12
BIT Sig
DECUS

DECUS 12 BIT SPECIAL INTEREST GROUP NEWSLETTER

May Number 28 1978

Contributions and correspondence should be sent to:

Robert Hassinger, Coordinator - 12 Bit SIG
c/o DECUS
129 Parker Street, PK-3/E55
Maynard, MA 01754

..or..

Liberty Mutual Research Center
71 Frankland Road
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 29-June-1978. 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.

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

DECUS. DIGITAL EQUIPMENT CORPORATION. and the editor assume no responsibility or liability for articles or information appearing in this document

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

Lee Nichols - see above

Micro-8 Working Group

Jonathan Lockwood - see above

Symposium Software Exchange Committee

Send copies of software you wish to exchange at the next US symposium to the appropriate committee member for preparation:

DECtapes
DEC floppys
AED floppys

Russell Overbey
PO Box Y Bldg. 92101-2
Oak Ridge National Laboratory
Oak Ridge, TN 37830

DECtapes
Magtapes
DEC floppys

Earl T. Elis, Jr.
USCG R & D Center
Avery Pt.
Groton, CT 06349
(615) 483-8611 Ext. 3-5176

LINCtapes

Larry Alber
FDA Room 1222
433 W. Van Burren
Chicago, IL 60607
(312) 353-5863

NEWSLETTER BACK ISSUES

It looks like the problem of getting back issues of the newsletter has finally been resolved. As an experiment, DECUS recently generated a set of microfiche masters for all the back issues of the 12 Bit SIG Newsletter (numbers 1 thru 27). Sets of copies were made available for free at the Spring Symposium. About a hundred sets were left at the end of the Symposium and when they find their way back to Maynard they will be available for the asking (one to a customer). No long term policy has been set on continuing to stock and give away back issue microfiche sets, but as long as the current supply lasts, they are available. Let me know how they work out and how they may be improved in the future. Given the cost of reproducing hundreds of pages in hard copy, (already seven fiche worth) this seems like the only practical way to provide full sets of back issues at an affordable cost. Requests should be addressed directly to the DECUS office in Maynard.

"SYMBIONT"

A "hook" is being designed into the next release of OS/78 (i.e. V2 with "commercial BASIC", due out before too much longer). The idea is to allow an interrupt driven routine such as a spooler or RTS/8 to co-exist with OS/78 in a VT-78 were the usual time share trap hardware is not available (i.e. normally you would use the time share trap to run a copy of OS/8 as the background task in an RTS/8 program).

The hook is this: Every program and CUSP, (including the Monitor and Command Decoder), that load into page zero of field zero are required to contain the following instructions in locations 1 and 2 of field 0:

```
        CIF 30
        JMP .-1
```

The effect is that if the software core size is set down to 12K or less, an interrupt driven routine is loaded in field 3 with a suitable entry at location 1, and the interrupts are enabled, this hook ensures that the correct instructions to route the interrupt to the interrupt service routines are always present.

Some of the restrictions on this are roughly as follows:

1) Only programs with the hook may be loaded or run while the interrupts are on.

2) The programs that are run while the interrupts are on cannot use or try to change the interrupts (this means there is a problem with FORTRAN IV - it should be fixable but it was not clear that it would be in V2, also, since FORTRAN "II" is not supported in OS/78, it has not been patched to meet these requirements so it is also not be usable).

3) At the moment, SYMBIONT is designed to work only on the VT-78. The basic restriction is that no data break mass storage device may be present in the system. The VT-78 has no data break capability so it is OK. On closer inspection, the real restriction is that, while the interrupts are on, no data break device may make any transfers into location 0 of field zero (because the transfer could come after an interrupt but before the return address is retrieved and saved from location zero, thus destroying the return path from the interrupt).

If, while interrupts are on, all I/O were to be done thru handlers or other code that could be sure this condition was met (i.e. turn off interrupts during transfers to location 00000 and perhaps do such I/O in intelegent handlers in the interrupt driven code called via fake device handlers ala RTS/8 background support) then the restriction might be circumvented.

4) This feature is expected to be in the V2 release of OS/78 but, of course, it is not in the current release of OS/8. At the Symposium, DEC

was not prepared to say what they would do for OS/8 in the future in this area. It is clear, however, that DEC wants to be able to generate OS/78 and OS/8 from the same sources so the most logical answer is that future releases of OS/8 will have the hooks patched into the system and CUSPs but, due to the possible presense of data break devices in OS/8 configurations, SYMBIONT will not be formally supported for OS/8. It would not suprise me if one or more users of OS/8 were to take advantage of the hooks on their own, perhaps by using some variation of the approach I outlined above. If that happens, I can imagine DEC eventually comming out with their own OS/8 version of SYMBIONT support.

This idea for providing an interrupt driven capability has been kicked around since the earliest days of PS/8. One of the reasons I originally pushed for and got the "software core size" feature was to provide a way to reserve an area of memory for such things. The key problems have always been data breaks to location 00000 and DEC support. I recall Richy Larry pointing out the data break problem many years ago during discussions with me of ways to remove some of the arcitectural limitations of PS/8-OS/8. The breakthru that makes it possible now for DEC to support the feature is the VT-78 which can ensure no data break devices will be present plus internal organizational support for development of features explicitly just for for the VT-78.

What the users have always needed was to have DEC commit to a standard hook that they would support (i.e. the users could not, as a practical matter, patch every new release of the monitor and every CUSP with something like this). Now that it looks like we will have a standardized, supported hook, many exciting things will be possible.

Anyone writing PDP-8 code should be aware of this hook and be sure to provide for it in any program that might be run under OS/78 and just in case, any program that will run under standard OS/8 also. PDP-12 users: what are the implications of trying to use SYMBIONT with code that goes into LINC mode when the interrupts are on - maybe such code will have to be modified.

12 BIT SIG AT THE SPRING SYMPOSIUM

In addition to the regular sessions at the Symposium in Chicago, several other activities took place. I represented the SIG in meetings of the DECUS/US Library, Publications and Symposium committees. Our Steering committee meet also. Among the actions at that meeting was the decision to invite Jonathan Lockwood to join the steering committee. We meet with Ernst Lopes Cardozo who attended the Symposium representing DECUS/Europe and the European 12 Bit SIG.

The committee agreed that the PDP-8 Product Line had been very generous in the hardware they brought to the Symposium and made available to us for program exchange and media conversion and we wish to thank the Product Line for its continuing support. Our committee on software exchange worked hard and long to keep some degree of order in these activities. The steering committee agreed, however, that we need to work on improving the way these activities are organized and supported.

In the PDP-11 area, software exchange was done only on 1/2 inch magtape by straight copy of tape to tape. Each of the major SIGs gathered all the software to be exchanged, went off and built one or more magtapes, then gathered stacks of user magtapes onto which they made straight copies of the master tapes in a batch operation run by SIG representatives at a few designated times on a system specially configured with a CPU, RK05 system disk, and two 75 ips tape drives (not a cheap system by the way). I did not see any self service software exchange or media conversion going on on the 11s. It appeared that if you wanted copies of the software you had to take them on magtape and you had to wait for the committee to make the copies. This makes for a simple software exchange with very few problems with media size (magtapes are BIG!) and speed (duplicating several thousand blocks only takes a few minutes with the system that was available.) On the other hand, people who do not have access to magtape are not well served I should think (for example I have a PDP-11/34 without magtape so I could not get access to the several tens of thousands of blocks of software available at the meeting).

Due to the belief that very few if any of the 12 bit users have access to magtape or any other single, really large capacity storage system, the 12 bit software exchange continues as it has in the past, based on smaller media (DECtape, floppy disk, and some RK05 disks). Also, a self service operation has always seemed the best way of doing the job to avoid having to ask a few volunteers to man the system 24 hours a day for four days. If you think that magtape would be practical as an exchange media at the symposia or if you think some other mode of operation would be better please let me and the Symposium Software Exchange Committee know. In the meantime, the committee, Stan Rabinowitz and I are looking into what we can do in the way of improved hardware and software capability to make the system work better, be more self-operating, more secure so that the actions or mistakes of one user will not damage the files for others, and provide greater capacity. I envision a system that would be able to avoid file name conflicts, solve the "small directory" problem, store the files in a relatively safe way (maybe a non-OS/8 file structure so that only special programs can alter the files) and maybe we can make progress on the capacity problem through the use of larger disks (like the new RL01), more organized mounting of several different disks - each with part of the files, or maybe even a magtape to store and back up the entire set of files with some sort of staging to disk as needed. This last is difficult to do because DEC sells very few magtapes for PDP-8s so it would be hard to get one on the system at the Symposium due to both cost and availablity. It would be a very attractive solution, however. Additional features might include automatic scheduling of requests and a seperate background capabilitiy for making directories, listings, and doing special, one time conversions and exchanges. These features would alleviate the problems with finding the system tied up by someone and not knowing when or if you will be able to get on it to make your copies. A separate background capability to brouse through descriptions of the various items and packages available, note what you would like copies of, and maybe calculate how much storage you will need would also improve the system. All this sounds like a real live development project but I think some reasonable subset can be achieved with the resorces available if we work hard. The main question is, is the goal worth the effort? Do enough people feel this is a really high priority at the Symposia? Will an improved system attract more attendees? Would a system with enough capacity to distribute the more popular DECUS library items and other material in addition to current contributions from Symposia attendees make a significent difference?

At the Spring Symposium we had about 15000 blocks of files on line (3 RK05 platters or 6 logical devices - mostly full) and there was nowhere near enough room for everything we could have put in the system. Several cases of lost files resulted. Earl Ellis estimates 30 RK05s, 80 floppy disks, and 25 DECtapes were filled during the meeting by people taking copies of the software. Considering the rather extraordinary amount of floppy to floppy copying that was done separate from the main system on the DECstation 78s, I would guess that the total count for floppies was really a good deal higher.

THE NEW RL01 DISK

The RL01 disk system was discussed at the Spring Symposium although no hardware was available for demonstration. It consists of a top loading, single platter drive that DEC has developed and will offer on a number of computers. It seems to be aimed at the low cost market and seems destined to replace the RK05.

The basic format of the disk is 10K data blocks of 512 8 bit bytes. As interfaced to the PDP-8, only three quarters of each block is used (as with the "six bit" format on floppy disks). As a result, a PDP-11 will record the same number of blocks as a PDP-8 but will actually access one third more bits. By comparison, the RK05 on the 11 has about 4800 blocks of 512 8 bit bytes or 256 16 bit words vs. the RK05 on the PDP-8 where you get about 6400 blocks of 256 12 bit words. In view of this it seems to me that on the 11 you see a full doubling of capacity which looks quite attractive but on the 8 you only get a 50% increase in capacity which is less interesting particularly when the format OS/8 is going to use is considered. OS/8 still is limited to a maximum of 4095 blocks on one logical device so the RL01 must be made to look like a minimum of three logical devices to access all of one platter (compared to two logical devices in the case of the RK05). DEC has chosen to break the disk into two 4K blocks sections and one 2K blocks section. Each cylinder has 40 (decimal!) blocks. 32 of the blocks in each cylinder go to make up the 4K parts (the outer cylinders make up one part and the inner ones make up the second part) and the remaining 8 blocks on all the cylinders go to make up the 2K part. Two-to-one interleaving is used due to the basic transfer speed of the disk. The bottom line on speed as compared to the RK05 is dependent on the exact interleaving scheme selected and was not quite clear, but it seemed clear that I/O on the 2K part was going to be slower than for the other two parts.

In order to minimize the system and media cost, the media is specified as being able to have a modest number of bad blocks. A bad block table is recorded on an otherwise unused area of the disk and it apparently contains information on what block to substitute for each bad block.

The greatest challenge in writing an OS/8 device handler for the RL01 was finding a way to deal with this bad block mapping within the constraints of OS/8. The results are two page system and non-system device handlers that just fit. We were told that they initialize themselves by reading the bad block table and storing the information inside their core image. They apparently map bad blocks on the fly. The result is that the bad blocks are invisable to an OS/8 user and he does not have to know anything about them. However, a transfer that you would expect to be done in one step could require multipule repositionings of the heads thus slowing down the transfer considerably.

Recognizing the speed penalty for bad block mapping, systems like RT-11 give you a choice of automatic bad block mapping or letting the system enter special files in the device directory that the system always leaves in the same place (even across "squishes" and "deletes"). They have the effect of preventing the system from allocating those blocks for files so the bad block mapping is avoided because you never try to access the bad blocks. OS/8 could also do this if PIP and FOTP knew enough to not move or delete files with some special file name extension. If this feature were added it could be extended in the same way RT-11 has done to provide for another special file name extension for system files that must remain in the same place. RT-11 uses this arrangement to place the system swap blocks, device handlers, and the monitor itself in files. The result is that RT-11 does not have to have a special, reserved area on system devices that is not present on non-system devices. New device handlers are installed by copying their files onto the disk and re-bootstraping (plus an "INSTALL" command for handlers the system does not already know to look for). The price for this is that during each bootstrap all these files must be looked up and and their locations recorded in the core image of the monitor and various precautions are needed as noted above to avoid moving the files or else to force a bootstrap when they are moved.

If OS/8 implemented this idea I don't know how you could retain the ability to start at 07600. It would seem that you would always have to do a bootstrap to start the system so you could ensure that OS/8 really knew where all the files were on the particular system device that was currently mounted. With core memory systems (i.e. non-volatile memory) you might be able to have a bootstrap routine loaded when you sign off so it would already be in core and ready to use the next time you started the system. This way you could at least avoid having to key in the bootstrap each time you start. This is a fairly reasonable approach because the only DEC configurations with volatile memory that would not be able to retain such a bootstrap are OMNIBUS machines that have low cost hardware bootstrap options readily available. Well, so much for this month's blue sky section.

RTS/8 WORKING GROUP NOTES FROM LEE NICHOLS

During the RTS/8 Workshop session at the Spring Decus Symposium, Steve Root described version 3 of RTS/8 which is scheduled for release this fall. (Steve is responsible for RTS/8 developmemt at Digital.) A detailed explanation of the new features and changes for version 3 appear below, followed by the 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 3 symposia. These informal meetings allow the RTS/8 users to discuss features they would like to see added or enhanced within the system. One of the major issues discussed in Chicago, was the compatibility problems created in converting RTS/8 to run under MACREL. The feeling of the group was that now is the time to make any syntax changes, since some minor editing will be required for current user tasks to run under version 3.

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.

During the Working Group meeting, several users expressed an interest in having an RTS/8 handler for the TD8E DECtape control. The TD8E is a programmed I/O device (rather than data break) and needs the interrupt off during data tranfers. While the interrupt off time may be deadly for high speed real time systems, it would be nice to be able to use the TD8E for OS/8 background support. Steve is very interested in this and would appreciate some comments from users who have tried to implement a TD8E handler. I know there have been several attempts at TD8E handlers with varying success. If anyone has a TD8E handler for RTS/8 or has tried to create one, would you please send me a copy of the source and/or a description of how it works and the problems or limitations you encountered. I will collect these and give them to Steve. Your help would be greatly appreciated.

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 file structured media (floppy, DECtape, cassette, or RK05).

File name	Bugs fixed
UDCICS	missing indirect JMS
OS8SUP	KRB, 2 ^C's in background
RTS/8 Exec	Swapper scheduler, powerfail for 8A, EAE, and RX01
MCR	date for 1978
PWRF	code for RX01 and 8A moved to RTS/8
RX01	powerfail code
CLOCK	date, tick race condition
PARAM	date, CLOCK parameter
NIP	date, "skimp" conditional
NSP	disconnect/connect, interrupt message

The two known bugs are in 1) KL8A when you have 3 interfaces, and 2) CLOCK when the schmitt trigger inputs are used with the DK8-EP programmable clock. I also have an RTS/8 source which will save and restore the EAE step counter for context switching and powerfail.

RTS8 V3 DESCRIPTION

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this manual.

The software described in this document is furnished under a license and may only be used or copied in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by Digital or its affiliated companies.

FUNCTIONAL DEFINITION

Since RTS/8 is already an existing product, the functional definition shall be a description of changes. 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 will be replaced by MACREL-LINKER. Backwards compatibility with PAL8 will be maintained if possible; however, this is not a primary goal. In any case, the new features of KT8A support and system build will not be supported for PAL8 assembled modules. Conversion of PAL8 programs to MACREL programs will require a variety of fairly small changes,

RTS/8 systems sources will have to be modified. The initial section of code that places values in the RTS/8 Executive tables by an absolute origin will have to be modified. RTS/8 sources that use the auto-increment registers will have to be modified to use the XSECT construction in MACREL. RTS/8 programs that use the non-resident feature will have to be modified to conform to the MACREL/LINKER overlay conventions.

These modifications apply, of course, to any user programs as well as the RTS/8 system sources. 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.)

The feasibility of a system change sub-section of the build procedure will be investigated. The basic premise is that it would be desirable to have a mechanism to make a system with a minimum of assembly time. Under a system change dialogue previous system binaries (possibly from a library) could be used for a majority of changed system.

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. A fairly minor modification to the present OS/8 support and TTY tasks by the development group will accomplish this goal.

The present design calls for the terminal to '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 must support both KT8A memory management and the old KM8E management. As a consequence, the RTS/8 executive, the OS/8 support task, and various I/O handlers will have to have conditional code, or separate modules. The build procedure will then 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. Unless unforseen technical problems arise, an OS/8 background may straddle a physical 32K boundary.

The feasibility of a mapped OS/8 background has been demonstrated on the KT8A prototype board. The software modules used need only slight modification to be placed into the Version 3 release.

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 must be modified to provide RUN, GET, SAVE and ODT support of images up to 128K.

EAE SUPPORT MODIFICATIONS

Under RTS/8 Version 3 there will be 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 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 committed.

First, when a clock request is cancelled, the clock handler shall remove that request from the clock queue. At present, the request is disabled, but left in the queue. This has, in some cases, been found to lead to queue overflow.

Second, implement an additional clock request type to post a specific event flag, in addition to that of the request itself. This leads to greater flexibility in the use of the clock. Note that the event flag of the request is posted immediately.

Third, code shall be added (conditionally) to the interrupt level such that clocks faster than 5 milliseconds are properly supported. At present, such fast clocks overflow the two word clock format before midnight. The solution would have several interrupts be required to make one tick.

Finally, a interrupt linkage facility will be 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 DISK 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 will be made to allow greater system flexibility.

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

The status list printout subsection of MCR will be modified to allow 127 system tasks. Other portions 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) will be placed in the executive, This allows individual tasks to place their names, eliminating a lot of silly dependencies.

MCR examine and deposit will be 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. If the presently planned MACREL/LINKER changes are feasible, the user will no longer have to type in the non-resident task names at start-up time.

The core images of the overlays will be part of the RTS/8 save file created by the LINKER. These core images will be modified during the execution of RTS/8 if they are declared as writable. It is the responsibility of the user to save a copy of the original system if he wishes an un-modified core image.

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 will be a -2 (impossible in the old format), followed by the extra argument word specifing the message field, followed by the three standard words. The new message field will apply to both the input and output message areas.

MONITOR SAFETY FEATURES

A small amount of code will be added to the RTS/8 executive to check for primary calling arguments outside of their normal range. At present 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. These safety checks will probably be conditionalized.

The offending task will have its non-existent and run-wait bits set. If there is a system task 'ERTRAP', its run-wait bit will be cleared. Thus, a user wishing more detailed diagnostic information can put it in the task ERTRAP. The PC of the offending task will be stored in the Task State Table. The MCR task will be modified to recognize the non-existent bit as a valid state.

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.

A short discussion of the terminal speed-up is warranted at this point. The speed up in over-all execution time is achieved at the expense of a slightly longer interrupt latency time. This issue may make it necessary to include two versions of the terminal handler, to be selected at build time.

Incorporated in the TTY source will be 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 will be 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. Aborted input will be terminated by a 4000 in the caller's buffer.

RX01 HANDLER

The present RX01 handler contains two bugs and a VT78 performance problem. By the time RTS/8 V3 reaches the field, it is likely that RX02's will have to be supported. All these items add up to a substantial amount of recoding for this handler.

OS/8F INTERLOCK

There are currently two problems with the mass storage interlock scheme involving OS/8 background support, and the OS/8 Files module for the foreground. The first problem is safety; there are cases that can lead to loss of data or corruption of the file system. Fixing the safety problem is difficult, and has been relegated to an optional modification.

The second problem is access; the present interlock prevents foreground access to any device while the background is using any device. The access problem will be fixed for RTS/8 V3.

SYMBIONT SUPPORT

The RTS/8 sources will be 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 will be incorporated into the Version 3 release.

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

Various system modifications have been suggested by the RTS/8 Working Group and others. There follows a partial list of these ideas. No commitment is 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.

Define assemble variables and parameters to start with a dollar sign and make them global.

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.

Modify OS/8 support so that a CONTROL Y will be recognized as a command to reload the virtual OS/8 system monitor heads in fields 0+1. Thus, the OS/8 background can be rerun after a crash without restarting the whole system. (This has been implemented, LHN.)

A customer has provided a system null job that allows 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.

Modify the OS/8 support task to send text messages to TTY instead of doing direct IOT's. It is no longer clear that this is a good idea.

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 text 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.

Adopt the Lee Nichols CLOSE with trim function for the foreground file module. Thus, when a file was closed, the extra space from its creation could be reclaimed.

Modify OS8F to return the largest empty file space when an ENTER request is made with a file length of 0.

Since nonresident tasks will be part of the RTS/8 save image with MACREL, maybe have the swapper's initialization code copy the program save image into a working file space for run time use. This would preserve the user's original save image, since writable non-resident tasks would be written into the working file space.

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.

Modify the RTS/8 executive to put the task number in the message event flag for a SEND CAL. It only sets a 1 now.

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.

Change OS8SUP to return the real status word after an I/O call.

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.

Have the FFP run in the background with the KT8A, If the background starts on a 32K boundary.

Maybe add some sort of I/O error logging or error counters.

Make the TTY handler do Rubouts properly across a TAB.

Implement the line feed function in the TTY handler to echo the keyboard input buffer.

KNOWN DESIGN FLAWS IN VERSION 3

Under MACREL/LINKER you cannot get 2 copies of a subroutine linked into a save image. If you use the same subroutine in two tasks which do not want to share the subroutine, the second copy must have a different name. This is a basic limitation in LINK.

The TC08 DECtape cannot support a background with the KT8A unless the background starts on a 32K boundary. The data break locations are the problem. This is also true for other 3 cycle data break devices.

CORRECTIONS TO MACREL VERSIONS OF RTS-8 SOURCES

A number of people who participated in Steve Root's sessions on RTS-8 at the Spring Symposium got copies of a pre-release version RTS-8 written in MACREL. Steve called to say that when he got back to Maynard he discovered a few bugs in what was distributed. The following are the corrections that need to be made to those sources.

Summary of Changes to Pre-release RTS8 MACREL Sources

```
RTS843
        Change all ACL's to MQA's (two instances)
        Remove CAM at UNTEM-3
        Move .GLOBAL KL8LOC within KL8A conditional at KL8LOC
OS8B31
        At XKSFWT, change DCA I LKSFEF to CLA IAC; DCA I LKSFEF
        Remove PAGE statement from end of .DSECT OSD
MCR19
        Add .EXTERNAL TSWFLG
PARM21
        after   IFNDEF  EAE     <EAE=0>
        add     IFNDEF  MQ      <MQ=0>
                IFNDEF  MQSV    <MQSV=0>
                IFNDEF  EAESV   <EAESV=0>
                IFNDEF  KT8A    <KT8A=0>
```

DECUS PROGRAM LIBRARY NEWS

The library is issuing a sheet of corrections and updates to the catalog. It reflects the fact that a number of existing library items are now available on floppy disk as well as other changes and corrections. I anticipate that the information will be circulated soon.

Recent Program Submissions

Bowling Record Tabulator - BASIC8-100

This two-part program will tabulate weekly records for a bowling league with twelve four man teams; but can be used for leagues with any number of teams and any number of bowlers. It will run under virtually any version of BASIC -- NO string handling capabilities are required.

Price codes: Writeup and listing - D01, Source on paper tape - G02

BATNUM: Battle of Numbers - BASIC8-101

The first question asked by the program (besides instructions) is "BEGINNING NUMBER?". The answer to this question should be an integer greater than 15. The next qustion is "MAXIMUM TO BE REMOVED?". The answer should be an integer between 4 and 14 inclusive.

You and the computer will take turns removing a number equal to or less than the number inputted as the maximum to be removed. The object of the game is to force the computer to remove the last number.

Note: This program was developed on an HP-2000 computer, but should run under any version of BASIC. The listing is less than one page.

Price codes: Writeup and listing - D01, Source on paper tape - G02

A Collection of Math and Demonstration Programs - BASIC8-102

This is a package of programs developed using EDU-50 BASIC. Programs include compute square roots, (accurate to fifteen significant digits) and common logarithms (accurate to ten significant digits); compose free verse poetry; plot circles and elipses on a terminal; find unknown terms in an arithmetic sequence; determine coefficients and exponents for a binomial expansion; and find the lowest terms fraction for a repeating decimal. There are eight programs included.

Price codes: Writeup and listing - D02

RAW INPUTS

I noticed a couple of interesting things in the documentation which just arrived with our RT-11 V03B update kit. In the list of supported devices is a referance to an "RX02" floppy disk in addition to the RX01. There is also a reference to a terminal called the "VT100". I could not find any further clues about the terminal but in the chapter on a new formatting program there were indications that this "RX02" was a floppy disk system that had a formatting capability and the ability to format disketts at either single or "double" density. DEC made no formal announcements in any of the 12 Bit sessions at the symposium relative to either of these items. I do not know if anything was said in any of the other symposium sessions. Perhaps by the time you read this, some announcement will have been made, at least for PDP-11s.

PIP /Y BUG

Jim Van Zee has reported that in the OS/8 V3D release (and presumably in OS/78 V1) there is a serious problem with PIP's "/Y" option. As I understand it, if you use a device handler that is not not co-resident with "SYS:" (i.e. permanently in the 07600 page) many bad things happen when you try to do a /Y operation. Jim reported the bug to DEC on an SPR and gave them a patch to fix it. When DEC responded to the SPR, they gave their own patch however. Jim arranged for copies of DEC's "official" patch to be distributed at the Spring Symposium. He has since discovered that the "official" patch has a serious problem (it creates an "invalid system head" I think) and it should NOT be used! I have passed this information on to DEC but the patch may have already been printed in a forthcoming issue of Digital Software News. If such an article does appear in the near future, it might be a good idea to implement it with considerable caution. Presumably the problem will be sorted out before long and a good patch published, until then be careful.

FUTIL PATCHES

The following item was forwarded by Earl Ellis.

This is material I have received from Ian Templeton and Jim Crapuchettes for FUTIL.SV. These will raise the patch level to Level "D". I assume that it will soon be in SOFTWARE NEWS.

1. Patch to fix a 'SHOW CCB' problem (from Jim Crapuchettes)

```
        12024/ 3242  5345     INSTALLED 21-JUN-78
            5/ 1642  7650
            6/ 7450  5236
            7/ 5236  1642

        12145/ XXXX  3242
            6/ XXXX  1242
            7/ XXXX  1351
           50/ XXXX  5225
            1/ XXXX  1175

        12520/ 0200  0300       /avail. in Chicago as FUTP7C.BI
```

This patch upgrades FUTIL Ver 7B to Ver 7C.

2. Patch to fix failure of ODT mapping on .LD modules. (from Ian Templeton) ✓

```
        12075/ 7040  7000

        12520/ 0300  0400
```

This patch upgrades FUTIL Ver 7C to Ver 7D.

NOTE: See 12 Bit SIG 27, page 12 for Ver 7A to Ver 7B

FUTIL is the OS/8 FILE UTILITY PROGRAM by Jim Crapuchettes and now distributed with the OS/8 V3D Extention Kit. It's operation is discribed in Appendex K of OS/8 Handbok Update (DEC-S8-OSHBA-A-DN4).

PROPOSED PATCH FOR THE OVER FULL DIRECTORY PROBLEM

Over the years there have been intermittent complaints about the strategy OS/8 uses when it adds a segment to a directory. Stan Rabinowitz called to tell me of a suggestion he recently received from Alistair Windrom that is designed to avoid the problem.

Basically, the problem seems to arise when the monitor's ENTER code is trying to find a place to enter a new file in a directory and it finds that it has reached the end of the last segment in use (i.e. the one with a zero link word) and there is not enough room left in the segment to enter the new file. The strategy implemented here is to allocate an additional directory segment (asuming any are left of course), and to move the last eight entries in the previous segment into the new one then to enter the new file after them in the new segment. (Note: I am not sure that this does not also apply any time an entry is to be made in any already full segment, with the whole rest of the directory being slid down a segment.) The reason this was done was to achieve certain optimizations in directory operations. If the eight name "breathing space" is not provided, certain later operations can cause inefficent operations. The problem is that it is possible that you can have two devices of exactly the same size with directories of exactly the same size (i.e. the same number of extra information words) but if the one has an "over full directory" (i.e. one where the "breathing space" slots have been filled by some combination of enters, closes and deletes) it is said that it is possible you will not be able to copy all the files from it to the second device. This is because as the second directory fills up, "breathing spaces" are created and when all the directory segments are full, OS/8 cannot recover the directory name slots set aside, at least not during something like a COPY *.*.

The proposed fix is to patch the third location of the ENTER overlay in the monitor from a -10 (octal) to a minus one. Stan says he has not yet verified the patch. I do not have it in the form of a FUTIL or EPIC patch at this time. Stan also says it can be expected to cause less efficient directory operations but it should be a solution to at least some of the overfull directory problems.

I suggest that only competent OS/8 system programmers try this patch until it has been verified and the implications more fully studied and documented.

FOCAL STANDARDS

Earl Ellis sent the a copy of the first draft of a standard for FOCAL that the DECUS/US FOCAL SIG is working on. It was reviewed by interested persons at the Spring DECUS Symposium. A considerable number of suggested changes were suggested and are under consideration. If you have any interest in this subject you should let Earl know and I would be interested in a copy of any inputs from our SIG. Earl notes that the editor of the FOCAL SIG's newsletter is Mark Woods, 9211 Cherry, Frankin Park, IL 60131.

FOCAL STANDARDS
19-APR-78 FIRST DRAFT - "A START"

The FOCAL Special Interest Group meeting at the Fall DECUS Symposium in San Diego appointed a special committee to draft standards for FOCAL. This is the first report of the committee and it should NOT be considered as final or all encompassing.

The number of installations and individuals using various versions of FOCAL continues to increase. The need for a standard is becoming urgent. The enclosed proposed standard draft will be presented to the FOCAL Special Interest Group meeting at the 1978 Spring DECUS Symposium.

An early draft based on material received principally from Jim van Zee, Douglas Wrege, and Rev. Geffery Chase was created. This material included van Zee's U/W-FOCAL manual and pocket card, D. Wrege's FOCFV7 and FOCFV9 material, and Father Chase's et al EFOCAL for E*T*O*S. Other material that aided in this document's preparation was DECUS FOCAL8-177 (OMSI PS/8 FOCAL 1971), write-up and listings of DEC-08-AJAE (FOCAL-69), and DEC-8E-LFOCA-A (FOCAL-8 also issued as FOCAL, 1971).

A very informative edit of the early draft was done by Jim van Zee. In addition, a copy of the FOCL-11 Manual (V 7) by West Virginia Medical Center, long replies from Jon Zellowitz and Dan Smith, and comments on JOSS by Jonathan Vaughn have lead to this draft.

Additional documentation, information, especially for other than PDP-8 FOCALs, and your comments are solicited.

Earl Ellis // Fred Strange // Chuck Meyers

FOCAL Standards Committe of FOCAL Sig
Attn Earl T. Ellis
USCG R&D, Avery point
Groton, CT 06340
(203) 445-8501 X 296

ETE 3/1/78 Copy to:

Fred Strange
Jim van Zee*
The 6502 Program Exchange
Chuck Meyers
TLF
Richard Gumbel*

ETE 4/1/78 Copy to:

Fred Strange
Jim van Zee*
Wayne Wall
Chuck Meyers
Charles Lasner
Dan Smith*

(* = Responded)

Douglas Wrege
Jon Zellowitz*

INTRODUCTION

FOCAL was developed around 1965 by Richard Merrill for the Digital Equipment Corporation's PDP-8. JOSS (Johnniac Online Simulatin System) for the Ran Corp.Johnniac and DEC PDP6 pre-dates FOCAL and may have influenced it's inception. MUMPS (Multi-User Medical Processing System), an ANSI standard language, is certainly a derivative of FOCAL.

There are a large number of FOCAL versions available for the PDP 5/8/12. These include FOCAL-69 and FOCAL-8 by DEC. OMSI PS-8 FOCAL and FOCLF are parallel developments based on a marriage of DEC FOCAL-69 and DEC FOCAL-D. This version ran on a system with a DF32 giving each of 7 users a 4K FOCAL-69 since the OMSI and FOCLF versions, there must be close to 25 known versons to run on 12-Bit computers. German and French versions of FOCAL exist. FOCAL runs on NOVA 1200s at Georgia Tech & Queens College, Dublin. Z80, 8080, and Wayne Wall's 6502 Micro-computer versions are running. Other versons are Rob Warnoch's for the PDP-10, West Virginia Univesity FOCL-11 for the PDP-11, Jon Zellowitz for the IBM-370, and of course DEC has PDP 9/15, RT-11 FOCAL, and MPS (8008) versions in addition to its 8 line. Many of these "FOCALS" incorporate laboratory functions peculiar to the equipment on which the language is to be used.

INTENT

FOCAL is intended to be easy for the novice to use. No feature is intended to compromise this ease. At present, there is no known FOCAL that meets the standard put forth in this draft. Therefore, all versions of FOCAL need to be examined to determine conflicts. If all implementors were to work towards the standards proposed herein, A FOCAL library could be developed at DECUS that anyone having a 'standard' FOCAL could use. This requires all FOCAL to be able to input and output ASCII code programs and follow a fixed command set.

A variety of FOCAL implementations will continue to be created for a variety of memory sizes, processors, and applications. It would be desirable if all these creations followed the philosophy and guidelines of a common standard even if they are a subset of the standard using only selected features. It is hoped that all the features of the standard could be implemented on a 16K 12 bit processor with mass storage and some operating system. In such systems, 4K is available for LIBRARY I/O, 4K is available for FOCAL, Functions, and a Math Package, 4K is available for the Program and 4K for Variables.

USING FOCAL COMMANDS

The 'greater than' (> (ASCII 276)) is the prompt which FOCAL will print to tell the user it is in command mode. Most FOCALs for the PDP-8 use the * as a prompt, but this conflicts with the Command Decoder. Others have used _ (FOCAL 5/69) and # (FOCLF). The > points to the command line. FOCAL‾commands may be direct or indirect. A direct command is typed after the greater than and is executed immediately. The form for the direct command is:

```
>COMMAND
```

An indirect command is always identified by a line number. Indirect commands are not executed until program control passes to the group/line number associated with the command. The format for indirect commands is:

```
>GG.ss COMMANDS
```

The range of line numbers (GG.ss) is from 00.00 to 99.99 (although some implementors may be limited to fewer say to 31.99 in 12-bit machines). It is not necessary to type leading or trailing zeros; E.g., 2.1 is equivalent to 02.10 and 2.010 is 02.01.

All FOCAL commands must be followed immediately with a space (ASCII 240) or other terminator (i.e.always a tab and sometimes a comma). All FOCAL command lines must be terminated with the RETURN key (ASCII 215). The LINEFEED key (ASCII 212) may be used to echo the input line. All indirect commands must begin with a line number and may be separated by a Semicolon (ASCII 273). The Comma (ASCII 254) is used to separate repeated calls to non-branching commands. Tables of equevalents to the above terminators should be developed.

All FOCAL standard commands are single letter commands. Single letter commands are a definate feature of FOCAL and should not be deleted. Double Word commands are allowed for expansion purposes.

FOCAL must be interruptable from the keyboard !!!

Control P (CTRL/P (ASCII(120))) is the BREAK CHARACTER.

FOCAL is a Restartable system program.

SINGLE LETTER COMMANDS

@ INDIRECT ON Uses string expressions as Commands

A ASK ["QUERY",X,:,!] Accept value of X from input device

B BREAK [L1] or [(E1)[L1,[L2]L3]] Exits from a FOR loop

C COMMENT Ignores the rest of the line

D DO [G1,L1, etc.] Calls a line or a group as a subroutine

E ERASE [G1] Delete all or part of the program

F FOR X=E1,[E2,]E3;(commands) Executes loop (E3-E1)/E2 times

G GOTO [L1] Branch to line L1

H

I IF(E1)[L1,[L2],L3] Transfers to L1,L2,L3 on sign of E1
[Note: No open enclosure & IF=INPUT (see
Two-letter commands)]

J

K

L LOGICAL (E1-E2)L1,L2 Logical IF; L1=YES,L2=NO
[Note: No open enclosure & LOGICAL=LIBRARY
(see Two-letter commands)]

M MODIFY/MOVE [L1,L2]/[G1,G2] Edit and/or Move lines or groups

N NEXT [L1] or (E1)[L1,[L2],L3] End FOR loop,resume at L1

O ON (E1) [G1,G2,G3] Calls a subroutine selected by sign of E1
[Note: No open parenthesis & ON=OPEN (see
Two-letter commands)]

P PLOT Reserved for analog or digital plotters

Q QUIT [L1] [,error code] Return to command mode and GOTO L1

R RETURN [L1] or (E1)[L1,[L2],L3] Exit from a subroutine (DO)

S SET X=E1[,Y=E2] or [X=Y=E1] Put value of E1 in variable X
if no =, evaluate/execute the expression

T	TYPE [E1,"TEXT",!, ,:,%,$]	Generate alphanumeric output
U	USER	Single letter command for user developement
V	VARIABLE	Reserved for Variable declorations
W	WRITE [E1,E2,E3]	Lists all or part of a program
X		
Y	YNCREMENT [E1,X,Y,E2,-Z]	Increment or decrement variables by E1, E1 defalt = 1
Z	ZERO [X,Y...]	Sets some or all of the variables to zero

Ln are Line Numbers from 00.00 to 99.99 - excluding integers
Gn are Line or Group Numbers from 0 to 99 (0 = next or all)
En are Arithmetic Expressions - - [] Enclose optional items

RULES for Group Calls

1) Line numbers ending in .00 (integer values) are interpreted by DO, ERASE, MOVE, ON, and WRITE to imply all lines in that group. Other commands treat such numbers as referring to only the single line XX.00 (if it exists.)

2) Negative Line numbers are interpreted by these same commands as Partial Group operations. ie Start here and continue; DO -1.5 = start at line 1.5 and continue to the end of the group.

3) Line numbers less than 1.00 refer to lines in the Current Group; GOTO .5 = GOTO 9.5 if executed in group 9; DO -.7 = start at XX.70 and finish the group (XX=CURRENT Group)

4) Line number 00.00 refers either to the entire program (DO or WRITE) or to the next command (IF, ON, NEXT, and most other commands)

FOCAL must be interruptable from the keyboard !!!

Control P (CTRL/P (ASCII(120))) is the BREAK CHARACTER.

FOCAL is a Restartable system program.

OPERATORS

A) ARITHMETIC

```
   ( ) [ ] < > [ ]    Four equivalent sets of enclosures
^  Exponentiation          Raise to positive / negative powers
*  Multiplication          Note especially that multiplication
/  Division                and division should have equal
-  Subtraction or Negation priority !
+  Addition
$  Concatenion            For string variables
```

B) ASK/TYPE

```
$  DOLLAR SIGN           Dump symbol table [as series of SETs]
,  COMMA or SPACE       Separates variables and/or expressions
"  String delimeter     Material in quotes will be printed out
'  Numeric value of ASCII                          Set E1 = 'A
!  Carriage return/linefeed  Starts new line for input/output
@  Input mode switch                     Input in OCTAL/HEX mode
#  Escape sequence                   Used to control CRT terminals
:  Tabulation                         TYPE :15 spaces to column 15
                                      ASK :-15 skips 15 characters
%  Format control %5.02 = 5 Digits, 2 decimal places maximum
    (for output only) %.04 = 4 digits in scientific notation
                      %0 = Output in OCTAL/HEX
```

c)MODIFY/MOVE

```
CTRL/P                      Aborts the command, line unchanged
CTRL/G (bell)                    Selects a new search character
CTRL/L (Form Feed)    Searches for next occurrence of character
CTRL/N                     Advance one character (N = Nibble 1)
System Line Kill Symbol     Deletes all characters to the left
RETURN                  terminates line at the current position
ESCAPE ($)      String Search $ replacement $[E1]$ do E1 times
CTRL/R                 Copies the remainder of the line unchanged
RUBOUT [or BACKSPACE (CTRL/H)]  Removes the previous character
                               but never corrupts the line # !
```

TWO-LETTER COMMANDS

Two-letter commands are determined by the first letter of two words/or two letters separated by a space. The letters "L", "I, and "O" will check for an enclosure, then check for a second letter. These Two-letter commands are used to interface FOCAL to the Host Monitor or Operating System.

In stand-alone FOCAL, these Library commands are not available, or have other meanings, such as LINK to a second program area, etc.

I. DIRECTORY COMMANDS

L A	List All [,Emptys]	List all files on a device
L P	Library Print [.Exten.]	List only FOCAL [or other] files
L L	Library Load	Load a FOCAL 'Save' file
L D	Library Delete name	Delete file 'name' from device

II. PROGRAM COMMANDS

L C	Library Call name [G1]	Call program as subroutine
L R	Library Run name [L1]	Load program and start at L1
L S	Library Save/Store name	Save the current program

[G1] indicates which line or group will be called by 'L C'
'L C' will save the current program as FOCAL.TM if unsaved.

III. INPUT/OUTPUT

The form of the file specification commands is not determined at this time other than the desirability of using INPUT and OUTPUT as the basic commands. FOCAL should be able to handle Variable File Names (supplied by the program) and multiple I/O files. Something like OPEN FILE NAME.EX or O F [E1].MAD and INPUT FILE 5 would be desirable. The commands INPUT and OUTPUT without any arguments should select the terminal for I/O. An ECHO feature is also desirable to allow multiple INPUT/OUTPUT simultaneously.

VARIABLES

Variable names may be of any length, but only the first four to six characters are stored. It is a restriction of many FOCAL's that use only the first two letters. Single subscripting is Standard. Double subscripting is essential feature. A method of storing different variable types (Integer or String for example) is also essential, however very little development work has been done to date. There is also a need for "local" variables where on assignment, values are made to the deepest level (most local) at which it exists, or the present level when it does not exist at any level. On retrieval, a variable has the value at the deepest level at which it presently exists. This would allow complete shadowing of higher level (more global) variables. At present, this feature is only encoded in IBM-370 FOCAL.

Variables will be stored in the format of the Math Package associated with FOCAL, unless it is flagged to be some other type of variable. Undefined variables should be automatically assigned the value of zero. Some form of "zero replacement" of variables is desirable. This means variables with the value of 'zero' only are stored as 'name', 'subscript', and 'zero exponent'. This saves up to 50% in storage.

Programmers should obtain a maximum amount of precision from FOCAL. In several machines, especially the PDP8, FOCAL is the only higher level language to give the user ten digits of accuracy.

FUNCTIONS

Function names are a set of letters begining with 'F'. Functions can be divided into two classes. Part one is mathematical in nature and essential to FOCAL. Part two does INPUT/OUTPUT and STRING manipulation. The letter 'F' followed by an enclosure shall be reserved for the FOCAL STATEMENT FUNCTION. F(G1,E1,E2,E3) will execute Group or Line 'G1', using E1,E2,E3 as local variables. The Function returns the results of the last arithmetic expression processed by the subroutine. The letter 'F' followed by an expression is FOR !

An essential part of FOCAL Documentation is sufficient information for a user to develop his own I/O Functions.

I. BASIC NUMERICAL

FABS(E1)	Returns the absolute value of E1
FINT(E1)	Returns the integer part of E1
FINR(E1)	Returns the rounded integer of E1
FRAC(E1)	Returns the fractional part of E1
FCOS(A)	Compute the Cosine of A
FSIN(A)	Compute the Sine of A
FTAN(A)	Compute the Tangent of A
FEXP(E1)	Exponential of E1 ([base e])
FP10(E1)	Exponential of E1 ([base 10])
FSQT(E1)	Square Root of E1
FLOG(E1)	Naperian logarithm of E1
FL10(E1)	Base 10 Log of E1
FASN(A)	Compute the Arcsine of A
FACS(A)	Compute the Arccosine of A
FATN(A)	Compute the Arctangent of A
FHTN(A)	Compute the Hyperbolic Tangent of A
FSGN(E1)	Sign = -1,0 +1
FRAN( )	Return pseudo-random number (0 to 1)

A is an angle. All together there are 12 basic Trig. Functions (SIN,COS,TAN)(ARC,HYP, and plain) and 12 more with CO in front! Could FOCAL do them all?

II. INPUT/OUTPUT FUNCTIONS

FIN([E1])	Read single character, return decimal ASCII value E1 = string variables declaration
FOUT(C)	Outputs character code C, returns zero
FIND(C)	Search for code C, return C or 0 if EOF
FTAB()	Read current print position

III. RESERVED I/O & STRING NAMES

FDAY	Works with System date
FADC	Analog to digital converter
FDAC	Digital to analog converter
FTIM	Time of day
FCLK	Ticks of the clock
FSR	Read switch registers
FSTR	Convert FP# to string #
FLEN	Length of string
FEQL	String comparisons
FVAL	Convert string to FP number
FPOS	Position locator in string
FMID	Find middle of string

and others

COMMENTS REGARDING NEWSLETTER 27

1) Re Dan Smith's "Hints", the "Brookstone" connectors are likely SPC Technology "A" Mode connectors. If not, try them; $40 gets 100 of them. Remind me of a modern version of the old Jones plug with a few spiffy additions. While not gold-plated, the SPC connectors ARE silver-plated and have stainless steel leaf springs in the contacts: surely worth $.40 ea! Newark Electronics used to sell them in packs of six and a hundred; they stopped sending me catalogs and I stopped ordering stuff from them. A dig thru a nearby EEM (invaluable to have nearby, but not worth purchasing) should yield an address and phone for SPC; you could then pump them for closer or cheaper distributors. I personally prefer the B1700 tactic of using fixed-size modules (they are talking about word size, not connectors) and just using multiples of them. Favorites are the 50-pin connectors used in key telephone sets and military AN connectors. The AN connectors are nice when a quarter-inch panel is involved; some of the blue language involved in hand-milling an oblong hole for DBM connectors returns in full force while trying to hand-solder the AN things. AN stuff is also multiple-sourced, extremely rugged, and often available as USG surplus. Looks hideous-will not impress visitors unless they're Signal Corps vets.

2) Re patch panels. We use 16 X 20 MAC panels to connect up analog instrumentation with our PDP-12. The panel runs full differential and contains a bit of electronics to ground-isolate single-ended stuff such as monitor 'scopes. We milled out room for more "Blue Ribbon" connectors at the bottom of our PDP-12 so there are five of 'em down there that give access to the AA50 DACs, the regular and extended relays (used the extra set of contacts on the boards; can't imagine 54 wires running to those 5-way posts), and the sense lines. Then we popped out the unused 'phone jack holes on the extended relay panels and put in 0.206" jacks for all the sense lines (gives some measure of "wrong plug" protection) and .250" jacks for 4 DAC outputs. These modifications make the PDP-12 much better for on-the-spot "tinkering" as well as keep down the "rat's nest" of wiring for long-term stuff. MAC panels, however, are not cheap. Our initial noise problems came from ground loops, from not taking advantage of the differential inputs offered by the PDP-12 (and any other ADC system worth its salt) analog and Schmitt inputs. Then came a round of isolating any single-ended auxiliary equipment. The remaining problem is a Syston-Donner time code system whose DTL reader to search unit cable radiates some noise into our analog coax. Both run in the same cable tray, so if the good 'ol square law holds, we can literally kick out that problem.

3) Re COMPUTER DESIGN: I enjoyed John Buckley's column in CD for some time, especially those showing some rather questionable tactics of the phone companies (the one on what a DAA device really "protects" is a classic). Lately, however, he seems off on evaluating phone tariffs to show folks who have sixty WATS lines how they can save money by using new DDD rates at late night, etc. It's hard to be enthusiastic about new COMSAT tariffs when you're still feeding a leased line at 1200 baud.

4) Re VELstraps: THANKS, DAN!!!

5) Re AA50 DACS: Don't try to feed them to Schmitt triggers 'cause they speak with spiked tongue! Need to deglitch them with a follow and hold ckt. or just RC filter them and pray, I guess.

6) Re my complaint about FLAP: It's been a long time since I used a macro assembler, folks, so it took a while to notice that REPEAT N does in fact repeat the line N times- it just doesn't SHOW that action.

7) Re FPP-12 interaction with PDP-12 display processor: Thanks again DEC. Discovered this little gem when a program with QANDA fired up the FPP. Glacier surprise! Characters lost pieces of themselves and everything looked really nasty. A desperate call to Bob Hassinger produced a vector to Thomas McIntyre and some electrical and historical background. I chose the software delay correction and used PUSHJ/REST instructions of the KF12B Automatic Priority Interrupt to hop to my delay:

```
                (...FIELD N)
                DSC 2           /DISPLAY 1/2 OF CHAR.
                IOB
                PUSHJ 3;PMODE   /CALL DELAY
                CTIMER          /IN FIELD 3
                LMODE;DSC I 2   /DISPLAY OTHER HALF OF CHAR
                IOB             /ANOTHER DELAY IF NEEDED, ETC...

                FIELD 3...
        CTIMER,         SRO I           /THIS INST. TAKES 3.2 USEC
                3737            /SKIPS EVERY SIXTH CYCLE
                JMP CTIMER      /THIS INST. TAKES 3.2 USEC
                IOB
                REST            /RETURN TO CALLING SEQUENCE
```

(I've really come to love the KF12B either in interrupt mode or just sailing around in memory- DEC should inflict it on all 8 users. Whenever a field or page or bank threatens to overflow, I scream "Mr. Wizzard, I don't want to be a minicomputer programmer anymore!" and PUSHJ somewhere else. Not exemplary programming, but it works and initially saves time) The loop as written above won't work: not enough delay! Now DSC takes 56 USEC max. to complete (all times are supposedly variable 20% either way), so if we allow 5 memory cycles each for the PUSHJ and REST to do their thing, the delay loop should kill 56-(10*1.6)=40 USEC. To execute the PUSHJ and REST instructions from LINC mode takes 5.9 USEC each: 56-[(10*1.6)+(2*5.9)]=28.2 USEC. O.K., so the SRO I loop uses up 38.4 USEC (loop of 6 was closest .GE. to desired delay time)- and still parts of characters get blown away. The empirical value is something greater than twice the above! Anyone got any pointers?

(The above material was unsigned, the return address was: Benignus, MD-58, Environmental Protection Agency, Research Triangle Park, North

Carolina 27711. The material looks as though it is from Brian Converse - RH)

Fun and Games With the FPP-12: Part II- Fixed-Point Mode for Signal Averaging

[Disclaimer: the signals being averaged here are 10 bits' precision one's complement ADC values of a PDP-12 (FPP-12 code does the 1's->2's complement conversion as signals are summed) and the heuristic employed may not succeed for 12 bits' precision values, or for physical processes other than human EEGs.]

The scheme I use most likely is an algorithmic process that can be parameterized and applied to the FPP-12 in such a way that any situation can be input and the necessary values cranked out mechanically. To me, it's a heuristic, if only because it resulted from a couple hours' work modeling the problem on a pocket calculator. Thus, this explanation is best followed by using a pocket calculator as a visual aid.

The basic problem in fixed point is that whenever a result is not a fraction (e.g. 1.000078), the FPP-12 neatly folds up shop and, if you've got the interrupt on, comes tugging at your sleeve for further instructions. Since the PDP-12 ADC values are 10 bits' precison and the FPP-12 exhibits about 23 bits' external precision, such things as summing for signal averaging take a good long time to overflow. Tell the FPP-12 they're fractions and stuff them into the 10 LSB's of the FAC fraction and forget it. For reasonable values of "N" (for averaging), it really dosen't matter what happens with the sum. Once calculation of the average starts, however, the quotient can shoot all over the place and very likely overflow the FAC. While what goes into the sum may not matter so much, what ends up there matters a great deal once division starts.

One approach would be to switch to floating-point mode, float the sum (an ALN and an FNORM does that), divide and fix. A judicious selection of the averaging sequence, perhaps a few locations extra for floating-point buffers, and a parallel program waiting to eat the result make this tack work. The nice part is that the floating point mode handles all the little nasties. The extra time involved occurs at a low duty cycle for reasonable "N"s and shouldn't be bothersome.

For various reasons (incl. perversity) not applicable to the discussion, the floating-point approach was not used. However, it is well worth considering in other situations. But the problem became, "how is this done in fixed-point mode?" An empirical assumption was made whereby the 8-digit octal fraction became two 4-digit octal numbers. A further assumption was made that this system could be modeled on an 8-digit decimal pocket calculator. A 6-digit "world" is assumed, broken into two 3-digit halves. Incoming data is 3 digits or less and is summed into the 3 least significant digits of each 6-digit summing "location". The "N" for the average is stored in the 3 MOST significant digits of its 6-digit "location". On a TI-1250 (a typical cheap 8-digit pocket calculator) then, consider some fake data: 21,122,25,19,-4, 17,23,21,21,25,31,22; N=12. The average, to 3 digits, is 29. But in this case, the data is summed as fractions: 0.000 021, 0.000 122, 0.000 025, etc., and the sum, 0.000 343, is divided by 0.012 000 to give an "average" of 0.029. Realistically, averages develop biases and such so that in some cases it's better to ALN the result one way or another. If the signal average is of a signal from Tau Ceti and nothing is there, fine. If they're sending us sine waves, fixed-point operations begin to exhibit a lack of dynamic range. For EEG behavioral work, the "average" often requires a boost (ALN by -3 or so; NOTE- ALN works on ONE's complement values, so LDX -3,7 followed by ALN 7 will arith. shift the FAC fraction left by 2, not 3 because FLAP will make a nice TWO's complement -3 for you...) to equal amplitudes of typical input signals. But note the result of the calculator experiment: the quotient is left in the most significant half of the fraction. By using the "trick" of indexed, offset FPP instructions described in newsletter 27, an N-point "average" can be fixed into an N+1 point single-precision buffer almost as part of the averaging process. The actual values in the FAC and in storage are irrelevant since they are composed of unrelated numbers forced to be "fractions". The summing/averaging process becomes a black box that "eats" single-precision data and sends back single-precision data.

(The above material was unsigned, the return address was: Benignus, MD-58, Environmental Protection Agency, Research Triangle Park, North Carolina 27711. The material looks as though it is from Brian Converse - RH)

FROM BRIAN CONVERSE

Bob - This is some more work to extend what I sent you recently on feeding long strings of 12-bit signed integers to the FPP-12 for processing. The difference in this printout is that I think I've solved the problem of doing a float operation for floating point numbers (I was personally only concerned with fixed-point values). My checking of the technique for nasties has been lacking and I can't even prove that the method is faster than successive XTAs, but the few values I try come out reasonable.

```
FLAP V  50  MAR 20, 70
            /ANOTHER FPP TEST PGM
            /*****NOTE! MUST FIX LINC CODE'S USE OF 500
            /AS "NLOC" BY PATCHING WITH ODC, ELSE PGM
            /SEES 0000 (FEXIT) AT LOC 500 AND CRASHES!
                    ORG 400    /FITS IN WITH LINC-MODE TEST PGM ELSEWHERE
00400 0006          STARTD     /MUST DO IN 2-WORD INDEXING=FIXED POINT
00401 0101          LDX -1,1   /PICKUP POINTER FOR SINGLE-WORD VALUES
00402 7777
00403 0102          LDX 14,2          /ALN INSTRUCTION CONSTANT=12 DECIMAL
00404 0014
00405 0103          LDX -1,3          /OUTPUT POINTER
00406 7777
00407 0510          FLDA BUFF,1+      /GET THE "0001" AND "7776"
00410 0540
00411 0012          ALN 2             /"ASR" THEM UNTIL 0001 IN LSW
                                     /(NOTE FINE DIAGRAMS IN FPP-12 MANUAL
                                    /THAT SHOW HOW "ALN" WORKS SO YOU
                                    /WON'T HAVE TO RUN TEST PROGRAMS
                                    /TO DOPE IT OUT)
00412 6530          FSTA FIRST,3+   /SAVE AS PROOF
00413 0550
00414 0410          FLDA BUFF+1,1   /NOTE FANCY INDEXING NOW GETS
00415 0541
00416 0012          ALN 2           /2ND AND 3RD WORDS AND WE "ASR"
00417 6530          FSTA FIRST,3+   /TO 2ND IN LSW OF FAC "MANTISSA"
00420 0550
00421 0510          FLDA BUFF,1+    /NOW WE GET 3RD AND 4TH WORDS
00422 0540
00423 0012          ALN 2           /ETC., ETC...
00424 6530          FSTA FIRST,3+
00425 0550
00426 0410          FLDA BUFF+1,1
00427 0541
00430 0012          ALN 2
00431 6530          FSTA FIRST,3+
00432 0550


            /NOW TRY TO DO SOME FLOATING-POINT LOADS
-->     FAC=0027;0000;0000
00433 0005          STARTF          /FLOATING-POINT MODE A MINIT
00434 0400          FLDA EXPOLY     /EXP->27 OCTAL (=23 DECIMAL)
00435 0700
00436 0006          STARTD          /THIS'LL HELP FAKE AN "XTA"
00437 0104          LDX -1,4        /JUST TOO WIERD, THIS INDEXING BIT
00440 7777
                                   /CAN'T JUST SWITCH MODES IN THE MIDDLE
                                    /OF A LIST, GOTTA USE ANOTHER INDEX
                                    /OR RESET CURRENT ONE OR USE ADDRESS
                                    /OFFSET.
-->     In fixed-point mode:
        mantissa changes but exponent survives
00441 0510          FLDA BUFF,1+    /FLOATING-POINT SETUP TRY
00442 0540
-->     This is what XTA, working off an index regester for input value
        would do:
00443 0012          ALN 2
-->     Optional:
00444 6530          FSTA FIRST,3+   /TO VERIFY FIXED-POINTNESS
00445 0550
00446 0005          STARTF          /HELPS IF U WANT TO DO FL. PT.
-->     This does rest of an XTA emulation:
00447 0004          FNORM           /ASSUMES EXP. WAS SOMEHOW=23!
00450 6540          FSTA SECOND,4+  /WE'VE ESSENTIALLY DONE AN XTA!
00451 0630
-->     Whether this beats zipping X0 around via SETX and doing XTAs,
        I'am not sure, but you could use fancy indexing and base
        registers with my way

00452 0400          FLDA EXPOLY          -->     Start next
```

```
00453 0700
00454 0006          STARTD          /(GETS LIKE LINC/PDP MODE...)
                                --> (i.e. STARTD...STARTF...etc.)
00455 0410          FLDA BUFF+1,1   /FL PT TRY ON POSITIVE VALUE
00456 0541
00457 0012          ALN 2
00460 6530          FSTA FIRST,3+
00461 0550
00462 0005          STARTF          /TOO BAD THERE ISN'T A "STARTS"
00463 0004          FNORM           /INSTRUCTION: SINGLE WORD VALUES
00464 6540          FSTA SECOND,4+  /AND INTEGER MATH WOULD BE NICE
00465 0630

00466 0400          FLDA EXPOLY
00467 0700
00470 0006          STARTD
00471 0510          FLDA BUFF,1+    /TRY A BIG
00472 0540
00473 0012          ALN 2
00474 6530          FSTA FIRST,3+
00475 0550
00476 0005          STARTF
00477 0004          FNORM
00500 6540          FSTA SECOND,4+
00501 0630

00502 0400          FLDA EXPOLY
00503 0700
00504 0006          STARTD
00505 0410          FLDA BUFF+1,1   /LAST  :110
00506 0541
00507 0012          ALN 2
00510 6530          FSTA FIRST,3+
00511 0550
00512 0005          STARTF
00513 0004          FNORM
00514 6540          FSTA SECOND,4+
00515 0630

00516 0400          FLDA EXPOLY     /TRY TO SEE WHAT'S HAPPENING
00517 0700
00520 0006          STARTD
00521 0410          FLDA BUFF+1,1   /GET THE 110 AGAIN
00522 0541
00523 0005          STARTF          /THIS TIME, DON'T ALIGN
00524 6540          FSTA SECOND,4+  /JUST SAVE IT TO SEE IF EXP
00525 0630
00526 0000          FEXIT           /ETC. IS OK
00527 0000          FEXIT

                    ORG 540
00540 0001  BUFF,   0001            /A TEST LIST OF FAKE SINGLE-WORD
00541 7776          7776            /VALUES WE WANT TO FLOAT
```

```
00542 1234              1234               /ALL 7 MAY NOT GET USED IN ABOVE
00543 5670              5670               /CODE. PDP-12 ADC VALUES MUST BE
00544 5252              5252               /CONVERTED TO 2'S COMPLEMENT OR
00545 2525              2525               /CONVERTED (VIA JLT, ETC.) IN THE
00546 3567              3567               /FPP CODE AS VALUES ARE LOADED
00547 0110              0110


                        ORG 550
00550 0000   FIRST,     0                  /THIS IS WHERE FIXED POINT FPP
00551 0000              0                  /INTEGERS APPEAR; APPROPRIATELY
00552 0000              0                  /STUFFED WITH ZEROS UNTIL VALUES
00553 0000              0                  /STORED BY FPP
00554 0000              ZBLOCK 40


                        ORG 630
00630 0000   SECOND,    0                  /WHERE OUR 3-WORD FL. PT.
00631 0000              0                  /VALUES SHOULD APPEAR
00632 0000              ZBLOCK 40


                        ORG 700
00700 0027   EXPOLY,    27;0;0
00701 0000
00702 0000

             /RESULTS:
             /LOC 550, FF
             /0 (550)
             /1 (551)
             /7777 (552, ETC.);7776
             /0;1234
             /7777;5670
             /7777;5252
             /0;2525
             /0;3567
             /0;0110
             /LOC 630, FF
             /0013 (630)         \_ = -2526
             /5252 (631)         /¯
             /0 (632)
             /0013 (633; ETC);2525;0000  -->     Seems O.K.
             /0013;3567;0000             -->     Ditto
             /0007;2200;0000 (LAST VALUE)
             /0027;0110;0000 (CHECK OF EXPON. SETUP)
             /0
```

from Dan Smith
EYE RESEARCH INSTITUTE
OF RETINA FOUNDATION
20 Staniford Street
Boston, Massachusetts 02114
(617) 742-3140

28-APR-78

ADDENDA: Brian Converse has identified the Brookstone S-4650 quick-connectors I like so much as SPC Technology "A" Mode connectors, listed on p. 428 of the 1977 Newark Electronics catalog #103. Roy A. Standing reports that the blue vinyl tabs that make DECtape stick to the reel are available as $M I.D. #84-9800-3766-1, Cat. No. 656 5/16x4"; 3M Company Data Recording Products, 350 South Lewis Road, Camarillo, California 93010, and recommends 3M as a source for fully-certified DECtape (481-3/4-260R42-DEC) I forgot to identify the Philips stepping motor chip in my last notes: SAA1027 Stepper Motor IC Driver, North American Philips Controls Corp., Cheshire, Conn.06410, (203) 272-0301, about $15 each or about $8 in quantities of ten; recommended motors, K82102-P2, K82201-P2, K82401-P2, K82701-p2, K82801-P2, all about $25-$30 each. K82801-P2 is the biggest and strongest, 7.5 oz-inches torque. Another interesting electronics source is Herbach and Rademan, 401 East Erie Avenue, Philadelphia, Pennsylvania 19134, the Rolls Royce of surplus junk houses (only their stuff generally isn't junk and generally comes with full technical information).

SCROLL V15: I certainly hope the present questions about SCROLL's future, especially on video terminals, are resolved, as SCROLL has the best human engineering of any editor I've ever used (and I've used nearly a score, beginning with "Expensive Typewriter" and Dan Murphy's original TECO on the PDP-1). For present PDP-12 SCROLL users, some very exciting improvements have recently been made by Eric Swanson and Jim van Zee, which Jim says they could make available to existing SCROLL "licensees." In their version, SCROLL is invoked by the CCL ".EDIT" command (hence commands are automatically remembered); files are always closed to the exact size on exit; files can automatically expand into empty blocks following the file; and options are available to create backup files and to guarantee empty space for expansion. In my experience so far, these are a very comfortable and usable set of options that have already saved me hours of frustration and mental anguish. A very nice mode is guaranteed expansion without backup; the first time this is invoked, the file will get copied into a place where it can expand, but usually thereafter it stays in the same place and just grows. There is also a feature to move command lines back and forth from the text area to the command-line area, permitting command lines to be edited and saved.

LISTINGS WITH FOTP: Re RH's wish for a directory listing capability in FOTP, I agree, but will mention that with a video terminal you can get a quick peek by specifying /L with no output; this lists all files of the specified type on the terminal and performs no transfers. No sizes or dates, of course.

OCOMP.SV Jim van Zee sent this to me along with his last HWF revision. He says it's by Dennis McGhie and he's not sure what its status is; it's in the DECUS catalog (8-609), but Jim thinks that may not be the latest version. Anyway, it is, in my view, a near-essential OS/8 CUSP which almost everybody needs, and we ought to try to persuade DEC to distribute it (a la FUTIL). It is for comparing binary files, and supports comparisons both by block number and word-within-block, and by core locations for .SV files. Can anybody out there get EPIC to work? I can't.

EXPIP In response to my FOTP wish list, Lars Palmer sent me a copy of EXPIP, which has existed since 1975, is in the DECUS library (#600c), and which contains virtually all of the requested functionality! However, at least in its present form, EXPIP cannot be recommended as a gold-plated essential (as DIRECT V5, FUTIL, and OCOMP are). For one thing, the version I received does not seem to handle the extended V3D dates, so that 1-APR-77 is treated as more recent than 1-APR-78, for example. EXPIP seems to have been developed independently and concurrently with FOTP, and as a result the EXPIP conventions and options are a little odd to a new user. My guess is that the most reasonable way to use EXPIP is to implement the desired options as CCL commands, as suggested in the writeup; for example, .UPDATE DTA1: transfers to DTA1: all files on the disk that exist on DTA1 but have a more recent date on DSK:. I haven't used EXPIP much because of the date problem; the documentation is full of warnings about options that cannot be combined, bad things that can happen if you CTRL-C out of EXPIP, etc. It is reassuring to have these things explicitly documented, but it makes me wonder if EXPIP, even if accessed only via CCL, is safe for an inexperienced user.

OS/8 and Terminal Device Codes: We just got an HP2648A graphics terminal, which is the greatest thing since sliced bread. It operates off our DP12B. A few brief words on a problem we had: it turned out that our PDP-12's backplane wiring was set for two stop bits, and the terminal at speeds over over 110 baud uses only one. When the option was installed, the HP terminal hadn't been delivered, so we borrowed a DECscope to check it out. The result of the stop-bit mismatch is that the DP12 can send successfully to a terminal, and can receive successfully single characters struck on th keyboard (even if, e.g. an automatic repeat is invoked, since these usuall only run 10-20 characters/second). It can also successfully send and receive to itself, hence pass the diagnostic. What it cannot do is receive a continuous stream of characters at 9600 baud. It happens that the HP terminal sends such streams, e.g. in response to status requests, in "block mode," etc. Anyway, the problem seems simple enough now, but the combined stupidity of yours truly together with some assistance from DEC and HP managed to make it incredibly difficult. The HP guy had heard of "an extra ROM that you need to use HP terminals on DEC machines" that he wanted to install (it turned out to be a ROM to implement a special communications protocol in PDP-11/70 IAS!), there was lots of finger-pointing, it was fun.

The DP12B is identical to the teletype interface except that it uses device codes 40/41 instead of 03/04. The question then arises of how and whether OS/8 can be modified accordingly, so that the terminal can be the console device. I am presently just slogging through the CUSPS with FUTIL, patching as I go. The answer seems to be that it is not very difficult, but there is no algorithm. I should add that I think there could be real problems if a teletype (03/04) were not available, so that during the transition period unmodified CUSPS would still be used from the teletype.

I think that DEC should address this problem seriously. I imagine the only solution is to have a reserved word in the system area or blocks that contain the device codes, and have CUSPs read them and patch themselves accordingly. It is beyond the scope of the SET command, because every OS/8 program has to access the keyboard if it is going to handle CTRL-C correctly; even if SET were smart enough, I believe all the patching could take unacceptably long on a floppy (and certainly on a DECtape).

OS/8 terminal device codes, cont.

I hope eventually to give a summary of patches, but this time will pass on what I've found out in general.

Virtually all CUSPS that output error messages on the terminal do so directly. This is a little disappointing; it is understandable that single-character inputs (like Y and N answers to FOTP /Queries) can't be done through the TTY: handler, but one might have thought that the outputs would be. I guess it's hard to find room for the buffer. The redeeming feature is that the output code is usually obvious; a FUTIL search with SET MASK 7770 for WORD 6040 will often turn up a consecutive 6046, 6041 and nothing else.

In present standard OS/8, it can be assumed that all terminal I/O will be via codes 6041, -2, -4, -6 and 6031, -2, -4, and -6 because 8I's and earlier didn't have any other codes. Now that DEC has casually decided that pre-omnibus machines don't exist (all you PDP-11 users reading the VAX announcements pay attention!) this may change, but it's true for the present. Hence any 603- and 604- 0's, 3's, 5's, and 7's are not terminal I/O can be ignored. 6030 crops up a lot because it's -1000(10) in octal! In the following, "making the straightforward patches" means changing the device codes in all instructions of the form listed above.

The command decoder, keyboard monitor, and ODT are easily converted by making the straightforward patches in system blocks 7-15 and 51-64. CCL patching is straightforward. When making the system patches, you will also doubtless want to install the video terminal rubout patches given in newsletter 24, pp. 11-13.

Patching handlers is fairly straightforward; remember that all of 'em check for CTRL-C on the keyboard, and hence all need patches if you want to CTRL-C from somewhere else. Contrariwise, the AS33 PTR: and PTP: device codes don't want to get changed unless the terminal also has a paper tape reader and punch on it. You can make the patches in the handler blocks themselves, using RESORC to tell you which is which, but I prefer to make the patches in BUILD so I won't have to repatch when I reBUILD. To patch BUILD itself, make the straightforward patches in field 0 only. To patch the handlers within BUILD, note that handlers begin at location 10400, appear in the order in which they list via $PR; all handlers are present, whether active or inactive; one-page handlers occupy one page and two-page handlers, two; the code in field 1 is the actual handler code itself (the names and entry points get loaded into tables in field 0). Best to use FUTIL to search for all terminal I/O codes in field 1, then use BUILD $ALTER and $EXAMINE for the actual modifications (that way you get less confused about which handler is where).

Modifications to FOTP are straightforward. Pal8 is modified by making the straightforward patches in locations 0-5777 only (don't patch the symbol table!)

DEC improves PIP. As everybody knows by now, DEC has greatly improved the directory listing options in PIP. Just in case it doesn't get into DSN, in time, I think everybody should be warned that they added an even more valuable enhancement to /Y: it bombs if the input handler is not coresident with SYS:. Jim van Zee has fixed this problem, and as you know he also has a patch which restores the directory listings with the correct V3D dates in the new format and without taking up any more disk space, but DEC wants to do things a different way and will release their own patch. In the meantime, I suggest that everybody hang on to their V3C PIP. (I had discovered experimentally myself, while trying to get from V3C to V3D, that new PIP /Y's bombed but old PIP /Y's worked fine; but in my case I kept getting the "bad system head" message, which confused me).

TWO PATCHES FOR FUTIL V7

1. Patch to fix a 'SHOW CCB' problem. (From Jim Crapuchettes.)

```
12024/ 3242  5345
    5/ 1642  7650
    6/ 7450  5236
    7/ 5236  1642

12145/ XXXX  3242
    6/ XXXX  1242
    7/ XXXX  1351
   50/ XXXX  5225
    1/ XXXX  1175

12520/ 0200  0300
```

This patch upgrades V7B to V7C.

2. Patch to fix failure of ODT mapping on .LD modules. (From Ian Templeton.)

```
12075/ 7040  7000

12520/ 0300  0400
```

This patch upgrades V7C to V7D.

Dear Mr. Hassinger:

I have now managed to contact Jim Crapuchettes, who has authorised my changes to FUTCCL. I therefore enclose both a reduced copy of the listing and a paper tape of the source.

He also told me of another bug in FUTIL, and agreed with the one that I found. He asked me to send in both, so I enclose them herewith. He authorised the 'patch numbers'. I have also, at his suggestion, sent a copy to Earl Ellis.

Yours sincerely,

I.M. Templeton

```
/FUTCCL.PA - CCL V1F MODS
/
/ORIGINAL BY:
/
/       JIM CRAPUCHETTES
/       MENLO COMPUTER ASSOCIATES
/       P.O. BOX 298
/       MENLO PARK, CALIF.  94025
/       (415) 323-3009
/
/
/MODIFICATIONS FOR CCL V1F BY:
/
/       IAN TEMPLETON
/       DIVISION OF PHYSICS
/       NATIONAL RESEARCH COUNCIL OF CANADA
/       OTTAWA CANADA K1A OR6
/       (613) 992-2113
/             .
/
/       THESE PATCHES SIMPLY MAKE USE OF VARIOUS
/  HOLES IN CCL VERSION 1F (**** NOTE THIS ****) LEFT BY
/  ZBLOCKS OR OTHER MEANS.  THEY IMPLEMENT THE FOLLOWING
/  ADDITIONAL CCL COMMAND FOR FUTIL:
/
/                           /L  [MODE=LOAD,DEFAULT EX="LD"]
/  .F(UTIL) DEV:FILE.EX/E/O=NNNN  [MODE=OFFSET,EX AS BELOW]
/                           /S  [MODE=SAVE,DEFAULT EX="SV"]
/
/  THIS COMMAND CHAINS TO FUTIL WHICH THEN LOADS THE
/  HANDLER FOR THE SPECIFIED DEVICE, SETS THE OPTION(S)
/  SPECIFIED BY THE SWITCHES AND THEN SIMULATES A "FILE"
/  COMMAND FOR THE SPECIFIED FILE.  ONLY ONE FILE NAME
/  MAY BE SPECIFIED AND IT MAY CONTAIN THE CHARACTERS "?"
/  AND "*", BUT THESE WILL NOT BE USED AS WILD CARD CHAR-
/  ACTERS, BUT AS ACTUAL CHARACTERS IN THE FILE NAME AND
/  EXTENSION.  THE COMMAND LINE IS REMEMBERED IN AN UNUSED
/  REMEMBRANCE LINE, ALLOWING CONVENIENT RECALL.
/
/       SWITCHES:  FOUR (4) SWITCHES ARE DECODED BY FUTIL--
/
/  /E   SET SHORT ERROR MESSAGES
/  /L   SET MAPPING MODE TO "LOAD", SET EXTENSION TO "LD"
/       IF NONE WAS SPECIFIED (& IGNORF FOLLOWING TWO
/       SWITCHES).
/  /O   SET MAPPING MODE TO "OFFSET", SET OFFSET VALUE TO
/       LOW 12 BITS OF "=NNNN" OPTION (& IGNORE FOLLOWING
/       SWITCH).
/  /S   SET MAPPING MODE TO "SAVE", SET EXTENSION TO "SV"
/       IF NONE WAS SPECIFIED.
/
```

```
            /       EXTENSIONS:  FUTIL PERFORMS FILE LOOKUPS WITH 3
            / DEFAULT EXTENSIONS, "SV", "LD" AND NULL.  THE FIRST THAT
            / SUCCEEDS WILL BE OUTPUT.  THIS IS ALSO TRUE FOR THE FILE
            / SPECIFIED TO THE CCL COMMAND.  HOWEVER, IF AN EXTENSION
            / IS SPECIFIED, ONLY ONE TRY WILL BE MADE, WITH THAT EXTEN-
            / SION.  NOTE THAT USING THE "/S" OR "/L" SWITCHES ABOVE
            / ARE THE SAME AS SPECIFYING AN EXTENSION AND ONLY ONE
            / LOOKUP ATTEMPT WILL BE MADE.

            /SOME DEFINITIONS FROM CCL:

      0023  NAME1=  10023   /TWO OF FOUR NAME.EX WORDS
      0024  NAME2=  10024

      3244  STAR=   13244   /"*" EXTENSION LOCATION

      4011  ISKP=   14011   /TO REPLACE (SKP & GAIN 1 SPACE

      5741  ARLOC=  15741   /SAVED POINTER TO "<"

      7000  REMEM=  17000   /REMEMBER SUBROUTINE

      7255  K4200=  17255   /PROVIDE PTR TO GNAME & GAIN 1 SPACE

      7321  FILLP1= 17321   /LABEL IN PATCHED SUBROUTINE

      6666  SEMSG1= 6666
      6675  SEMSG2= 6675
      6703  SEMSG3= 6703
      6712  SEMSG4= 6712

            /
            /.COMP FUTCCL
            /.LO SYS:CCL.SV/I$
            /*FUTCCL$
            /.SA SYS CCL;12001=2003
            /
```

```
/FUTCCL.PA - CCL V1F MODS                   PAL8-V10A 07-APR-78 PAGE 3

      0001            FIELD 1 /ORDER AS IN CCL SOURCE

      2535  *12535

12535 5354            FILDMY  /NEW POINTER

      3121  *13121            /ADD OVER "ZBLOCK 13"

13121 4470  XFUTIL,   4470    /DO CD; ADD "<"; REMEMBER
13122 3244            STAR    /DO SPECIAL MODE DECODE ++
13123 0000            0       /NO DEFAULT SWITCHES
13124 4127            FUTSUB  /SUB. TO DO--REMEMBER & SETUP
13125 6436            YFUTIL  /CHAIN TO IT


            /  ++ NOTE:  BECAUSE "<" IS ADDED TO COMMAND AND A SPECIAL
            /  MODE DECODE IS DONE (ALLOWING ONLY 1 OUTPUT), ONLY ONE
            /  (1) FILE MAY BE SPECIFIED OR A SYNTAX ERROR WILL OCCUR!


      3240  *13240            /ADD OVER "ZBLOCK 4"

13240 3121            XFUTIL  /POINT TO ABOVE ENTRY


      3571  *13571

13571 7351            SEMSG-1 /NEW POINTER


      4003  *14003

14003 1211            TAD ISKP

      4127  *14127            /OVERWRITE LOVE, FILDMY & SEMSG

14127 0000  FUTSUB,   0
14130 1747            TAD I ARLOCI     /GET SAVED POINTER
14131 3347            DCA ARLOCI
14132 6201            CDF 0
14133 3747            DCA I ARLOCI     / & ZAP BACKARROW
14134 6211            CDF 10
14135 4744            JMS I REMEMI     /REMEMBER COMMAND LINE
14136 0006              6
14137 1350            TAD SDEV1
14140 3745            DCA I A7606      /MOVE DEVICE NAME TO C.D.
14141 1351            TAD SDEV2
14142 3746            DCA I A7607      / SAVE AREA.
14143 5727            JMP I FUTSUB
14144 7000  REMEMI,   REMEM
14145 7606  A7606,    7606
14146 7607  A7607,    7607
14147 5741  ARLOCI,   ARLOC
14150 0000  SDEV1,    0
14151 0000  SDEV2,    0
```

```
/FUTCCL.PA - CCL V1F MODS                 PAL8-V10A 07-APR-78 PAGE 4

14152  3351  FILLP2, DCA SDEV2       /SAVE DEVICE NAME
14153  1023          TAD NAME1       /FOR "FUTIL ..."
14154  3350          DCA SDEV1
14155  1024          TAD NAME2       /CONTINUE AS
14156  5776          JMP I FIL1PI    /BEFORE PATCHED.

       4176  *14176

14176  7321  FIL1PI, FILLP1  /WAS (SKP


       5354  *15354          /WRITE IN SPACE

15354  0425  FILDMY, FILENAME DUMMY.SV       /MOVED
15355  1515
15356  3100
15357  2326


       5413  *15413

15413  7000          NOP     /WAS JMS I LOVE


       7322  *17322          /PATCH 'GETSPC'

17322  4655  FILLP,  JMS I K4200     /PTR TO GNAME

       7342  *17342

17342  5761          JMP I FIL2PI    /AC= C(NAME2)

       7352  *17352          /WRITE IN SPACE
17352  6666  SEMSG,  SEMSG1  /MOVED
17353  6675          SEMSG2
17354  6703          SEMSG3
17355  6712          SEMSG4

       7361  *17361

17361  4152  FIL2PI, FILLP2  /WAS (GNAME

/FUTCCL.PA - CCL V1F MODS                 PAL8-V10A 07-APR-78 PAGE 5

       0000          FIELD 0

       0572  *572    /ADD KEYWORD TO COMMAND TABLE OVER "ZBLOCK 600-."

00572  0665          0665    /F UTIL ["F" IS ONLY REQUIRED CHAR]
00573  6451          6451
00574  5400          5400


       6436  *6436           /OVERWRITE LOVMES

06436  0625  YFUTIL, FILENAME FUTIL.SV
06437  2411
06440  1400
06441  2326

                     $=$=$=$=$        /THE END
```

National Research Council Canada | Conseil national de recherches Canada

Division of Physics | Division de physique

100 Sussex Drive, Ottawa

File Référence

March 10, 1978

Dr. R. Hassinger
Liberty Mutual Research Center
71 Frankland Road
Hopkinton, Ma. 01748
U.S.A.

Dear Dr. Hassinger:

I have had a number of enquiries about my DECSYK program (Decus 8-872) which you described on p. 9 of newsletter #26. I wrote it quickly to get V3D on floppies, so it wasn't exactly tidy or sufficiently general to be of immediate use to all "unbuffered Sykes" users. The enclosed update represents quite an improvement, I think. Also, I don't remember if I sent you a note about BATCH spooling, so I'll enclose a copy.

Yours sincerely,

Ian Templeton

I. M. Templeton

BATCH SPOOLING - NOTE FROM IAN TEMPLETON

THOSE OF YOU WHO WERE FORTUNATE ENOUGH TO RECEIVE 'SOFTWARE NEWS', JAN/78, MAY HAVE SEEN THE S.P.R. CONTAINING A CRY FOR HELP WITH BATCH SPOOLING. MY OWN OBSERVATION IS THAT THIS IS VERY POORLY DOCUMENTED. BATCH RECOGNISES A NON-FILE-STRUCTURED DEVICE FOR SPOOLING ONLY IF IT IS SPECIFIED TO THE COMMAND DECODER AS AN OUTPUT DEVICE EITHER BY DEV:< OR VIA A CCL SWITCH. THUS, WITHIN A SPOOLED BATCH JOB, .DIR WILL NOT SPOOL BUT .DIR-T AND .DIR TTY:< WILL. IN FRTS, BATCH WILL NOT RECOGNISE THE INTERNAL TTY OR LPT HANDLERS, NOR WILL *LPT:/3 SPOOL; HOWEVER, *LPT:</3 WILL! IT DOES NOT APPEAR TO BE POSSIBLE TO SPOOL OUTPUT FROM FORTRAN II OR BASIC.

DECSYK

PDP8/E PROGRAMS TO READ AND WRITE DEC FORMAT DISKETTES VIA SYKES 7100/7200

I. M. TEMPLETON
DIVISION OF PHYSICS
NATIONAL RESEARCH COUNCIL OF CANADA
OTTAWA, CANADA K1A OR6

TEL. (613) 992-2113

TWO PROGRAMS, DTODEV & DEVTOD, ARE PROVIDED. THEY MAY BE CONDITIONALLY ASSEMBLED EITHER TO ALLOW RUN-TIME CHOICE OR TO ASSIGN A PERMANENT SYKES UNIT NUMBER FOR THE DEC DISK AND A PERMANENT COPY DEVICE NAME. THE COPY DEVICE MAY BE ANY FILE STRUCTURED OS/8 DEVICE WITH AT LEAST 494 (DEC.) BLOCKS.

THESE PROGRAMS ARE AVAILABLE DIRECT FROM THE AUTHOR ON STANDARD SYKES (12-BIT) FORMAT FLOPPY DISK.

DECSYK

THE DATA ON A DEC DISKETTE IS WRITTEN IN 3/4 FILLED, INTERLEAVED SECTORS. IT APPEARS TO BE IMPOSSIBLE TO UNSCRAMBLE THESE VIA AN OS/8 HANDLER WITHIN THE TIMING RESTRICTIONS OF THE UNBUFFERED SYKES SYSTEM, SO THE PRESENT APPROACH HAS BEEN USED. THE 'READ' PROGRAM READS SECTORS SEQUENTIALLY, TWO TRACKS AT A TIME, AND DEPOSITS THE DATA IN THE CORRECT (INTERLEAVED) PLACES IN A 6400-WORD BUFFER (IN FIELD 3 AS WRITTEN). THIS BUFFER IS THEN COPIED, VIA A STANDARD OS/8 HANDLER, TO AN ALTERNATE FILE STRUCTURED DEVICE. THE 'WRITE' PROGRAM DOES THE OPPOSITE, FIRST FILLING THE BUFFER FROM THE SOURCE DEVICE, AND THEN COPYING THAT TO A BLANK DISKETTE IN SUCH A WAY AS TO REPRODUCE THE CORRECT INTERLEAVING WHILE WRITING SEQUENTIALLY.

BECAUSE THE DEC DISKETTE CONTAINS 494 BLOCKS = 76 TRACKS (WITH TRACK 0 UNUSED) THE FINAL 'READ' OR 'WRITE' OPERATION WILL LEAVF THE SYKES HEAD AUTOMATICALLY STEPPED TO THE (NON-EXISTENT) TRACK 77 POSITION. THE RESULTING ERROR IS IGNORED, AND THE HEAD IS STEPPED BACK BY ONE TRACK TO AVOID SUBSEQUENT HANGUP.

THE UNIT NUMBER FOR THE DEC DISK, AND THE COPY DEVICE NAME, MAY BE CHOSEN AT RUN-TIME, OR, BY SETTING THE CONDITIONAL EXPRESSIONS UNIT, DEVA & DEVB TO THE REQUIRED UNIT NUMBER AND (IN 2 PARTS) 6-BIT DEVICE NAME RESPECTIVELY, THE PROGRAMS MAY BE ASSEMBLED WITH PERMANENT ASSIGNMENTS WHICH ARE AUTOMATICALLY DISPLAYED ON THE TERMINAL AT RUN TIME.

BOTH 'READ' (DTODEV) AND 'WRITE' (DEVTOD) PROGRAMS MAKE EXTENSIVE USE OF THE MQ REGISTER: IT IS NOT CERTAIN THAT ALTERNATIVE (SLOWER) OPERATIONS WILL NECESSARILY MEET THE SYKES TIMIMG CRITERIA.

```
        .COMP DTODEV
        .LO DTODEV
        .SA SYS DTODEV 2000-2777;2000=6003
```

$$$

VERUS INSTRUMENTS INC. • BOX 122 • FORT ERIE, ONTARIO • (416) 871-0733

March 27 1978

Dear Bob,

I've recently discovered a peculiarity in FOTP when used with non-file structured handlers. I also learned a lesson about 2 page handlers and thought I should pass them on to the 12 bit SIG.

I have never been able to use FOTP to transfer ASCII files to my lineprinter. Running FOTP directly or via the CCL .LIST command would result in a halt on page zero. I discovered FOTP attempts to close all output files even on non-file structured devices. This is harmless in most cases but does not follow the "initialize convention" for non-file structured handlers. Some non-file handlers expect to be called with a zero block number the first time to set flags, do form feeds, rewinds or other once only initialization. FOTP's first call to the handler is with a non-zero block number so the initialization code is skipped. If you use the initialize code to setup page links in a 2 page handler like I did, your handler might use a zero page link to bomb on page zero.

I modified the handler to setup the page links on every entry and the problem disappeared.

Ian Templeton and I found an undocumented feature in SET. Curious users may try .SET TTY GAG at their own risk. (Don't worry it's harmless)

I've been working on a few things of potential interest to 12 Bit SIG readers.

I recently put a buffered serial link into COS-300. (the DEC commercial operating system) It allows data files to be transmitted to a COS system at high speed via a standard RS-232 link.

I've also completed work on an interupt driven graphics package for the KV8E or KV8I graphics system for use with Fortran IV. It was adapted from a non-interupt version of unknown origin. Points, vectors and characters are supported. The interupt support made a dramatic increase in speed even with buffers as small as 30 locations. The package may also be a usefull example of Fortran IV interupt support for those attempting similar projects.

I've have also written a set of Fortran IV routines to plot points on a VT8E terminal in graphics mode. They are still in very preliminary form and use one EAE Mode A divide instruction. This could be easily bypassed for use on non-EAE machines.

Has anyone considered adding double precision support to FRTS for systems without an FPP ? If the Lineprinter, HSR, HSP code was purged there might be enough room if field 0. If not then maybe a RALF FIELD1 section would do it.

I have also included some patches to make the Command Decoder and Keyboard Monitor accept commands in lower case characters. Lower case characters are folded to upper case and echoed as upper case. The patch overlays portions of the "backslash rubout" and "linefeed playback" features of the monitor and should only be used after .SET TTY SCOPE has been specified to the monitor. These simple patches should very handy for terminals without a caps lock key or for those people like me who forget to use it until the monitor has rejected a lower case command.

This letter was prepared using Dewar Information Systems document generator ACID. I have made extensive modifications to it to support the features of a Diablo word processing printer. Justification is done in .008" increments and line feed size is adjustable in .020" increments. Subscript, superscript, backspace, overstrike, boldface and plotting capabilities have been added.

Anyone interested in any of the above may reach me at (416) 871-0733.

Sincerely yours,

John Youngquist

John Youngquist

The following patch will allow the Keyboard Monitor to accept lower case commands.

```
.R FUTIL

11.233/ 1344    1207                      /IGNORE LINEFEEDS

11.263/ 2020    1034    TAD NM1           /GET THE CHARACTER
11.264/ 5267    5344    JMP LFEED         /GO HANDLE THE CHARACTER

11.344/ 4561    1354    TAD (-340         /TEST FOR LOWER CASE
11.345/ 3413    7500    SMA               /SKIP IF UPPER CASE
11.346/ 1163    1353    TAD (-40          /CONVERT TO UPPER CASE
11.347/ 4423    1352    TAD (340          /RESTORE CHARACTER
11.350/ 1160    3034    DCA NM1           /SAVE CONVERTED CHARACTER
11.351/ 3017    5270    JMP PRNT+6        /NOW ECHO IN UPPER CASE
11.352/ 1417    340     340               /CONSTANTS
11.353/ 7450    7740    -40
11.354/ 5324    7440    -340

WR
```

The following patch will allow the Command Decoder to accept lower case commands.

```
53.050/ 5366    5207                      /IGNORE LINEFEEDS

53.156/ 2024    1025    TAD NAME1         /GET THE CHARACTER
53.157/ 5362    1220    TAD (-340         /TEST FOR LOWER CASE
53.160/ 1101    7500    SMA               /SKIP IF UPPER CASE
53.161/ 4466    1367    TAD (-40          /CONVERT TO UPPER CASE
53.162/ 3024    1370    TAD (340          /RESTORE CHARACTER
53.163/ 1025    3025    DCA NAME1         /SAVE CONVERTED CHARACTER
53.164/ 4466    1025    TAD NAME1         /GET THE CHARACTER
53.165/ 5755    4466    JMS I PRINT       /NOW ECHO CONVERTED CHARACTER
53.166/ 4312    5755    JMP I PRNT        /EXIT PRNT ROUTINE
53.167/ 3415    7740    -40               /CONSTANTS
53.170/ 1100    340     340

WR
```

THE PENNSYLVANIA STATE UNIVERSITY
417 BRUCE V. MOORE BUILDING
UNIVERSITY PARK, PENNSYLVANIA 16802

College of the Liberal Arts
Department of Psychology

Room 535

Area Code 814
865-1725

April 4, 1978

Robert Hassinger - Coordinator, 12-bit SIG
Liberty Mutual Research Center
71 Frankland Road
Hopkinton, Mass. 01748

Dear Bob,

I am writing to indicate that we have support available for the KL8-A multiplexor under OS/8 in the form of a 2-page (single-device-at-a-time) handler, and as a modification to UTILTY.SB in FORTRAN II. The system device handler does not have all of the features of the KL8E.PA handler, but FORTRAN II works quite well.

I would also like to take this opportunity to urge that people with programs of general utility (i.e. compilers, file mgmt., etc.) that run under RTS-8 submit them to DECUS. I have been working on (attempting) a version of PIP to run under RTS-8, but have no interest in reinventing the wheel.

Sincerely,

Alex W. Frediani

THE
GEORGE
WASHINGTON
UNIVERSITY
MEDICAL CENTER

676-2692

*Office of Computer Assisted Education / 2300 Eye Street, N.W. / Washington, D.C. 20037 / (202)*xxxxxxx

April 14, 1978

Robert Hassinger, Coordinator
12 Bit Sig
Liberty Mutual Research Center
71 Franklin Road
Hopkinton, MA 01748

RE.: Switched relay control of KL8E baud rate

Dear Mr. Hassinger:

We have recently installed Magnecraft relays (W118DIP-2) on a number of our KL8E boards which allow us to control the baud rate of the KL8E remotely at our terminal sites without stringing any additional cabling or having to power down our system to change on-board jumpers.

The heart of the modification is the fact that we take advantage of an unused line in the standard DEC RS232 KL8E cable and the current loop mode power source to operate the relay. We mount the relay, which comes in an IC-14-PIN DIP-size package, on the top of the KL8E board. Power for the relay (it draws 31.25 milliamps when powered) comes from the KL8E from an on-board circuit designed to control relays in a teletype; thus while our power drain is negligible even in our heavily loaded system, we take the power from a circuit designed to handle the drain of off-board power-hungry relays.

The relay is mounted in a standard IC-14 wire wrap pin socket; the center three pins on each side are not used. The socket is mounted on the top of the KL8E, above the baud rate jumpers, so that the socket leads come down on each side of the KL8E. A small nut and bolt passed through the center three socket pins and one of the eyelets of the plastic board grips holds the assembly in place. The assembly does not protrude beyond the grips either vertically or horizontally.

The relay socket is wired as followed:

Lead	Connection on KL8E	Purpose
1	baud rate jumper 5	common
2	coil +5 volts	emitter side of transistor Q3*
3	coil reference	BERG connector pin R (CINCH connector pin 17)**
4	baud rate jumper 6	normally open (300 baud)
5	--	not used
6	--	not used
7	--	not used
8	baud rate jumper 2	normally closed (1200 baud)

*This lead changes sides of the board through an unused hole which is a convenient place to tap the +5 volts.

**This lead is unconnected on the KL8E, however the standard DEC KL8E cable includes a lead BERG(R) to CINCH(17).

The remote control is accomplished with a single pole double throw switch which shorts pin 7 (protective ground) with pin 17 on the KL8E cable. We had an unused lead in our ceiling cables, so we made a connector cable with an in-line switch (this could also serve as a null modem cable) which is between the KL8E/ceiling cable and the terminal. Naturally, the terminal side does not have pin 17 wired.

We are currently using these KL8E's with CRT's which run at 1200 baud. When we attach a 300 baud slave printer for local copy we simply flip the switch and the CRT to 300 baud. For our text processing work (using RUNOFF and producing output on Diablo 1620-3's) it is invaluable.

Sincerely,

Roy A. Standing
Programmer

RAS:syy

WESTERN RESERVE ACADEMY
HUDSON, OHIO 44236

20 March 1978

Mr. Robert Hassinger, Coordinator
12 Bit SIG
Liberty Mutual Research Center
71 Frankland Road
Hopkinton, MA 01748

Dear Mr. Hassinger:

I believe that by far the best feature of the PDP-8 -- and, indeed, the only one that really makes it preferable to a microcomputer -- is that there is a tremendous amount of exciting software available for it from DEC, DECUS, and other sources. Unfortunately, I have found that it is very difficult to get information about much of it. You would think people were trying to hide their software products!

I discovored ETOS, a system which the Academy is now considering purchasing, quite by accident at a conference of math teachers. The existence of that most elusive program, MULTI-8, was revealed to me by the title of a paper listed in the catalog of DECUS proceedings I was sent when I joined. From the preliminary program for the Spring DECUS Mini/Midi Symposium I deduce the existence of a SNOBOL 8.2 (presumably not the same thing as DECUS' POLY SNOBOL) and something called SLANG. I have heard of a FORTRAN called FIV, again presumably not the same as OS/8 F4. An OS/8 COBOL is rumored. Just two days ago I discovered, in my 1973 Small Computer Handbook, of all places, the existence of PDP-8 TRAC and SAIBOL. (Incidentally, a 1966 article in CACM mentions a TRAC for the 5. I wrote to the author, who has a trademark on the word TRAC and thus control over the language, but he seemed to think that there wasn't and never was such a thing.) I could go on and on but I think you can see my point.

Even DEC maintains an aura of mystery around their software. Their software catalog does not tell one what programs do or cost, only their order codes. A wide variety of fascinating-sounding systems is listed -- PAMILA and TSAR/INA, for example. (I gather that OS/8 has been bundled, which is rather unfortunate for us, as we are TECO- and BATCHless, and stuck with V3C, inoperable date-wise after 1977.)

My ignorance may be due in part to the fact that I do not go to conferences and/or due to the minimally-supported nature of our installation. However, I imagine that there are many other users (especially of the 6100) who are in the same boat. What I suggest is that the 12-Bit SIG, or DECUS, or someone, publish an exhaustive list of all the PDP-5/8 software that has ever been written, excepting programs in the DECUS library, which is well-documented. For each entry, the list might give a short abstract like those in the DECUS catalog, with minimal hardware and software needed to use the program, and how much it cost from whom, if it were available. Obsolete software should not be ommitted, and it's worth tracking down a lot of the stuff that's floating around in the public domain but not in the DECUS library. (I got a copy of CHEKMO-II, for example, long before the program was submitted to DECUS; I gather that it was orignally written by DEC as a whiz-bang demo.)

I think that many users might even be willing to pay real money ($$$) for such a publication; and ad space might be sold to the likes of Educomp, who would probably be glad to advertise ETOS, etc. It could probably be self-supporting, and would be a boon to all concerned -- DECUS, DEC, other software houses, and most of all the user

Yours Sincerely,
David Chapman
David Chapman

P.S. You don't have to answer this letter -- just do something about it. You might print it in the SIG newsletter, though.

UNITED STATES
DEPARTMENT OF THE INTERIOR
GEOLOGICAL SURVEY

Branch of Isotope Geology (MS 18)
345 Middlefield Road
Menlo Park, California 94025

March 21, 1978

Mr. Robert Hassinger, Coordinator
12-Bit SIG
Liberty Mutual Research Center
71 Frankland Road
Hopkinton, MA 01748

Dear Bob:

I am interested in the possibility of using one of the new small and inexpensive "smart" plotters on one of our PDP8/E or A systems. The one that looks particularly attractive at the moment is H-P's 7221A four-color Graphics Plotter. I have read over the 7221A programming manual and it looks like it should be quite easy to drive it with either OS/8 BASIC or FORTRAN IV without writing any assembly language routines, but I would like to get in touch with someone who has actually tried this. If there is anyone out there in PDP8-land who can help, I would appreciate hearing from them.

We have several virtually identical PDP8 systems that we use to operate and take data from our mass spectrometers, and to reduce the data. Our primary source language is OS/8 BASIC. I chose BASIC primarily because I could write the assembly language routines we required in PAL8 and interface them with BASIC very easily. I found RALF to be a bit confusing for the occasional programmer like me. Because we use BASIC for nearly everything else, I am interested in using BASIC for plotting just to keep everything simple. I am also not interested in writing any long assembly language plot routines because it is just not economical for me to do that unless absolutely necessary.

Incidentally, this letter was produced on a new WS/78 word processor. It is a well-thought-out and very impressive system.

Sincerely,

G Brent Dalrymple

G. Brent Dalrymple
Geologist

14 Doncaster Road
Cherry Hill, NJ 08003
(609) 424-2765
April 20, 1978

Dear Mr. Hassinger,

First, let me compliment you on the consistently superb newsletter. It is a regular fountain of 12 bit thoughts, tips, and arcana.

I'm a senior at Cherry Hill High School East. Our connection to the 12 bit world is our PDP-8/L with 4K words and ASR-33. We run CHEDU-10 BASIC (a modified version of EDU-10 imported from Cold Spring Harbor, Long Island), FOCAL 5-69, and UCONN EAP.

The main reason I'm writing is that two other seniors (Jim Hennessy and Nelson Ronkin) and I have written a FOCAL for the IBM 370. We decided to write FOCAL-78 because we felt it would be interesting to throw a neat, efficient language on a powerful machine. It is based on Jim van Zee's excellent U/W-FOCAL, however, it was written from the ground up (or top down, if you're a Structured Programmer). FOCAL-78 contains a few neat features which would be impossible on a PDP-8.

Arrays may be of any dimension. We recognize, of course, that only about 63 subscripts fit on a 132 character line, and that over 6 or 10 subs are purely academic, but fun is fun. No secret variables are used- the arrays are truly multidimensional.

Names have eight significant characters. All names are hashed, and arrays take up only one hash table entry.

Because single character I/O is impossible on the 370, we have dropped the standard editing commands and written a line oriented editor facility into the direct command section.

Character strings (char arrays are coming soon) are available, with lengths from 0-255 chars. This eliminates the FIN/FOUT business.

FSFs are called by name, and have two syntaxes. A FORTRAN function style:

```
     FRUP(N)=FITR(N+0.5)
```

and a PL/I procedure type:

```
     FACT(NUM); SET F=1; FOR I=2,NUM; SET F=F*I; NI; RET
```

They are both called like normal built in functions:

```
     SET Q=FACT(FSQT(Q)); TYPE Q,FACT(FRUP(Z)),!
```

Localization of variables is available in FOCAL-78. Procedure-type FSFs will automatically nest variables one generation deeper upon being called. This provides true subroutine and recursion capabilities. For manual localization of names, we have added two commands, LOCAL and LEVEL (the traditional L commands have been eliminated, as well as the O commands). LEVEL takes a numeric argument, and moves the nesting level more local (positive integer), more global (neg int), or returns to the most global level (zero). LOCAL has a character argument containing the names of variables to localize at the most current level. Reference to a name uses the most local value at which the name is defined; assignment is to the most local definition or to the present level if the name is completely undefined. This allows easy garbage collection for negative LEVELing.

A new command pair, WHILE and END, adds two handy capabilities. WHILE with no arguments causes FOCAL-78 to ignore all subsequent end-of-lines until the corresponding END is hit. This is most useful in extending FOR loops. WHILE followed by a condition (WHILE X<3) loops to the corresponding END whenever the condition is true. This allows easy iterative loops or tests.

The math expression evaluator accepts relational and logical operators: < > = >= <= ~ <> & ¦ @ (last two are "or" and "exclusive or").

The VARIABLE command declares attributes of variables. EX:

```
VARIABLE A(3,5,8)=FLOAT,NAME=CHARACTER
```

Those are the major modifications for FOCAL-78. Future plans call for file access, probably as a declared correspondence between a variable name and a file name (like in IBM's APL.SV). We realize, of course, that many of our modifications cause FOCAL-78 to be rather non-standard, to say the least! We hope that they will be seen as sincere attempts to add to the language, rather than just another wierd mutation of FOCAL.

We would like to thank Jim van Zee for his invaluable help and support.

On the lighter side, I enclose a tape of a rather amusing PDP-8 assembler program. I wrote it at the end of my sophomore year, and it's therefore rather primitive. It clears core and then erases itself. It starts at 0001, and takes about 30 seconds. It may not be very useful, but the concept of a self erasing program is interesting. PDP-8 hackers might like to try writing their own before looking at this one.

This letter was printed through SCRIPT/370, IBM's version of RUNOFF. It has one feature which amuses me: you can specify the use of roman numerals for page numbers!

Best wishes,

Jon von Zelowitz

Jon von Zelowitz

```
/*********************VACUUM*******************
/SELF-ERASING CORE ZEROER
/V 2.0, 6/29/76
/WRITTEN BY JON VON ZELOWITZ
/+++++++START AT 0001+++++++
*7777
        JMS I 12        /LOOP TO GET RID OF 13-23
*0000
        DCA I 12        /LAST INSTR:CLEARS SELF&12
        CLA CLL         /STARTING LOCATION
        DCA I 11
        TAD K2          /HERE, WE AVOID DELETION
        TAD 11          /OF LOCATION 7777.
        SZA CLA         /SKIP IF 7776 CLEARED
        JMP 1           /ELSE CLEAR NEXT LOC.
        JMP 13          /GO TO 13 IF 7776 DONE
K2,     2
        23              /THESE ARE LOCS FOR
        12              /THE DEPOSITS TO START.
        DCA 1
        DCA 2           /THESE DCAS CLEAR OUT THE
        DCA 3           /LOCATIONS 1-11:
        DCA 4           /THE PART WHICH
        DCA 5           /CLEARS MOST OF
        DCA 6           /CORE INITIALLY.
        DCA 7
        DCA 10
        DCA 11

/THE NEXT INSTRUCTION WILL PROVIDE AN AUTOMATIC
/START FROM RIM LOADER.
*7757
        JMP 1           /JUMP TO THE START OF VACUUM

$
```

DEPARTMENT OF THE ARMY
WALTER REED ARMY INSTITUTE OF RESEARCH
WALTER REED ARMY MEDICAL CENTER
WASHINGTON, D.C. 20012

REPLY TO
ATTENTION OF:

SGRD-UWI

1 May 1978

NETWORK Special Interest Group
c/o DECUS Office
129 Parker Street, PK3-1/E55
Maynard, Mass. 01754

Dear Sirs:

I am a DECUS member (DD05911) with need of input and output device handlers for the DB8-EA interprocessor buffer to be used with two PDP 8/e's under OS8. As a special interest group specializing in networks, one of your members may be able to supply this software. Any suggestions will be most appreciated.

Sincerely yours,

Steve R. Hursh

STEVEN R. HURSH, Ph.D.
Chief, Physiology & Behavior
Branch
Department of Medical Neurosciences
Division of Neuropsychiatry

SRH/sd

MOVING OR REPLACING A DELEGATE?

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

( ) Change of Address
( ) Delegate Replacement

DECUS Membership No.: ____________________

Name: ____________________

Company: ____________________

Address: ____________________

____

State/Country: ____________________

Zip/Postal Code: ____________________

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

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

DECUS

DIGITAL EQUIPMENT COMPUTER USERS SOCIETY
129 PARKER STREET, PK3/E55
MAYNARD, MASSACHUSETTS 01754