTLY THE FIRST ELEVEN CHARACTERS OF THE FULL ALPHANUMERIC KEY INCLUDING UP TO A MAXIMUM O 44 CHARACTERS. IN THIS CASE, WHEN TWO DIFFERENT FULL KEYS GIVE THE SAME ADDRESS, IT RESULTS A "COLLISION" THAT MUST BE TREATED BY ANOTHER SUBROUTINE. IF THE "COLLISIONS" ARE CORRECTLY MANAGED, WE ARE SURE TO OBTAIN DIRECTLY RANKED RECORDS.

BIBLIOGRAPHY :
****

Y. Y. LUM, P. S. T. YUEN, M. DODD (1971) : KEY-TO-ADDRESS TRANSFORM TECHNIQUES : A FUNDAMENTAL PERFORMANCE STUDY ON LARGE EXISTING FORMATTED FILES. (COMMUNICATIONS OF THE A. C. M. APRIL, VOL. 14, NO. 4 PP. 228-239).

D. E. KNUTH (1973) : SORTING AND SEARCHING, THE ART OF COMPUTER PROGRAMMING (3). (ADDISON-WESLEY, READING, MA.).

R. BAYER (1974) : STORAGE CHARACTERISTICS AND METHODS FOR SEARCHING AND ADDRESSING. (INFORM. PROCESS. -NORTH-HOLLAND PUBLISH. COMPANY PP. 440-443).

P. CLAPSON (1977) : IMPROVING THE ACCESS TIME FOR RANDOM ACCESS FILES.
(COMMUNICATIONS OF THE A. C. M. MARCH, VOL. 20, NO. 3, PP. 127-135).

E. GOTO, T. IDA (1977) : PARALLEL HASHING ALGORITHMS. (INFORM. PROCESS.
LETTERS FEBRUARY, VOL. 6, NO. 1, PP. 8-13).

R. SPRUGNOLI (1977) : PERFECT HASHING FUNCTIONS : A SINGLE PROBE
RETRIEVING METHOD FOR STATIC SETS. (COMMUNICATIONS OF THE
A. C. M. NOVEMBER, VOL. 20, NO. 11, PP. 841-850).

W. LITWIN (1978) : INFORMATIQUE. -UNE NOUVELLE METHODE D'ACCES PAR
CODAGE DECOUPE A UN FICHIER. (C. R. ACAD. SC. PARIS, T. 286, 24 AVRIL,
SERIE A, PP. 695-698).

R. L. RIVEST (1978) : OPTIMAL ARRANGEMENT OF KEYS IN A HASH TABLE.
(J. OF THE A. C. M. APRIL, VOL. 25, NO. 2, PP. 200-209).

R. FAGIN, J. NIEVERGELT, N. PIPPENGER, H. R. STRONG (1979) : EXTENDIBLE
HASHING-A FAST ACCESS METHOD FOR DYNAMIC FILES. (A. C. M. TRANS. ON
DATABASE SYSTEMS, VOL. 4, NO. 3, SEPTEMBER, PP. 315-344).

PRINCIPLES OF THE METHOD :
****

1). COMPRESSION : THE FIRST 44 CHARACTERS OF THE STRING CAR()
ARE DIVIDED INTO FOUR SETS OF ELEVEN CHARACTERS CONVERTED INTO FOUR RADIX50
REAL*8 A(I) VALUES. THAT INCREASE THE VELOCITY WHEN IDENTITY IS MET FOR
THE PREVIOUS A(I)'S. IT IS NOT NECESSARY TO COMPARE CHAINS OF ELEVEN
CHARACTERS FROM LOW TO HIGH WEIGHTS AND THE COMPRESSION FACTOR IS 25 %,
THAT IS AN ECONOMY OF STORAGE. THE ORIGINAL SETS OF CHARACTERS
(CAR()) CAN BE RESTORED BY THE INVERSE TRANSFORM FUNCTION : RADIX50 TO ASCII.

2). LINEARITY : THERE IS A LINEAR RELATIONSHIP BETWEEN THE NEPERIAN
LOGARITHM OF THE RESULT A() OF THE CONVERSION RADIX50 OF THE STRING CAR()
AND ITS DECIMAL VALUE, THAT INVOLVES A LINEAR RELATION BETWEEN THE
CALCULATED ADDRESS : "HDAN". THE RELATION IS GIVEN BY THE FORMULA :

HDAN=(DLOG(DABS(A(1)))+89.14405060)*(NADR-1)/5.56083+1.5

A GRAPHICAL EXPRESSION OF THIS RELATION IS GIVEN IN FIG. 1.

LISTING OF THE FORTRAN IV SOURCE :
****

```
      C
      C  ********************************************************************
      C
0001        FUNCTION HDAN(CAR,A,I,NADR)
      C
      C  RETURN THE ADDRESS FOR THE KEY CAR() CONTAINING ASCII CHAR.
      C
      C  (THIS "HASH-CODING" FUNCTION IS NOT WEIGHTED BY
      C  THE PROBABILITY DISTRIBUTION OF THE FIRST CHARACTER OF CAR).
      C
      C  ********************************************************************
```



```
      C
      C  CAR() IS TREATED LIKE THE 44 FIRST CHAR. OF CAR() ARE USED
      C  AS IN THE FOLLOWING FORMAT :  4*(<SPACE>,11 CHAR.).
      C  EXAMPLE :
      C   FIRSTFIRSTF SECONDSECON THIRDTHIRDT FOURTHFOURT
      C  -          -          -          -
      C  IS THE RESULT OF THE TREATMENT OF THE ORIGINAL CAR() :
      C  FIRSTFIRSTFSECONDSECONTHIRDTHIRDTFOURTHFOURTXXXXXX...XXX
      C
      C  AND THE 4*(1+11)=48 CHAR. ARE REAL*8 RADIX 50 CONVERTED AND MOVED IN A()
      C
      C  AT THE END, THE ADDRESS IS CALCULATED BY MEANS OF A(1).
      C
      C
      C  I     : ACTUAL NUMBER OF CHARARACTERS DETERMINED.
      C  NADR  : NUMBER OF POSSIBLE ADDRESSES.
      C
0002        BYTE CAR(128)
0003        REAL*8 A(4)
      C
      C  <SPACE> COMPLETION.
0004        I1=I+1
0005        DO 1 J=I1,48
0006  1     CAR(J)=32
0007        K1=I/12+1
0008        IF(K1.GT.4)K1=4
0010        DO 4 K=4,1,-1
0011        I1=11*K
0012        I2=I1-10
0013        IF(K.GT.K1)GO TO 3
0015        DO 2 J=I1,I2,-1
0016  2     CAR (J+K)=CAR(J)
0017  3     M=I2+K-1
0018        CAR(M)=32
      C  RAD50 TRANSFORMATION
0019        J=IRAD50(12,CAR(M),A(K))
0020  4     CONTINUE
      C  ADDRESS CALCULATION FROM A(1).
      C  CAR(2) (AT THE BEGINNING : CAR(1)) MUST BE INCLUDED IN [A-Z].
      C  IF NOT, WE DO HDAN=NADR.
0021        HDAN=NADR
0022        IF(CAR(2).GT.90.OR.CAR(2).LT.65)RETURN
0024        HDAN=(DLOG(DABS(A(1)))+89.14405060)*(NADR-1)/5.56083+1.5

      C  I IS RETURN TO THE VALUE I-K1
0025        RETURN
0026        END
```

```
***** COMPILATION STATISTICS *****
*                                 *
*------- COMPILER  TABLES -------*
* SYMBOLS:           00223 WORDS *
* PROGRAM:           00203 WORDS *
*                                 *
***********************************
```

ADVANTAGES AND DISADVANTAGES OF THE METHOD :
****

IN ASSOCIATION WITH A SUBROUTINE WHICH CORRECTLY TREATS THE "COLLISIONS", THIS METHOD PERMITS TO OBTAIN INFORMATION DIRECTLY SORTED. THE OPERATIONS OF COMPRESSION INCREASE THE VELOCITY OF THE COMPARISONS WHEN AN IDENTITY OR A "COLLISION" ARE MET. THE EFFICIENCY, EXPRESSED BY A LOW RATE OF "COLLISIONS", CAN BE AFFECTED BY THE NATURE OF THE SOURCE (FRENCH, ENGLISH, DICTIONARY, ANY INFORMATIC LANGUAGE,...), IF A LARGE PERCENTAGE OF AGGREGATED AND EMPTY ZONES FOLLOW THAT IS THE RESULT FROM A BAD DISTRIBUTION OF THE MONOGRAMS OR BIGRAMS. IF NECESSARY, A PRIOR MONOGRAM OR BIGRAM ANALYSIS CAN BE PREVIOUSLY REQUESTED TO WEIGHT THE RESULT OF HDAN().

APPLICATION :
****

WE HAVE APPLIED THIS "HASH-CODING" FUNCTION FOR QUANTITATIVE CONTENT ANALYSIS OF A BIBLIOGRAPHIC ASCII FILE (10000 ITEMS) WITH THE PROGRAM : DOST1.

THE RESULTS ON THE DIFFERENT FIELDS FOR THE FIRST 100 ITEMS WITH A LOADIN FACTOR OF 100% ARE :

DICTIONARY OF KEY-WORDS : COLLISION RATE : 75.6%
DICTIONARY OF REVIEWS : COLLISION RATE : 57.6%
DICTIONARY OF AUTHORS : COLLISION RATE : 49.5%

IN THIS CASE, IT WAS NECESSARY TO WEIGHT THE FUNCTION TO ELIMINATE THE EFFECTS OF THE AGGREGATES. THIS EXAMPLE SHOWS THAT THE FUNCTION IS PERTURBED DIFFERENTLY BY THE NATURE OF THE INFORMATION, AND THE USER SHOULD STUDY THE DISTRIBUTION OF THE MONOGRAMS OR BIGRAMS OF THE SOURCES TO IMPROVE EFFICIENCY IN THE APPLICATION.

CONCLUSION :
****

WHEN NO PRE-SORTING IS NEEDED, WE THINK THAT IT IS PREFERABLE TO USE A "HASH" FUNCTION THAT IS INDEPENDENT OF THE SOURCE ITSELF. WE WILL CONSIDER THIS PROBLEM AND ITS SOLUTION IN A LETTER PUBLICATION ON THE "HASH-CODING" FUNCTION : HASDAN. THE FUNCTION HASDAN IS A UNIVERSAL NO-DEPENDENT OF THE SOURCE "HASH-CODING" FUNCTION THAT USES ALL INFORMATION (1 TO N CHARACTERS) AND IS NOT SENSITIVE TO AGGREGATES.

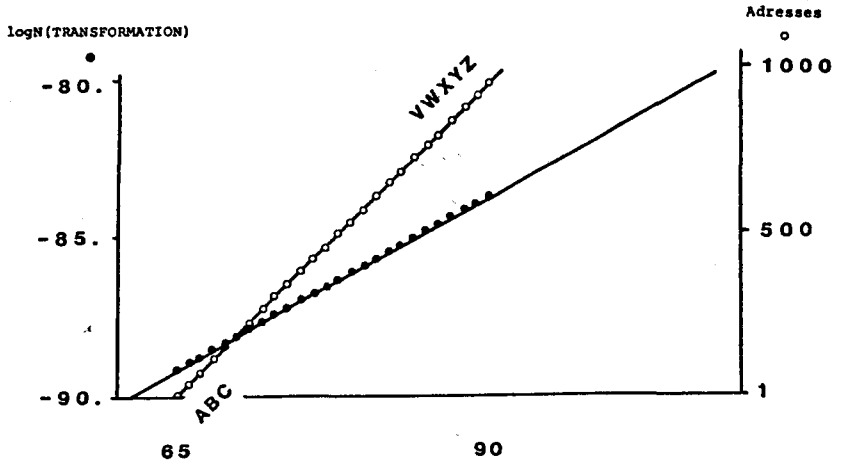


FIG.1. Linear relations between :
logN of the real transform RADIX50 of the characters sets in function of the decimal ASCII values.
adresses in function of the decimal ASCII values.

USER REQUESTS

On behalf of one of our customers we are trying to locate an RT-11 emulator to run under RSX-11 V3.2. The customer in question has purchased a software system called MINITAB.

MINITAB under RSX-11M requires a FP11 floating point unit since the taskbilder (RSX-11M SIOTKB) can not cope with MINITAB unless FP11 is installed. The RT-11 linker can, on the other hand, link MINITAB without a FP11.

We are aware of RTEM-11 from DEC but are looking for a more economical solution.

If you can help us please inform me how I can obtain an emulator.

VKS SF Computer Systems Software Development Consulting
Skeggjagata 2 105 Reykjavík Iceland Phone 2 80 22

Yours truly,

Friðrik Marteinsson
Systems Analyst

Hús verzlunarinnar,
108 Reykjavík,
Iceland.

User request

1. We are very interested in logical disk subsetting. Currently we use the "virtual" disk handler and its utility from the FALL 81 DECUS tape. It implements virtual disk at Queue element level. We use the Crapuchette version. It works fine, except we get always the message: "ASLOOK?-W-assignment table not found"(Anyone found the bug?). As DEC will provide logical disk subsetting under V5, could a preliminary overview of the V5 implementation be given, and the differences with the DECUS version?

2. The VM: is a very nice fast device for under SJ and FB. The VM.MAC code suggests that is supports PDP 11/70 22-bit. But who has a PDP 11/70 available under RT-11? Does the VM code also support other 22-bit systems?

3. RT-11 does not check or can check if a job references memory outside its jobspace. However when you specify a buffer address outside the jobspace in a .READ/WRITE you get an error. I could not find documentation on this subject in the manuals. Are there also other protection schemes where most users are not aware of? What is the benefit of this protection, apart form specifying a wrong buffer address by mistake? It does limit job communication. How many code does it occupy in RMON?

I hope the DEC RT-11 support group reads these requests! I'm sure they could give good answers.

4. Would anyone report on DISK-DATA CACHING. Experience, strategies to follow, existing program code, etc.

5. We use the "Transparent Output Spooler" found on DECUS Proc. tape, developed by Greg. L. Adams. Although we are very satisfied with it, we would suggest some enhancements and extensions. As the author , as stated in the sources, strongly recommends to contact him, before changing anything, we would like to know his current address.

CPU LOAD MEASUREMENT

Measuring CPU load may be useful to get insight how a program uses the CPU. It informs you if you can put additional processing in your program. In a multijob environment it may be important to know the CPU load of each job.
Under version 2/3 we did the measurement by replacing the idle-loop routine by one, equal in size, just incrementing a 2-word counter. By reading this counter at the begin and end of a program, CPU load could be calculated. However, having software support, modifying the monitor, is less elegant and can give problems with DEC. Fortunately I discovered that under V4 the FB monitor with system job support (the one we use normally), had an idle-loop, one-word counter. I found the counter in the RTMON program (produced by DEC, distributed on DECUS tape), which is indeed a very nice program. This V4 idle-loop counter however overflows about every 10 sec., so it must be read regularly when measuring over longer intervals. Two different sets of programs are supplied to measure CPU load. One for measuring CPU load over the integral time period during which program executes and one for the CPU load within small, say 1 sec., timeslices.

1. PEM.MAC, PEMMON.MAC, PEMDON.MAC

PEM contains two subroutines: PEMSTR and PEMSTP for start and stopping CPU load measurement. The program layout is as follows:

```
CALL PEMSTR
-program code-
CALL PEMSTP
```

Example printout of PEMSTP:

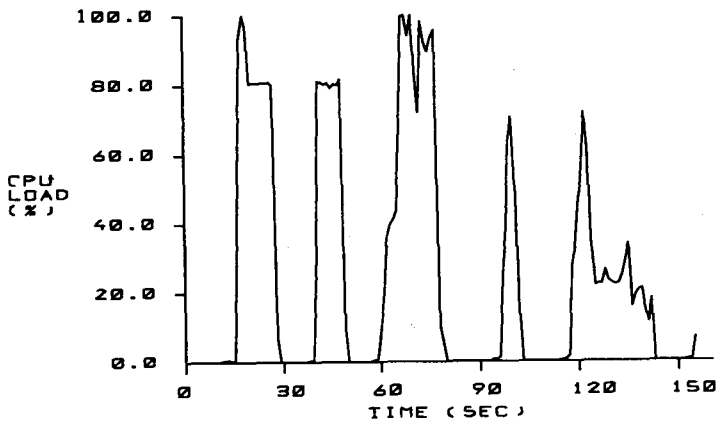
```
Current   time: 16 HOURS 51 MINUT 12 SECND 14 TICKS
Execution time: 00 HOURS 00 MINUT  6 SECND 43 TICKS
Free CPU  time: 00 HOURS 00 MINUT  2 SECND 47 TICKS

CPU usage ratio in PROMILLE:  572
```

PEMMON is a main program to run in the Foreground. It calls PEMSTR, then waits for a Background message, and, when this is received calls PEMSTP and aborts. The background message may be send by the program PEMDON.

2. PEMLOG.MAC, PEMLOG.FOR

The Macro version is a program to run in the Foreground. After initialisation it measures CPU load every timeslice and loggs it in the file SY:PEMLOG.TMP. The FORTRAN program is an example of how this file may be read and printout of data can be done. This program could be extended with e.g. graphics code. The figure below shows an example of CPU load for the command sequence DIR/ALPHA (for RK05), DIR, MAC PEMLOG, FORT PEMLOG, @PEMLOG.LNK. Timeslice is 1 sec. PEMLOG starts and stops with a message from the Background. Therefore PEMDON may be used if PEMLOG is started with the command: .FRUN/NAME:PEMMON PEMLOG
Note: all programs just read the idle-counter in the monitor. If you have not the correct null counter offset in the monitor, you probably always measure a CPU load of 100%. Watch out for the DECUS RTMON program, it also clears the counter, and if you have not the correct offset, the monitor may trap at mysterious occasions.



```
        .TITLE  PEM
        .MCALL  .QSET,.TWAIT, .PRINT, .GTIM, .MRKT, .CMKT
        .GLOBL  JMUL, JDIV, CVTTIM
        .ENABL  LC
;DAT 21-JUL-77  V2C/7-APR-82  V4        H. HAENEN-NKG
;       *** Program for CPU usage MEASUREMENT ***
;Note: *FIRST CODING IN USER PROGRAM: CALL PEMSTR
;       *  END CODING OF USER PROGRAM: CALL PEMSTP
;       *MODULE MUST RESIDE IN ROOT WHEN OVERLAYING!
;       *TIMER ID. 16001. IS USED!
;PRINTOUT: EXECUTION TIME/WAISTED CPU-TIME/CPU USAGE RATIO
RMON=54
NULCNT=420
PEMSTR::CLR     NULLO
        CLR     NULHI
        TST     TICK                    ;.QSET already
        BNE     1$                      ;               done?
        .QSET   #XQUE,#1                ;Reserve 1 Q-element
1$:     MOV     #100.,TICK
        MOV     @#RMON,R1
        MOV     NULCNT(R1),R2           ;Get value idle counter.
        .TWAIT  #AREA,#TIME             ;Wait 2 sec.
        CLR     R4
        MOV     NULCNT(R1),R5           ;Get new idle value
        SUB     R2,R5
        DIV     TICK,R4                 ;100. ticks= 2 sec.
        MOV     R4,REFIDL               ;Store reference idle value.
;------------------------ Schedule timer --------------------------------;
        CLR     TIME                    ;High order
        MOV     #300.,TICK              ;Low order
        .MRKT   #TAREA,#TIME,#COMPLT,#16001.    ;Schedule nulcnt runout.
        BCC     2$
        .PRINT  #MRKMES
2$:     .GTIM   #AREA,#TOLD     ;GET CURRENT TIME
        MOV     @#RMON,R1
        MOV     NULCNT(R1),SAVE ;GET NULL COUNT
        RETURN
TIME:   .WORD   0
TICK:   .WORD   0               ;Interval ticks. Also flag for .QSET
XQUE:   .BLKW   10.
NULLO:  .WORD   0
NULHI:  .WORD   0
REFIDL: .WORD   0
SAVE:   .WORD   0
MRKMES: .ASCIZ  /?PEM-F-No Q-Element?/
        .EVEN
;------------------- Timer completion routine ---------------------------;
COMPLT: MOV     @#RMON,R0
        MOV     NULCNT(R0),R1           ;Get new idle value
        SUB     SAVE,R1
        CLR     R0
        DIV     REFIDL,R0
        ADD     R0,NULLO
        ADC     NULHI
        MOV     @#RMON,R1
        MOV     NULCNT(R1),SAVE         ;GET NULL COUNT
        .MRKT   #TAREA,#TIME,#COMPLT,#16001.    ;SCHEDULE NULCNT RUNOUT
        BCC     1$
        .PRINT  #MRKMES
1$:     RETURN
```



```
PEMSTP::.GTIM   #AREA,#TNEW     ;GET CURRENT TIME
        MOV     TNEW,R2         ;HIGH ORDER PART.
        MOV     TNEW+2,R3       ;LOW ORDER PART.
        MOV     R2,TCURR        ;STORE CURRENT TIME, HIGH
        MOV     R3,TCURR+2      ;                    LOW
        SUB     TOLD+2,R3       ;SUBSTR LOW ORDER PART.
        SBC     R2              ;SUBSTR BORROW FROM HIGH PART.
        SUB     TOLD,R2         ;SUBSTR HIGH ORDER PART.
        MOV     R2,TNEW
        MOV     R3,TNEW+2
        .CMKT   #AREA,#16001.,#TIME ;Cancel timer
        MOV     @#RMON,R1
        CLR     R4
        MOV     NULCNT(R1),R5   ;Get new idle value
        SUB     SAVE,R5         ;Substract start val.
        DIV     REFIDL,R4
        ADD     R4,NULLO
        ADC     NULHI
        .PRINT  #MSGHEL         ;HEADER PRINTOUT
        .PRINT  #MSG0           ;PRINT CURRENT TIME
        MOV     #TCURR,LIST+2
        JSR     PC,OUTPUT
        .PRINT  #BUF
        .PRINT  #MSG1           ;PRINT EXECUTION TIME
        MOV     #TNEW,LIST+2
        JSR     PC,OUTPUT
        .PRINT  #BUF
        MOV     TNEW,R3         ;Re-arrange high and low
        MOV     TNEW+2,TNEW     ;               for INTEGER*4
        MOV     R3,TNEW+2       ;                       support.
        .PRINT  #MSG2
        MOV     NULLO,TOLD+2
        MOV     NULHI,TOLD
        MOV     #TOLD,LIST+2    ;PRINT 0-JOB TIME
        JSR     PC,OUTPUT
        .PRINT  #BUF
        MOV     NULLO,TOLD
        MOV     NULHI,TOLD+2
        MOV     #VERM,R5
        JSR     PC,JMUL         ;INTEGER*4 multiply.
        CMP     #-2.,R0         ;CHECK ON OVERFLOW
        BEQ     OVF
        MOV     #DELEN,R5       ;INTEGER*4 divide.
        JSR     PC,JDIV
        MOV     #1000.,R2
        MOV     R2,TNEW+2       ;CALCULATE EFFICIENCY
        SUB     TNEW,TNEW+2     ;PRO MILLE INVERT
        .PRINT  #MSG3
        MOV     TNEW+2,R1
        BGE     OK1
        CLR     R1
        BR      OK2
OK1:    CMP     R1,R2
        BLE     OK2
        MOV     R2,R1
OK2:    MOV     #HRS,R2
        JSR     PC,BINDEC       ;PRINT CPU USAGE RATIO
        MOVB    #12,(R2)+
        CLRB    (R2)+
        .PRINT  #HRS
        RTS     PC
```

```
OVF:    .PRINT  #OVFMSG
        RTS     PC
;------------- Convert INT*4 to ASCII Hrs Min. Sec. Tick. ---------------;
OUTPUT: MOV     #LIST,R5
        MOV     R2,-(SP)
        MOV     R3,-(SP)
        JSR     PC,CVTTIM       ;HRS,MINS,SECND,TICKS
        MOV     #BUF,R2
        MOV     #4,R3
        MOV     #HRS,R4
LOOP:   MOV     (R4)+,R1
        JSR     PC,BINDEC       ;Decimal-ASCII's
        ADD     #6,R2
        DEC     R3
        BNE     LOOP
        MOV     (SP)+,R3
        MOV     (SP)+,R2
        RTS     PC
VERM:   .WORD   3,TOLD,CONST,TOLD ;JMUL argum. (1)*(2)=(3)
DELEN:  .WORD   3,TOLD,TNEW,TNEW  ;JDIV argum. (1)/(2)=(3)
CONST:  .WORD   1000.,0         ;PRO MILLE
TAREA:  .BLKW   4
AREA:   .BLKW   5
TOLD:   .WORD   0,0
TNEW:   .WORD   0,0
TCURR:  .WORD   0,0
OVFMSG: .ASCIZ  /0-JOB time EXCEEDED MAX./
MSGHEL: .ASCII  /       *** Results TIME measurements and CPU usage ***/
        .BYTE   15,12,12,200
MSG0:   .ASCII  /Current   time: /<200>
MSG1:   .ASCII  /Execution time: /<200>
MSG2:   .ASCII  /Free  CPU-time: /<200>
MSG3:   .ASCII  <012>/CPU-usage ratio in PROMILLE: /<200>
        .EVEN
BUF:    .ASCII <0><0><0><0><0><0>/ HOURS/<0><0><0><0><0><0>
        .ASCII / MINIT/<0><0><0><0><0><0>
        .ASCII / SECND/<0><0><0><0><0><0>
        .ASCII / TICKS/<0><0>
LIST:   .WORD   5,0,HRS,MINS,SECNDS,TICKS
HRS:    .WORD   0
MINS:   .WORD   0
SECNDS: .WORD   0
TICKS:  .WORD   0
DUMMY:  .WORD   0
        ;SUBROUTINE FOR CONVERSION OF ONE BIN. WORD TO 6 ASCII CHARS
        ;ASSUMES THE WORD TO BE IN R1, POINTER TO STR. BUFFER IN R2
BINDEC::MOV     R3,-(SP)
        MOV     R4,-(SP)
        ADD     #6,R2
        MOV     #5,R3           ;SET UP COUNT
BLOOP:  CLR     R0
        DIV     #10.,R0         ;RESULT IN R0,REMAINDER IN R1
        ADD     #'0,R1
        MOVB    R1,-(R2)
        MOV     R0,R1
        DEC     R3
        BNE     BLOOP
        MOVB    #' ,-(R2)       ;ADD BLANK
        ADD     #6,R2           ;CORRECT POINTER
        MOV     (SP)+,R4
        MOV     (SP)+,R3
        RTS     PC
        .END
```



```
        .TITLE PEMMON processor usage monitor

        .MCALL  .EXIT,.LOOKUP,.READW,.CLOSE,.PRINT
        .GLOBL  PEMSTR,PEMSTP
START:  .LOOKUP #AREA,#0,#JBBLK         ;Open a message channel.
        BCC     JOBOK
        .PRINT  #NOJOB
        .EXIT
JOBOK:  CALL    PEMSTR                  ;Start measurement.
        .READW  #AREA,#0,#BUF,#1        ;Wait until someone sends
        BCC     2$                      ;       us a message.
        .PRINT  #MSGERR
2$:     CALL    PEMSTP                  ;Stop measurement.
        .CLOSE  #0
        .EXIT
;---------------------------- Storage -------------------------------
BUF:    .WORD   0,0
AREA:   .BLKW   4
JBBLK:  .RAD50  /MQ/
        .WORD   0,0,0,0                 ;Accept messages from all jobs
NOJOB:  .ASCIZ  /?PEMMON-F-Fatal JOB LOOKUP/
MSGERR: .ASCIZ  /?PEMMON-F-Message READ error/
        .EVEN
        .END    START



        .TITLE PEMDON sends dummy message to PEMMON

        .MCALL  .EXIT,.LOOKUP,.WRITW,.CLOSE,.PRINT
;-------------------- Send message --------------------;
START:  .LOOKUP #AREA,#13.,#JBBLK       ;PEMMON running?
        BCC     JOBOK
        .PRINT  #NOJOB
        BR      EXIT
JOBOK:  .WRITW  #AREA,#13.,#BUF,#1      ;O.K. give message!
        BCS     1$
        .PRINT  #MSGOK
        BR      EXIT
1$:     .PRINT  #MSGERR
        BR      EXIT
;----------------------- Storage-----------------------;
BUF:    .WORD   1
AREA:   .BLKW   4
JBBLK:  .RAD50  /MQ/
        .ASCIZ  /PEMMON/<0>
NOJOB:  .ASCIZ  /?PEMDON-F-PEMMON not running/
MSGOK:  .ASCIZ  /?PEMDON-I-Message Transmitted!/
MSGERR: .ASCIZ  /?PEMDON-F-Message WRITW error/
        .EVEN
;----------------------- Finish ------------------------
EXIT:   .CLOSE  #13.
        .EXIT
        .END    START
```

```
        .TITLE PEMLOG
        .MCALL .TWAIT,.EXIT,.ENTER,.LOOKUP,.PRINT,.QSET
        .MCALL .READW,.READC,.WRITW,.GTIM,.CLOSE,.GVAL
        .ENABL LC

;  PEMLOG.MAC CPU usage logging, storage in file SY:PEMLOG.TMP
;  H.H. NKG-AZG JUN/JUL 82
RMON    = 54            ;Pointer to monitor base.
SYSOP   = 372           ;Sysgen options
STASK$  = 40000         ;System task support present
NULCNT  = 420           ;Monitor offset to null-counter, check accurately!
                        ;Monitor must support SYSTEMJOBS!
TIMEP=50.               ;Time slice(ticks) for each measurement.
        .PSECT  INIPEM
START:
BUFFER::.PRINT  #INIT1
        .TWAIT  #AREA,#TIMINI           ;Wait 4 sec. to relax CPU
        MOV     @#RMON,R1
        MOV     NULCNT(R1),R2           ;Get value idle counter.
        .TWAIT  #AREA,#TIMINI           ;Wait 4 sec.
        MOV     NULCNT(R1),R0           ;Idle value
        SUB     R2,R0                   ;Substract start val.
        ASR     R0                      ;/2
        ASR     R0                      ;/4
        MOV     #NONUL,R2               ;R2 -> Error message
        MOV     R0,REFIDL               ;Store reference idle value.
        BEQ     FATAL                   ;Wrong NULCNT offset?
        .PRINT  #INIT2

;------------------ Open channels on device/job -----------------------;
        MOV     #INIERR,R2              ;Default error message.
        .QSET   #QUE,#1                 ;1 extra element.
GETFIL: .ENTER  #AREA,#0,#FILE,#0       ;CH# 0 = SY:PEMLOG.TMP
        BCS     FATAL
        .LOOKUP #AREA,#1,#JBBLK         ;CH# 1 = MESSAGE CHANNEL
        BCC     JOBOK
FATAL:  .PRINT  R2
        .EXIT
JOBOK:  .READW  #AREA,#1,#MESS,#1       ;Wait for start message.
        BCS     FATAL
        .READC  #AREA,#1,#MESS,#1,#STOP ;Look out for stop message.
        BCS     FATAL
        MOV     #NOST,R2                ;R2 -> Error message
        .GVAL   #AREA,#SYSOP            ;Get monitor sysgen options
        BIT     #STASK$,R0              ;System task support present?
        BEQ     FATAL                   ;If eq, no - can't run
        CLRB    FLAG                    ;Reset STOP flag
        .GTIM   #AREA,#TSTART           ;Get current time.
        .WRITW  #AREA,#0,#BUFFER,#256.,#0 ;Write header block.
        BCS     3$
        JMP     RUN                     ;OK jump to
3$:     JMP     IOERR                   ;       RUN code!

FILE:   .RAD50  /SY PEMLOGTMP/          ;LOG-FILE
JBBLK:  .RAD50  /MQ/
        .WORD   0,0,0,0                 ;Accept message from everyone.
```



```
INIT1:  .ASCIZ  <11><11>/PEMLOG: Wait 10 s.!/
INIT2:  .ASCII  <11><11>/PEMLOG: Init O.K.!/<015><012><012>
        .ASCIZ  <11>/R PEMDON to start, R PEMDON again to stop!/
INIERR: .ASCIZ  /?PEMLOG-F-Init error/
NOST:   .ASCIZ  /?PEMLOG-F-No system job support/
NONUL:  .ASCIZ  /?PEMLOG-F-No null count value/
        .EVEN
TIMINI: .WORD   0,200.                  ;4 sec. interval.
;       ------ Up to here, code may not exceed 500 bytes ------
        .=BUFFER+500.                   ;Set near end of buffer area.
REFIDL: .WORD   0                       ;Reference CPU idle value
TSTART: .WORD   0,0                     ;Start time PEMLOG
        .WORD   NULCNT                  ;Null count monitor offset.
        .WORD   TIMEP                   ;Measurement timeslice
BUFEND: .WORD   0                       ;Last word of buffer/startcode

;--------------End of initializing code and buffer area-----------------;
        .PSECT RUNPEM
RUN::
        CLR     R4                      ;Set block nr.
NEWBUF: MOV     #BUFFER+2,R3            ;Pointer for null-count values.
        CLR     R2                      ;Nr. values.
NEXTS:  MOV     NULCNT(R1),R5           ;Get start val.
        .TWAIT  #AREA,#TIME             ;Wait some time.
        MOV     NULCNT(R1),(R3)         ;Get new idle value
        SUB     R5,(R3)+                ;Difference
        INC     R2                      ;Count the values
        TSTB    FLAG                    ;Can we go on?
        BNE     1$
        CMP     R3,#BUFEND
        BLOS    NEXTS
1$:     MOV     R2,BUFFER               ;Say how many there are in.
        INC     R4                      ;Next block.
        .WRITW  #AREA,#0,#BUFFER,#256.,R4
        BCS     IOERR
        TSTB    FLAG                    ;Can we go on?
        BEQ     NEWBUF
EXIT:   .READW  #AREA,#0,#BUFFER,#256.,#0
        .GTIM   #AREA,#BUFFER           ;Store END time.
        .WRITW  #AREA,#0,#BUFFER,#256.,#0
        .CLOSE  #0                      ;Close LOG-FILE.
        .CLOSE  #1                      ;Close message channel.
        .EXIT

IOERR:  .PRINT  #HRDERR
        BR      EXIT
STOP:   MOVB    #1,FLAG
        RETURN
;--------------------------- Storage ------------------------------;
AREA:   .BLKW   5
QUE:    .BLKW   10.
TIME:   .WORD   0
TICKS:  .WORD   TIMEP                   ;Once a TIMEP/50. second
MESS:   .WORD   0,0
FLAG:   .BYTE   0
HRDERR: .ASCIZ  /?PEMLOG-F-IO error/
        .EVEN
        .END    START
```

```
C
C  PEMLOG.FOR Read out of file SY:PEMLOG.TMP, H.H. NKG-AZG Jun-82
C
       DIMENSION IBUF(256)
       BYTE FILNAM(12),SUCCES
       DATA FILNAM/'S','Y',' ','P','E','M','L','O','G','T','M','P'/
C
       CALL GETFIL(FILNAM,ICHAN,NBLOK,SUCCES) !Allocate channel, .LOOKUP
C
       IF (SUCCES) GOTO 10
       STOP 'File NOT found'
10     IST=IREADW(256,IBUF,0,ICHAN)            !READ HEADER
       IF(IST.NE.256) WRITE(7,100)IST
100    FORMAT(' READ ERROR, STATUS:' ,I5)
       REF=IBUF(251)
       REF=REF/100.
       NULCNT=IBUF(254)
       SLICE=IBUF(255)
       WRITE(7,110)IBUF(251),NULCNT,SLICE/50.
110    FORMAT(' *** PEMLOG ***'/' Ref. val.=',I6,' Monitor
       1 offset=',O4,' Timeslice(s.):',F7.2/)
C
C START/END/EXECUTION TIME:
C
       CALL CVTTIM(IBUF(1),IHRSE,IMINE,ISECE,ITICKE)
       CALL JJCVT(IBUF(1))
       CALL JJCVT(IBUF(252))
       CALL JSUB(IBUF(1),IBUF(252),IBUF(1))
       CALL JJCVT(IBUF(1))
       CALL JJCVT(IBUF(252))
       CALL CVTTIM(IBUF(252),IHRSS,IMINS,ISECS,ITICKS)
       CALL CVTTIM(IBUF(1),IHRS,IMIN,ISEC,ITICK)
       WRITE(7,101)IHRSE,IMINE,ISECE,ITICKE
       WRITE(7,102)IHRSS,IMINS,ISECS,ITICKS
       WRITE(7,103)IHRS,IMIN,ISEC,ITICK
101    FORMAT(' Current time (H-M-S-T):',4I3)
102    FORMAT(' Start   time (H-M-S-T):',4I3)
103    FORMAT(' ----------------------------------'
       1/' Execution    (H-M-S-T):',4I3/)
C
C  Read DATA words
C
       DO 150 IREC=1,NBLOK-1
       IST=IREADW(256,IBUF,IREC,ICHAN)
       IF (IST.NE.256) WRITE(7,100)IST
       NVAL=IBUF(1)+1
       DO 150 J=2,NVAL
       VAL=100.-FLOAT(IBUF(J))/REF
       WRITE(7,140)(IREC-1)*255+J-1,VAL,IBUF(J)
140    FORMAT(' Time slice nr.',I6,' CPU usage %:',F7.1,'(',I5,')')
150    CONTINUE
C
       CALL CLOSEC(ICHAN)
       CALL IFREEC(ICHAN)
       STOP 'PEMLOG'
       END
```


USER RESPONSES

1. In response to the user request about an alternative for RTFILE I would suggest MUMPS available from the University of Tennessee or FM-11 from MC2 Corporation(201 West Pine Street, Rome, New York). If anyone has already experience with these packages, would she or he report? Also DECUS Proceedings mention user developed data base software. (For acquiring MUMPS binairies on floppies: send a $25,= cheque to:
   Kevin C. O'Kane
   Rt. 4 Box 634
   Louisville, Tennessee 37777, USA
   and specify RX01 or RX02 format.)

2. The RM02 handler is also available from a Dutch firm for about $300,=. Contact me if necessary.

H.T.M. Haenen
Dept. Clinical Neurology
University Hospital Groningen
P.O. Box 30.001
9700 RB GRONINGEN
The Netherlands

WISH LISTS

European RT-11 SIG Wishlist

This wish list is that generated by the European RT11 SIG during the 1982 European symposium held 7-10 th. September in Warwick University, UK. The responses are those given by Jim Williams (RT11 developer) during the feed-back session.

1.

Request: RE-ENTER to KED after crash or ^C.

Response: Maybe

2.

Request: Prevent stack pointer overflowing vectors.

Response: Consider

3.

Request: Create and publish fixed offset for clock ticks to save EMT's.

Response: Consider

4.

Request: LIBR to list .IDENT from all modules and show library file name (a la TKB).

Response: Think about.

5.

Request: KMON to pass command to FG job as in "FRUN PROG COMMAND".

Response: Maybe in V6

6.

Request: Improve speed of RENAME/PROTECT

Response: Look at.

7.

Request: SQUEEZE/BAD

Response: Consider

8.

Request: For a driver loaded for FG use only, SHOW DEV is incorrect.

Response: Bug, maybe already be fixed.

9.

Request: DELETE/NEW

Response: V5

10.

Request: SHOW JOB shows FG as running (?) and DONE but can't unload.

Response: Bug in RESORC.

11.

Request: DIR to show tentative files, i.e. DIR/TENT.

Response: Look at.

12.

Request: Improve wild card RENAME capabilty (e.g. A%% --> B%%) and only specify device name once.

Response: Try

13.

Request: COPY for special directory devices ignores file length (PIP always uses 0)

Response: We'll check

14.

Request: COPY/SINCE /BEFORE etc.

Response: V5

15.

Request: Lower case to CSI.

Response: We'll check.

16.

Request: Link to AST for .SCCA

Response: We have severe problems here.

17.

Request: Virtual CUSPS for XM.

Response: Maybe some (in V5).

18.

Request: FORTRAN 4+/77

Response: Looking into.

19.

Request: FORTRAN listings to contain file name.

Response: See FORTRAN group.

20.

Request: PASCAL for RT.

Response: See F77 question.

21.

Request: Last line editing for VT11/GT41.

Response: Not on VT/GT screen.

22.

Request: IND should load drivers.

Response: Try.

23.

Request: V5 MOUNT should load driver.

Response: Consider

24.

Request: 2 QUEUE's

Response: Look at.

25.

Request: Transparent spooling

Response: Maybe

26.

Request: Programs which produce listings (MACRO, LINK, FORTRAN etc.) to use QUEUE if available.

Response: Maybe, problem would be solved by transparent spooling.

27.

Request: Wild card unit numbers and/or devices.

Response: Unlikely.

28.

Request: ASSIGN logical device to another logical device (e.g. ASS SY: ABC:)

Response: Look at.

29.

Request: IND to use channel 17(8)

Response: Check.

30.

Request: Check that MFPT (Move From Processor Type) is memory size independent on an 11/44.

Response: We'll check.

31.

Request: RSX style IND for RT11.

Response: Yes, in V5

32.

Request: Console logging control by single key (e.g. ^T)

Response: Look at.

33.

Request: Software write lock for mag-tapes.

Response: Consider.

34.

Request: Removed from list.

Response:

35.

Request: Removed from list.

Response:

36.

Request: TU58 directory caching.

Response: This is very difficult and potentially a can of worms!

37.

Request: MAKE and MUNG on SJ monitor.

Response: No.

38.

Request: DIR and INIT to default to /VOL.

Response: Maybe.

39.

Request: Set date and time from within a program.

Response: Yes, use DATTIM.SAV in your startup file

40.

Request: Replace KED by EDT.

Response: No.

41.

Request: Improve KED to use better features of EDT in video mode.

Response: Maybe.

42.

Request: Someway to handle extra directory words.

Response: Already possible, see MINI-TASKER.

43.

Request: Improve FILEX to make it useful, i.e. allow it to deal with modern devices.

Response: No plans but a good idea.

44.

Request: Documentation to always refer to EXTENDED memory not Virtual, since with the advent of the VAX, virtual means something different.

Response: Look at.

45.

Request: Change date of a file by program.

Response: Solved.

46.

Request: User input when using .COM files.

Response: Use IND in V5.

47.

Request: VM: to be supported.

Response: Yes, in V5.

48.

Request: Support FB TT: features in SJ.

Response: Probably not.

49.

Request: Abilty to specify unit numbers for TT: (e.g. COPY FOO.BAR to TT3:)

Response: Look at but a problem.

50.

Request: Patching programs to update file date.

Response: V5.

51.

Request: E (=examine) command to high memory.

Response: Can do, feature patch in SYSGEN or Software Bulletin.

52.

Request: Ability to check if USR in memory without calling it.

Response: Probably can do already.

53.

Request: KED display to give full 80 character lines.

Response: Problem with cursor position, maybe 79.

54.

Request: Address watchdog for ODT.

Response: Maybe.

55.

Request: Symbolic debugger (Please!!)

Response: Maybe.

56.

Request: LINK to identify module(s) referencing undefined globals.

Response: Linker cross-reference, in V5.

57.

Request: Removed -- Programmer error!

Response:

58.

Request: Increased functionality in SYSLIB routines.

Response: Some in V5.

59.

Request: Allow jobs to access other jobs' devices.

Response: No.

60.

Request: CSI to handle * and %.

Response: It does!!

61.

Request: Volume changed flag.

Response: No, because no hardware support.

62.

Request: VM: to run with low-memory to 30k.

Response: No, but maybe get it to check and refuse to run.

63.

Request: Add parity memory support to VM: (report as bad blocks?)

Response: Maybe.

64.

Request: Use Supervisor and User and Kernel MMU modes.

Response: No, but you could do it yourself under SJ.

65.

Request: Removed from list.

Response:

66.

Request: Set magtape characteristics by unit number.

Response: Look at, but maybe hardware problems.

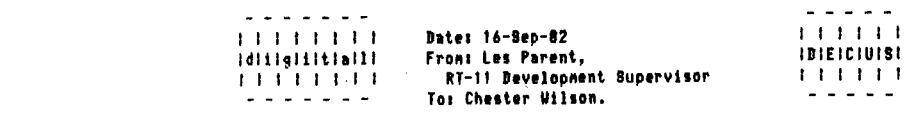
67.

Request: INIT/BAD:ONLY to write FILE.BAD's in <empty> areas only.

Response: Good idea, look at it.

Items saying removed from list were either erroneous, irrelevant or forgotten due to bad note taking. Such entries are entirely the responsibilty of the SIG chairman and are not due to any influence by Digital. Their inclusion does however, allow the request numbers to be kept as they were during the feed-back session.

RT-11 Wish List Australian DECUS 1982
with replies from Les Parent



Greetings. How goes life "down under"? What follows are responses to your latest Wish List, given to me by Bill Andrus a few days ago. I also received another one from you some time ago, and I trust you'll accept my apologies for not responding to it; V5 has taken over my life, and I just haven't had the time for much of anything else! Here goes ...

FILEX:
---

1. If a file be written to an interchange format discette using filex, and the file to be written is the exact lenth of the remaining space on the discette, gerbage is written into spare directory segments on track 0. This does not affect reading the discette under RT-11, but fouls on IBM systems.

Reply: **Sounds like a bug - we'll fix it in V5.**

LINK/LIBR:
---

2. LINK should be able to provide symbol tables for debugging programmes, INCLUDING sets of local symbols. This should be supported by Fortran and Macro also (currently only Bob Denny's "C" provides local symbols in the .OBJ files).

Reply: **Huh?**

Retort: **Someone in the TOPS-10 group sit him down for a couple of hours with BDT and its beautiful handling of local symbols please.**

3. Symbol cross-reference ability within LIBR for providing details of globals defined within and referenced by modules comprising library files.

Reply: **Extract the modules and use LINK to produce global symbol x-ref (in V5).**

4. Global Patching facility (cf RSX GBLPAT) in LINK.

Reply: **Nice idea - good midnight project, but no official plans.**

5. Switch to suppress use of SY:SYSLIB.OBJ for debug use.

Reply: **Another good idea, worth consideration, but no promises for V5.**

OTHER UTILITIES:
---

6. .DIRECTORY dev1:,dev2:,dev3: (no reason for just 1 device per string!).

Reply: **Nope, sorry.**

7. Non-overlaid version of PIP for use with long command files on floppy discs.

Reply: **A boon to floppy users indeed, but no plans. Buy the sources and do it yourself!**

Retort: **All Golden Casket tickets gratefully accepted ...**

8. SYSLIB support for wildcards.

Reply: No plans, but we do supply ULBLIB.MAC with the sources, in which there is a wildcard matching subroutine.

9. Why does DIR read the home block even if /VOL is not requested? Makes life difficult with discs with a bad home block.

Reply: To see if it's an RT-11 disc - of course!

Retort: You care?

KMON:
---- (ah, a subject near and dear to your heart, eh?)

10. Allow customization of command options (eg DIR/VOL default, DIR/ALPH, DEL/NOQ etc) - ? by a utility.

Reply: Nice idea - good midnight project, but no official plans.

11. Ability for a programme to specify whether or not it wants lines passed to it from KMON to be translated to CSI format.

Reply: V5 via User Command Linkage (UCL).

12. Ability to know whether the next line to be read by the programme will come from a command file or from the user's terminal. (Currently INDFL pointers do not know that the end of a command file has been reached until the request for the next line is being processed.).

Reply: We've nearly totally re-written the @file processor in V5 to support chain .EXITs from command and IND control files. No promises that INDFL status will be any different, but it may!?

13. ".R filename string" similar to ".RUN filename string". RMON should be smart enough to pass this on to KMON if it can't deal with it itself.

Reply: Maybe in V5 - may fall out of merging R/RUN and F/Srun code to RUN virtual jobs from other than SY:.

14. .EXIT with a list of commands to KMON currently obliterates any operating command files. Can it optionally merely increase the nesting depth of these? (The vote here was about 50/50, some finding it useful as it is, others finding it a considerable nuisance!).

Reply: See 12 above!

15. ".ASSIGN logical logical" as well as ".ASSIGN physical logical". The assignment should use the present physical equivalent of the first logical device, and need not change should this be later reassigned. (See TSX).

Reply: V5

16. EXECUTE should be able to cope with .OBJ files if sources not present. Search orders for extensions under EXECUTE should be rationalized.

Reply: No plans to change the "intelligence" (or lack thereof, you say?) of EXECUTE. It's a can of worms ...

17. SET CLOCK 50 and SET CLOCK 60

Reply: No plans.

RMON:
----

18. Ability to wake up a job when terminal input is available (eg run a coroutine when either a line or a pre-defined character is typed).

Reply: Nope ... never ... to many context problems, especially in XM!!!

Retort: Keep working on 'em, lads! Even if it's just a .READC from TT: with a zero word count!

19. Multi-terminal support allowing access in the form TT0:-TT7:.

Reply: No!

Retort: Any more donations of Golden Casket tickets please?

20. .CSTAT under SJ monitor.

Reply: Done (finally) in V5.

21. Ability to extend SET command to use same mechanism for modifications to SAV files (? if .SYS file not found). Eg SET KEB WRAP=75 or equivalent in user's programmes.

Reply: See 4 above ...

22. Extended time & date routines and commands so as to get date and time expanded and all together a la RSX.

Reply: No plans, since we already support individual calls to get date and time.

FILE STRUCTURE:
---

23. Optional creation time in directory.

Reply: Do it yourself with extra word (see 24).

24. Standardized supported method for accessing additional words in a directory (V4 technique probably adequate).

Reply: It'll stay as is, though it is a bloody [the original from Les spelt this "bloddy". Sorry mate - your Australian visit's wearing off. Just have to come back again, won't yer?] pain that the utilities don't support it. You can always COPY/DEVICE though.

Retort: Some utilities are being written which can cope with these.

25. Bytes within home block reserved for users under RT-11 (preferably about 100 please).

Reply: Can't promise you that we won't want to use them ourselves (we took some more in V5 for BUP (Backup Utility Programme). Sorry, you'll have to take your chances!

26. A couple of bits within the file status word within directories reserved for users (eg archiving bit or other purposes).

Reply: See 25.

GENERAL:
---

27. Support for RT-11 in the year 2000.

Reply: Do you mean DEC's support or that the time won't break in 2000 A.D.?

Retort: Yes.

28. RT-11 on PC300 series.

Reply: No "current" plans, but one never knows, does one!?

29. Utilities to transfer data from RT to PC (CX, DX, Radial-serial or something similar).

Reply: See 28, but one goes with the other, obviously.

- o 0 o -

And now, for the RT-11 Development Group's Wish List:

1. SET USER FRIENDLY

2. SET USR LARGE

3. SET MONITOR NOBUGS

4. SET SPR NONE

5. SET KNOW READABLE

6. SET TT NOTYPO

7. SET SY NOCRASH

8. SET DD MILLPOND (obtuse reference - the Mill Pond here in Maynard is to where all Software Engineers condemn less than desireable hardware ...)

9. SET PAYCHECK LARGER

So, you see we're all human after all ...

Regards,

Les.

DECUS

DIGITAL EQUIPMENT COMPUTER USERS SOCIETY
ONE IRON WAY, MRO2-1/C11
MARLBORO, MASSACHUSETTS 01752

BULK RATE
U.S. POSTAGE
PAID
PERMIT NO. 129
NORTHBORO, MA
01532

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
One Iron Way, MRO2-1/C11
Marlboro, Massachusetts 01752 USA

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