TT

# DECUS 12 BIT SPECIAL INTEREST GROUP NEWSLETTER

January Number 32 1979

Contributions and correspondence should be sent to:

Robert Hassinger, Coordinator - 12 Bit SIG
c/o DECUS MR2-3/E55 ..or.. Liberty Mutual Research Center
One Iron Way 71 Frankland Road
Marlboro, MA 01752 Hopkinton, MA 01748

DECUS/Europe .contributions are solicited through:

Lars Palmer
DECUS/Europe 12 Bit SIG Newsletter Liaison
Hassle
Fack
S-431 20 MOLNDAL 1
SWEDEN

(Please include reference to Newsletter number and page when inquiring about material published.)

## NEWSLETTER SUBMISSIONS

The Newsletter is currently published bi-monthly in the odd months. The deadline for each issue is the last Friday of the preceding even numbered month. Submissions are accepted at all times and are normally used in the next issue to go to press regardless of date of receipt. The deadline for ready-to-use material for the next Newsletter is 2-March-79 (note the extra week). Material requiring editing/re-typing should be in earlier. Ready-to-use material should use an area 6 1/2 inches (16.5 cm) wide by no more than 9 inches (23 cm) long on each page. It should be single spaced on white bond paper whenever possible and must be reasonably clean, legible and sufficiently dark for good photographic reproduction.

Material submitted in machine readable form is particularly desirable because it can be edited and incorporated into the newsletter format more easily. Higher quality reproduction is also possible this way. Contact the editor (Bob Hassinger) for further details on acceptable media and formats if you plan to make a submission in machine readable form.

## LATE NEWSLETTER

This issue of the newsletter was delayed at the request of the SIG Steering Committee in order to allow inclusion of program information for the Spring Symposium.

©1979, 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.

## SIG COMMITTEES AND WORKING GROUPS

### Steering Committee:

Robert Hassinger - address above - (617) 435-3452

Jim Crapuchettes
Menlo Computer Associates, Inc.
P.O. Box 298
Menlo Park, Calif. 94025

(415) 323-3009

Lee Nichols
E. I. Du Pont
Experimental Station
Building 357
Wilmington, DE 19898
(302) 772-3839

Jonathan Lockwood
Harris Semiconductor
PO Box 883
Melbourne, FL 32901
(305) 724-7542 M.S. 54-40

Special Steering Committee Advisors:

Tom W. McIntyre

Stan Rabinowitz

### RTS-8 Working Group and US Symposia Committee Representitive

Lee Nichols - see above

### Micro-8 Working Group

Jonathan Lockwood - see above

### COS-310 Working Group

Lawrence H. Eisenberg
17141 Nance Street
Encino, California 91316
(213)-788-0351

## FROM LEE NICHOLS

### SPRING DECUS SYMPOSIUM IN NEW ORLEANS

The DECUS Symposium Committee met on January 10 to schedule the Spring symposium. The 12 BIT SIG and the DEBUG SIT representatives coordinated the scheduling of their respective sessions to minimize conflicts for the various PDP-8 users: OEM, commercial, industrial, word processing, etc. Below is a condensed list of the sessions of general interest to the PDP-8 community along with a brief description of some, to provide the flavor of the Symposium. The varity of sessions, workshops, panels, and user presentations, will provide all PDP-8 users with an informative, and productive Symposium.

## Tuesday, April 17

Introduction to DEC and DECUS for New 12 Bit Users
This session will prepare the new PDP-8 user to get the most benefit from the Symposium.

12 BIT Road Map
An introduction to the scheduled sessions and the people in the SIG and DEC who are responsible for the PDP-8 presentations.

DEBUG Road Map

Advanced COS 310 Programming

12 BIT SIG Meeting and Short Notes

INFPAK - An Affordable Information Retrieval System

PDP-8 Product Panel
A description of product futures covering the whole PDP-8 family: OS/8, OS/78, COS 310, and word processing.

COS 310 Workshop

OS/8 and OS/78 Workshops

All About DIBS Session

## Wednesday April 18

EDP Application Session
A detailed comparison of Commercial BASIC and DIBOL: the strengths and weaknesses of each language.

Use of RTS/8 for WS and WD 200 Systems
How OS/8 and new peripherials are added to the 200 series systems.

DEBUG Swap Workshop

MACREL/LINKER Workshop
A tutorial session in how to use this new assembler

RTS/8 Workshop

Differences Between DIBOL-8 and DIBOL-11

Micro-8 Sessions
Application, use and generation of development systems using the 6100 processor chip.

Thursday April 19

Word Processing Sessions
Details of new software and communication options for COS 310 and WPS systems.

COS 310 Insurance Agency Package

Commercial OEM and Field Service

Firday April 20

COS 310 Communication Products

Commercial OEM Competitive Workshop

COS Under OS/8

DECUS ACTIVITIES IN GERMANY

Rudi Stange writes: "Dear Bob, may I take the liberty to inform you and the readers of your Newsletter that Germany has, at last, formed its own DECUS group under the name of "DECUS Munchen". The chairman is Dr. Thomas Kokott from the Physics institute of the Bonn University.

"At the same time nine different SIG's were formed: 12-Bit-SIG, 10/20-SIG, BIOMED-SIG, EDU-SIG, RSX11-SIG, 18-Bit-SIG, 16-Bit-SIG, RT11-SIG, and micro-11-SIG.

"The second German Symposium will be held March 28, until March 30, 1979 at the University of Bonn. Registration will begin February 1979."

Rudi enclosed a copy the first issue of the DECUS MUNCHEN BULLETIN, a very nice 12 page typeset publication. Although I do not read German, it is easy to see that the 12-Bit section is the the most extensive. The following address is listed for the 12-Bit-SIG: Dipl.-Ing. H.P. Stoehrel, Institut fur Modellstatik, Universitat Stuttgart, Pfaffenwaldring 7, D-7000 Stuttgart 80. Rudi's address is: R605, Digital Equipment GmbH, Wallensteinplatz 2, D-8000 Munchen 40, West Germany.

DECUS PROGRAM LIBRARY NEWS

The following is information on recent submissions and revised writeups in the DECUS library. Check with them for further information.

DECUS 8-825 and 12-212 - ALPHA V03C- Revised 20 Dec 1977 with changes for OS/8 V3D plus several new features.

DECUS 8-861 - BASIC User-Defined Functions And A Multi-Channel Data Acquisition And Control System For Mass Spectrometers - This version (called USER5) is a revision for OS/8 V3D and OS/8 BASIC V5.

DECUS 8-754 - NUMBER and REDATE - Revised to conform with OS/8 V3D. REDATE now accepts up to 5 input specifications, which may include 'wildcards'. In addition, there are switches to select input files by their having today's date or no date at all. Output options include put on today's date, ask me (i.e. specify date word in octal) and get date from keyboard for each file to be redated.

DECUS 8-876 - OS/8 System Device Handler For Sykes 7520 Floppy Disk - Packs two 12 bit words into three eight bit bytes on a Sykes 7520.

DECUS 8-881 - DEC-10 System Controller - This is Bob Phelps DEC-10 that has been documented in the Newsletter before. It allows an OS/8 based PDP-8 system to act as a terminal on a PDP-10. Simple commands allow files to be transferred between the 10 and the OS/8 devices on the 8. Available on both DECtape (along with 8-879 and 12-213) and on LINCtape (along with 8-897). Requires 12k and KL8J type interface for connection to the 10.

DECUS 8-884 - CHISQR chi Square Test With Yate's Continuity Correction - Written in FORTRAN-II, needs 8k.

DECUS 8-886 - ADC: A general Purpose Analog To Digital Conversion System For Processing Biological Data - Keywords: AD Conversion, Signal Collection, EEG Data Collecton; written in PAL-8, needs 16k, converters and real time clock.

DECUS 8-887 - FUTCCL: CCL Modifcation - Binary overlay for OS/8 V3D CCL to implement the FUTIL command.

DECUS 8-888 - TKPLOT: Graphic Display On Tektronix Terminal - Enables a Tektronix 4010 Series terminal to be used as the plotting device with a PDP-8 or 12 under OS/8 FORTRAN IV. The Graphic Input capability is used to annotate the display from the keyboard. A Hard Copy Unit may be used under program control. A file may be designated to receive all the information necessary to reproduce displays on other graphic devices. An incremental plotter may be used to reproduce the saved displays. Written in RALF, PAL8 and FORTRAN IV, requires the FORTRAN IV Plotter Routines and the Tektronix Terminal cannot be the console terminal.

DECUS 8-897 - EDUSYSTEM-25 BASIC Patche - ODT patch to EDU25 (DEC-S8-ED25B) to cause distinction between permanent and non-permanent programs and data files to be transparent for most operations. The use of the $ symbol becomes unnecessary, except for operations in which the permanent program or data file of a given name must be selected in preference to a non-permanent file of the same name. Writeup only is enough, it lists the ODT patch and explains it.

DECUS 8-896 - RESEQ.PA - 16k PAL8 program to replace OS/8 BASIC program for resequencing BASIC program lines numbers. "Ten times faster and vastly more reliable".

DECUS 12-208 - DIAL-MS Utility Programs and DIAL-MS for the RK8F - Revised 11 March 1978 - Corrections for the DIAL system area (SPR files), DIAL utility programs or modifications of other available programs, and a 12k version of "FOCAL-12" which has 23 added functions.

DECUS 12-213 - "LIFE" for the PDP-12 - A version of the popular game of LIFE written in PAL8 for an 8k PDP-12. Will use a 32 by 32 or a 64 by 64 grid.

DECUS 12-214 - VERIFY: LINCtape Format Verification Program - PAL-12, 8k.

DECUS FOCAL8-301 & 12-216 - U/W FOCAL V1A - This is the early version of U/W FOCAL (see notes elswhere in this issue about more recent versions available from author) with OS/8 V3D date fix.

DECUS BASIC8-106 - LAB 8 Evoked Potential Analysis Programs - Uses LAB-8e system, OS/8 BASIC, DECUS BASIC-56 overlay, written in BASIC.

## THE VT-100 AND OS8/OS78

Most of us got our first chance to see the new VT-100 terminal at the Fall DECUS Symposium. It is priced just about the same as the VT-52 and has many new features. It looks as though DEC is going to phase out the VT-52 and entirely replace it with the VT-100. It seems as though new systems are being supplied with it and some people were saying that already they could not order systems with the VT-52. This should be just fine because you can set the VT-100 to run in "VT-52 compatible" mode. What this turns out to mean is that the same control characters and escape sequences work in the same way on both terminals. Life never seems to be quite that simple however!

The symposium was the first chance anyone, including the software development and maintanance people, had had to try a VT-100 with OS8/OS78. There were some problems that showed up. Under some conditions the output would garble or the programs would get bad input using the new terminal. All the results are not in yet but this is what I have found out so far: The most recent releases of OS8/OS78 have been patched to work with the VT-52 ("scope mode"). This means that the TTY handler recognizes control-S and control-Q to start and stop output to the terminal and the backspace, space, backspace form of rubout is implemented in both the TTY handler and elsewere in the system software. The fact that control-S and control-Q support is not implemented in several parts of the system that do their own terminal I/O (as opposed to using TTY:) seems to be source of the trouble.

It appears that while the VT-52 and VT-100 both use ^S and ^Q in the same way, the VT-100 tends to use them much more. Any time the terminal cannot keep up with the output it is receiving, it sends ^S to delay further output until it catches up then it sends ^Q to restart the output. When the VT-100's"smooth scroll" feature is on, for example, ^S and ^Q seem to be used extensively to slow down the effective output rate to what was reported to be about 600 baud. It may be that even during VT-52 style scrolling operation, the terminal does not really accept output at the full 9600 baud continuously.

The terminal at the DECUS symposium was connected to a PDP-8A using an interface set for 9600 baud. Apparently, when the terminal tried to control the flow of output from programs that do not support ^S and ^Q (like the Keyboard Monitor, Command Decoder, ODT, etc.) some of the output would be lost or garbled and the program would get the control characters the terminal was sending and not know what to do with them. It looks as though this has not been a problem with the VT-52 because it rarely tries to send the control characters to these programs because they typically output only a little at a time, giving the terminal time enough to keep up.

Steve Root called, after the Symposium, to say that he had tried a VT-100 running at 4800 baud and that as long as the smooth scroll feature was turned off, OS/8 worked for this speed. I guess this is OK for now as a way around the problem, but a permanent restriction of the console terminal speed to less than 9600 baud is very undesirable. However, I anticipate that, if the above analysis is reasonbly acurate, a complete fix may be very hard to implement. It took some doing to fit in the patches for rubout support in several of the monitor areas. Finding room for full ^S and ^Q support in the monitor could prove to be very difficult.

One way around the problem that users might experiment with is to run their system under RTS8 using the OS/8 background task. The RTS8 Version 3 release will have support for using a single terminal to control the system and communicate with OS/8 so now, finally, single terminal configurations can use the OS/8 background feature. You should be able to modify the terminal I/O code in the OS/8 support task to handle the ^S and ^Q problem outside of OS/8 proper.

Incidentally, if you have RTS8 and enough core, this is one of tne easiest and cleanest ways to switch redirect OS/8 console terminal I/O to other than the standard 03 and 04 device codes. This way you do not have to go around patching the monitor, all the device handlers and every program that does direct I/O to the terminal with TSF; JMP .-1; TLS, etc. It is also a good way to intercept other I/O requests and redirect them to alternate devices. For example, I have a line printer interfaced through the paper tape punch device code so none of the programs that do direct line printer I/O (like OS/8 EDIT, etc.) can access it because they are coded to use the standard 66 device code. A line or two of code modified in the OS8 background support task and a version of the RTS8 LPT driver task modified for my printer interface fixes the problem.

## TECO NEWS AND VTEDIT

I had a long talk with Stan Rabinowitz a few days ago about the future of TECO. Stan has been the principal maintainer and developer of OS/8 TECO at DEC for several years. He has also been one of most active proponents and defenders of TECO on all DEC systems. Stan is now moving out of the PDP-8 software group to go on to bigger and better projects elsewere in DEC.

Since the last release of TECO (with OS/8 V3D last year), Stan and others have been adding features, some potentially valuable to a great many users. The problem is this: There are no near term plans for a software release through which the new version of TECO could reach the users. Pending approval of everyone concerned, Stan hopes to be able to submit TECO V6 to DECUS to get around this problem. Hopefully he will be able to submit the documentation and sources also so that the user community can maintain the advanced version and continue to enhance it. Stan has preliminary indications that he will be allowed to do this, however, long experiance suggests that we wait and see what happens.

In the meantime, Stan anticipates that copies of TECO V6 and VTEDIT.TE will be available at the Spring Symposium to try and to copy.

TECO V6 contains VT52 support code (written by Jim Roth). An explanation of what this is, and how to use the VTEDIT editing macro, is given below. (Note: I have noticed that the code contains a section that is intended to also provide VT05 support. I have not seen this work yet though and have no reports as yet on it. - R.H.)

TECO V6 also has the following additional features:

1. All patches to TECO V5 have been included.
2. Symbiont support has been added, so that this version of TECO can run with OS/78 version 2.
3. The ^P command and the ^P execution time command have been removed.

4. :ER and :EB work. These return a value, -1 if the file was found, 0 if the file was not found.

5. F has been put back in.

6. The ^L command has been implemented. Upon execution, it types a form feed.

7. The ^A command may now be @-sign modified.

8. The 2ED command causes Y and _ to always work (i.e. it disables Yank protection). 0ED brings Yank protection back, and is the default condition.

9. The search buffer has been expanded by 7 characters (except on PDP-12's or systems using VT52 support).

10. ET bit 4 (128's bit) if set, causes TECO to abort (return to OS/8) after any error. This bit is always cleared by TECO when it prints its prompt.

11. "= is now the same as "E .

12. ET bit 3 (256's bit) if set means that VT52 support should truncate displayed lines to the width of the terminal (80 characters). The default is that this bit is not set, which means that lines greater than 80 characters in length overflow onto the next line on the screen.

13. ET bit 2 (512's bit) is a 1 initially if VT52 support is available.

14. VT52 support is made available if the OS/8 monitor scope bit is set (via a .SET TTY SCOPE command) and if your terminal is a VT52 or a VT100 and if you have 20K or more (or if you have 16K and BATCH is not running).

15. When VT52 support is available, the W command becomes usable.
-1W creates a window into the text buffer (refreshes the screen).
    0W resets cursor line to default position.
    nW if n>0 sets cursor to line n.
    nW if n<-1 'forgets' top |n|-1 lines of screen (and causes them to be refreshed on the next -1W).

16. If VT52 support is available, then bit 0 of the ET flag if on means trap CTRL/C. A typed CTRL/C will be read as any other character. This bit is reset to 0 by TECO's prompt and by the acceptance of a CTRL/C.

17. The ER command now accepts a /S switch to mean ignore end-of-file on this file. (SUPERTECO mode)

18. ED bit 1 if off means that an up-arrow in a search string has special meaning. It means convert the next character to its corresponding control character. The default is that this bit is on.

19. ^R has the same meaning as ^S in search strings.

20. When TECO is chained to, before parsing the command given to it, it will effectively execute the teco macro in file SYS:TECO.INI if one is present. If this macro returns a value, then Q-register X will be executed after the command is parsed.

VTEDIT.TEC is a TECO macro that was origianaly written by Herb Jacobs and Mark Bramhall for use on the PDP-11. It has been modified by Jim Roth and Stan Rabinowitz for OS/8 TECO V6. This macro provides immediate mode support for the VT52 and VT100 video terminals (updates occur as they are entered). This macro makes editing with the VT52 extremely easy and accurate, but will initially take some editing to get used to. This macro is very complete and allows for almost any editing situation. Before VTEDIT will work, assembly language support for it must be built into TECO. This is what Jim Roth has done for V6. See items 13,14 and 15 above.

VTEDIT.TEC resides in q-register I. It may be loaded by ERVTEDIT.TEC$YHXIHK$$. Whenever TECO will accept a command MI$$ is used to enter VTEDIT. (Note: the TECO.IN feature (item 20 above) lets you start VTEDIT automatically. A command like .TECO FOO.PA will start TECO, load and start VTEDIT, and do the normal EB and Y on FOO.PA if the correct TECO.IN file is present on SYS:)

Any characters typed to VTEDIT are immediately put into the text buffer at the current position of the text pointer, except for the single character commands described below. Because the typed text is immediately displayed in the text buffer VTEDIT stops terminal echo.

The commands for VTEDIT.TEC are:
(named character commands are not on the keypad)

| Keypad Character | Teco equivalent | Description |
|---|---|---|
| -> | C | Move text pointer forward 1 character |
| <- | R | Move text pointer backward 1 character |
| 0 | L | Move text pointer down 1 line |
| 1 | 0J | Move text pointer to beginning of text |
| 2 | ZJ | Move text pointer to end of text |
| 3 | 0L | Move text pointer to begining of current line |
| 4 | -L | Move text pointer up 1 line |
| 5 | D | Delete 1 character following the text pointer |
| 6 | | Deletes string just searched for |
| 7 | | Insert 1 blank line and position before it; esthetically pleasing for inserting text lines |
| 8 | P | Perform 1 page command |
| 9 | | Read the next input character literally, not as a command |
| Delete | -D | Delete 1 previous character |
| Backspace | L2R | Position the text pointer at the end of the line |
| Control-U | 0K | Delete the text from the begining of the line to the text pointer |
| Control-K | K | Delete the text from the text pointer to the end of the line. |
| Control-D | | Removes the text from the text pointer to the end of the line. |
| Control-C | | Exit from this macro to normal TECO mode The macro is started again by MI$$ |
| Control-Z | | Same as Control-C but no error message is given. |
| ENTER | | Accept a new search argument to be used in conjunction with the keypad '.' key. This argument may be edited with DELETE and Control-U and is terminated with the keypad '.' key which performs the search. The search string may not end with an altmode, but may have embedded altmodes. |
| .(period) | | Search for next occurence of ENTERed argument If search fails, text pointer is moved to top of page |
| ^(uparrow) | | Move the text pointer up 1 line to the same column |
| v(downarrow) | | Move the text pointer down 1 line to the same column |
| blue key | | Save 1 line of text for moving, each successive key strike adds another line to movable portion; the first line saved is from the cursor to the end of line |

| | |
|---|---|
| grey key | Retrieve saved text (from blue key) at current text pointer position |
| red key | Enter extended command mode |

Extended command mode allows any TECO command to be executed directly while remaining in VTEDIT. The text buffer is not affected except as the possible result of the executed TECO command. The TECO command to be executed is displayed at the top of the VTEDIT screen and allowed to be edited with the following commands:

| | |
|---|---|
| Control-U | Abort extended command mode, return to normal insert mode. Nothing is executed. |
| Delete | Delete previous character entered. |
| Escape Escape | Execute the command just typed and then return to normal insert mode. Commands are executed with the text pointer at its last position in normal insert mode. If the command causes an error, INSERT has to be continued by executing MI$$ . Executing another macro is O.K. as long as it doesn't destroy any of VTEDIT's q-registers. |

A numeric argument (of the form ESC number) may precede the following commands:

Paste, open line, page, quote, up, down, left, right, up line, down line, delete char, search.

This argument makes the obvious modification to the command (i.e. the command is repeated the number of times specified).

I have used a variant of this macro on our PDP-11 with the VT-11 graphics display for a long time with very good results. It is very easy to teach, powerful and still gives access to the full capabilities of TECO if you want them. The most compelling reason to be interested as far as I am concerned is that, with this macro, the same easy to teach and use display oriented editor is avaiable under several operating systems on both PDP-8s and PDP-11s (and VAX 11/780, too, I think). Most PDP-8 users have never had the luxury of a display oriented editor and will not appreciate how nice it is until they have a chance to see one run. You see about 20 lines of your text buffer centered on the current location of the editing pointer which is displayed as a cusor. You just type to enter new text with the additions added to the display as each key is struck. As you type the special keys above, the indicated changes such as deleteing characters, etc. take place immediately and are display as you go. There is no need to keep a mental picture of what you are doing or to keep giving commands to output one or more lines to see where you are or what has been happening.

If only for the sake of getting VTEDIT out to the users, I certainly hope Stan is successful in his efforts. When this software is combined with a VT52 or VT100 equipped PDP-8 and OS/8 RUNOFF (discussed in the last Newsletter), a very powerful text processing capability results that is easy to teach to non-computer oriented people like secretarys and the great mass of people out there who need to generate reports, theses, manuals, newsletters, and so on.

## U/W FOCAL

In November, Jim van Zee sent along a copy of the release notes for his U/W FOCAL Version 4E. Unfortunately, it arrived just a few days after the last issue of the Newsletter went to press. Because DECUS is now exerting great presure on me and the other newsletter editors to control and reduce our page counts, I feel I cannot include the entire document. Here is a brief extract however:

"Version 4E, a minor revision of V4D, is the last planned release of UWF. It incorporates one major new feature: The treatment of '=' signs as arithmetic operators, and a number of other minor improvements such as a 'character value' operator and XON/XOFF (i.e. ^Q/^S - RH) control of the terminal. Operation of the standard version under the MULTI-8 or MULTOS/8 timeshare systems is now guaranteed and there is also a special version available for use with the ETOS system. The 'USER' command in V4C (January '78) has been dropped in favor of an expanded form of 'QUIT' and 'JUMP' has been given an additional duty required by the non-OS/8 versions. Speaking of these, the non-system versions have now been integrated into a single binary which contains all options and allows automatic expansion from 8-32K using the Switch Register to select various features (nice for 6100 systems). Back in the OS/8 world, the format of the Output Date command has been changed to 'day.month.year' to agree with the new form of the monitor .DATE command (and to please my European users).

"For those who missed the upgrade from Versions 3R/3S (February/April 1977), the arrival of Version 4 (September 1977) brought significant changes to UWF's 'addressing' capabilities. We now have 'relative line numbers' which permit 'group-independent' coding, 'negative line numbers' which permit 'sub-group' and 'multiple-entry point' calls, and a line number option in the RETURN command which turns 'DOs' into 'GOTOs'. The '@' character was added to the command table and a 'user error trap' facility (modified in V4D) installed. Various other improvements to keep abreast of changes in the monitor system were also added: support for V3D date algorithm and use of 'scope mode rubouts' for video terminals. Somewhere along the way operation under BATCH and/or RTS8 became possible.

"Documentation: A complete manual for V4C and one set of summary cards were printed and have been partially distributed. Editing difficulties with the main manual have shifted most of the newer material into the 'beginners' section which is currently being reprinted as a separate volume. This document specifically describes the non-OS/8 version, but all versions are identical, except for the 'L' and 'O' commands and the addition of a few system-related functions in the OS/8 version. New summary cards are required since the rewrite of 'EVAL' necessitated quite a few error code changes."

...

"With this release I am ending further development of UWF - you will finally have a stable version! It has been fun, and a struggle too, during the last six years or so, and there are things left undone, such as an improved scheme for allocating handler space, string variables, logical IF's, etc., etc.

"However, what we have is very thoroughly debugged, quite powerful I think, and fairly well documented. I have learned something about the difficulty of maintaining programs and realize the effort it would require to implement even just one of the items on the 'wishlist' above. Before giving up, however, I wanted to add the replacement operator since that seemed to have such far-reaching implications, yet fit within the current framework of the language.

"I would like to thank all the users who have welcomed these developments and who, by their own enthusiasm for such things, have supported me over the years. Many of you I have now met personally and the pleasure of your acquaintance is certainly part of the reward. I will be available at least through next June to answer questions or to supply updates. Please address correspondence to me in care of: LAB DATA SYSTEMS, 10230 Ravenna Avenue N.E., Seattle, Washinton 98125."

## FILE TRANSFER BETWEEN OS/8 AND RT-11 ON RK05 DISKS

About two years ago we developed a need to transfer files from our OS/8 system to a newly installed RT-11 system. The only common media between the systems was RK05 type disks. There was no information available on how to make such transfers, however. After considerable study of the problem, a relatively simple approach was selected. In response to several inquiries, the following outline of how this scheme works is offered.

Media: The RK05 type disk is a single platter, two surface, hard sectored disk. The hard sectors are implemented as slots in the hub that define the start point for each sector on the disk. The media for the RK-8e and the RK-11 differ physically in just one respect. For the PDP-8 the disk is divided into sixteen sectors. For the PDP-11 the disk is divided into twelve sectors. Other than this difference, implemented as alternate hubs, the media are physically identical.

Disk Format: When a new disk is first used, a special data format is written on it to define the sectors for the hardware. In the cases of both the RK-8e and the RK-11, this format is approximately as follows. Immediately after a sector slot in the hub is detected, a header area is established. This header contains various synchronizing information and bits to identify the cylinder and sector. The numbering of the cylinder and the sector are separate. This proved to be very convenient for our purpose. The cylinders are numbered from the first to the last in sequence and the sectors are numbered starting from zero on each surface of each cylinder. After the header area is the data area. It is recorded as a serial stream of bits. In the case of the RK-8e there are 256 times 12 bits (3072 bits) in the data area. In the case of the RK-11 there are 256 times 16 bits (4096 bits) in the data area. The total data on one track is 12 sectors times 4096 (49152 bits) for the RK-11 and 16 sectors times 3072 bits (also 49152 bits) for the RK-8e. Following the data area in each sector is a checksum word. On both the 8 and the 11 this is a sixteen bit CRC checksum of the same type DECnet uses. The hardware automatically computes and records the checksum when doing a write and it reads and checks it when doing a read. The fact that the same CRC checksum is used by the 8, the 11, and DECnet proved useful in minimizing the work involved in this project.

Problem: The problem in transfering data using the RK05 is this. On the one hand the RK-11 controller hardware is physically incapable of addressing sectors beyond the twelfth on each track so it is impossible for it to access one forth of the sectors on a PDP-8 type disk. On the other hand, the RK-8e controller hardware is physically incapable of reading or writing more than 3072 bits of data in a sector so it cannot read or write the last quarter of a PDP-11 data area or the CRC checksum which follows.

In our case no hardware changes were to be made due to a lack of technicians and a desire to avoid possible maintenance difficulties with DEC. The solution chosen also had to minimize the software effort required due to limited resources and higher priorities. Therefore, design tradeoffs requiring less programming effort which resulted in slower, less convienient operation were acceptable.

Hardware Solution: The solution to the problem of physically incompatible disk formats and controller hardware selected was as follows. We defined a disk format that accommodates the PDP-11 hardware by only using 12 sectors and which also accommodates the PDP-8 by only using 3072 data bits per sector with the CRC checksum immediately following. This format is used on PDP-11 type (12 sector) disks so that there is enough room in the sector for the RK-11 to write its full sector size (4096 bits) without spilling over into the next sector.

Software: With these decisions made, we are left with the question of how to use this "hardware compatible" disk format. On the OS/8 side we note that each sector is exactly one OS/8 block in size. By simply implementing a device handler that knows there are only twelve sectors per track (rather than the normal 16), the special disk can be accessed as a normal OS/8 device. The only special consideration is that the length of the device is different from a normal OS/8 RK8e disk. Our implementation was a cheap and dirty, one hour effort that added a second page to a copy of the standard OS/8 RK-8e non-system device handler. An elementary divide by twelve routine was located in the second page and patched into the normal handler code in place of the usual divide by 16 code that computes the cylinder and sector numbers from the requested block number. The handler worked the first time and has never been touched since. Because it is a non-system handler, an alternate system device is needed to use it. In our case we have only one RK drive so we use DTA0: as SYS:. A more elegant solution would be to speed up the handler (it seems to use a full disk revolution per block due to the time to do the divide by twelve) and to make it into a system device handler. All the standard OS/8 software such as FOTP and DIRECT works normally with this handler. For fully correct implementation, OS/8 PIP should be patched to reflect the correct length of this device although this has not been a problem so far. I have chosen to call this handler "ZZ" (a name that is compatible between OS/8 and RT-11).

On the RK-11 side the situation was more involved. Fortunately RT-11 FORTRAN was available and, with SYSLIB, proved to be able to do everything that was needed. The result is slow and has pointed out a number of deficiencies in various releases of the compiler but I doubt if we could have afforded the time to to do the job in MACRO-11. The first problem to be addressed is that the CRC checksum calculated and written by the RK-8e controller falls after the first 3072 data bits in the sector. As far as the RK-11 controller is concerned, this is still part of the data, with a quarter of the sector yet to go. It does not know the sector has ended, so it goes on reading bits and forming them into words until 4096 bits have been read. Then the next 16 bits are taken as the CRC checksum. Naturally, this turns out to be invalid. Fortunately, the hardware still allows the data read to be transferred to memory and just reports the error to the handler. It proved possible, using SYSLIB routines, to read such a block of data into a FORTRAN program and to anticipate and ignore the error that was reported back. A software CRC routine looks over the received data and computes the correct checksum for the first 3072 bits then it compares this against the next 16 bits which is where the RK-8e wrote its checksum. The balance of the data block is ignored. The write routine (which has been coded but not tested) simply sets up a block of 3072 bits, computes a CRC, puts it in the next 16 bits where the RK-8e hardware will look for it, clears the rest of the RK-11 size sector, and writes it.

The RK-11 writes a complete 4096 bit sector followed by a checksum computed by the hardware. The software computed checksum is in the middle of the data block, at the point the RK-8e will look for it so that when OS/8 reads this block, the RK-8e will find exactly the checksum it expects and it will respond just as if the block had actually been written on a PDP-8.

A routine was needed to convert between the RK-8e and the RK-11 addressing schemes for the disk. This is due to the fact that the numbering of the surfaces of the disks is reversed in the two systems. A more involved problem was figuring out how to unscramble the bits, bytes and words. The conventions for how data is mapped onto the disk are different in almost every possible respect on the two machines. It was very difficult to fully solve this puzzle analytically so the eventual solution was to write known, identifiable patterns of data on a disk with FUTIL on the 8 then to look at what came out on the 11. Eventually analysis plus trial and error found the right combination. A routine was coded in FORTRAN to go from what RT-11 delivered to FORTRAN into correctly sorted out PDP-8 words. That is, 256 scrambled 16 bit words go in and 256 12 bit words, (one per 16 bit integer) come out. One more routine is needed to do the 3 bytes from 2 words unpacking of the OS/8 ASCII format. This takes 256 12 bit words (one per 16 bit integer) and returns 384 8 bit bytes. These routines, like all of the PDP-11 code, were written in FORTRAN which is slow and reluctant to do the operations needed for bit masking and shifting. Upwards of one second per block is now needed to unscramble, reformat, and write a block to an RT-11 file. For low volume work this is acceptable but it needs to be improved. Obviously these routines should be recoded in MACRO for speed but we have been able to tolerate the slower FORTRAN implementation so far.

Given the facilities listed above, it was a simple job to write two programs to use them. The first, "ZZDIR", reads the OS/8 directory blocks and prints a formated directory of the disk. The second, "ZZ2RT", accepts requests to transfer files from the "ZZ" disk to arbitrary RT-11 files using more or less the same command syntax (i.e. standard CSI) as RT-11 PIP. With a little effort to write a routine to do ENTERS and CLOSES in the OS/8 directory, an "RT2ZZ" program could be written if someone needed to get back to the 8 from the 11. Rather than write programs for doing DELETEs and SQUASHes on the 11, we just use the standard OS/8 utilities for all our file maintenance operations on the "ZZ" disks.

The disks used by this package are standard PDP-11 type RK05s that have been formatted using the standard PDP-11 procedures. They must then be ZEROed in the usual way under OS/8 to establish an OS/8 directory on them.

In the present implementation, it is impossible for the "ZZ" disk to be an RT-11 system disk so an alternate system device is required. We use RK0: for the system device and RK1: for the "ZZ" disk. On other configurations appropriate provisions would be required.

Conclusions: Given appropriate hardware configurations, this package successfully moves OS/8 files to RT-11 and could be extended with some software work to move files back from RT-11 to OS/8 on RK05 type disks. It uses an approach that maximizes the use of standard, existing software and minimizes the effort required for specialized software.

Where fast serial or parallel interfaces are available for interconnection on both machines and they are close enough together, direct, "network type" transfers would be more suitable and faster. Appropriate software (i.e. DECnet for example) is not yet generally available for OS/8 to RT-11, however. Certainly, we are not aware of any suitable software in the public domain at this time.

Where the distance between the communicating machines is great enough to preclude high speed connections, and the amount of data to be transfered is sufficient, the effective bandwidth and reliability of physically moving data via RK05 disks (or other media such as magtape) remains very attractive. For example, the "ZZ" format disk holds a little over 1.5 megabytes. If that much data were transferred over a 300 baud phone line it would take over a month, 24 hours a day, seven days week assuming a better than average phone line and effective error checking and recovery with low network overhead. At that rate, an airplane trip to deliver a disk or tape is faster and probably cheaper! Where the long term volume does not justify magtape for both machines, data transfer via disk is relatively fast, cheap and easy. Only the availability of compatible, DECnet class facilities for OS/8 and RT-11 at low cost would induce us to abandon the present scheme. The near term prospects for such a capability look dim. DECnet is much too expensive and getting more so and DECs enthusiasum for implementing a fully functional, compatible version for the PDP-8 seems discouragingly suspect.

## NOTE FROM DICK MURPHY

"Here are two patches which might be of use for users of RK8e/RK05 OS/8 systems. The first is a modification of the system handler to avoid tampering with the date on bootstrap. The second change allows the use of the "B" portion of a disk as a system device (i.e. RKB0, RKB1, etc.).

The easiest way to install the patch for the date change is with a source modification. For those of you who do not have the source of the handler, the change can be installed with FUTIL as shown below:

> [On the below, all user input is underlined, the character string <lf> means line feed, while <cr> means carrage return.]

```
.R FUTIL<cr>

SET DEVICE SYS<cr>

0.2/ 1414  7005<lf>
0000.0003\ 6211 1013<lf>
0000.0004\ 3415 7640<lf>
0000.0005\ 6201 5000<lf>
0000.0006\ 1013 1414<lf>
0000.0007\ 7640 6211<lf>
0000.0010\ 5000 3415<lf>
0000.0011\ 5437 5016<cr>

0.16/ 0000  1015; 1040; 7640; 5024<lf>
0000.00022\ 0000 2014; 2015; 6201; 2041; 5006; 5437<cr>
0.40/ xxxx  0113<lf>
0000.0041\ xxxx 7600<cr>

WRITE<cr>
EXIT<cr>
```

Note that this patch uses an operate microcode (IAC RAL) which is illegal on non-OMNIBUS machines (i.e. 8/I, 8/L, etc.). This can be hacked out with the following change:

```
0.2/1414 7001 (was 7005 above)
0.377/xxxx 0200 (for 1978-1986. use 0400 for 1987-1995, etc.)
```

The rest of the patch is the same as above.

There is a way to patch this into BUILD so that the changes can be saved. This way, the system does not need to be patched each time it is rebuilt. To do this, the RK8E handler MUST be the last handler in the BUILD tables. (This is seen when you do a PRINT command to BUILD.) To force the RK8E handler to be last, do the following:

```
.RUN SYS BUILD<cr>
$UNLOAD RK8E<cr>
$LOAD dev:RK8ESY<cr>     (where "dev" is the name of the device where
                          the handler binary (RK8ESY.BN) is stored.)
$INSERT RK8E:SYS<cr>     (and RKA0 and RKB0 if you want.)
$^C                      (control-C, of course.)
.SAVE SYS BUILD<cr>
```

Now run FUTIL to patch in the changes.

```
.R FUTIL<cr>
SET DEVICE SYS<cr>
FILE BUILD.SV<cr>
BUILD.SV ...........
SET MODE SAVE<cr>
STRING 7740,1412,3413,1414<cr>

0000.0YYYY:  7740  1412  3413  1414
```

Where 0yyyy is some address (above 7000 in field 0) where the string desired is found. If there are two or more matches, pick the highest one in field zero.

Now patch the bootstrap block:

```
yyyy/ 7740  7736 <lf>
0000.0xxxx\ 1412 <lf>
0000.0xxxx\ 3413 <lf>
0000.0xxxx\ 1412  7005 <lf>
0000.0xxxx\ 6211  1013 <lf>
0000.0xxxx\ 3415  7640 <lf>
0000.0xxxx\ 6201  5000 <lf>
0000.0xxxx\ 1013  1414 <lf>
0000.0xxxx\ 7640  6211 <lf>
0000.0xxxx\ 5000  3415 <lf>
0000.0xxxx\ 5437  5016 <lf>
0000.0xxxx\ 0177 <lf>
0000.0xxxx\ 7577 <lf>
0000.0xxxx\ 0046 <lf>
0000.0xxxx\ 7646 <lf>
0000.0xxxx\ 0000  1015; 1040; 7640; 5024; 2014; 2015; 6201 <lf>
```

```
0000.0xxxx\ 0000  2041; 5006; 5437 <lf>
0000.0xxxx\ 6741 <lf>
0000.0xxxx\ 5030 <lf>
0000.0xxxx\ 0035 <lf>
0000.0xxxx\ 3436 <lf>
0000.0xxxx\ 5000 <lf>
0000.0xxxx\ 0006 <lf>
0000.0xxxx\ 0342 <lf>
0000.0xxxx\ 7605 <lf>
0000.0xxxx\ xxxx  113 <lf>      (old contents probably zero)
0000.0xxxx\ xxxx  7600 <lf>

WRITE<cr>
EXIT<cr>
```

In the above, all the addresses are shown as "xxxx" as the address of the boot block can vary with the number and types of handlers loaded into BUILD.

The next patch is a simple one-word change to the RK8E handler to allow RKBO to be SYS:. This is done within BUILD.

```
.RUN SYS BUILD<cr>
$ALTER RK8E,13<cr>
5224/ 5223<cr>
$DCB SYS<cr>
4231/ 4401<cr>
$BOOT<cr>
WRITE ZERO DIRECT?Y<cr> (unless you have a system-type directory
SYS BUILT               on RKBO already, which is unlikely.)
.SAVE SYS BUILD<cr>
```

The bootstrap for the RKBO system is different from the normal RK05 bootstrap. The new boot is listed below.

| LOCATION | CONTENTS |
|---|---|
| 0026 | 1032 |
| 0027 | 6746 |
| 0030 | 6743 |
| 0031 | 5031 |
| 0032 | 6260 |

| | |
|---|---|
| 27 | 1032 |
| 30 | 6743 |
| 31 | 5031 |
| 32 | 6260 |

The RKBO system can be invaluable during a crash situation by allowing swift reload of OS/8, allowing examination and correction of the crash.

Here's hoping that some of this mess will be useful to some of you. All of the procedures have been tested, but you can contact me if you have any troubles.

Here is a source compare of the differences in the RK8E system handler for the date patch.

```
SRCCOM V4

1)      /3 RK8E SYSTEM HANDLER FOR OS/8 BUILD
```

```
2)      /4 RK8E SYSTEM HANDLER FOR OS/8 BUILD

1)002           VERSION="C77
1)              *0
****
2)002           VERSION="D77
2)              *0
********
1)003           BOOT-BLAST
1)              RELOC 0
1)      BOOT,   TAD I BOOTX1
1)              DCA I BOOTX2
1)              TAD I BOOTX3
1)              CDF 10
1)              DCA I BOOTX4
1)              CDF 0
1)              TAD BOOTX2
1)              SZA CLA
1)              JMP BOOT
1)              JMP I B7605
1)      BOOTX1, 177
1)      BOOTX2, 7577
****
2)002   /
2)      /2.     MODIFIED BOOTSTRAP CODE TO NOT CHANGE DATE ON
2)      /       BOOTSTRAP.
2)      /       CODE BY R. MURPHY-PDP8 PRODUCT SUPPORT-D.E.C
2)      /                         MAYNARD MASS PK3-2/S20
2)      /
2)003           BOOT-BLAST      /LENGTH OF BOOTSTRAP (FOR BUILD)
2)              RELOC 0
2)      BOOT,   TAD I BOOTX1    /RELOCATE 200 TO 7600
2)              DCA I BOOTX2
2)              IAC RAL
2)              TAD BOOTX2
2)              SZA CLA         /COVER LOCN 07777!!
2)              JMP BOOT
2)      BOOT1,  TAD I BOOTX3    /RELOCATE 47 TO 17647
2)              CDF 10          /MOVE TO FLD 1
2)              DCA I BOOTX4
2)              JMP BTCHK       /SEE IF DONE
2)      BOOTX1, 177             /MUST BE HERE TO BE AUTO-INDEX!!
2)      BOOTX2, 7577
********
1)003           ZBLOCK  30-.    /DSKP GOES OVER 30
1)              DSKP
1)              JMP .-1
****
2)003   BTCHK,  TAD BOOTX4
2)              TAD M7665       /SEE IF 17666
2)              SZA CLA         /WHICH IS THE LOCATION OF THE DATE WORD.
2)              JMP BOOT2       /IF IT IS, SKIP THIS LOCN.
2)              ISZ BOOTX3      /SKIP OVER.
2)              ISZ BOOTX4
```

```
2)      BOOT2,  CDF 0
2)              ISZ BTCTR       /CHECK IF DONE.
2)              JMP BOOT1
2)              JMP I B7605     /BOOT TO OS/8
2)              ZBLOCK  30-.    /DSKP GOES OVER 30
2)              DSKP            /LOADS OVER TOGGLE-IN.
2)              JMP .-1
********
1)003   BLAST,  RELOC
****
2)003   M7665,  -7665
2)      BTCTR,  7600
2)      BLAST,  RELOC
********
```

P.S. I have a version of ADVENTURE for the PDP-8 which I am about to submit to DECUS. It will require 32K of memory, unfortunately."

Dick is with the PDP-8 Product Support, Digital Equipment Corp., 129 Parker St.
PK3-2/S20, Maynard, MA 01754

## PROM PROGRAMMER

R. D. Oestreich writes as follows: "I have noticed a substantial interest in microprocessor development (cross assemblers, etc.) in the past few Newsletters. Perhaps this information will be of use to some of your readers.

"We have developed an OS/8 based PROM programmer with the following features:

1. OS/8 core image format input and output on any 1 or 2 page handler file structured device
2. Console ODT type commands to examine and alter files
3. PROM read, write, test, and compare capabilities
4. Listings on console or LPT: in Octal, Hex or Binary
5. Automatic PROM type decoding from file name
6. Extremely fast burn times

"The programmer hardware consists of an M-1709 board for control logic and interfacing plus an external cabinet-mounted drawer containing power supplies and chip sockets. Other hardware required is an Omnibus compatible OS/8 system with 16k and a real time clock. Line printer and an additional terminal (phone coupler) are optional.

"The programmer presently accepts TI 74S-188, 288, 287, 387, 470, 471, 472, 473, 474, 475 and TMS-2708, 2716, 2732 PROMS. Immediate expansion plans include TI 2516 and Intel 1702 PROMS.

"Depending on the response from your readers, we are considering making the hardware and software available as a package. If anyone interested would write to me, I will respond with some more detailed specifications."

The address is: Hartman Engineering, 66 School Street, Victor, N.Y. 14564 - (716) 924-3075.

## PRINTRONIX PLOTTING

Professor Ronald P. Larkin writes: "I would appreciate your including ... my request to get in touch with anyone having interest or experience in plotting with a Printronix impact matrix line printer on a PDP-8 series computer. I am interested in either (1) writing general-purpose plotting software for the unit or (2) interfacing the Printronix directly with a Tektronix storage-tube display terminal."

The address is: The Rockefeller University, 1230 York Avenue, New York, New York 10021.

## RTS/8 WORKING GROUP NOTES FROM LEE NICHOLS

During the RTS/8 Working Group meeting at the Fall DECUS Symposium, Steve Root described version 3 of RTS/8, which is now scheduled for release at the end of March. A detailed explanation of the new features and changes for version 3 appear below, followed by the current RTS/8 Wish List.

The Wish List and some of the new features for version 3 have resulted from meetings of Steve and the RTS/8 Working Group at the last 4 Symposia. These informal meetings allow RTS/8 users to discuss features they would like to see added or enhanced within the system. The length of the Wish List indicates an active user interest in the continued development of RTS/8. If you have ideas you would like to see added to the list, please send them to me.

At the Symposium, Steve gave me the following RTS/8 V2B files which have been updated with bug fixes. I will distribute these files to RTS/8 users who will send me a formatted OS/8 media (floppy, DECtape, cassette or RK05).

| File name | Edit level and bugs fixed |
|---|---|
| UDCICS | (1) missing indirect JMS |
| OS8SUP | (2) KRB, 2 C's in background |
| RTS8 Exec | (3) Swapper scheduler, powerfail for 8A, EAE, and RX01 |
| MCR | (2) date for 1978+ |
| PWRF | (3) code for RX01 and 8A moved to RTS8 |
| RX01 | (1) powerfail code |
| CLOCK | (3) date, tick race conditions |
| PARAM | (5) date, CLOCK parameter |
| KL8ASR | (1) page overflow for 3 interfaces |
| NIP | (2) date, "skimp" conditional |
| NSP | (2) disconnect/connect, interrupt message |

Since RTS/8 is already an existing product, this functional definition is a description of the changes for version 3. The RTS/8 User's Manual describes the current RTS/8 Version 2B.

### MACREL-LINKER

In RTS/8 Version 3, the assembler-loader pair of PAL8-ABSLDR is replaced by MACREL-LINKER, and backwards compatibility with PAL8 has not been maintained. For this reason, the new features of KT8A support and System Build will not be supported for PAL8 assembled modules.

Conversion of current PAL8 programs for assembly with MACREL will require a variety of fairly small changes. The inconvenience of these source changes is minor compared to the convenence of writing programs in a relocatable rather than absolute environment. For example, the system build described below depends explicitly on the ability to create relocatable code.

## SYSTEM BUILD

A major goal of the Version 3 release of RTS/8 is to make an automated system build procedure, whereby a user of moderate experience can bring up a working RTS/8 system in less than an hour. In contrast, at present it is not uncommon to have an experienced user spend a day to bring up a system. Even then, the system may not function correctly without debugging the build.

MACREL-LINKER is a key component in creating such an automated build procedure. At present, the programmer spends time during the build procedure acting as a linking loader. He must decide where each absolute binary must fit in memory. With MACREL-LINKER, the LINKER now efficiently places the relocatable binary modules into position.

Another key component of the build procedure is a dialogue. At present a parameter file is supplied with the system sources. The programmer must edit the parameter file to create a specific parameter file for his particular system. This editing procedure is a error-prone process, expecially for someone not extremely familiar with the system. Furthermore, debugging an incorrect parameter file by looking at the incorrect system produced by such a file is at best a haphazard process.

RTS/8 Version 3 will have a dialogue, asking the user specific, coherent questions about his intended system configuration. If the user is unsure of a particular question, he may then request additional information. The experienced user may select a less verbose form of the dialogue to reduce his build time.

It is a design goal that answering all questions with a carriage return will produce a viable default system. That default system will be a 16K PDP-8A with clock, MCR, and console terminal. This default system will also include a demo task called TEST.

The build dialogue results in the production of a parameter file, a batch file for MACREL, and a batch file for the LINKER. The parameter file will be similar to the present parameter file. The information contained therein will be more globals affecting linkage than equates affecting assembly. The first batch file can be invoked to carry out these assemblies, and the second to carry out the system load. Thus, when changing an existing system, it might be easiest to re-assemble one module, and run the LINKER batch file. An experienced user who wishes to rebuild his system with a small change also has the option of editing the parameter and batch files directly.

Another important feature of the automated build procedure is an option to provide automatic task priority assignment. In the present procedure, a user must assign a priority number to each task in the system. This is somewhat confusing to an inexperienced user. The build procedure can assign all task priorities in the system.

For a majority of systems, this will be entirely adequate. The experienced user has the option of providing any or all priorities himself. To speed system modification at a later time, a user may provide one or more dummy tasks with priorities. In later development, real tasks can then be substituted in with a minimum of system change. (A system reload is still necessary.)

A user who is creating a system with non-resident tasks must still provide the information as to which tasks share a certain non-resident memory area. However, the LINKER will take care of the problem of placing the non-resident area into the memory map, just like any other system module.

## BACKGROUND-FOREGROUND

There is, at present, an OS/8 support module in RTS/8. At a separate terminal from the foreground terminal, the user can run OS/8 programs in a protected mode. This protected mode prevents a program malfunction under OS/8 from destroying the RTS/8 operating system.

The term Background-Foreground, or B/F, is meant here to describe the shared use of a single terminal by the foreground and the OS/8 background. The terminal will 'belong' to either the background or foreground at any specific time. The typing of CONTROL B shifts the ownership of the terminal to OS/8 (if not already so owned). The typing of a CONTROL F shifts the ownership of the terminal to the foreground. Terminal output generated by the non-owner will hang until the ownership of the terminal is shifted.

A foreground user may have an important message that should break through into the background. He should send the message not to TTY, but to the emergency message facility. See the section on TTY modifications below.

## KT8A SUPPORT

The KT8A is a memory control module that extends the addressing and I/O capability of the PDP-8 from 32K to 128K. The KT8A also provides relocated addressing for a program running in user mode. Furthermore, the KT8A implements in hardware mapped CDF, CIF, and RDF instructions.

This combination of KT8A features means that the OS/8 support task under RTS/8 can be brought up to nearly the stand-alone speed of OS/8. The present performance problem with the OS/8 support task is that the CDF, CIF, and RDF IOT's which are executed by the OS/8 CUSP's must be trapped to the support task, slowing execution speed. In addition, under the KM8E, code residing between a CIF and a follwing JMP must be software simulated at a considerable time cost.

The RTS/8 Version 3 system will support both KT8A memory management and the old KM8E. As a consequence, the RTS/8 executive, the OS/8 support task, and various I/O handlers will have conditional code. The build procedure will provide the appropriate conditionals to the parameter file.

RTS/8 will be supported to a size of 128K. A single OS/8 background of up to 32K will be supported. The backgound may have a real starting address of 8K or larger, so long as the background does not extend beyond the 128K limit, but may not straddle a physical 32K boundary.

KT8-A support includes the issues of linking, loading and debugging RTS/8 images of greater than 32K. MACREL/LINKER V2 will handle the linking task. OS/8 has been modified to provide RUN, GET, SAVE and ODT support of images up to 128K.

## EAE SUPPORT MODIFICATIONS

Under RTS/8 Version 3 there are three separate levels of EAE support. The choice is made at system build time. These levels of EAE support apply solely to context switching. Any interrupt code that must use the EAE is responsible for doing its own save and restore. The system cannot be burdened with a general interrupt save-restore of the EAE.

The first level of EAE support (EAESV=0, MQSV=0) is none at all, that is, none of the EAE data is saved in the job context area. The size of the job context area will remain at the present 4 locations per job. Note, for the remainder of this section, 'saved in the job context area' implies that the appropriate restore operation also takes place.

The second level of EAE support (EAESV=0, MQSV=1) will save only the MQ. Again, there is no change in the size of the job context area.

The third level of EAE support (EAESV=1) saves the entire EAE status in the job context area. The size of the context area will, in this case, be increased to 5 locations per job. For OMNIBUS machines, the MQ, the step counter, A/B mode status, and greater-than-flag will be saved. For pre-OMNIBUS machines, the MQ and step counter will be saved.

For PDP-8A and VT78, EAESV=1 is not allowed, as there cannot be EAE. Note, that the PDP-8A/620 is really a PDP-8e packaged as a PDP-8A.

## CLOCK HANDLER MODIFICATIONS

Several relatively minor changes to the system clock handler have been added.

First, when a clock request is cancelled, the clock handler shall remove that request from the queue. In version 2, the request is disabled, but left in the queue. This has in some cases lead to queue overflow.

Second, an additional clock request type has been added to set a specific event flag, in addition to that of the request itself. This leads to greater flexibility in the use of the clock.

Third, code has been added (conditionally) to the interrupt level such that clocks faster than 5 milliseconds are properly supported. In version 2, such fast clocks overflow the two word clock format before midnight. This solution will have several interrupts be required to make one tick.

Finally, a interrupt linkage facility has been provided. For a given number of hardware ticks, a user subroutine will be called at interrupt level from the clock interrupt handler. At the end of the interval, the linkage will be automatically removed. This feature is utilized at the user's own risk, since the operation of his routine is with interrupts off. Furthermore, the user must exit back via the clock handler to maintain system integrity.

## RL01 SUPPORT

RL01 support under RTS/8 will follow the OS/8 file handling conventions. In addition, a track-sector mode will be provided (no bad block checking) to allow a user to access the entire disk or read foreign media.

## MCR MODIFICATIONS

Several minor modifications to MCR have been made to allow greater system flexibility.

The time functions under MCR has a field for seconds, as well as the present hours and minutes.

The status list printout subsection of MCR has been modified to allow up to 127 system tasks. Other protións of the system can already handle this expansion. Note that this large number of tasks is still entirely optional; the user still controls his number of tasks at build time.

MCR has been modified to allow dates beyond 1977. Dates input from OS/8 function thru 1999, which is the present OS/8 limit. Dates generated thru RTS/8 are good thru 2067.

The MCR Name Table (Task Name Table) has been changed into a DSECT, and uses 6 character task names. The task names are put in the name table by the .Task Macro provided by the System Build.

MCR examine and deposit has been modified to reference 128K.

## SWAPPER MODIFICATION

The RTS/8 V3 overlay facility will be a combination of the present RTS/8 non-resident task facility, and the MACREL/LINKER overlay driver. With the MACREL/LINKER changes, the user no longer has to type in the non-resident task names at start-up time, they are entered during System Build.

The core images of the overlays will be part of the RTS/8 save file created by the LINKER. The LINKER will also generate a separate area to hold each overlay when it is swapped out. Therefore a RTS/8 program will execute any initialization code in the overlays each time it is restarted.

## EXPANSION of TTY and LPT MESSAGE FORMATS

Pressure from users and the MACREL implementation has made it desirable to provide the option that the buffer for a TTY or LPT message be in a field different from that of the executive request itself. In order to provide compatibility with existing software, the TTY and LPT handlers will still accept the old format. The new format is a -2 (impossible in the old format), followed by the extra argument word, followed by the three standard words. The new message field applies to both the input and output areas.

## MONITOR SAFETY FEATURES

A small amount of code has been added to the RTS/8 executive to check for primary calling arguments outside of their normal range. In version 2, such an argument can bomb the system in unpredictable and potentially dangerous ways. Examples include illegal task numbers, non-existent tasks, and out-of-range arguments. In addition, the OS/8 support task will check whether the time-share option switch is properly set before running. If an invalid argument is detected, the current task is suspended and both run wait and non-existant wait bits are set.

## TTY MODIFICATIONS

Modifications have been made to the TTY handler to substantially reduce the overhead (by about a factor of three) of handling a terminal. With 9600 baud terminals in loaded systems, or with the VT78 processor, this reduction in overhead becomes quite significant. The speed up in over-all execution time is achieved at the expense of a slightly longer interrupt latency time.

Incorporated in the TTY source is an additional task (optional at build time). This task is a receiver for emergency messages to TTY (output only). This emergency message task (EMT) will sieze control of TTY when an emergency message request is received. The emergency message will be output. EMT will then restore the TTY handler to its previous state. Emergency messages will come thru independent of B/F ownership, assigns, or TTY input wait. Since emergency messages may break into the middle of an output or input operation, it is the responsibility of the user to provide whatever separators (carriage returns, line feeds, bells, asterisks) seem appropriate.

To alleviate the problem on hung input on TTY suppressing ordinary (as opposed to emergency) output, the following mechanism has been implemented. When a message to TTY expects input, the TTY handler will issue a CLOCK wait. At the completion of the CLOCK wait, if there is another TTY request queued up and no input has been typed, the present input will be aborted. If input is not aborted, the CLOCK wait is started up again. Aborted INPUT will be echoed by U, and terminated by a 4000 in the caller's buffer. This time out feature is a Build time option.

## RX01 HANDLER

The version 2 RX01 handler contains two bugs and a VT78 performance problem, and has been recoded. Version 3 also contains a handler for the RX02 double density drives. The RX02 handler will read and write both single and double density floppies.

## OS8SUP CHANGES

The OS8SUP task has been separated from the OS8F task. A CONTROL Y command has been implemented to reboot the OS/8 background should OS/8 hang. This feature actually reads in the OS/8 head from the system device and reinitializes it. Because the address of the OS/8 background initialization code is saved on startup, squising the system device may cause an error and a "boot error" message. The RTS/8 program must be restarted to recover the OS/8 background.

The device handler status word now passes through OS8SUP correctly, with 4000 octal ORed in when an error is detected.

## OS/8F MODIFICATIONS

The version 2 interlock in OS8F prevents foreground access to any file structured device while the background is using any OS/8 device. The interlock has been changed in version 3 so that only the device OS/8 is actively using is interlocked, leaving other devices available for RTS/8 to use.

The ENTER function has been changed so that if ENTER is called with a zero block length, it will return the address and length of the largest empty area.

A CLOSE call has been added. This call can be used to shorten the file length specified in the device directory to the actual length used. The file must have been opened by a previous ENTER as in version 2. If the CLOSEd length is zero, the file is deleted.

## SYMBIONT SUPPORT

The RTS/8 sources has been modified to allow RTS/8 to exist in fields other than 0. This modification allows concurrent use of RTS/8 with a modified OS/8 in a VT-78 environment.

## BUG FIXES

Bug fixes from previous releases of the operating system have been incorporated into the Version 3 release.

## OTHER MODIFICATIONS, THE RTS/8 WORKING GROUP "WISH LIST"

Various system modifications have been suggested. Below is a list of these ideas. While no commitment has been made to incorporate these ideas into the next release. If time permits, the implementors may install such of these ideas that seem both feasible and useful.

Create a library of standard system call MACRO's. For example, a SENDW call would be .SENDW TASK, MSGADR.

Create a handler for the TD8E DECtape control.

Convert mass storage call format to accept 2 word block numbers. A bit in the first word of the calling message will specify whether the block number is one or two words. The RL01 handler can use 2 word block numbers.

A customer has provided a system null job that allow dynamic display in the MQ of any memory location while the system is running. A debugging null job has been developed to display the amount of unused CPU time. These null jobs could be combined into a single null job in one page of memory. If this null job is implemented, the system build would have to provide an option for its inclusion in a system.

Install a task-ordered table, optional at build time. Each time the clock interrupts, the table entry for the active task is incremented. This data can be used to monitor system activity.

Add conditional code to the MCR command decoder such that an unrecognized command can be passed on to another task. This change would allow expansion of the MCR capability without having to modify MCR. This technique was used for the EXIT task in Version 2B.

Create a unified multiterminal TTY handler.

Split the current TTY handler into two separate tasks. This would allow output to continue through hung user input. For instance, a scope style TTY may be being used to maintain a dynamic data display.

Change the clock handler so that RTS/8 updates the OS/8 date at midnight.

Save the state of the link across an executive request. If this feature is not installed, the documentation should note the present destruction of the link.

Install new Executive Requests to retrieve job and table information. This would decrease the probability of a runaway program bringing down the system, because the executive table pointers would not exist in the program.

Install and support the George Lord foreground editors.

Allow dynamic installation of a core image. No I/O rundown is provided. This feature is used at one's own risk.

Modify TTY and LPT to support a message format compatible with mass storage messages. The new messsage format would start with a -1, followed by the mass store format call. The mass storage handlers would also have to be modified so as to ignore the leading -1. Character packing and unpacking routines would also be included as needed.

Implement a full interlock scheme between the OS/8 background and the OS/8F foreground file handling module. A table with an entry for each device would have four bits for read and write for both foreground and background. Modification of USR would cause the OS/8 support module to be notified when a file on a device was opened (for read or write) and subsequently closed. Similarly, from the foreground files side an additional CLOSE function would have to be implemented to serve the same function. Protection would exist over the entire time that the file would be open for input or output. Background and foreground reads would be allowed simultaneously. Finally, modifications would be made to PIP to notify the interlock scheme when a squish was to take place.

Add paper tape support in the OS/8 background support.

Create a task that would run every 10 seconds or so to enable interrupts in devices like TTY and LPT. These devices can sometimes end up with the interrupt disabled, when the printer runs out of paper for instance.

Define an IOT for the OS/8 background to use to request something from the foreground. The function code of the request would be in the accumulator. For example, OS/8 could request the time of day. The 6000 IOT might be good for this. Code would be added to OS8SUP for this.

Allow MCR to use the DATE command even when a clock is not present.

The TCF command is a NOP in OS8SUP, remove it.

Modify non-system handlers supporting OS/8 to look for C.

The data the VT78 sends to the line printer is in 1's complement form, and the LPT handler needs a fix to support this.

Allow the special function bits to pass through OS8SUP to RTS/8 for I/O calls.

### FROM JIM VAN ZEE

It is occasionally necessary to construct short delay loops in a program, and in such cases the usual method is to employ an ISZ instruction to run out an appropriate counter. Another method is to just load the AC and then count up until you reach zero. However many of the 'operate' instructions generate a fixed length sequence of values, one of which is zero, when they are called in a loop such as:

```
instruction
SZA
JMP .-2
```

The case where 'instruction' = IAC clearly generates 4096 cycles when entered with the AC=0, but other instructions generate shorter cycles. Some of these can be controlled by the initial setting of the link, which can therefore be used as a 'switch'. The following table summarizes the possible values for such loops:

| Instruction | octal | L=0 | L=1 | Instruction | octal | L=0 | L=1 |
|---|---|---|---|---|---|---|---|
| IAC | 7001 | 4096 | 4096 | CMA RAL | 7044 | 13 | 13 |
| IAC BSW | 7003 | 127 | 127 | CMA IAC RAL | 7045 | 37 | 1 |
| IAC RAL | 7005 | 14 | 13 | CMA RTL | 7046 | 13 | 13 |
| IAC RTL | 7007 | 14 | 13 | CMA IAC RTL | 7047 | 1415 | 1 |
| IAC RTR | 7013 | 14 | 13 | CMA RAR | 7050 | 13 | 13 |
| STL IAC RTL | 7127 | 8 | 8 | CMA IAC RAR | 7051 | 37 | 1 |
| STL IAC RAR | 7131 | 13 | 13 | CMA RTR | 7052 | 13 | 13 |
| CMA STL IAC RAL | 7145 | 13 | 13 | CMA IAC RTR | 7053 | 1415 | 1 |
| CML RAL | 7024 | 25 | 1 | CMA CML RAL | 7064 | 2 | 26 |
| CML IAC RAL | 7025 | 26 | 26 | CMA CML IAC RAL | 7065 | 1 | 26 |
| CML RTL | 7026 | 25 | 1 | CMA CML RTL | 7066 | 2 | 26 |
| CML IAC RTL | 7027 | 1743 | 189 | CMA CML IAC RTL | 7067 | 1 | 26 |
| CML RAR | 7030 | 25 | 1 | CMA CML RAR | 7070 | 2 | 26 |
| CML IAC RAR | 7031 | 25 | 1 | CMA CML IAC RAR | 7071 | 1 | 26 |
| CML RTR | 7032 | 25 | 1 | CMA CML RTR | 7072 | 2 | 26 |
| CML IAC RTR | 7033 | 7 | 1743 | CMA CML IAC RTR | 7073 | 1 | 26 |

The longest cycle is of course 4096; the second longest is 1743. The largest difference between L=0 and L=1 is 1736 while the largest ratio is 1415. Other interesting observations are left to the reader.

DECUS 12 BIT SPECIAL INTEREST GROUP NEWSLETTER
NUMBER 32 JANUARY 1979

Harris Semiconductor
M.S. 54-40
PO Box 883
Melbourne, Fl. 32901

## NOTES FROM MICRO-8 WORKING GROUP

By Jonathan Lockwood Phone: (305) 724-7542

### FALL-78 SYMPOSIUM

The symposium was again very profitable for the end user as well as the OEM. It was difficult at times to keep up with the hectic 8:30 am to 11:00 pm pace, however you had the following week to recover.

### NEW PRODUCTS

The RX02 was finally anounced along with new versions of the DECstation. The RX02 diskette is the same as the RX01 diskette, being reformatted for RX02 mode by the user. Both RX01 and RX02 modes can be used on the drive. The controller has both a software density protect and a software write protect.

The following new DECstation configurations were announced:

78/50: Basic DECstation-78 with RX02 floppy disks, OS/78-V2B **AND NOW** COS-310. With the addition of a Class-A COS-310 license, the cost has gone up considerablly.

88/50: PDP-8A205 processor with 32K words of memory, Dual RX02 drive, VT-100 terminal with out the advanced video features, a 30" cabinet, OS/78-V2B, and COS-310 V-8.

88/80: PDP-8A205, 32K memory, dual RX02, one RL01, a 40" cabinet, VT-100, and OS/78-V2B with both RX and RL support.

88/90: PDP-8A205, 32K memory, dual RL01 drives, 40" cabinet, VT-100, and OS/78-V2B.

88/92: An 88/90 with a PDP-8A425 processor for more expansion room.

88/97: An 88/90 with a PDP-8A425 processor, KT8/A memory management, and 64k words memory.

There many questions about the possible availability of COS-310 for previous DECstation purchased. There was no commitments made by DEC concerning this issue. Historically, COS-310 has been part of Commercial Products group and has been tightly bundled with the hardware. Mountains had to be moved internally within DEC just to bundle OS/78 with COS-310. The next major hurdle is to bundle in the Word Processing software. From an outsider's viewpoint, this task is equivalent of moving Mount Everest to the Pacific Ocean. There is hope however because within DEC there are several very persistent souls who are dedicated to make this miracle happen.

If you currently have an OS/8 license and want Commercial Basic, you can now order OS/78 cross license for a "low cost" (about $200 J.L.). Also if you have OS/78 and want TECO or FUTIL, you can order these under a cross licnesing agreement.

### RX02 ALTERNATE SOURCE

I just received some literature on a new Data Systems Design floppy disk drive that is compatible with the rx02. the dsd 440 is packaged in a low profile 5 1/4 inch chassis and records data is both dec double density and ibm 3740 formats. for more info call: george fink (408)-249-9353.

DECUS 12 BIT SPECIAL INTEREST GROUP NEWSLETTER
NUMBER 32 JANUARY 1979

# ANNOUNCING FORMATION OF THE COS-310 WORKING GROUP

LAWRENCE H. EISENBERG
17141 Nance Street
Encino, California 91316
(213)-788-0351

## COS-310 COORDINATION

In response to a substantial clammer for the creation of a subsection to the Decus 12-Bit SIG to afford primary attention to Dibol (COS-310), we now will have a Dibol coordinator (that is, if you all out there show some interest and some participation). The main reason for the creation of this Working Group is DEC's decision to include COS-310 with all future DECstation-78 and 88 systems, ie., you can't order a system with only OS-78! Another reason is provide a single forum ,the 12-BIT SIG, for all the PDP-8 operating systems.

Briefly -- I am an attorney, and have no formal education in computer programming, science, engineering, or the like. I got myself into this pickle because I allowed my secretary to talk me into purchasing a Dec Data System 310 (primarily for word processing), which just happened to come complete with COS-310 and OS/8. Needless to say, I'm now hooked. But, all of you gurus out there might keep a couple of things in mind:

(a) I can't answer really technical questions -- but I am learning where to find the answers and, if requested, we'll print your inquiries and make our own efforts to find the answers;

(b) I hope to make this column an educational and learning experience for all of you out there (especially those of us who can't find it in the manual and have no Dibol brains on the payroll);

(c) This column will be dedicated primarily to COS-310 and the Dibol language; and

(d) We need your input, so that you can share with the output. I.e., please share your programming hints and experiences. Sure, we have all expended hundreds of hours to figure out that simple routine, but by sharing the results with others we can all pick up something helpful.

## NEWS FROM THE FALL 1978 SYMPOSIUM - "THE EIGHT IS DEAD! LONG LIVE THE EIGHT!"

Over 2600 participated in the Fall-78 Symposium (which included an awful lot of Dec-10/20 people who will not -- we are assured -- be going to the New Orleans Symposium). Even so, the PDP-8 received considerable exposure so far as workshops and sessions were concerned.

The growth of the Word Processing ᴗystem ("WPS"), which is primarily a PDP-8 development line, has undoubtedly breathed new life to the Eight. Along with WPS, however, has come considerable feuding within DEC's management. (More on this below.)

With some 25,000 WPS units sold in 1977, and a projection of over 125,000 during 1980, the "dead" Eight is now a very much alive Eight.

According to Mr. Jack Gilmore, chief of the WPS marketing for DEC, Dibol will now be standard, no cost option (Category D "license only") for all Word Stations, and will be supported by the Word Processing Systems. (OS/8 and OS/78 are, of course, also standard features of the data processing systems which aid in the sales of the WPS.)

There were several workshops available for Dibol at the Symposium. Unfortunately, many were poorly described and a few were difficult to find. Hopefully, by next Fall we can overcome that problem. The first session of the symposium, which I am told was the best (Advanced COS-310 Dibol) did not take into consideration San Francisco's fog. My plane was delayed several hours; and I missed too much. Also, I was advised of my reporting functions late Thursday (without notes, this column won't be too great).

There were, however, some highlights and a very excellent programming aid (never before published, I am assured) which will make nearly all of you feel that although you may have wasted some time reading all of this, you got something well worth it in return.

## DIBOL - VERSION 8

It's official. Version 8 is now out and available. (How many of you knew that there was a Version 7? Those who did not were blessed, as the poorer features of 7 are now overcome by some really significant improvements in V-8. V 6.05 is still the prevailing standby.)

Mike Dougherty of DEC (Phone: (603)-884-5537), who wrote V 8, gave one of the finest presentations of any of the sessions. (Whoever said that engineering wizards lacked public speaking capabilities?) While most of the improvements are for our big brothers (CTS 500) many are applicable to COS-310.

V-8 supports the LA 120 printer and the new RX02 ("double density") drive. It will not support the RL01, and there are no plans, currently, to implement such use. (But, see the WPS discussion, below.) It also supports the new VT-100 terminal. (Next issue we'll have some escape sequences for that terminal, which can be programmed in a Dibol program, for rather interesting display and graphic work.) The use of the MR-78 on the DECstation-78 as a hardware lock has been eliminated in V-8; it was added in V-7 to prevent copies to unlicensed machines.

The documentation package has been completely rewritten. There are three new manuals, each written for a different level of user, as well as a new System Reference Card. The order numbers for these manuals are:

| NAME | ORDER NUMBER |
|---|---|
| COS-310 New User's Guide | AA-0758A-TA |
| COS-310 System Reference Manual | AA-D647A-TA |
| COS-310 Release Notes and Installation Guide | AA-D759A-TA |
| COS-310 System Reference Card | AV-D757A-TA |
| COS-310 Divider Set | AV-H205A-TC |
| COS-310 2 1/2" Binder | 12-13565-02 |

These manuals as well as other PDP-8 manuals can be ordered directly from one of the three U.S. Technical Documentaion centers. Delivery is about three weeks if you send them a check. The addresses are as follows:

NORTHEAST/MID-ATLANTIC
Technical Doc. Center
Cotton Road
Nashua, NH. 03060
Phone: (800)-258-1710
In NH call: (603)-884-6660

CENTRAL REGION
Technical Doc. Center
980 East Remington Road
Schaumburg, IL. 60195
Phone: (312)-640-5612

WESTERN REGION
Technical Doc. Center
2525 Augustine Drive
Santa Clara, CA. 95051
Phone: (408)-984-0200

**BUILD** has been deleted, as obsolete, especially since the CRT is now almost universally the standard for operator input.

**SYSGEN** (SYStem GENeration) now only has two modifiers: /B which creates a new system device [replaces /C]; and /C which changes the system handlers [incorporating part of old /C]. Logical unit tables and assignments are no longer handled with SYSGEN, but are now created and read with DFU (which replaces DFDIR).

**DFU** (Data File Utility) is a new which allows the user to designate and examine logical unit assignments. For the OEM this utility would be included with the end user's package rather than SYSGEN because it is smaller and does not allow hardware reconfiguration of the system.

**FLOW** is a charming new utility. It prints a flowchart, using standard flowchart symbols, from a source program. Not only is it pretty to look at, but should provide an interesting debugging tool.

**PATCH** provides a needed auto-patching capability. It works from a short command file which can be called from disk. PIP has also been modified to work from a similar command file to simplify common tasks.

**MENU** is the last "new" program. It displays a predefined form on the screen, accepts an operator response, and executes a predetermined Monitor or Editor command based upon the response. Its primary use will be for non-technical personnel who would prefer to use a menu-driven program (but it only acts on the first command). It is called with "RUN MENU" (which can be changed to any title, of course).

Several other changes were announced, but reference to the new manual will be required to cover them all. The only programming change (i.e., change which must be made to run an older program on V 8) which would be noticed in the new version is with NEGATIVE NUMBERS IN THE "FORMS" STATEMENT. They are no longer recognized and the results are not predictable.

DATE format is changed, and there is no error message (needed room for future patches) for incorrect input: DAte dd-mmm-yy. (Note the three characters for the month!) This date will work out to 1999.

"SYSGENNING A COS-310 SYSTEM"

Norm Farrington, of Computer Applications Corporation (Phone: (515)-232-8181), also gave an interesting talk on some of the new applications. His subject was: "Sysgenning a COS-310 System". It was far more interesting than that. Some of his sage advice follows.

When using the new SYSGEN, especially with the new RX02 drive, be sure to run SYSGEN at the end of any program runs. It would be a good idea to make "RUN SYSGEN/C" as the last BATCH stream program.

If you run PIP/E (e.g., to compress the Directory), on any version, be careful! If you have defined any logical units on the system device you may destroy your system and all programs unless you redefine the System device to show no logical (i.e., data files) units residing there. If you want a technical explanation I would be happy to provide it, but get a better answer from Norm.

WHEN IS "DOUBLE" ONLY 150%

When DEC decided to call their new RX02 drive "double density" they may have been somewhat misleading. The new drive (which will work with single density as well as double density diskettes) does not use both sides of the diskette, but compresses more information on the same diskette. The gain is a mere 50% (so, why "double" density?). The RX02 contains 61 segments per diskette versus 41 for the RX01.

Double density diskettes must be formatted (RUN DYFMT) under Version 8. Regular diskettes may be operated on the RX02 drives without further modification. Transfers between the two diskettes may be accomplished.

EDITORIAL - WAR OF THE WORDS

Unfortunately DEC is undergoing a troublesome policy dispute as between WPS and DP management. The chism is already having a predictable result.

WPS absolutely refuses to support double density mode on the RX02; claims that it is too unreliable. All drives will be hardware straped for RX01 mode. On the other hand, WPS has introduced the RL01 (5 meg disk drive) for delivery in January (?) and claims that it will support Dibol, SIMULTANEOUSLY WITH THE WPS, on the RL01. (Such support requires a configuration of at least one dual RX01 and two RL01s. The theory involves use of the "cx" mode via channel 0 and running Dibol in the background while WPS is running in the foreground. Only one terminal at a time can be dedicated to the Dibol, although both (or all) can operate the WPS.

Commercial Products absolutely refuses to support the RL01; claims that there is no funding and probably no need. DP also says that Dibol will not work under the WPS plan. [WPS says that they are already shipping and that it does work. There were no displays at the Symposium, so we can't say for sure.]

This type of bickering between two factions, who should have similar motives (e.g., let's keep the "8" breathing), makes no sense at all. Surely a driver for the RL01 cannot be so expensive a project that DP has to ignore it. Also, since the RX02 can operate without modification on both single and double density diskettes it seems irresponsible for WPS to take a firm position that WPS units will not have RX02s.

We believe that DEC's top management should look into the matter and, possibly, place their DP and WPS managers in a locked room until they can come out and resolve their skirmish.

## PROGRAM NOTES

How often have you wanted the ability to append (or insert) SOURCE FILES, such as sub-routines which are necessary for one program, but not another? Having been assured that this information has never before been published, and having been sworn to absolute secrecy (with consent to release to those with a need to know), we present a FIVE STEP PROGRAM TO APPEND OR INSERT SOURCE FILES:

1. FEtch FILEB; REM This is the file to be appended or inserted into FILEA.

2. 0001 LN 2000,1; REM Must be the first item in the program. The line number and the LN can be separated only by a single space. Be sure to check FILEA so that the first line number of FILEB (2000 in the example) will start after the end of FILEA (or, if to be inserted, that there is sufficient space for the insert).

3. WRite FILEB/Y; REM Return the file for temporary storage and for call under the BAtch call.

4. FE FILEA; REM Places the first file in the buffer. FILEB will be appended or inserted into FILEA on the next command.

5. BAtch FILEB

IT WORKS!' For whatever reason, FILEA will be in the buffer and ends, say, on line 1400. Using the above example, the editor will read the first line of FILEB (LN 2000,1) and operate on it, thereby renumbering all of the lines of FILEB as 2000, 2001, .... The BAtch program will then halt and you will have your two system files appended (or inserted).

CAUTION: Be sure that your line numbers are proper - both the last number of FILEA and the new number of FILEB; also be sure that you won't overflow the TEXT AREA. (If you do you will receive an error message.) Also, be sure to remove the LN command from FILEB, or you'll forget later.

You can perform this utility upon as many source files as you wish, until you run out of text area or line numbers, but you have do one at a time. (Remember to WRite your new file before FEtching a new source file.)

Now you can keep all of your sub-routines on separate source files. (All donations will be accepted most gracefully.)

## NEW WPS TO COS CONVERSION UTILITIES

Many of us "scalpers" have known for some time of the fact of an "unofficial" DEC utility for converting WPS to COS (and vice versa), and have had the program (also available is a SORT and MATH utility). It works, but is awful slow. (Several private sources have developed much faster versions. If anyone wants a reference, we will be happy to refer you.)

RUMOR now has it that last month DEC developed a much faster VERSION 2 of the Utility. Maybe someone out there can help me in getting a copy.

The following quote from DEC's most recent "Word Processing Software" flyer may be revealing upon this subject:

> "WPS-8 document floppy disks can be read and written under COS 310 (Commercial Operating System for the DEC Datasystem 310 small business computer). This facility permits integration of business data processing applications with the word processing power of WPS-8."

In other words, it sounds as though DEC is going to support these utilities officially.

## ESCAPE SEQUENCES

Dibol allows for the use of escape sequences, although the programming manual does not discuss it at all. This is a very powerful programming aid. Next issue we will include several examples. For now, we'll merely mention how it's done.

The ESCAPE MODE is entered with a DISPLAY command:

```
        DISPLAY (0,0,27)
```

The next command is the ESCAPE CHARACTER which may be entered in decimal or by letter. E.g.:

```
        DISPLAY (0,0,Z)    or
        DISPLAY (0,0,90)
```

The first two numbers (e.g., 0,0) are cursor positioners and do not necessarily have to be "0".

## CALL FOR HELP

We understand that the Release Notes for CTS 300 V5 contain an excellent documentation for menuing (a new word) or page displays. Anyone able to send a copy will receive our very much appreciated thanks. We hope to include additional information regarding the escape mode, and the menu capabilities, in the next issue.

Also, next time we will include a routine which will allow for password entry (i.e., the ACCEPT statement does not echo the password, so that the screen will remain blank while the password is entered).

How about sharing some of your favorite routines? We will provide credit (unless requested otherwise) and you will be helping all of us.

## NEW WORD PROCESSING SIG

In gestation is a new Word Processing SIG (actually a TEXT SIG) which was in the formation stage at the Fall-78 Symposium. This SIG plans to organize all users of TEXT EDITING (from office word processing all the way to the biggies, such as typesetting, etc.).

For further information contact:

Ms. Kathy Tamer
Technology Inc.
17511 East Camino Real
Houston, Texas 77058

(713) 483-3471

Anyone having made it this far, we thank you. Please feel free to make suggestions and send inquiries and program hints to me at my address above.

## MODS. TO FRTS ERROR ROUTINE

MANY FORTRAN IV USERS HAVE BEEN ANNOYED BY THE LOSS OF OUTPUT FROM A LONG CALCULATION WHEN AN UNEXPECTED ERROR OCCURS IN THE NEXT STEP. THE PATCH TO FRTS.SV DESCRIBED BELOW SEEMS TO CLEAR UP THIS PROBLEM.

```
C         **************************************
C
C         PROGRAM TO TEST LPT FINISH DESPITE ERROR
C
          WRITE(3,100)
100       FORMAT('0  THIS IS A TEST LINE')
C
C         NEXT LINE CONTAINS ERROR, BUT TEST LINE PRINTS
C         IF FRTS.SV IS MODIFIED AS SHOWN BELOW
C
          Z=SQRT(-1)
          STOP
          END
C
C         **************************************
```

A FEW LINES ARE STOLEN FROM THE LPT RING-BUFFER:

```
          0003/5166 6600
          0267/5166 6600
          7577/5166 6600
```

THEN A SMALL PATCH WAITS FOR THE BUFFER TO EMPTY AFTER AN ERROR:

```
          5052/1136 5366
          5166/0000 1403
          5167/0000 1004
          5170/0000 7640
          5171/0000 5366
          5172/6600 1136
          5173/XXXX 5253
```

IAN TEMPLETON<br>
PHYSICS DIVISION<br>
NATIONAL RESEARCH COUNCIL OF CANADA<br>
OTTAWA, CANADA K1A 0R6

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

**MOVING OR REPLACING A DELEGATE?**

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

( ) Change of Address
( ) Delegate Replacement

DECUS Membership No.: ____________________

Name: ____________________

Company: ____________________

Address: ____________________

____________________

State/Country: ____________________

Zip/Postal Code: ____________________

Mail to: DECUS - ATT: Membership
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.