THE mini-tasker

DECUS RT-11 SIG NEWSLETTER

JUNE 1980 VOL. 6 NO. 3

Contributions to the newsletter should be sent to:

Ken Demers
MS-48
United Technologies Research Center
East Hartford,Connecticut 06108
203 727-7241

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

---

RT-11 SIG STEERING COMMITTEE

---

| | |
|---|---|
| John T. Rasted<br>JTR Associates<br>58 Rasted Lane<br>Meriden, Ct 06450<br>(203) 634-1632 | RT-11 SIG Chairman |
| Ken Demers<br>MS-48<br>United Technologies Research Center<br>Silver Lane<br>East Hartford, Ct 06108<br>(203) 727-7241 | Newsletter Editor<br>DECNET Contact |
| Mark Bartelt<br>California Inst. of Tech.<br>MS 356-48<br>1201 California<br>Pasadena, Ca 91125<br>(213) 795-6811 EXT 2663 | RT-11 LUG Coordinator<br>RT/RSX Contact<br>West Coast<br>161-2662 |
| Nick Bourgeois / 1736<br>Sandia Laboratories<br>P.O.Box 5800<br>Albuquerque, NM 87185<br>(505) 844-8088 | Tape Copy Coordinator<br>MACRO Contact |
| James R. Cutler<br>Space Physics Research Lab<br>2455 Hayward<br>Ann Arbor, Mich 48105<br>(313) 763-9940 | Standards Coordinator<br>RT/RSX Contact<br>Midwest |
| Art Hermes<br>MIT/LNS Bates Linear Accelerator<br>P. O. Box 95<br>Middleton, Ma 01949<br>(617) 245-6600 | RT/RSX Contact<br>East Coast |
| Shirley Hooper<br>Applied Research Laboratories<br>9545 Wentworth Street<br>Sunland, Ca 91040<br>(213) 352-6011 | RT LUG Library<br>Coordinator |
| Carl Lowenstein<br>University of Calif, San Diego<br>Marine Physical Laboratory of the<br>Scripps Institution of Oceanography<br>San Diego, Ca 92152<br>(714) 294-3678 | RT-11 HHK Coordinator<br>SIG Hardware Contact |
| Eric Morton<br>Sanders Tech. Inc.<br>P.O.Box 1226<br>Nashua, NH 03061<br>(603) 882-1000 EXT 253 | RT-11 DECUS Library<br>Coordinator<br>C Contact |
| Susan S. Rasted<br>Software Dynamics Inc.<br>1000 Yale Ave.<br>Wallingford, Ct 06492<br>(203) 265-2226 | FMS-11 Contact |
| Marilyn Runyon<br>39 Locust Point Rd.<br>Locust, N.J. 07760 | Wish List |
| Edwin J. Stevens<br>EMDA Inc.<br>899 East Green St.<br>Pasadena, Ca 91106<br>(213) 795-5991 | MU-BASIC Contact |
| Ray Strackbein<br>Chalfront Communications<br>73-680 Highway 111<br>Palm Desert, CA 92260<br>(714) 346-3987 | Symposia Coordinator |
| J. W. Tippie<br>Argonne National Labs<br>9700 S. Cass Ave<br>Argonne, Ill 60439 | CAMAC Contact |

©1980, DECUS
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.

Ron Trellue
Division 1523
Sandia Labs
Albuquerque, NM 87185
(505) 844-2115

FORTRAN Contact

Greg Woodbury
2213 Elder Street #2
Woodbury-Wolfe Prog. Ltd.
Durham, N.C. 27705
(919) 286-3578

BASIC Contact

---

USER INPUT

---

# A short history of RT-11

Ian Hammond
HAMMOND-software, Am Feldborn 22
D-34 Goettingen, West Germany

RT-11 is almost seven years old now and in 'computer time' thats enough to warrant at least a short history.

After some general data the RT-11 origins are discussed and a detailed rundown of RT-11 V1 is given. V2 and V3 are presented as enhancements to V1. Separate sections deal with program requests, device handlers, support/documentation and RT-11 layered products. A general summary and a look at the future conclude the paper.

The presentation is not always strictly chronological. For example, the discussion of the V1 monitor structure includes features that were not introduced until V2 or V3. There are some references to V4 which appeared just before the final draft.

The focus is on the RT-11 monitor components - and the options, such as FORTRAN or BASIC, are mostly ignored. These options are independent of both the RT-11 architecture and development group.

It is assumed that the reader has a working knowledge of RT-11. All memory sizes are given in kilowords rather than in kilobytes. Thus 10k means '10 kilowords' and not '10 kilobytes'.

This 'history' is based on old documentation, listings, recollections, rumours, conversations and wild guesses; I must confess that the history lapses into personal opinion at times. Clearly, there will be major errors and omissions. If readers would care to write me corrections then perhaps I could present a 'revised' version later.

I would like to thank Frank Tolkmitt, Howard Schultens and Dieter Michael for their comments and corrections to the first draft.



- General data

The following version release dates are derived from the printing dates given in the RT-11 system reference guides and system release notes.

| | | |
|---|---|---|
| V1 September 1973 | | |
| V2 October 1974 | V2B June 1975 | V2C January 1976 |
| V3 October 1977 | V3B March 1978 | |
| V4 June 1980 | | |

The RT-11 development group varies between five and eight people. The individual members remain anonymous in the sources and authors are only identified by their initials. Some members of the development group appear to have remained with RT-11 since Version one.

There are over 15,000 RT-11 systems in the field - at one point RT-11 was DEC's fastest selling operating system. RT-11 is popular in many areas and is often used as a base for layered products both by DEC (e.g. MINC) and other manufacturers.

- RT-11 origins

DOS, the first PDP-11 system, lacked balance. It was too complex for a small system but too limited for a large multi-user system. RT-11 was written to replace DOS and provide a fast, simple single-user system. The following table shows some of the simplifications achieved by RT-11.

| | DOS/BATCH | RT-11 |
|---|---|---|
| Directories per device | multiple | single |
| Directory operations | init/open close/release | open close |
| Steps to run program | login and run | run |
| File specification | device,name,type,user-id | device,name,type |
| File structure | linked and contiguous | contiguous only |
| Overlay call | special program request | user transparent |
| Program format | LDA and core image library | image |
| Program requests | enhanced set | essential set |
| CSI decoding | many operations | single operation |

CAPS-11, the DEC cassette operating system, was simply an enhancement of the standard PDP-11 paper tape software. CAPS-11 and RT-11 were developed at the same time and shared some source code (such as V1 EDIT). When Floppy diskettes replaced cassettes as the low-cost system device, support for CAPS-11 fell away. Many CAPS-11 users moved to floppy-based RT-11 systems.

Since the release of RT-11 both DOS/BATCH and CAPS-11 have been de-emphasised by DEC. In the 1976 'PDP-11 Software Handbook', which presents a summary of PDP-11 operating systems, there is no mention of DOS. In the 1978-79 issue CAPS-11 is no longer listed.

The original version of RSX-11 developed out of DOS but with RSX-11D the RSX-11 architecture was redefined. RSX-11 provided the names and models for some RT-11 V2 and V3 program requests:

| | | |
|---|---|---|
| General | V2 | GTIM, SFPA |
| Timer support | V2 FB | CMKT, MRKT |
| Foreground/Background | V2 FB | RCVD, RSUM, SDAT, SPND |
| Extended memory | V3 XM | ASTX, CRAW, CRRG, ELAW, GCMX, MAP, UNMAP |

The true predecessor of RT-11 was the PDP-8 single-user system OS-8. In the early days of RT-11 you could predict the way RT-11 would develop by looking at the development history of OS-8. OS-8 is still alive and has a vigorous SIG and newsletter ('The 12 bit SIG newsletter'). Some of the OS-8 development group moved to the original RT-11 development group. Indeed, the RT-11 V1 sources contained PDP-8 emulating macros (such as 'AND x,y').

OS-8 provided the model for many RT-11 structures. The monitor layout of KMON/USR/CSI/RMON/handlers comes from OS-8. 'SAVE' files, the JSW and the RT-11 file structure all originated with OS-8. The KMON commands for OS-8 are: ASSIGN, DEASSIGN, GET, SAVE, ODT, RUN, R, START, DATE. The OS-8 extended console command language included commands such as BOOT, COMPILE, COPY, DIRECTORY, EXECUTE etc.

Basing RT-11 on an established system made sense. OS/8 had shaken down the basic structures of the operating system and version one of RT-11 was already a mature and balanced system.

- Version one

In September 1973 DEC released V1 of RT-11. I remember hearing about RT-11 just as we were about to purchase DOS/BATCH. I didnt believe that RT-11 would deliver and I was wary of 'Version one' software, so we purchased both DOS and RT-11. They arrived together. RT-11 was up and running an hour later and we never even tried to get DOS up. However some Europeans tell me they never got V1 to run. There was no 'B' release for V1, but some components did arrive later, like the software support manual, and some initial VT-11 support.

V1 was certainly a minimal operating system. For example the list of useful V1 utilities was: EDIT, PIP, MACRO, LINK, PATCH and ODT and a single 375 page reference manual described the whole system. But V1's minimality implied efficiency rather than insufficiency - it was like a fast, no-frills sportscar.

RT-11 V1 system components.

| | |
|---|---|
| Device handlers | SY, DK, DT, TT, LP, PP, PR, RK |
| Monitor | Single-user, Single-job monitor (MONITR.SYS) |
| Program requests | DATE, CLOSE, CSIGEN, CSISPC, DELETE, DSTATUS<br>ENTER, EXIT, FETCH, HRESET, LOCK, LOOKUP, PRINT<br>QSET, RCTRLO, READ, READC, READW, RELEAS, RENAME<br>REOPEN, SAVESTATUS, SETTOP, SRESET, TTYIN/TTINR<br>TTYOUT/TTOUTR, UNLOCK, WAIT, WRITE, WRITC, WRITW |
| Program format | Memory image 'save' file, '.SAV'. |
| Console commands | DATE, INIT, ASSIGN, CLOSE, GET, EXAMINE<br>BASE, SAVE, RUN, R, START, REENTER |
| Error messages | Brief and mystic: FIL NOT FND. |
| Utilities | EDIT, PIP, ODT, PATCH, PIPC, MBUILD |
| Language processors | MACRO, EXPAND, ASEMBL, LINK |
| Documentation | 'System reference manual' (375 pages).<br>Software support manual in November 1973. |
| Options | BASIC, FORTRAN (following year). |

RT-11's basic monitor structure has remained much the same since V1. The V1 monitor layout and some V2/V3 extensions are shown below:



| | |
|---|---|
| | System device handler (SY:) |
| RMON | Resident monitor (V1 RMON=1.3k, V4 BL=1.6k FB=4k)<br>I/O queueing/completion routines<br>EMT dispatcher<br>CSW and RT-11 tables |
| | [ V2: VT-11 scroller]<br>[ V2: foreground job (FG)]<br>[ V3: Resident handlers ]<br>[ V3: command files] |
| USR | User Service Routines (USR=2k)<br>CSI processing<br>Directory operations (LOOKUP etc.) |
| KMON | Keyboard Monitor (KMON V1=1.5k, V3=3.75k)<br>[ V2: KMON overlay (KMOVLY=.5k) ] |
| BSTRAP | Bootstrap (BSTRAP V1=.5k, V3=1.5k) |

- RT-11 Monitor layout

Because the DOS monitor resided in low memory, programs had to be relocated to run above the monitor, which forced relinking of programs for different monitor configuration sizes. RT-11 solved this problem by relocating the monitor to the top of memory, which allowed RT-11 programs to be written and run with an zero origin. This feature allows standard RT-11 programs to run on any RT-11 configuration without relinking.

The RT-11 bootstrap first loads the monitor into the low 8k of a machine and then relocates it to the top of memory. This means that RT-11 systems do not require a systems generation session to install - RT-11 is bootstrappable as delivered. This feature, and other factors, forced the monitor to be written in quasi-position-independent code.

Since memory was expensive in 1973 a goal of RT-11 was to provide a good service in 8k (OS-8 has a minumum of 8k also). The 1.3k word resident monitor (RMON) was supplemented by the 2k 'swapping monitor' (USR) for file oriented operations (lookups, enters etc.) and KMON for console command support.

The keyboard monitor (KMON) runs contiguous with the USR. Now, KMON could have been implemented simply as a 'special' RT-11 program. The reasons why it was not so implemented do a lot to explain RT-11's layout: RT-11, and OS-8, were designed to give a good performance on Dectape. This meant minimising the amount of Dectape activity required by the monitor. Relocating KMON as high as possible in memory permits KMON to stay in memory across the execution of small programs, and thus reduces i/o. The V1 KMON was less than 2k and this was not difficult.

If RMON is known as the 'resident' monitor and the USR as the 'swapping' monitor then KMON can be called the 'sliding' monitor. With V2 and V3 it became necessary to keep some secondary monitor components (handlers, scroller etc.) in memory across program runs. An understanding of the 'sliding' mechanism involved in allocating and reclaiming memory space for these components is crucial to an understanding of the RT-11 implementation.

The reason for the sliding can be explained by considering the KMON LOAD command. Since a program may cause both KMON and the USR to be swapped out of memory, the only place for the handler to go is in between the USR and

RMON (see the figure below). KMON makes space for a handler by first sliding both itself and the USR down in memory. When a handler is UNLOADed, KMON releases the handler and slides itself and the USR back up again. The 'slide' routines are located at the beginning of KMON and at the end of the USR. The following figure illustrates the sliding mechanism.

```
   (1) Before loading        (2) After loading           (3) During program

       RMON                      RMON                             RMON
       USR  (slide USR/KMON)   handler                          handler
       KMON                      USR  (swap-out USR/KMON)       program
                                 KMON
```

Note that life can become complicated in a sliding environment. For example, if KMON slides while in a subroutine then the subroutine(s) have to relocate their return path.

Another distinct peculiarity is that the KMON console line buffer goes backwards. This takes advantage of the RT-11 auto-decrement addressing mode (e.g. -(RØ)) but one wonders if the few words saved have been worth the headaches it probably caused.

- V1 file structure

The file structure defined by V1 has survived all versions and presented no problems, other than those inherent in a contiguous file structure. RT-11's contiguous file structure is fast and simple and recovering files from a crashed disk with a contiguous file structure is much simpler than doing it with a linked file structure.

RT-11 keeps a copy of the most-recent directory segment in USR buffer. This causes problems if you change disk cartridges and RT-11's still got a copy of the old directory. Later versions of RT-11 solved this problem partially by ignoring the current-segment for .ENTER operations, which at least stopped destroying disks. Also, RT-11 will crash if you try to perform a directory operation on media that hasnt been INITIALIZEd.

- V1 program image format

OS-8 language processors produce formatted binary (LDA) output files and the OS-8 user does 'linking' manually: First, the 'Absolute loader' is used to load the required LDA program segments into memory. Secondly the console 'SAVE' command is used to store the memory image on disk.

The RT-11 linker performs the loading and saving steps for the RT-11 user but the OS-8 file type '.SAV' stuck. Apparently the implementors were not sure about this automatic linking and provided RT-11 with a SAVE command anyway.

- V1 Utilities

RT-11 V1 provided the bare minimum of utility programs. PIP is a standard utility on all DEC operating systems. RT-11 EDIT was weaker than the papertape or DOS editors. ODT, the debugger, was probably adapted from existing software. PATCH was supplied to enable the installation of DEC update patches.



MACRO was adapted to RT-11 and LINK was a completely new. It is not surprising that, being new, the linker had the most problems. One fudge with early versions of LINK were the error messages in the map file. LINK apparently ran out of room, and instead of using a separate buffer for map file error messages, each error message required a complete block, which was padded out with nulls. LINK was substantially rewritten in later releases.

A major advantage of RT-11 was the simplicity of the overlay scheme, which makes recompiling of programs in overlayed or non-overlayed forms simple. The decision to overlay or not to overlay is made at LINK time and the source code requires no special calls.

- Weird utilities

RT-11 is committed to compatibility between versions where possible. However each release of RT-11 has included a some weird utilities to perform special tasks. V1 provided a modified version of PIP for cassette operations called PIPC. Systems generation from cassette involved another utility called CBUILD, itself a further modified version of PIPC. EXPAND and ASEMBL provided a two stage MACRO assembler for 8k systems. GTON and GTDIAL, were provided (after the release of V1) to supply basic VT-11 scrolling support. These utilities were superseded by the monitor 'GT' command in V2.

- Version two

V2 followed V1 by little more than a year and fleshed out the bare bones of V1. The main features of version 2 were a fuller choice of commands and utilities. To use DEC terminology, RT-11 V2 had a lot more 'functionality': CREF, LIBR, FILEX, SRCCOM, DUMP, BATCH, SYSLIB, timer support and foreground support.

Foreground/Background (FB) support is a standard extension to single user systems and most of the new commands and program requests in V2 were for the FB monitor. With V1 there were a lot of errors that would unconditionally trap a program back to the monitor. V2 allowed for more program control with the .SERR, .HERR, .SFPA, .TRPSET, and .CHAIN requests.

New V2 system components.

| | | |
|---|---|---|
| Device handlers | | CR, MT, RF, VT-11 scrolling, clock, SET. |
| Monitor | | Foreground/Background monitor (FB) |
| | | MONITR.SYS renamed to RKMNSJ.SYS or RKMNFB.SYS |
| Program requests | | CDFN, CHAIN, GTIM, GTJB, HERR, SERR, PURGE |
| | | SFPA, SPFUN, TRPSET |
| (non-EMT) | | ..V2.., REGDEF, INTEN, SYNCH |
| | FB | CHCOPY, RCVD, RCVDC, RCDW, SDAT, SDATC, SDATW |
| | FB | CMKT, CNTXSW, CSTAT, DEVICE, MRKT, MWAIT |
| | FB | PROTECT, RSUM, SPND, TLOCK, TWAIT |
| Program format | | FG 'relocatable file', '.REL' |
| Console commands | | GT, TIME, LOAD/UNLOAD, SET, FRUN, SUSPEND, RSUME |
| Utilities | | CREF, LIBR, FILEX, SRCCOM, DUMP, PATCHO |
| | V2B | BATCH, SYSLIB |
| Documentation | | A larger system reference guide. |
| | V2C | Reorganised documentation set, new cover design. |

Incidently, it was only during the period covered by V2/V2B that DEC listed RT-11 as a trademark in the reference manuals. By way of contrast RSX and OS-8 have always been listed as trademarks.

V1 was, like most operating systems, a fairly sober affair. The spirit of V2 is well illustrated by the quotations that comment many FB monitor routines. Heres a short sampler:

Just before the code that slides KMON/USR around:
; "The awful shadow of some unseen power floats, tho' unseen, among us"
;   - Shelly, "Hymn to intellectual beauty"

Just before the call to the fatal error halt:
; "Extreme remedies are very appropriate for extreme diseases"
;   - Hippocrates, "Aphorisms"

At the fatal halt code itself:
; "The death of God left the angels in a strange position."
;   - Barthelme, 'On angels'

Just before the .READ routine:
; "I'm quite illiterate, but I read a lot."
;   - J. D. Salinger, "Catcher in the rye"

Just before the .WRITE routine:
; "Their manner of writing is very peculiar, being neither from left to
; right like the Europeans; nor from the right to the left like the
; Arabians; nor up to down, like the Chinese; nor from down to up, like
; the Cascagians"
;   - J. Swift, "Gullivers Travels"

- V2 monitor growth

V2 fitted in fairly well with the original design of RT-11 and found few conflicts with the existing software. The FB implementation was only apparent in RMON - the USR and KMON were substantially the same for both SJ and FB. The size of the V1 RMON, now called the Single-job (SJ) monitor, increased by only 25 words.

The FB monitor is not only some 2k bigger than the SJ monitor it's also noticably slower for some i/o tasks (especially with floppies). Job arbitration, scheduling and swapping add substantial overhead to the FB monitor.

The FB monitor provided message requests (.SDAT/.RCVD) for BG/FG communications with a psuedo-handler in RMON. V2 established a habit of pushing all new functionality into the monitor. Messages could have been handled by a separate handler, which would have simplified the monitor and allowed users who didnt require the message support to save the space involved. The FB monitor also moved the console handler (TT:) into RMON.

The new console commands and the FB monitor caused expansion of KMON which was handled by providing KMON with a 2 block overlay area 'KMOVLY' (V2 used only a 1 block area for SJ). The KMON overlay is not implemented using the standard LINK overlay facility and requires special calling formats between the KMON root section and KMOVLY.

- RT-11 exception handling

One weakness of RT-11 is the inconsistancy of exception handling. This table shows the exceptions and their handling:



| | | |
|---|---|---|
| Illegal address/instruction | TRPSET | interrupt routine |
| Floating point exception | SFPA | interrupt routine |
| Severe request error | SERR/HERR | negative error code |
| Exit from program | DEVICE | address/data list |
| Control C detected (V3) | SCCA | user status word |

In V1 the hardware cleanup after a program exit was handled with a hardware RESET instruction. This was not tenable with the FB monitor, since it would kill the other job. The solution was to allow a program to specify a list of 'DEVICE' addresses and the values to fill them with on EXIT. With V3 it was clear that a single list was insuffucient, and a new .DEVICE request was implemented that allowed the specification of multiple 'device' lists.

The expansion of the acronym 'SCCA', in V3, is 'Set Control-C Ast'. I think support was first planned as an interrupt routine but changed to a flag word because of problems associated with the XM monitor.

- The VT-11 scroller

V2 provided a console scroller for the VT-11 graphics processor. The console command 'GT ON' switches console output to the VT-11. The scroller had to be adapted to the existing console handler and RT-11 used a VT-11 interrupt at the end of each scroller buffer to pick any new output for the scroller; this had two consequences: If a user disables the scroller, then RT-11 'hangs' when the output ring buffer fills. Secondly, if only a couple of characters are being displayed, then the scroller interrupts so frequently that it dominates CPU time. RT-11 has compensated for this by putting dummy 'dark' vectors into the scroller display to slow it down.

- FG program format

The .REL FG relocatable format is a super-set of the BG .SAV layout and the file has relocation information prepended to it. When the program is loaded, the FRUN command uses the relocation information to relocate the program in memory. Overlays are relocated in the image file itself and thus require that the original values of relocated addresses be stored for separate program runs.

The resulting program in memory is identical with a .SAV file and this means that the decision to run a program in the BG or FG is made at link time. This flexibility is similar to the RT-11 overlay scheme which also defers the overlay decisions to link time.

The FG job has a higher priority than the BG job. The idea is that the FG is used for high priority event driven tasks, while the BG is used for program development. Internally, the RT-11 structure allows for upto 128 jobs, however it is unlikely to support anywhere near this number.

The FG job did not provide RT-11 with a multi-user environment, in the sense that a separate user can not use the FG to run program development tasks.

- V2 utilities

V2 offered the RT-11 user a larger set of utilities. These utilities enhanced the language processors (CREF,LIBR) and general file operations (FILEX, SRCCOM, DUMP).

CREF provides MACRO with a Cross-Reference listing facility. MACRO builds a temporary intermediate file with the cross-reference information. When MACRO has completed an assembly it .CHAINS to CREF. CREF processes the intermediate file to produce a cross-reference listing. The format of the intermediate file is quite general and it is possible for other language processors to utilise the CREF facilities.

LIBR, the standard DEC librarian, was provided to complement FORTRAN. V2B followed up LIBR by supplying 'SYSLIB' (system library) which extended the use of program requests to FORTRAN programs.

V2B provided BATCH. BATCH has a lot of overhead which tends to make it unsuitable for a small systems but BATCH is often used to run production sequences and its style of operation is well known. A lot of BATCH users work almost exclusively with BATCH 'RT-11 mode', which is similar to command files.

- Wierd utilities

MBUILD, adapted from PIP, was supplied to handle systems generation from magtape distribution kits. PATCHO supplied object (.OBJ) file patching facilities. Both these utilities were replaced in V3 (MDUP, PAT).

- Version three

DEC was unhappy with the 'required' V2B update and pushed the RT-11 group to do fuller field testing of V3 before its release. Thus V3 had a longer development time than V2. However a V3B update was still required.

Its difficult to categorize V3. On the one hand it provided a huge number of new services and a new standard of console support and documentation. On the other hand it had many bugs and much of V3 was later rewritten or dropped. V3 was not simply an extension of V2, it was more like a new operating system. V3 aimed to make an operating system much easier to use than its predecessors.

V3 illustrates very clearly a difference between hardware and software engineering. When a hardware engineering group builds a major new version of a computer they discard the previous implementation and start again from scratch. Conversly, new versions of operating systems are usually built by enhancing the previous implementation. You can visualize the kind of problems that arise by imagining an LSI-11 built out of PDP-11/20 components - a lot of things just dont fit.

But, V3 suprised us all; the RT-11 group had produced a new user interface with a high standard of software engineering. Suddenly RT-11 started talking to us in English rather than in acronyms. Commands prompted for missing specifications and RT-11 queried potentially dangerous commands. The system 'talked' more, but could also be muted.

V3 monitor changes

| | |
|---|---|
| Device handlers | DL, DM, DP, DS, DX, MM, NL, (V3B DY) |
| Monitor | Base-line SJ monitor (BL) |
| | Extended Memory Monitor (XM) |
| | FB/XM Multi-teletype option (MTT) |
| | SJ/FB/XM Error logging/Memory parity option (EL) |
| | Ansii or DEC escape sequence support |
| (V3B) | Swap blocks moved to SWAP.SYS |
| | Monitor reorganized into separate assemblies. |



| | | |
|---|---|---|
| Program requests | | GTLIN, GVAL, SCCA, UNPROTECT |
| (non emt) | | FORK, MFPS, MTPS |
| | XM | CRAW, CRRG, ELAW, ELRG, GMCX, MAP, UNMAP |
| (options) | MTT | MTACH,MTDTCH,MTGET,MTIN,MTOUT,MTPRN,MTRCTO,MTSET |
| (options) | SJ | MRKT, CMKT, FORK |
| (changed) | | CSIGEN, CSISPC, DATE, DEVICE, EXIT, GTJB, SYNCH |
| Program images | XM | Virtual job, Priveleged job |
| Console commands | | Command files, factoring, line continuation |
| | | Minimum abbreviations, Prompting |
| | | APL, BOOT, COMPILE, COPY, DEASSIGN, DELETE |
| | | DIBOL, DIFFERENCES, DIRECTORY, DUMP, EDIT |
| | | EXECUTE, FOCAL, FORTRAN, HELP, INITIALIZE |
| | | INSTALL, LIBRARY, LINK, MACRO, PRINT, REMOVE |
| | | RENAME, RESET, SHOW, SQUEEZE, TYPE |
| (replacements) | | RESET/INIT, RESUME/RSUM |
| Error messages | | Extended format: ?KMON-F-File not found |
| | | SET ERROR NONE/SEVERE/ERROR/WARNING |
| Utilities | | DIR, DUP, ERRUTL, PSE, SYE, TECO, SYSGEN |
| (replacements) | | MACRO-11/MACRO PIP/PIP, MDUP/MBUILD, PAT/PATCHO |
| (improved) | | LINK, LIBR, BATCH, EDIT, FILEX, DUMP, PATCH |
| | V3B | RESORC, FORMAT |
| Documentation | | New set : 5 large ring binders, HELP command |
| Derivatives | | Run-time RT (RT squared), MINC |

- Terminology

Some system terms were redefined with V3. A file 'extension' became a file 'type'. Device 'handlers' are sometimes referred to as 'drivers', which is the RSX-11 term. The command language has been variously called DCLS (Digital Command Language Standard), DCL, CCL (Console or Concise Command Language) and the 'interactive console command language'.

- V3 monitor growth

RT-11 had grown considerably with V3 but was committed to continuing support for a minimal 8k system. V3 clarified this by defining the Base Line (BL) system. Options cost memory and V3 gave the user the choice of including or excluding many new options with SYSGEN. Thus, many system configurations were no longer 'bootable as delivered'.

The RT-11 design constrains the size of the USR to 2k words. The new V3 .GTLIN command, enhanced CSI requests and command line handling complicated the USR and they ran out of room! They saved some space by moving some commands into the USR buffer. More room was found by deleting the USR device tables for DSTATUS and using a disk access to pick up the device table information.

Let's review the effect of these changes: The USR must be re-read from disk whenever a command in the USR buffer is executed. Many DSTATUS requests now require a disk access - and DSTATUS is used quite heavily by programs like PIP to identify tape-like devices (MT:, CT:). Both these request types cause more USR re-reading of the 'current directory segment' which is usually 'cached' in the USR buffer. (With V4 .DSTATUS requests no longer require a disk access.)

However the biggest expansion of the monitor was in KMON, which handles all the monitor console command language and command file setup. KMON was originally under 2k - with overlays it must be more than 12k now. The DCLS interpreter in KMON is very complex and must remain position independent and slide during execution. It's amazing that they got it to work at all.

The RT-11 command file implementation only allows access to command files via the CSI requests (.CSIGEN/.CSISPC/.GTLIN). This denies command file access to programs that rely on .TTYIN. The implementation itself is scattered thruout KMON, KMOVLY and USR. This scattering was dictated by the sliding of KMON, the KMOVLY overlay structure and to minimise the number of overlay reads required in a base-line 8k system.

The command file implementation also triggered one of RT-11's biggest bugs: The routines that caused KMON/USR to slide were all in KMOVLY in V2. Now KMON always locked the USR before calling KMOVLY so that this sliding never affected a FG program. In V3 the command file support called the slide routines from outside KMOVLY, without locking the USR. A FG job could interrupt the slide and call a fractured USR, or have it slid while it was blocked in the USR. It was literally a case of having the carpet slid out from under your feet. The best way to trigger the bug was to type ^C^C to a BG command file while the FG was in the USR.

- The extended memory monitor

Welding eXtended Memory (XM) support onto the FB monitor was not easy and RT-11/XM has had problems. Simply put, if you want to use extended memory it should be part of the initial monitor design, since it affects the way the monitor deals with all communications with the jobspace. The idiosyncracies of RT-11's program requests and rather loose structuring of RT-11's monitor/jobspace accesses further complicate the XM monitor.

Two kinds of job are defined with XM: A 'privileged job' is compatible with the FB BG jobspace. This compatibility jobspace is used to run existing RT-11 utilities (MACRO etc.). A 'Virtual job' can map upto 32k words of virtual memory in either the BG or FG.

The XM monitor demands that the root section of a program and certain other program elements (queue and channel areas) be placed in the low 28k area. This shows one problem with RT-11 most clearly; as the monitor gets bigger the BG jobspace gets smaller. With a FG job and a number of handlers loaded and the USR NOSWAP the BG can get very small.

- Monitor options

V3 provided multi-teletype (MTT) handling, escape sequence support and error logging as monitor 'options'. To control monitor options RT-11 provided a Systems Generation (SYSGEN) facility and the monitor sources (stripped of comments). RT-11 SYSGEN sessions are fairly straight forward question and answer dialogues. Installation managers use SYSGEN to select the options they require and save memory space by eliding the options they dont need. V4 speeded up the SYSGEN process and provided even more control.

The Multi-terminal option extended the RT-11 console program requests (.TTYIN, .PRINT etc.) to multiple consoles (.MTIN, .MTPRNT etc.). Support was included for a wide variety of console types. The RT-11 group were directed to provide error logging support - the result was not wonderful. and error logging was rewritten for V4 using 'system tasks'. I dont know anyone who used the V3 escape sequence support and it seems to have disappeared in V4. Some older RT-11 features are not supported by monitors that include the V3 options. For example, VT-11 scrolling is not supported with MTT.



- V3B monitor reorganisation

Until V3B, RT-11 distribution kits included a prebuilt monitor for each monitor type (BL,SJ,FB,XM) and system device (RK,DY etc.) combination. However, with the growing number of monitor types and system devices this began to take up a lot of space. RT-11 V3B alleviated this problem by moving the 24 monitor swap blocks into a separate file (SWAP.SYS); this saved 24 blocks per monitor. They also saved space by not distributing the XM monitors and handlers.

The V3 RT-11 monitor was a huge piece of software and took hours to compile, depending on the processor and systems devices used. In V3B the monitor was reorganised to allow for faster, and more modular, assemblies. V4 solves this problem further by separating the monitor and the system handler. Each handler that can be used as a systems device includes the basic bootstrap code for the device.

Before V3 the name of the running monitor file was always 'MONITR.SYS'. V3 changed the bootstrap to allow monitors to be booted under their systems identification (e.g. RKMNFB.SYS, DYMNSJ.SYS) and also allowed bootstrapping of a monitor file directly, without first copying the monitor file into the bootstrap blocks. V4 simplifies the monitor names to RT11SJ.SYS, RT11FB.SYS and RT11XM.SYS.

- V3 console commands

RT-11 got a lot of cosmetic attention with V3. DEC has defined a 'command language standard' for all their operating systems to allow more user portability between systems. VAX/VMS and RSX-11M/plus both support this command language.

V3 uses english, rather than acronyms, which makes commands easier to remember. 'Minimum abbreviations' allow more confident users to specify commands quickly and economically. 'Line continuations' allow commands to be specified on more than one line. RT-11 allows a command to be fully specified in one line or the user can choose to be 'prompted' for missing specifications. In general DCLS serves both the novice and the professional with the kind of services they require.

Some console commands (e.g. SET, LOAD) are handled entirely by KMON and others (e.g. EDIT, HELP) use .CHAIN to pass information and control to the associated utility. RT-11 implements most DCLS commands by translating them into 'QUIET' command files which run a utility in V1/V2 mode to actually perform the operation (e.g. COPY runs PIP).

V3 supported command files, a long felt need by RT-11 users. RT-11 enhanced the bootstrap sequence by automatically running a start-up command file which relieved the user of the tedious job of loading handlers and setting up assignments.

- V3 utilities

Implementing DCLS affected many RT-11 utilities. Utilities were rewritten to conform to a system wide error message standard and to report errors back to the monitor. PIP was rewritten as three separate programs; PIP retained its basic file copy, delete and rename functions. DIR was written to handle directory listings. DUP (Device Utility Program) got all the device dependent tasks of bootstrapping, initialization, device copying and squeezing.

V3 PIP and DIR both support extended wild-card and wild-character capability. PIP changed the default copy mode to separate files, rather than to concatenating files. PIP was reorganized for V4 and DUP was more or less rewritten for V4.

V3 provided RT-11 with RSX-11 MACRO-11. DEC had been making an effort to reduce the number of dialects of all their language processors. DEC handles compatibility between different operating systems in much the same way with all programs, that is, a number of 'system dependent' modules are defined which interface the program with the local operating system. TECO and FORTRAN-IV also use this technique.

V1 and V2 utilities were written exclusively in MACRO but V3 produced three variants on this: DIR was written in a structured MACRO programming language called 'SUPER-MAC'. However, SUPER-MAC was implemented using many layers of macros and had very long compile times. When V3B was released half of DIR had been rewritten using standard assembler.

The V3 HELP command was implemented as a TECO program, this was cute but meant a fairly long response time for HELP commands. HELP was rewritten for V4 and provides faster and simpler support.

RT-11 SYSGEN sessions are driven by 'SCRIPT' files. The SCRIPT interpreter is implemented using FORTRAN. Once again, SCRIPT was modified for V4 by defining a less verbose 'ABBREVIATED' mode for SCRIPT.

- RT-11 program requests

With RT-11, the 'program request' is the method of communication between program and monitor. RT-11 program requests are easy to use and have minimal complications and most program requests are implemented using the PDP-11 EMT instruction. Here's the basic EMT instruction format:

```
                      15          8 7            4 3            Ø
EMT instruction:   [EMT code: 1Ø4][function code][channel/subcode]
```

A warning that the EMT program request formats would change with V2 was included with V1 documentation. V2 did supply an alternate EMT format but the planned desupport of V1 forms never eventuated and RT-11 has supported two different EMT structures ever since. Apparently only 3Ø words would have been saved by desupporting the V1 EMT format and this saving did not justify the incompatibilities it would cause to both existing DEC and user V1 programs. Supporting two EMT formats has complicated the system macro library (SYSMAC.SML) which often has to check for V1 or V2 program request types.

V1 passed program request parameters to the monitor on the stack. V2 EMTs call the monitor with the parameters in RØ (EMT 374 group), or in a list pointed to by RØ (EMT 375 group). The major parameter in a V2 call contains the EMT subcode and channel:

```
15     8 7     Ø
[subcode][channel]
```

V1 used 4 bits in the EMT instruction to specify a channel number, which limited the number of channels to 16 and also forced channels to be defined at assembly time. The V2 EMT style specifies the channel in a parameter byte which allows upto 256 channels to be defined. V2 EMT forms also enhanced existing program requests. For example, V2 directory operations (LOOKUP etc.) are permitted to pass tape positioning information to Magtape.



Most of the new V2 program requests were required to support the FB monitor. Keeping the SJ and FB monitors compatible also meant that many of these request had also to be implemented or gracefully ignored by the SJ monitor. The FB monitor provided timer support with the MRKT (mark time) program request group. The existing 'i/o completion routine' structure was expanded to handle timer expiration routines.

V3 added 19 program requests however the XM and MTT requests used the V2 channel byte for further sub-coding. V4 adds only 2 program requests. The XM monitor forced RT-11 to clean up the monitor/jobspace interface. The .DATE program request was changed from a direct monitor-table access to an EMT call and .GVAL was defined to access other fixed monitor offsets.

In contrast to the console commands, the names of the new program requests have become more acronymic and obscure. For example the documentation never explains what '.SCCA' even stands for! Related program requests are not systematically named. For example, all the following requests perform a 'get status' function: .DATE, .GTIM, .GTJB, .GVAL, .GMCX, .DSTATUS, .CSTAT, .SCCA. The arbitrary names given to program requests causes a lot of wasted time cross-checking the spelling with the user guide.

- RT-11 device handlers

With few exceptions the RT-11 device handlers have always been very clean pieces of software. Each version of RT-11 has supplied new handlers and simplified the task of writing and interfacing handlers to RT-11.

Each handler is prefaced by a short table which describes the handlers interrupt entry, and two queue pointers which are maintained by the monitor. When a handler is called, one of these queues points to the Current Queue Entry (CQE), which describes the i/o operation to be performed. The handler executes the operation described and when done jumps to the i/o completion code in the monitor. This simple scheme allows a good deal of flexibility and allows users to add their own i/o handlers.

The V2 FB monitor had to maintain separate BG and FG stacks and another system stack for interrupts. With V2 handlers must declare an interrupt to the monitor with .INTEN (which I guess stands for INTerrupt ENtry). The monitor swaps to the system stack if necessary and returns to the handler. The .SYNCH program request was defined to allow an interrupting process to synchronise with either the BG or FG job. The SJ monitor provided dummy .INTEN and .SYNCH routines to stay compatible with the FB monitor.

Setting up the characteristics of a device handler was changed from a patching operation to a console command operation in V2. Each handler that accepts a SET command places interfacing code in previously unused space in block zero of the handler. The monitor handles most of the SET command decoding, so that the handlers have little work to do. The monitor/handler interface was well defined and has presented no problems.

The V2 magtape (MT:) handler and message handlers extended the definition of handlers. MT: and CT: are called 'special devices' and are called to handle their own directory operations, i.e. LOOKUP, ENTER, DELETE, CLOSE. (V3 added RENAME to this set). The 'Message' handler (for .SDAT/.RCVD) required the definition of a new handler type that is always called on any program abort (to purge undispatched messages).

V3 and V4 followed the RT-11 tradition of making it easier to write local device handlers by supplying a series of macros which defined most of the handler interface to the monitor. With V1 and V2 the procedure to install a new handler consisted of patching the monitor tables concerned; a tiresome and error prone procedure. V3 removed this problem with new console commands to INSTALL and REMOVE handlers.

The XM monitor requires that handlers deal with 18 bit addresses, rather than the 16 bit addresses. A series of monitor routines are provided which make most of this work trivial, if time consuming.

RT-11 supplied handlers usually present no problems. However, the V3 DL: and DY: handlers had some serious bugs. The DL: seems to have been produced in a hurry, and indeed, even the allocation of the device vector was changed. The initial DL: handler supplied with V3 simply didnt work. The DL: and DY: handlers were rewritten for V4.

A special 'primary bootstrap' handler is required for each of the RT-11 system devices. Until V4 the primary bootstraps were contained in BSTRAP.MAC and this required a separate system generation session for each system device. V4 solved this problem by moving the primary bootstrap code for each device to the end of their associated handlers. DUP takes care of combining handler and monitor code to build a full bootstrap for a device.

- Software support and documentation

Software support has remained fairly constant thru-out all versions of RT-11, though the format of the Software Performance Report (SPR), or the RT-11 software dispatch seems to change a couple of times each year. DEC's software support group seem to be disorganised at times; I have wrongly received software dispatches for RSTS, RSX-11 and PDP-8's. At other times I have not received RT-11 dispatches or received multiple copies.

The uniform nature of the software performance reports disguises the relative severity of the bugs they document. Some are trivial, some must be very embarrassing. But lets face it; programming is a constant exercise in humility. With thousands of users to satisfy RT-11 has done well. Indications of the pressures caused by a user audience this large can be read in the software reports For example, not many users need to print 2^15 console characters without a linefeed, but someone did:

'Several SJ console terminal problems are corrected. It is now possible to print more than 32,767 characters without a line feed.' - RT-11/3B release notes.

The RT-11 binary update service provides an installation with RT-11 system updates, including a complete system distribution once a year. However a 'systems update' does not include installed patches, thus a user can purchase RT-11 and have to install patches that are over a year old. DEC only installs intra-version patches when they must: The RL01 patches were installed only on RL01 distribution disks. V4 promises an 'auto-patch' service, which means that patches are distributed on machine-readable media and installed using command files.

RT-11 documentation has grown from a single, rather thin, manual to a set of five rather cumbersome ring binders. One gets the feeling that the V1 documentation was padded out, since one third of the guide was taken up with appendices. V3 documentation won DEC an award and DEC does try to make its documentation clear and accurate. (DEC have a pocket guide on technical writing called 'Writing for the Reader'.)

The only problem for users with V3 documentation is simply that there is so much of it, which makes it difficult to locate the information required. FORTRAN programmers have the most trouble since they must deal with many layers of documentation: RT-11/FORTRAN release notes, RT-11/RSTS FORTRAN extensions, DEC standard FORTRAN, RT-11 system message manual, SYSLIB, LINK and LIBR and the local FORTRAN guru who keeps track of the folklore ("I know thats what they say in the manual, but just do such-and-such and it'll work").

The V1 system reference guide was supplemented with a software support manual'. This manual detailed various structures and formats, including queue formats, file structures, fixed offsets and device handler layouts. V2 appended flow charts of the complete monitor to this manual. When V3 was distributed users received an empty ring binder with the title 'software support manual'. Now the software support manual usually follows any release by a couple of monthes, but the V3 manual simply never arrived. Over a year later a supplement to the 'advanced programmers guide' was announced with a small subset of the software support manual. V4 supplies the software support manual with the distribution kit.

DEC has an extensive 'self-paced course' available for RT-11 in five volumes (CONCEPTS, MACRO, FORTRAN, BASIC, DIBOL) which provides detailed information about the monitor and its language processors. (The missing parts in the V3 software support supplement could be found in the MACRO volume of this set)

- RT-11 layered products

RT-11 has spawned and supports a large range of products both from DEC and other companies. Many hardware manufacturers supply their products with ready-to-run RT-11 device handlers and software libraries. The most common products in this area are graphics and laboratory products. This section lists some of the products available at present.

- DEC products

| | |
|---|---|
| APL-11 | A Programming Language, popular with mathematicians. |
| BASIC-11 | Beginners All-Purpose Instruction Code, used very widely. |
| Graphics | VT-11 graphics library. |
| Laboratory | Laboratory applications modules. |
| Industry | Industrial BASIC available. |
| MU-BASIC | Multi-User Basic, supports upto eight users. |
| COS-350 | Commercial system, includes DEC's language DIBOL. |
| DECNET | DEC NETwork tools to implement DDCMP intra-computer links. |
| FOCAL-11 | FOrmula CALculator, a DEC developed BASIC-like language. |
| FMS-11 | Forms Management System. |
| FORTRAN-11 | FORmula TRANslator, the classic computer language. |
| Graphics | FORTRAN-extensions for VT-55. |
| | DECgrahphic-11. |
| | PLOT |
| Scientific | SSB, Scientific Subroutine Package |
| System | SYSLIB makes the RT-11 program requests available . |
| GAMMA-11 | Nuclear medicine application. |
| LA-11 | Laboratory Applications package. |
| MINC | Modular INstrument Computer. BASIC-11 superset/RT-11 subset. |
| LA-11 | Lab Applications routines and SPARTA supervisor. |
| PDL/RT-11 | Programmable Data Logger for medical laboratories. |
| REMOTE/RT-11 | REal-time Multi-processor OrienTed Editor. A multi-user editor with limited satellite down-line loading capability. |
| RT-11/2780 | IBM 2780 remote batch terminal emulator. |

Other DEC RT-11 implementations

| | |
|---|---|
| RSTS-11/E | The DEC RSTS system supports an RT-11 jobspace. |
| RSX-11 | RT-11 support under RSX has long been a rumour. |
| RTSIM,SIMRT | FORTRAN/REMOTE provide limited stand-alone RT-11 support. |
| RT squared | Run Time RT-11 (RTRT) - RT-11 for a run time environment. |

Non-DEC multi-user RT-11 implementations

| | |
|---|---|
| STAR-eleven | A 'plural-computer' extension of RT-11 for 8 satellites. |
| TSX | Time Sharing eXecutive supports 12 RT-11 workplaces. |

The DECUS program library and the RT-11 symposium tapes provide other software sources. The most popular DECUS programs tend to be the games, such as ADVENTURE and DUNGEONS. The most recent symposium tape includes some nice system software for V4, including a system job that displays a continuous update of the status of all the jobs in the system and a VM: handler which treats extended memory as a file-structured device (VM: also supports PDP-11/70 22 bit addressing and permits RT-11 to be bootstrapped from VM:).

- Summary

The original goal of RT-11 was to deliver a small, fast single-user system and seven years later RT-11 will still provide service in 8k. Each version of RT-11 has delivered more console commands, program requests and a new job type (FG, XM, System jobs). Each version has provided more device handlers and simplified the task of writing device handlers.

V1 was spartan, the minimal operating system. V2 fleshed out the basic structure and followed paths mapped out by OS-8 and other single-user systems. RT-11 broke new ground with V3 but it also got a bit too big for its bootstrap. V3 added a lot of code to support DCLS, XM, MTT and was a monster to SYSGEN. With V4 the RT-11 group seem to have taken a good look at RT-11 and have begun to clean it up. The main theme of V4 is making services faster, smaller and cleaner, rather than adding new services. The additional services offered by V4 (system jobs, virtual overlays) integrate easily with RT-11.

The problems of V3 should be balanced against the impressive new services it provided and one should keep in mind the pressure under which a development group has to work. Management, marketing, hardware and customers all tend to push the operating system. For example: V3 error logging was forced on RT-11; The XM monitor was required for some commercial applications; DCLS is part of a general plan of DEC to provide all operating systems with the same console language; The DL: and DY: handler problems arose partly because the release of V3 coincided with the release of these devices. Additionally, if you compare the RK: handler with the DL: handler you'll see how hardware engineers are cutting corners and leaving more work to be done in software.

The goals of V1 have been replaced by goals oriented to new software technology and much cheaper memory. One illustration of the V1 structures inappropriateness to V3 is the 'sliding' KMON designed for small memory sizes and slow systems devices. More recent operating system designs tend to run their KMON-equivalent as standard program with few special priveleges (e.g. the UNIX Shell, the IAS PDS system).



A habit of RT-11 has been to pack all new services into the monitor. For example the FB message services (.SDAT/.RCVD) and Multi-terminal support were implemented in the monitor rather than in handlers. Using handlers more would have left the monitor simpler, reduced the number of occasions a SYSGEN was required, saved the effort required to rewrite FB services for the SJ monitor and allowed services to be selected by simply loading a handler rather than swapping monitors. The six system jobs allowed by V4 provide RT-11 with an environment outside the monitor for further system expansion and V4 already uses system jobs to set up error logging and spooling.

Here's some extremely crude estimates of RT-11's proportions: The operating system and utilities may have 100 man years work in them. Users may have done around 20,000 man-years programming and produce about 1000 lines of code per hour. RT-11 probably supports a total of 5 billion (english) words of primary memory.

- The future of RT-11

Most users have some 'new feature' they would like to see in RT-11 - the wishlists are endless. However the advantage of RT-11 is its simplicity. It boots up and starts working and requires almost no supervision. At every level it's easier to use than its multi-user alternatives. RT-11's low profile allows you to bend the system to new applications without too much of a fight and RT-11 provides a highly predictable program environment for real-time tasks. Some writers have noted that 'extended monitor features' often end up being part of the problem set, rather than being part of the solution.

The future of RT-11 is tied, in a more general context, to the future of single-user operating systems. There is a prevalent feeling that cheaper memories and disks will gradually lead installations to use RSX-11-type systems more eventually spell the end for the single-user system. Working against that notion are the constantly falling prices of computers: Whats the point of sharing a $200 processor?

One of the planned RT-11 sessions at this years DECUS Europe symposium is: 'RT-11 versus RSX-11M .. upgrading to RSX and RT-11 emulators'. This session is indicative of a general feeling that single-user RT-11 is just a poor cousin of multi-user RSX-11M. I disagree. As a user I would view a move to RSX-11 as 'down-grading'.

There is a strange contradiction between the way many feel about programming languages and operating systems. With 'structured programming languages' we equate simplicity and modularity with efficiency and predictability. The opposite of a structured program is the unstructured, unpredictable 'monolithic' program.

We can extend the principles of structured programming to operating systems. Looking at operating systems in this way we see that the single-user system is analogous to the modular approach: the single-user environment is simple and predictable. Like a module in a structured program, it aims to peform one task well. On the other hand the multi-user system is analogous to the monolithic program: complex and unpredictable.

If you examine the multi-user machine a little more closely you'll find that what its really trying to do is emulate a number of single-user computers! As I said above, whats the point of sharing a $200 processor? Structured programming has proved that 'simple' does not mean 'simple-minded' and the same point can be made for operating systems. I hope that in the near future we'll start hearing about how to upgrade from RSX-11 to RT-11!

Some programmers may find the opinion expressed here hard to accept. I think its worth remembering that when structured programming first appeared a lot of us found it hard to believe that we'd end up programming without the GOTO. The situation here is analogous: GOTO's complicated programs and multi-user operating systems complicate the programming environment.

Another trend working in favour of single-user systems is the emergence of relatively low-cost/high-performance local networks. A 'single-user computer' in a 'local network' is analogous to a 'module' in a 'structured program' (or a'task' in a 'multi-tasking system'). These new local networks combine the best of multi-user systems (shared system peripherals and databases) with the best of single-user systems (simple, predictable and independent environment).

RT-11's simplicity not only allows it to be adapted to users needs easily, it also allows RT-11 to be adapted to the 'future' and V4's streamlining makes that future more certain. Perhaps we should also discuss services we could afford to drop in our RT-11 V5 wishlists.

## RT-11 HANDLERS AND PSUEDO-HANDLERS

By Ray Strackbein

Over the years at the DECUS symposia there have been several users who addressed themselves to the fact that handlers are super places to hide data or jobs which you may want to hold memory-resident at all times. One merely assigns the 'handler' a vector in order to be able to find its location in memory once it is loaded. Mark Bartelt of Cal-Tech presented a beautiful paper which was submitted to the Minitasker some issues back about how to use a psuedo-handler to allow one set of Fortran OTS subroutines to be memory resident and used by both the background and foreground jobs. James Williams of West Virginia University presented two handlers at a DECUS meeting a couple of years ago which allowed the user to turn the system clock on and off and also allowed the user to make a crash dump on demand. Once you get the hang of it I think you will see some of the posibilities that handlers can present for solving problems.

I have out of necessity over the years embedded a power-fail bootstrap and a crash dump program in my perpetual foreground jobs. Recently, I have started changing my foreground jobs around and I no longer have a single job which runs all of the time. I have re-written the code to make a single handler function as a re-booting program, a power-fail rebooting program, and as a crash dump program which always remain memory resident.

Because the PDP-11/05 I have used for years has no bootstrap ROM, I learned early how to toggle the console switches to boot up the system. I also became very tired of the process. My RT-11 monitors have been modified for a long time to REBOOT on an error instead of to HALT. Do you have any idea just how great it is to have the monitor announce a fatal error and to print ton the console that it is going to HALT, but instead to re-boot itself automatically? When the bootstrap was imbedded in the foreground job, I never even missed a clock tick! But, alas, my handler has not yet reached that level of sophistication.

And so, for your entertainment and enjoyment, may I present a crash-dump, re-booting handler and possible monitor modifications to allow the system to automatically take a panic snapshot core dump and reboot itself on serious monitor or program errors.

Too good to be true? I'm affraid so. It only works on a RK05 system device.



And now the program:

```
        TITLE PF
;+
; THIS HANDLER HAS BEEN IN USE IN PRINCIPLE FOR SEVERAL YEARS
; BUT HAS JUST BEEN REWRITTEN FOR GENERAL USE AND AS SUCH IS CERTAI
; NOT YET PERFECTED IN ITS PRESENT STATE.  IT HAS BEEN DEBUGGED.
; THIS HANDLER IS WRITTEN FOR USE ON A MACHINE USING AN RK05 AS A
; SYSTEM DEVICE.  THIS HANDLER MAY NOT SUPPORT THE XM MONITOR IN IT
; PRESENT STATE.  I DON'T HAVE AN EXTENDED MEMORY MACHINE TO TEST I
; IT WILL DEFINATELY HAVE PROBLEMS UNDER VERSION 4 XM MONITOR IF TH
; RE-BOOT OR CRASH DUMP IS CALLED FROM EXTENDED MEMORY.
;
; ANY PROBLEMS SHOULD BE REPORTED TO:
; RAY STRACKBEIN
; P O BOX 1457
; PALM DESERT, CA 92260
;
;-
        .MCALL  .DRBEG,.DRFIN,.DREND
        .IIF NDF TIM$IT,TIM$IT=0
        .IIF NDF ERL$G,ERL$G=0.
        .IIF NDF MMG$T,MMG$T=0
        $PNAME  =<<<'P-100>*50>+<'F-100>>*50+0
PFSTS   =20377  ; SEE P. C-34 IN RT-11 ADVANCED PROGRAMMER'S GUIDE
EOFBIT  =20000
        .DRBEG  PF,24,0,PFSTS,PFTAB
; THE FOLLOWING IS STOLEN FROM THE NULL HANDLER (NL.SYS) JUST IN
; CASE SOME FOOL TELLS THE MACHINE TO "COPY/DEVICE RK0: NL:" THEN PRE
; THINGS WILL HAPPEN --- NOTHING BAD.
PFENTY: MOV     PFCQE,R5
        TST     6(R5)
        BMI     PFEXIT
        BIS     #EOFBIT,@-(R5)
        BR      PFEXIT
        RTS     PC
;+
; THE ROUTINE DUMPIT CAN BE CALLED BY A "MOV @#260,PC" SINCE VECTOR
; 260 POINTS TO DUMPIT.  DUMPIT TAKES A SNAPSHOT DUMP OF MAIN MEMORY
; AND STORES IT ON THE LAST 32K WORDS OF THE RK05 FROM WHICH THE SYS
; WAS BOOTED.  DUMPIT THEN PASSES CONTROL TO PFINT WHICH REBOOTS THE
; SYSTEM RK05.  THE DATE AND TIME ARE LOST. (ALWAYS LEAVE ROOM FOR
; FUTURE ENHANCEMENTS.)
;-
DUMPIT: RESET                   ; TURN OFF INTERRUPT ENABLE BITS
        MOV R0,SAVREG   ; AND SAVE THE REGISTERS
        MOV PC,R0
        ADD #SAVREG-.,R0
        TST (R0)+
        .IRP X,<R1,R2,R3,R4,R5,SP>
        MOV X,(R0)+
        .ENDM
LOGIT:  MOV @#54,R1     ; LETS GET THE UNIT NUMBER OF THE SYSTEM DEV
        MOVB 275(R1),R1 ; AND STORE IT IN R1
        CLC
        BIC #177770,R1
        ROR R1
        ROR R1
        ROR R1
        ROR R1
```

```
        MOV #177406,R0  ; POINT TO RKWC
2$:     TSTB -6(R0)     ; IS THE CONTROLLER READY?
        BPL 2$          ; LOOP IF NOT
        MOV #-100000,(R0)+      ; AND WRITE 32K WORDS
        CLR (R0)+       ; STARTING AT WORD 0 IN CORE
        BIS #14124,R1   ; AND STORE IT STARTING AT THE LAST 32K WORDS ON
        MOV R1,(R0)     ; THE SYSTEM DEVICE
                        ; (WOE BE TO HE WHO FORGOT TO PUT A FILE THERE!)

        MOV #3,-6(R0)   ; AND START WRITING
10$:    TSTB @#177404   ; LOOP TILL THE CONTROLLER IS READY
        BPL 10$
        TSTB @#177400   ; LOOP TILL THE DRIVE IS READY
        BPL 10$
        BIT #100,@#177400       ; THEN LOOP TILL THE DRIVE IS READY AGAIN
        BEQ 10$
        BR PFINT
SAVREG: .BLKW 7         ; THE REGISTERS BE STORED HERE FOR YOUR CONVENIENCE
;+
; PFINT IS THE START OF THE POWER FAIL RE-BOOT ROUTINE. IT MATTERS NOT
; WHAT YOU SPECIFIED FOR A POWER FAILURE MESSAGE IN THE SYSGEN ... THIS
; PIG TAKES OVER REGARDLESS!!!  PLEASE NOTE THAT THE REGISTERS ARE
; STORED IN THE PRECEEDING SEVEN WORDS.  THEY ARE EASY TO FIND, NO?
; IF YOUR MACHINE HAS CORE MEMORY, THEN THIS ROUTINE WILL RE-BOOT THE
; RK UNIT WHICH WAS THE SYSTEM DEVICE WHEN THE POWER WENT OFF.  IF YOUR
; MACHINE HAS RAM MEMORY WITHOUT BATTERY BACK-UP POWER, THEN YOUR CORE
; WILL COME UP WITH ZEROS AND CAN'T POSSIBLY INVOKE THIS ROUTINE.
; BUT NO MATTER WHAT TYPE OF MEMORY YOU HAVE, THE REBOOT PART OF
; THIS HANDLER WILL WORK JUST FINE WHEN USED TO RECOVER FROM FATAL
; ERRORS AND MONITOR HALTS.
;-
PFINT::
        MOV PC,R5               ; SAVE THE CURRENT LOCATION
        RESET                   ; AND SHUT OFF ALL INTERRUPTS!
        TSTB @#177562           ; TURN OFF THE CONSOLE KEYBOARD READY BIT
        CLR R0
10$:    INC R0                  ; A PAUSE FOR THE CAUSE
        BNE 10$
;+
; OK, LET'S EXPLAIN THE NONSENSE ABOVE ...  YOU SEE, DEC CHOSE TO
; DEFINE A SINGLE POWERFAIL VECTOR.  AND THE SAME INTERRUPT GETS
; TAKEN WHETHER THE POWER IS GOING DOWN OR COMING UP.  NOW THE LAST THING
; WE WANT IS TO TRY TO READ THE BOOTSTRAP RIGHT OVER OUR POWER FAIL
; VECTOR WHILE THE POWER IS GOING DOWN.  THEN WE WOULDN'T KNOW WHERE
; TO GO WHEN THE POWER COMES BACK UP AGAIN.  THAT IS THE TROUBLE WITH
; WRITING THIS POWER FAIL ROUTINE AS A HANDLER -- HANDLERS MOVE AROUND
; IN MEMORY DEPENDING ON A LOT OF THINGS FROM ONE BOOTSTRAP TO ANOTHER,
; SO I CAN'T BUILD SOME PERMINENT ADDRESS INTO THE POWER FAIL VECTOR IN
; THE BOOTSTRAP.  ANYWAY, THAT COUNT TO 64K (ABOVE) EATS UP ENOUGH TIME
; SO THAT WE WON'T DO A DISK READ AS THE POWER IS GOING DOWN AND WALK
; ALL OVER THE START-UP DIRECTIONS.
;-
        MOV @#54,R1             ; FIND THE MONITOR IN MEMORY
        MOVB 275(R1),R1         ; AND GET THE SYSTEM UNIT NUMBER
11$:    CLC
        BIC #177770,R1
        ROR R1
        ROR R1
        ROR R1
        ROR R1
```



```
12$:    RESET
        MOV #177406,R0          ; THIS BOOTSTRAP IS TAKEN FROM THE POCKET GUI
        MOV R1,4(R0)
        MOV #177400,(R0)

        TST -(R0)
        TSTB (R0)
        BPL 12$
;+
; THE BOOTSTRAP LOOPS HERE UNTIL THE DRIVE IS READY. BUT IF THE
; OPERATOR GETS IMPATIENT, OR IF THE DRIVE NEVER GETS READY, THE OPERATOR
; CAN TYPE A DIGIT AND THE CORRESPONDING UNIT WILL TRY TO BOOT.
;-
1$:     TSTB @#177560   ; KEYBOARD CHARACTER?
        BPL 13$         ; BR IF NOT
        MOVB @#177562,R1        ; MAKE IT A UNIT NUMBER
        BR 11$          ; AND SEE IF WE CAN BOOT THAT ONE
13$:    TSTB -4(R0)     ; IS THAT DRIVE READY YET???
        BPL 1$          ; LOOP TILL IT IS
        MOV #5,(R0)     ; LETS READ THE BOOTSTRAP
2$:     TSTB (R0)       ; AND WAIT TILL READ COMPLETE
        BPL 2$
        CMP #240,@#0    ; THE FIRST WORD OF A GOOD BOOTSTRAP IS 240
        BNE PFINT       ; TRY AGAIN IF NOT A GOOD BOOTSTRAP
        MOV #24,R4      ; WE JUST WROTE THE BOOTSTRAP OVER THE POWER-FAIL
                        ; VECTOR -- LET'S RESTORE IT BEFORE WE GET HIT AGAIN
        MOV R5,(R4)+
        MOV #340,(R4)
        CLR PC          ; EXECUTE THE BOOTSTRAP
PFTAB:  .WORD 24
        .WORD PFINT-.
        .WORD 340
        .WORD 260       ; THE OLD CASETTE VECTOR (AND I DON'T EVER EXPECT TO
                        ; RUN CASETTE)
        .WORD DUMPIT-.
        .WORD 340
        .WORD 0
PFEXIT: .DRFIN  PF
        .DREND  FF
        .END
```

THE POWER FAIL VECTOR (24) IS USED TO BOOTSTRAP THE SYSTEM FROM THE
FROM WHICH THE SYSTEM WAS LAST BOOTSTRAPPED; THE CASETTE VECTOR(260) IS USE
POINT TO THE CRASH DUMP AREA. A CRASH DUMP AUTOMATICALLY RE-BOOTS THE SYS
TO CALL A CRASH DUMP, MERELY USE "MOV @#260,PC" IN THE PROGRAM. NOTICE
THIS IS A TWO WORD INSTRUCTION. IT SEEMS THAT THE RT-11 MONITOR USES A TWO
INSTRUCTION TO HALT THE MACHINE ON SERIOUS ERRORS: 0 (HALT) NAD 776 (BR
UNCONDITIONALLY BACK TO THE HALT.)

THERE IS AN UNDOCUMENTED FEATURE IN PATCH FOR RT-11 V3 AND V3B.
FEATURE IS THE "SEARCH" COMMAND (';S'). TO FIND ALL OF THE HALT-POINTS IN
COPY OF THE MONITOR, MERELY RUN PATCH LIKE THIS:

```
 .R PATCH
 FILE NAME ---
 *RKMNFB.SYS/A
 *776;S
```

(AND PATCH PRINTS OUT ALL OF THE LOCATIONS FOR THAT UNCONDITIONAL BRAN
PATCH ALSO RUNS RIGHT OFF OF THE END OF THE MONITOR BECAUSE OF THE '/A' WE
IT AND THAT CAUSES A READ ERROR -- NOTHING TO WORRY ABOUT, REALLY. NOW A
WITH THE LOCATIONS OF ALL OF THE '776'S, RUN PATCH AGAIN WITH THE '/A' AND C

to make sure there is a zero in the word prior to each 776. Replace each occurance of a zero followed by the next word of '776' with a '13707' and a'260' (MOV @#260,PC). Simple, huh?

Now we must make sure that the the last 32K words on the RK05 (all which may be run with the PF handler) have a dummy file name to keep other data and programs from being clobbered by the crash dump program.

```
.R DUP
*RK0:DMPSPC.BAD=/C:11100:200
*
```

That reserves space on the disk where the PF handler is going to put its picture of the core when something awful happens. I use the extention ".BAD" because it doesn't get moved by a squeeze of the disk. It does get renamed to ' FILE.BAD' which is a little inconvenient because one has to patch the directory to remove or rename FILE.BAD. Delete DMPSPC.BAD before you squeeze the disk or you too will become an expert at patching directories. Also, don't forget to use DUP to put the dummy file back on the disk after squeezing it.

It does appear to be possible to re-write the handler so that it can find the dump area on the disk no matter where it happens to be and to also at the same time have the handler be able to dump and to boot any legal device at all. It would make the handler bigger. It now works just fine for me (with the restrictions mentioned) and so some one else can change it.

The program DUMP can be used to inspect any portion at all of the file DMPSPC.BAD. A sample of memory 0-1000 and 143000-144000 is shown below. Notice that the registers are stored in the seven words just below the word pointed to by the power fail vector (24).

```
     DMPSPC.BAD/O:0
BLOCK NUMBER  00000
000/ 040000 104350 147122 000340 147122 000341 000000 000000 *.@H.RN'.RNA.....*
020/ 000000 000000 143102 000340 147754 000000 000000 000000 *....BF'.LO......*
040/ 004134 001000 000060 001612 015502 000000 143310 000000 *...0...B...HF..*
060/ 163744 000340 166000 000340 000000 000000 000000 000000 *DG'..L'.........*
100/ 157636 000341 000000 000000 000000 000000 000116 000000 *.A.........N...*
120/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
140/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
160/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
200/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
220/ 167572 000340 000000 000000 000000 000000 000000 000000 *ZO'.............*
240/ 000000 000000 157572 000340 000000 000000 000000 000000 *....Z'.........*
260/ 142722 000340 000000 000000 000000 000000 000000 000000 *RE'.............*
300/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
320/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
340/ 163744 000341 166000 000341 163744 000342 166000 000342 *DGA..LA.DGB..LB.*
360/ 163744 000343 166000 000343 000000 000000 000000 000000 *DGC..LC.........*
400/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
420/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
440/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
460/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
500/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
520/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
540/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
560/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
```



```
600/ 000000 000000 000000 000000 000000 000000 000000 000000 *.........
620/ 000000 000000 000000 000000 000000 000000 000000 000000 *.........
640/ 000000 000000 000000 000000 000000 000000 000000 000000 *.........
660/ 000000 000000 000000 000000 000000 000000 000000 000000 *.........
700/ 160040 160054 156312 162160 162160 156334 146676 154534 * ','JPDPD
720/ 001000 001000 144316 166004 160040 160054 163734 146346 *....NH.L
740/ 151774 155400 146336 000000 144400 146336 152652 142644 *!S.[L...I
760/ 142644 000120 142517 000003 000000 142520 141002 000010 *$EP.DE...

     DMPSPC.BAD/O:143
BLOCK NUMBER  00143
000/ 006001 012700 177406 105760 177772 100375 012720 100000 *..@...P.Z
020/ 005020 052701 014124 010110 012760 000003 177772 105737 *..AUT.H.P
040/ 177404 100375 105737 177400 100372 032737 000100 177400 *..>....Z.
060/ 00176L 000407 000060 146454 146454 144450 146736 000000 *V...0.,M,
100/ 167406 010705 000005 105737 177562 005000 005200 001376 *.DE....R.
120/ 013701 000054 116101 000275 000241 042701 177770 006001 *A.,.A.=.!
140/ 006001 006001 006001 000005 012700 177406 010160 000004 *........@
160/ 012710 177400 005740 105710 100366 105737 177560 100003 *H...'.H.V
200/ 113701 177562 000751 105760 177774 100367 012710 000005 *A.R.I.P.!
220/ 105710 100376 022737 000240 000000 001323 012704 000024 *H.~.% ...
240/ 010524 012714 000340 005007 000024 177630 000340 000260 *T.L.'....
260/ 177442 000340 000000 010704 062704 177406 013705 000054 *".'...D.D
300/ 000175 150434 160054 160674 000167 014540 000000 001050 *>..0,'<AW
320/ 000112 000014 001000 000000 000000 000000 000000 000000 *J........
340/ 000000 000000 000000 000000 000000 000000 000000 000000 *.........
360/ 000000 000000 000000 000000 000000 000000 000000 000000 *.........
400/ 000000 000000 000000 000000 000000 000000 000000 000000 *.........
420/ 000000 000000 000000 000000 000000 000000 000000 000000 *.........
440/ 000000 000000 000000 000000 000000 000000 000000 000000 *.........
460/ 000000 000000 000000 000000 000000 000000 000000 000000 *.........
500/ 000000 000000 000000 000000 000000 000000 000000 000000 *.........
520/ 000000 000000 000000 000000 000000 000000 000000 000000 *.........
540/ 000000 000000 000000 000000 000000 000000 000000 000000 *.........
560/ 000000 000000 000000 000000 001002 000000 000000 132664 *.........
600/ 156370 000000 001000 001003 104001 000000 177560 177562 *X........
620/ 177564 177566 177777 006444 146336 000002 156656 036377 *T.V...$.L
640/ 170017 000303 000000 140077 170063 000000 170377 000000 *.PC...?@3
660/ 000000 134664 000320 000000 000445 000432 000002 100000 *..49P...%
700/ 000000 022004 010000 001414 005710 015364 177400 000000 *...$....H
720/ 001020 000000 164400 000000 113664 013400 000000 113664 *.....I..4.
740/ 000000 000000 000000 143744 146736 167144 103621 000001 *......DGM
760/ 106746 000240 105066 000001 016646 000002 016666 000002 *F. .6....
```

I hope I have stimulated your thinking a little. It is very diff adequately cover this kind of thing in a small space.

Subject: Double Height/Width for FMS-11

Dear Mr. Demers:

An entry for user requests:

I understand a Canadian RT User has modified FMS-11 to take advantage of the VT100 features for double height/ double width. If this is true, who is the person?

Thanks.

ICI Americas Inc.

WILMINGTON, DELAWARE 19897
(302) 575-3000

Sincerely yours,

Roland R Burgess

R. R. Burgess
Senior Scientific/
Programmer Analyst

"If anyone has experience comparing TSX with RSX, in either a development system or real-time environment, please contact me:

Lyle Ryan
Cetus Corporation
600 Bancroft Way
Berkeley, California 94710
(415) 549-3300 extension 394"

Gentlemen:

I am having problems using the VT52 support program for the TECO editor supplied with my RT-11 V3B software. I wish I could specify the problem more specifically, but generally the VT52 editor seems to insert line feeds and carriage returns in a seemingly random order on our VT55. I have enclosed a copy of the initial display which occurs when our VT52 editor is called, as well as a copy of the display screen as it should appear. I would appreciate any help that you can provide concerning this problem.

Sincerely,

Mark Meinrath

WAKE FOREST UNIVERSITY
WINSTON-SALEM, NORTH CAROLINA 27109

Mark Meinrath
Instructor



**A. "This is the display image that appears when TECO and VEG.TEC are called up using the commands: .R TECO**
***ERVEG.TEC$YXZMZ$$**

```
          Keypad layout                           Other keys
 _____________________________________________
! "BLUE"  ! "RED"    ! "GREY"   !   "^"     !    CTRL/C  Exit from m
!crSave   ! TECO     ! Unsave   !  Up in    !    CTRL/D  Kill rest
!  text*  ! command  ! text     ! column*   !    CTRL/K            of line*
!_________!__________!__________!___________!    CTRL/U  Kill start
!  "7"    !  "8"     !  "9"     !  "v"      !    CTRL/Z  Exit from  of line
! Open    ! Page*    ! Quote    ! Down in   !    BK SP   Go         macro
!o line*  !          ! next*    ! column*   !    DELETE  D      end of line*
!_________!__________!__________!___________!    2 ESC's Repe    previous*
!t "4"    ! "5"      !  "6"     !  ">"      !                    TECO command
!  Up     ! D        !          !           !
!leline*  !character! Dlaste    ! Cursor    !    Arguments
!_________!__________!__________!___________!    All starred
!  "1"    !  "2"     !  "3"     !  "<"      !    optionally take commands
!  Top    ! Bottom   ! Start    ! Cursor    !    entered          an argument
!sof page*! of page  ! of line  !  left*    !              <ESC><digits>
!_________!__________!__________!___________!
!       "0"          !  "."     ! "ENTER"   !
!    Down line*      ! Search*  ! Search    !
!                    !          !argument*  !
!_
_____________________!__________!__________!
      File name:LETTER█
```

**(Note: random insertion of line feeds and control characters)**

**B. This is a copy of the display image as it should appear:**

```
 I        Keypad layout                          Other keys
 ____________________________________________
! "BLUE"  ! "RED"    ! "GREY"   !   "^"     !    CTRL/C  Exit from macro
! Save    ! TECO     ! Unsave   !  Up in    !    CTRL/D  Kill rest of line*
! text*   ! command  ! text     ! column*   !    CTRL/K  Kill line*
!_________!__________!__________!___________!    CTRL/U  Kill start of line
!  "7"    !  "8"     !  "9"     !  "v"      !    CTRL/Z  Exit from macro
! Open    ! Page*    ! Quote    ! Down in   !    BK SP   Go to end of line*
! line*   !          ! next*    ! column*   !    DELETE  Delete previous*
!_________!__________!__________!___________!    2 ESC's Repeat TECO command
!  "4"    !  "5"     !  "6"     !  ">"      !
!  Up     ! Delete*  ! Delete   ! Cursor    !    Arguments
!  line*  !character! last     ! right*    !
!_________!__________!__________!___________!    All starred (*) commands
!  "1"    !  "2"     !  "3"     !  "<"      !    optionally take an argument
!  Top    ! Bottom   ! Start    ! Cursor    !    entered as <ESC><digits>
! of page*! of page  ! of line  !  left*    !
!_________!__________!__________!___________!
!       "0"          !  "."     ! "ENTER"   !
!    Down line*      ! Search*  ! Search    !
!                    !          !argument*  !
!____________________!__________!___________!

        File name:
```

****************************************

SPRING 80 DECUS RT-11 TAPE COPY

Since the Fall 79 DECUS symposia in San Diego and especially since the February 80 'Mini-Tasker' has been distributed I have been inundated with tape copy requests. I have copied the following:

25 Spring 78 Chicago
25 Fall 78 San Francisco
24 Spring 79 New Orleans
35 Fall 79 San Diego

I have been and am still willing to continue, but now I have some help. Future requests for tape copies should be made of your nearest LUG or from the nearest all volunteer copy center listed below. A list of LUG's was published in the February 80 'Mini-Tasker'.

The rules are still quite simple. A tape in reusable packing along with return label and postage (not cash or check) is required. Include a note stating which tape or tapes you want. Any media arriving without the reusable packing, label and postage will be treated as a gift to the copy center. Before requesting copies on any media other than tape I suggest that you contact the copy center for confirmation.

Ray Kaplan
Electrical Engineering
Building 20
University of Arizona
Tucson, AZ 85721
(602) 626-4462
Media: RK05, RX01, RX02

Gary Siftar
CISCO, Inc
4135 S 100th E Ave
Tulsa, OK 74145
(918) 665-2110
Media: RX01, RX02, MT(800/1600)

Alfred H Scholldorf
Physics Dept
SUSB
Stony Brook, NY 11794
(516) 246-7110
Media: RL01, RX02, MT(800/1600)



Nick Bourgeois / 1738
Sandia National Laboratories
P O Box 5800
Albuquerque, NM 87185
(505) 844-8088
Media: MT(800)

At Chicago we built our master tape for the Spring 80 DECUS symposium. The US Chapter will distribute copies to the various LUGs that requested tapes and to those who ordered copies through the DECUS office at the symposium. The LUGs will be free to distribute as they see fit. Later, copies may be purchased from the DECUS library at their then current rates.

I want to thank both Joseph Lachman of Lachman Associates and Jerome Martin of First Computer Corp for the use of their time and facilities. They were both most cooperative and helpful in putting the Chicago-80 tape together.

The Chicago-80 tape contains 2997 blocks in 136 files. The annotated directory that was posted in the RT-11 campground at the symposium was incomplete. I later managed to get the preliminary FORTRAN OTS structure information added. A copy of the full annotated directory follows:

SPRING 1980 DECUS          CHICAGO          RT-11 SIG TAPE

ANNOTATED DIRECTORY

*********************************************************

SUBMITTED BY: RAY STRACKBEIN / 74-148 SANTA ROSA CIR / PALM DESERT, CA 92260 / 714-346-4656

ART HERMES, III / MIT/LNS BATES LINAC / P O BOX 95 / MIDDLETON, MA 01949

CHICAGO DECUS (1980) RT-11/PDP-11 ASSEMBLY LANGUAGE TUTORIAL: COMPLETELY DOCUMENTED PROGRAM TO ILLUSTRATE THE USE OF ASSEMBLY LANGUAGE UNDER RT-11.

```
                                    CRASH .DOC     23 23-Apr-80
ARCIVE.SAV      9 23-Apr-80         ARCIVE.MAC     52 23-Apr-80
ARCIVE.DOC     16 23-Apr-80
 5 Files, 100 Blocks
```

*********************************************************

SUBMITTED BY: GARY M SELZER
PRINCETON GAMMA TECH
P O BOX 641
PRINCETON, NJ 08540

RT-11 VERSION OF RATFOR TO FORTRAN TRANSLATOR. MODIFICATION OF PROGRAM PRESENTED IN 'SOFTWARE TOOLS', BY KERNIGHAN AND PLAUGER WHICH ORIGINATED IN BELL LABS.

| | | | | | |
|---|---|---|---|---|---|
| RATFOR.DOC | 16 | 23-Apr-80 | RATFOR.SAV | 79 | 23-Apr-80 |
| RATFOR.TEC | 1 | 23-Apr-80 | RATFOR.RFR | 44 | 23-Apr-80 |
| RATMOD.RFR | 50 | 23-Apr-80 | RATDEF.RFI | 7 | 23-Apr-80 |
| RATCOM.RFI | 8 | 23-Apr-80 | RATSTK.RFI | 1 | 23-Apr-80 |
| RATFOR.FOR | 72 | 23-Apr-80 | RATMOD.FOR | 90 | 23-Apr-80 |
| RATFOR.OBJ | 194 | 23-Apr-80 | RATMOD.OBJ | 110 | 23-Apr-80 |
| RATFOR.LNK | 1 | 23-Apr-80 | | | |

13 Files, 673 Blocks
*******************************************************

SUBMITTED BY: ROBERT B DENNY
CREATIVE SYSTEM DESIGN CO
3452 E FOOTHILL BLVD SUITE 602
PASADENA, CA 91107
213-355-6836

SOURCES AND BUILD COMMAND FILES FOR RATFOR/RT-11 V20.

| | | | | | |
|---|---|---|---|---|---|
| RAT1ST.COM | 3 | 23-Apr-80 | RATBLD.COM | 3 | 23-Apr-80 |
| GETSTR.FOR | 2 | 23-Apr-80 | PUTSTR.FOR | 2 | 23-Apr-80 |
| RATRT .FOR | 72 | 23-Apr-80 | RAT1 .FOR | 49 | 23-Apr-80 |
| RAT2 .FOR | 46 | 23-Apr-80 | RAT3 .FOR | 55 | 23-Apr-80 |
| STRLIB.FOR | 78 | 23-Apr-80 | RATHLP.MAC | 5 | 23-Apr-80 |
| CCHAR .RAT | 2 | 23-Apr-80 | CDATIM.RAT | 1 | 23-Apr-80 |
| CDEFIO.RAT | 1 | 23-Apr-80 | CFOR .RAT | 1 | 23-Apr-80 |
| CFUNC .RAT | 1 | 23-Apr-80 | CLINE .RAT | 1 | 23-Apr-80 |
| CLIST .RAT | 2 | 23-Apr-80 | CLOOK .RAT | 1 | 23-Apr-80 |
| COUTLN.RAT | 1 | 23-Apr-80 | CPRTLN.RAT | 1 | 23-Apr-80 |
| CSTR .RAT | 1 | 23-Apr-80 | CUCLC .RAT | 1 | 23-Apr-80 |
| DEFIN .RAT | 6 | 23-Apr-80 | DEFINS.RAT | 5 | 23-Apr-80 |
| RATDEF.RAT | 6 | 23-Apr-80 | RATRT .RAT | 60 | 23-Apr-80 |
| RAT1 .RAT | 37 | 23-Apr-80 | RAT2 .RAT | 36 | 23-Apr-80 |
| RAT3 .RAT | 42 | 23-Apr-80 | STRLIB.RAT | 68 | 23-Apr-80 |
| TEST1 .RAT | 1 | 23-Apr-80 | TEST2 .RAT | 1 | 23-Apr-80 |
| TEST3 .RAT | 2 | 23-Apr-80 | TEST4 .RAT | 1 | 23-Apr-80 |
| TEST5 .RAT | 1 | 23-Apr-80 | CFILES.RT | 2 | 23-Apr-80 |
| RATFOR.RNO | 114 | 23-Apr-80 | | | |

37 Files, 711 Blocks
*******************************************************

SUBMITTED BY: ANTON CHERNOFF
DEC ML5-5/E76

BAMBI.TXT HAD A BAD BLOCK ON THE SUBMITTED MEDIA.

| | | | | | |
|---|---|---|---|---|---|
| VTECO .SAV | 51 | 11-Mar-80 | VOTECO.SAV | 50 | 19-Mar-80 |
| MTYSET.MAC | 18 | 07-Apr-80 | MTYSET.SAV | 3 | 07-Apr-80 |
| RTMON .MAC | 33 | 12-Mar-80 | RTMON .SYS | 7 | 22-Feb-80 |

6 Files, 162 Blocks
*******************************************************

SUBMITTED BY: MARK BARTELT
CALIFORNIA INSTITUTE OF TECHNOLOGY
MS 356-48
PASADENA, CA 91125
213-795-6811 EXT 2663



TOOLS FOR BUILDING & USING RESIDENT LIBRARIES UNDER RT-11 (AS DESCRIBED IN FOSTER PAPER AT SPRING '79 SYMPOSIUM, AND REPRINTED IN PROCEEDINGS AND MINI-TASKER).

| | | | | | |
|---|---|---|---|---|---|
| RESLIB.DOC | 6 | 17-Apr-80 | MAKELB.FOR | 11 | 04-Apr-79 |
| LSHIFT.MAC | 2 | 21-May-79 | MAKELB.SAV | 16 | 02-Mar-80 |
| LBMAIN.MAC | 6 | 06-Apr-79 | | | |

5 Files, 41 Blocks
*******************************************************

SUBMITTED BY: MARK BRAMHALL
DEC

TECO-11 V36 FOR RT-11.

| | | | | | |
|---|---|---|---|---|---|
| TECO .SAV | 51 | 17-Apr-80 | TECOV .SAV | 50 | 17-Apr-80 |
| TECO .INI | 8 | 17-Apr-80 | VTEDIT.TEC | 23 | 17-Apr-80 |
| LOCAL .TEC | 3 | 17-Apr-80 | SEARCH.TEC | 4 | 17-Apr-80 |
| SQU .TEC | 10 | 17-Apr-80 | TYPE .TEC | 7 | 17-Apr-80 |
| TECO .TES | 27 | 17-Apr-80 | TECOIN.TES | 18 | 17-Apr-80 |
| VTEDIT.TES | 32 | 17-Apr-80 | LOCAL .TES | 3 | 17-Apr-80 |
| SEARCH.TES | 8 | 17-Apr-80 | SQU .TES | 15 | 17-Apr-80 |
| TYPE .TES | 17 | 17-Apr-80 | CRTASM.COM | 1 | 17-Apr-80 |
| CRTPRE.MAC | 1 | 17-Apr-80 | CRTRUB.MAC | 47 | 17-Apr-80 |
| TECLNK.COM | 1 | 17-Apr-80 | TECO .OBJ | 46 | 17-Apr-80 |
| TECOV .OBJ | 1 | 17-Apr-80 | TECOIO.OBJ | 9 | 17-Apr-80 |
| TIOFET.OBJ | 1 | 17-Apr-80 | TIOEIO.OBJ | 6 | 17-Apr-80 |
| TIOIAS.OBJ | 2 | 17-Apr-80 | SCREEN.OBJ | 16 | 17-Apr-80 |
| SCRINS.OBJ | 4 | 17-Apr-80 | SCROLL.OBJ | 4 | 17-Apr-80 |
| CRTRUB.OBJ | 7 | 17-Apr-80 | TIOENC.OBJ | 4 | 17-Apr-80 |
| TIOINI.OBJ | 11 | 17-Apr-80 | TIORFS.OBJ | 4 | 17-Apr-80 |
| TIODCD.OBJ | 11 | 17-Apr-80 | | | |

33 Files, 452 Blocks
*******************************************************

SUBMITTED BY: GREG L ADAMS
DEPT OF NATIONAL DEFENCE
1813-641 BATHGATE DR
OTTOWA, ONT, CANADA
613-993-9624

THIS CONTAINS THE SOURCE MODULES FOR THE RT-11 SPOOLER PACKAGE DISCUSSED IN THE PAPER "A TRANSPARENT OUTPUT SPOOLER FOR RT-11", CHICAGO 1980.

| | | | | | |
|---|---|---|---|---|---|
| SPX .SYS | 2 | 21-Apr-80 | SP .SYS | 2 | 21-Apr-80 |
| SPOOLX.REL | 7 | 21-Apr-80 | SPOOL .REL | 8 | 21-Apr-80 |
| SPBLD .COM | 1 | 19-Apr-80 | SPLINK.COM | 1 | 19-Apr-80 |
| SP .MAC | 11 | 20-Apr-80 | SPINIT.MAC | 24 | 20-Apr-80 |
| SPOOL .MAC | 9 | 20-Apr-80 | SPINT .MAC | 25 | 20-Apr-80 |
| SPOOL .OBJ | 2 | 21-Apr-80 | SPINT .OBJ | 6 | 21-Apr-80 |

12 Files, 98 Blocks
*******************************************************

SUBMITTED BY: GREG L. ADAMS
DEPT OF NATIONAL DEFENSE
1813-641 BATHGATE DR.
OTTAWA ONTARIO CANADA
(613) 993-9624

FLECS TRANSLATOR:

THIS IS AN RT-11 IMPLEMENTATION, INCLUDING DOCUMENTATION, OF THE UNIVERSITY OF OREGONS PUBLIC DOMAIN FLECS TRANSLATOR.

| | | | | | |
|---|---|---|---|---|---|
| FLECS .DOC | 152 | 20-Apr-80 | FLXDOC.RNO | 121 | 20-Apr-80 |
| FLXREN.DOC | 3 | 20-Apr-80 | | | |

3 Files, 276 Blocks

| | | | | | |
|---|---|---|---|---|---|
| FLXMIN.SAV | 77 | 20-Apr-80 | FLXREG.SAV | 77 | 20-Apr-80 |
| FLXGEN.COM | 2 | 04-Apr-80 | FLXHDR.FLX | 3 | 07-Mar-80 |
| FLXBLK.FLX | 1 | 05-Mar-80 | FLECS .FLX | 51 | 20-Apr-80 |
| FLXIO .FLX | 19 | 20-Apr-80 | FLXGO .FLX | 18 | 20-Apr-80 |
| FLXGOS.FLX | 18 | 20-Apr-80 | FLXANL.FLX | 30 | 05-Mar-80 |
| FLXLST.FLX | 58 | 05-Mar-80 | FLXSUP.FLX | 9 | 05-Mar-80 |
| DATIME.FLX | 4 | 07-Mar-80 | DATEIN.FLX | 5 | 07-Mar-80 |
| TIMEIN.FLX | 6 | 07-Mar-80 | | | |

15 Files, 378 Blocks

****************************************************************

SUBMITTED BY: JAMES A. KRUPP
MIDDLEBURY COLLEGE
MIDDLEBURY, VT

PRELIMINARY DOCUMENTATION FOR THE SESSION ON FORTRAN IV OTS: ITS STRUCTURE AND EFFECTIVE USE.

| | | | | | |
|---|---|---|---|---|---|
| READM4.DOC | 1 | 24-Apr-80 | RTALK .RNO | 107 | 24-Apr-80 |
| RTALK .DOC | 122 | 24-Apr-80 | CHAIN .HLP | 15 | 24-Apr-80 |
| GETCC .MAC | 2 | 24-Apr-80 | CHAINX.MAC | 1 | 24-Apr-80 |
| CHAIN .MAC | 6 | 24-Apr-80 | CCLRUN.MAC | 2 | 24-Apr-80 |
| TRACER.MAC | 25 | 24-Apr-80 | RCIMOD.MAC | 3 | 24-Apr-80 |
| OTSDOC.DOC | 98 | 24-Apr-80 | | | |

11 Files, 382 Blocks

**********************************************************************

SPRING 80 DECUS RT-11 USER APPLICATION WORKSHOP

Our workshop was just great until we all just ran out of steam. We had three informal presentations followed by some exciting exchange from the floor. The informal presentations were:

Ken Demers: A graphics package for Tektronix 4010/4014 terminals.

Ian Darwin: A network workstation application using LSI-11's.

Jon Melvin: LSI-11 based earthquake prediction system and other general lab uses.

Next time in San Diego I hope that our workshop does not again fall at the tail end of RT-11's busiest day. We all might be able to survive somewhat better.

Nick Bourgeois



Dear Ken,

I'm writing to follow up on my submission to the Spring DECUS RT-11 SIG tape. I put a version of Dave Sykes' RATFOR preprocessor on the tape. The RSX-11M version has been evolving over several years under Dave's guidance, and it is in a very workable state. Earlier this year, I had the occasion to convert it to run under RT-11. I did not, however, have the time to perform thorough testing, so it was put it on the tape as a "prototype".

The manual which was supplied with the kit is the RSX-11M manual without changes. There is sufficient information in another text file and in the comments at the head of RATRT.RAT for most anyone to use the RT-11 version.

I am most interested in hearing from anyone who has used this version of RATFOR. I plan to baseline it this summer, keeping it functionally identical to the RSX-11M version, and release it to the Fall tape as a "final" version.

In addition, I have just finished converting the latest RSX-11M version of RUNOFF (M02) to run under RT-11. I plan on putting it on the Fall tape also, but I am looking for a few test sites who can take it on mag tape, use it, and give me feedback before then. I have updated the documentation to include the RT-11 specifics. It has all of the functionality of the RSX version, including hyphenation. In the future, RUNOFF should be maintained in a standard version for all PDP-11/VAX operating systems.

Finally, there is currently an effort going on to standardize and baseline David Conroy's excellent C compiler and assembler with versions for RSTS, RSX-11M and RT-11. There will be separate runtime libraries (of course) for each operating system, but the interface from the language side will be nearly identical and very UNIX-like with a "vanilla" interface to the operating system on the other side. We hope to include system-specific libraries which will give the C programmer access to programmed requests from C. This project is in it's formative stages right now, and I will be writing again when there is a need for test sites.

My hope is that with the conversions of RATFOR and C for RT-11, that the RT-11 community will participate in and contribute to the growth of the structured languages SIG and it's efforts. This will give RT-11 users compatible (nearly) versions of Pascal, RATFOR and C.

3452 E. Foothill Boulevard
Suite 601
Pasadena, Ca. 91107
213-355-6836

Sincerely,

Bob Denny

# HAMMOND software

o Announcing STAR-eleven V4 :

STAR-eleven is a 'general-purpose plural-computer system' that provides upto eight satellite computers with a full, fast RT-11/SJ environment. Fast host to satellite connections (1.2 megabaud), low overheads and a software cache system provides satellites with a quality of service that often exceeds a stand-alone RT-11 environment. STAR-eleven solves the 'multi-user RT-11' problem for many installations.

V4 of STAR-eleven is primarily an upgrade to RT-11 V4. Enhancements include support for the RT-11 V3/V4 DCLS commands set (COPY, EXECUTE etc. plus some extensions). Satellite thruput is further improved with a 'select lookup cache' on the host for commonly used programs (PIP, MACRO etc.).

o Announcing DISKS-eleven V1 :

Many RT-11 users have patched handlers to divide large disks into a number of smaller 'virtual disks'. DISKS-eleven provides this 'virtual disk' support on a completely general basis without requiring any handler patches. Here's an example session with DISKS-eleven that splits an RK: up into four 1000. block 'virtual disks':

```
.INSTALL VD                              !Install the virtual device
.LOAD RK                                 !Load the physical handler
.RUN DISKS                               !The DISKS-eleven utility
*RK1:*=VD0:,VD1:,VD2:,VD3:/C/S:1000.     !Create the virtual disks
*^C
.INIT VD0:                               !Init the virtual disks
.INIT VD1:
 etc.
```

Virtual disks are created as standard RT-11 files (e.g. RK1:VD0.BAD). The design is completely general and supports all RT-11 file-structured devices. The virtual disk operation is transparent to RT-11 and all commands and program requests are supported (excluding BOOT and FORMAT).

'Virtual disks' are used to provide RT-11 with user separation (each user has a separate virtual disk), to store more files on huge disks and to reduce the time spent by RT-11 searching long directories. Since DISKS-eleven does not require any handler patches its easy to new versions of RT-11 and does not interfere with V4 auto-patch operations.

o For more information :

RT-11 users who are interested in STAR-eleven or DISKS-eleven can contact us us for more detailed information and a copy of the user guides at :

HAMMOND-software
Am Feldborn 22
D-34 Goettingen
West Germany

.1. 0551/23828



BOMP11 - BILL OF MATERIAL PROCESSOR

The BOMP11 system is designed to organize and maintain a central information system linking product structure records with master (inventory type) records. Single level and multi-level explosion and implosion options are provided which enable the operator to produce the most desirable type of document for his requirements. These listings may be directed to either the crt or to the line printer if a hard copy is wanted. The package is written in Fortran IV and has been configured to run under the RT-11 operating system, V3B or later.

For further information contact:

Software Dynamics Inc.
1000 Yale Avenue
P.O. Box 500
Wallingford, CT 06492
(203) 269-4998

SPR'S

Dear Ken,

Enclosed is a copy of a Software Performance Response that I recently received from the DIGITAL SPR center. I believe the content is sufficiently general to warrant publication in the Mini-Tasker, especially since DEC has known about the problem for some time and has yet to publish it in the Software Dispatch. I have voiced my concern about this to DEC. The workaround mentioned in the Response is quite straightforward. The user creates a subroutine that issues a mark time request, either from FORTRAN or MACRO-11. The subroutine then enters a loop that tests a flag word. The completion routine for the mark time request clears that flag word, allowing the subroutine to exit from the loop and return to the calling program.

I word also like to comment on the letter from Mr. Geoffrey Griffin that was published in Vol. 6, No. 1 of the Mini-Tasker. I have not experienced the SAME problem as he has with the V2.1 FORTRAN IV compiler, but have other gremlins crop up when compiling a program using INLINE code. To be honest, there have only been two occasions when the compiler did not create the correct INLINE code. The second did use mixed-mode arithmetic. One case invloved a flag variable in a COMMON block used for communication with an asynchronous routine. The main program would set this to either -1, 0, or 1. A simple 'I=1' assignment statement was used which was improperly compiled when INLINE code was chosen. The program executed correctly when THREADED code was used! After several edits to the program, none to the assignment statement, the problem disappeared! The other occurrance involved an array in a DO loop. Certain values of the array were modified in the loop and the index was calculated using mixed-mode (integer/real) arithmetic. The array contained garbage after coming out of the loop. Again, THREADED code gave the correct result.

My conclusion, based on my experiences and bolstered by the experiences of Mr. Grinton, is that there are definite bugs in the V2.1 compiler. I have made it part of my debugging process to recompile programs as THREADED if the problem is not immediately obvious. I have found that rearranging the order of the FORTRAN statements will sometimes allow the program to be compiled correctly with INLINE code. To the credit of DEC software designers, I have only observed these two instances, out of more than a hundred programs, where the compiler was actually at fault. However, having the COMPILER fail you even once is a very unnerving experience, even for an experienced programmer, and is enough to cause the novice to bolt at the sight of a computer!



SOFTWARE PERFORMANCE RESPONSE 

SPR NUMBER: 11-30245 DATE: 2-MAY-80

SOFTWARE SYSTEM(S) AFFECTED: RT-11 VERSION(S): V3B-00

SOFTWARE COMPONENT(S) AFFECTED: SYSF4 VERSION(S): V3B-00

**STATEMENT:** **The ISLEEP routine provided in SYSF4 does not function under the SJ monitor with timer support**

**RESPONSE:** **Thank you for submitting your SPR. The problem you desgrib has been known about for some time and is to be considered a restriction. The restriction will remain in RT-11 Versio but the documentation will call it out specifically. The w around using a .MRKT request that you suggest is precisely the method we have recommended to other users reporting th problem.**

Mark Gregory Liverman
Rice University
Department of Chemistry
P.O. Box 1892
Houston, Texas 77001

Sincerely,

Mark Gregory Liverman
(713) 527-4014

---

LAST MINUTE ADDITION

---


Marine Physical Laboratory of the
Scripps Institution of Oceanography
University of California, San Diego
San Diego, CA 92152

19 June 1980

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

Dear Mr. Demers:

I have recently been working with the RATFOR V20 files which were on the Spring 1980 RT-11 SIG tape. There were a number of interesting surprises which ought to be made public. The first surprise was that all the files were in RSX-like text format, with character counts per line and no <cr><lf>'s. This is not too hard to cope with, and I have a small utility program RSXCNV (copy enclosed) for that purpose.

The second surprise was that there was a garbage block in the middle of file RATRT.FOR. By looking carefully at RATRT.RAT, and hand-compiling a few lines, I was able to reconstruct that block, as later proved by running the whole compiler.

The third surprise was that the documentation was in RSX RUNOFF source format, which can also be coped with if you happen to be a RUNOFF expert.

The fourth surprise was that essentially everything works, as long as one stays away from inline code in the compiled RATFOR pre-processor.

Bob Denny, who submitted the files for the SIG tape, says that the RSX-like format came from doing all the file transfers in a hurry on a VAX. In any case, I can distribute a properly RT'ed version of all this stuff on a couple of floppy disks, given the usual protocol. (Requestor supplies the disks, the mailing container, and the return postage.)

Carl D. Lowenstein



```
            PROGRAM RSXCNV
      C
      C     CONVERT RSX ASCII FILES TO SOMETHING READABLE
      C         INPUT FROM INP:, OUTPUT TO DK: WITH SAME FILE NAME
      C
            IMPLICIT INTEGER (A-Z)
            BYTE INFILE(20),OUTFIL(16),IARRAY(512)
            EQUIVALENCE (INFILE(5),OUTFIL)
            DATA INFILE/'I','N','P',':',16*0/
      C
      1     IFIRST = .TRUE.
            TYPE 1000
      1000  FORMAT('$FILE NAME: ')
            ACCEPT 1001,OUTFIL
      1001  FORMAT(16A1)
            OPEN (UNIT=10, NAME=INFILE, TYPE='OLD', ACCESS='DIRECT',
           + RECORDSIZE=128, ASSOCIATEVARIABLE=IVAR)
            OPEN (UNIT=11,NAME=OUTFIL,CARRIAGECONTROL='FORTRAN')
      C
            IVAR = 1
      10    READ (10'IVAR,END=900) IARRAY
      C
            CHRPTR = 1
      20    IF (CHRPTR .GT. 512) GOTO 10
      C         AT END OF BLOCK, READ ANOTHER
            IF (IARRAY(CHRPTR) .EQ. "136) GOTO 10
      C         END OF DATA WITHIN THE BLOCK
            IF (.NOT. IFIRST) WRITE (11,1002)    ! MAKE A LINE FEED
      1002  FORMAT( )
            IFIRST = .FALSE.
            DECODE (4,1003,IARRAY(CHRPTR)) LENGTH
      1003  FORMAT(I4)
            IF (LENGTH .GE. 5) GOTO 25
            WRITE (11,1004)
            GOTO 30
      25    WRITE (11,1004) (IARRAY(CHRPTR+J-1),J=5,LENGTH)
      1004  FORMAT('+',128A1)
      30    CHRPTR = CHRPTR + LENGTH
            GOTO 20
      C
      900   CLOSE (UNIT=10)
            CLOSE (UNIT=11)
            GOTO 1
            END
```

DECUS

DIGITAL EQUIPMENT COMPUTER USERS SOCIETY
ONE IRON WAY, MR2-3/E55
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, MR2-3
Marlboro, Massachusetts 01752 USA

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