DECUS 12 BIT SPECIAL INTEREST GROUP NEWSLETTER

September Number 30 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 27-October-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 U.S. 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. Ellis, Jr.
USCG R & D Center
Avery Pt.
Groton, CT 06340
(203) 445-8501 Ext. 296
(FTS) 642-7274 Ext. 296

LINCtapes

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

ENGINEERING SPECIAL INTEREST GROUP

Walter V. Dixon wrote to say that he is putting together the first issue of a newsletter for the Engineering SIG. He is interested in identifying the interests of the engineering user community and how his newsletter can best serve them. He invites comments and contributions. His address is Mechanical Technology Inc., 968 Albany-Shaker Road, Latham, New York, 12110 - phone: (518) 785-2211.

DECUS 12 BIT SPECIAL INTEREST GROUP NEWSLETTER
Number 30 - September 1978

FALL DECUS/US SYMPOSIUM

The preliminary schedule for the Fall Symposium shows about 16 hours of 12 Bit related sessions. Details on the sessions were not available at press time but the following session titles are listed:

- 12-Bit SIG
  - Road Map
  - Meeting
  - Short Notes Session
- DECSTATION 78/88 Software Workshop
- 12-Bit Microprocessor
  - Applications
  - Hardware Paper
- PDP-8 Programming Tools Workshop
- RTS-8 Papers
- PDP-8 Educational Instructions Panel
- PDP-8 Product Panel
- OS/8 Papers
- 12-Bit Wrap Up

The TECO Tutorial should also be of interest to 12-Bit users

More detailed information should be in the mail soon from DECUS/US to U.S. members and others who request it. See you there.

OS/78 VERSION 2

I recently ordered and received the OS/78 V2 update kit. When ordered as an update for those who have OS/78 V1, it cost something like $20. The SPD says something about an upgrade kit for OS/8 V3D owners but I do not have any details or costs. The main reason an OS/8 user might want the OS/78 V2 kit (available only on floppy disks of course) is to get access to the new version of BASIC. This is a major upgrade of the existing OS/8 BASIC. The official description of the features are in the SPD and the manuals. You should check them for full details, but here are some of the features I have found interesting.

> The VT-52 scope is supported much better. The proper rubout sequences are used and Control S and Control Q are supported as well as the SET TTY PAUSE feature for controlling output on fast CRTs.
>
> The BASIC editor, and programs written in BASIC can both handle the full seven bit upper and lower case ASCII character set rather than the old 6 bit, 64 character, upper case only set.
>
> The RS command is the same as the RUN command, except that before the program starts, a report is printed to show how much free space is left. This allows you to evaluate the size of your programs better than before. This same feature is available with the CCL EXECUTE command as the /S option.

LIST, LISTNH and DELETE now work on selected ranges of line numbers.

There is an EDIT command that will search for a character string on a line and replace with a new one. This removes the need to completely retype every line with an error. It also allows copying lines to new line numbers without having to retype them.

The SEQUENCE command sets a starting line number and a line number increment. In this mode the system automatically supplies the line numbers as you enter the program.

The WEAVE command will read a program in over one already in core (i.e. an OLD that does not clear the previous program). This simplifies combining sections of code in separate files.

The old hacks involved in numeric input (i.e. the need for inputting dummy variables at the end of lines, etc.) are gone. It looks as though BASIC finally works the way it should.

ON-GOTO and ON-GOSUB have been added to transfer control based on the value of an expresion.

A new IF OPEN # statement has been added. It is now possable to tell if the open of a file was succesful and to avoid a program abort. Unfortunatly, there is still no way to avoid a warning message that is typed when a file open fails, however.

A form of PRINT USING has been added. It allows considerable control over output formats. It is not as general as some more advanced versions such as in the PDP-11 BASIC-PLUS but it gives the controls needed for doing business reports and checks and so on.

The TAN and ATN functions have been added.

The CAP$ function converts lower case characters to upper case. This is handy because the comparison operations on strings do not ignore case.

The CCL command allows a program to exit and pass a command to CCL for execution. This hook has many interesting possibilities and reduces the need to run under batch in some kinds of applications, thus saving space.

The OCT function returns the decimal value of a string of octal digits.

The OCS$ function returns an 8 digit string with the octal value of of the variable argument.

The PNT function and the documentation allow control of the special features of the VT-52 terminal and the CUR$ function automatically sends the correct sequence of controls to do direct cursor positioning on the VT-52.

The AND and IOR do the bitwise logical AND and Inclusive OR operations on the binary representations of decimal numbers.

The KEY$ function automatically inputs one character from the keyboard. If an escape sequence is received this function handles it.

The PMT$ function allows changing BASIC's standard "?" prompt character to any 0 through 7 character string.

Commercial arithmetic (i.e. greater than 6 digit precision) is supported via string arithmetic. You can do the four basic operations on values stored as digits in strings. The values can be integers and/or fractions and up to 15 digits long.

DIRECT RECORD I/O is provided for storing, retrieving, and updating individual records in mass storage files. This feature is similar to the FORTRAN direct access mode but it is much better. Data records are of fixed length and are stored in the file in standard OS/8 ASCII character format, complete with a carrage return and line feed at the end of each record and a control-z at the end of the file so it can be accessed with standard software. Even better is the fact that the records are fully packed, and they automatically span block boundaries. This means that there is no wasted space as in OS/8 FORTRAN IV where at most one record per block is recorded.

New, improved facilities are provided for compiling and saving programs so they may be run without recompiling every time. On the relatively slow DECstation 78 this is very valuable. Unfortunatly, there seems to be a bug that cause problems if your program depends on information that loads in the top page of field 2. For example, the executable code could be larger than one field (it loads from the top down), or you might depend on data storage being initialized to zero (it loads down from the executable code). The bug has to do with the handling of one versus two page system handlers at compile and at run times. DEC knows about this bug already but no fix has been published yet and it is not documented in the release materials.

The new release includes a "Multifunction Operation" feature. This is the "Symbiont" mentioned in a previous newsletter. You have the normal OS/78 operation in the first three fields and at the same time a second task that runs on interrupts can be in field 3. The symbiont task supplied with V2 is a print spooler. You have new monitor commands to start and stop the symbiont, to pass the spooler task a list of files to be listed on the printer, and to check on the status of the spooler. If you want, you are permitted to write your own symbiont task.

The following is the full explanation of what rules a user written program must observe to allow it to coexist with a symbiont.

1) If the program loads into page zero of field zero, then it must contain the following code:

   ```
   FIELD 0
   *1
   CIF 30
   JMP .-1
   ```

   Also, locations 0,1 and 2 must not be used as scratch or as data.
2) If the program does not load into or use page zero of field zero, then no modifications are necessary.
3) The program must not use page zero of field zero as a buffer or data area. However, you may swap the OS/78 command decoder in this area.
4) The program must never turn interrupts on or off.
5) The program should not modify the software core size.
6) The program must not require the use of field 3.

In general the Symbiont feature will only work under OS/78 V2 on a DECstation-like configuration. This is because no Data Break devices may be used, you must be able to disable interrupts from all devices, in particular the console terminal that comes up enabled and which can not be disabled on the older PDP-8 family machines, and because you must have the new OS/78 versions of all the system programs that have been modified to observe the above conventions.

OS/78 V2 also includes support for multicharacter switchs in CCL commands. For example you can say "DIR /BRIEF" rather than "DIR /F". This is a rather nice feature. The multicharacter switches are often easier to remember and teach. The ":" character can also be used in place of "=" for setting numeric values after multicharacter switches. For example "/I=24" can be expresed as "/IMAGE:24". Users of other DEC systems that support DCL rather than CCL will recognize these forms. Since DCL is DEC's new standard command language that is replacing CCL in most systems, as much compatibility as possible is desirable.

Unfortunatly, in OS/78 DEC is not providing the source of CCL so you can not change or extend the list of multicharacter switches and commands. I think this is a serious problem and I hope it does not propagate to the next release of OS/8 if and when it ever comes. Although it was the intention of the new CCL code to pass through all single character switches just as always, there seems to be a bug, at least in the DIRECTORY command. If one tries to pass the /A switch, it does not get through but seems to be mapped to some other, strange bit. I hope this gets fixed so we can use the Alphabetize feature of the user enhanced version of DIRECT.

NOTE FROM JIM VAN ZEE

Jim writes to say that he and Carl Appelof have just managed to transfer the source files for ADVENTURE (DECUS 11-340) from an RT-11 floppy disk to an OS/8 floppy disk. He notes that so far the program will not run (it needs a lot of modifying to fit and run under OS/8 FORTRAN IV) but the system used to do the transfer might be interesting. Jim writes:

"The basic ingedients are an assembly language program which Carl wrote for the 11 and the byte-mode handler developed by Steuart Dewar for the 8 which was described in the last Newsletter (#29, p15). Dewar's handler uses a very straight-forward sector interleaving scheme (as opposed to that used by Lynch's handler, for example) which makes it relatively simple to read/write on the 11. We have made no attempt to create a directory, so to recover the files we just used U/W-FOCAL to copy them since this program allows file input from specific block numbers. One could also just write a rather simple program to do the same thing, or better still, just use something like PIP or FUTIL (or even UWF !) to create the directory after-the-fact, providing one left room for it at the beginning. ..."

"I don't know if this is useful to anyone else or not, or if (as seems likely) someone else has already developed a better way to move files between 11's and 8's. Anyone interested (or better informed!) can write me c/o Dept. of Chemistry, University of Washington, Seattle, WA 98195. Anyone who wants a copy of ADVENTURE on an OS/8 medium is welcome to a copy, but please include ample return postage."

Having gotten sources to both the RT-11 and PDP-10 versions of ADVENTURE via other means, and having looked at the conversion to OS/8 FORTRAN IV, I can report that it is going to be quite a job and need most of 32k. I have run the game a good bit on an RT-11 system and it is great fun. The conversion would be worthwhile for people with a big 8 and no access to an 11 who like this sort of thing. It is far better and more complex than STARTREK for example.

On the subject of moving data files between 8's and 11's, I may have mentioned in the Newsletter that a year or two ago I managed to develop a means for moving files from the 8 to the 11 on an RK05 disk (note: the hardware makes that imposable but I did it anyway). The disk is organized as an OS/8 file structured device and I have programs written in RT-11 FORTRAN to access the files via the directory. The only reason I do not go from the 11 back to the 8 is that I never got around to writting the FORTRAN routine to create files in the OS/8 directory. If ever there is an overwelming demand, I might do a Newsletter article on how all this is done and/or resurrect the code and make it available (I don't know if a bug free version of the sources still exists, it might have to be worked up again from backup versions). (RH)

DECSYSTEM 8 ENHANCEMENTS

Lyle P. Bickley sent a note to say that he has been working with Don Harmer on enhancements to the version of CCL that goes with DECsystem 8 for OS/8 V3D. He has added the following features:

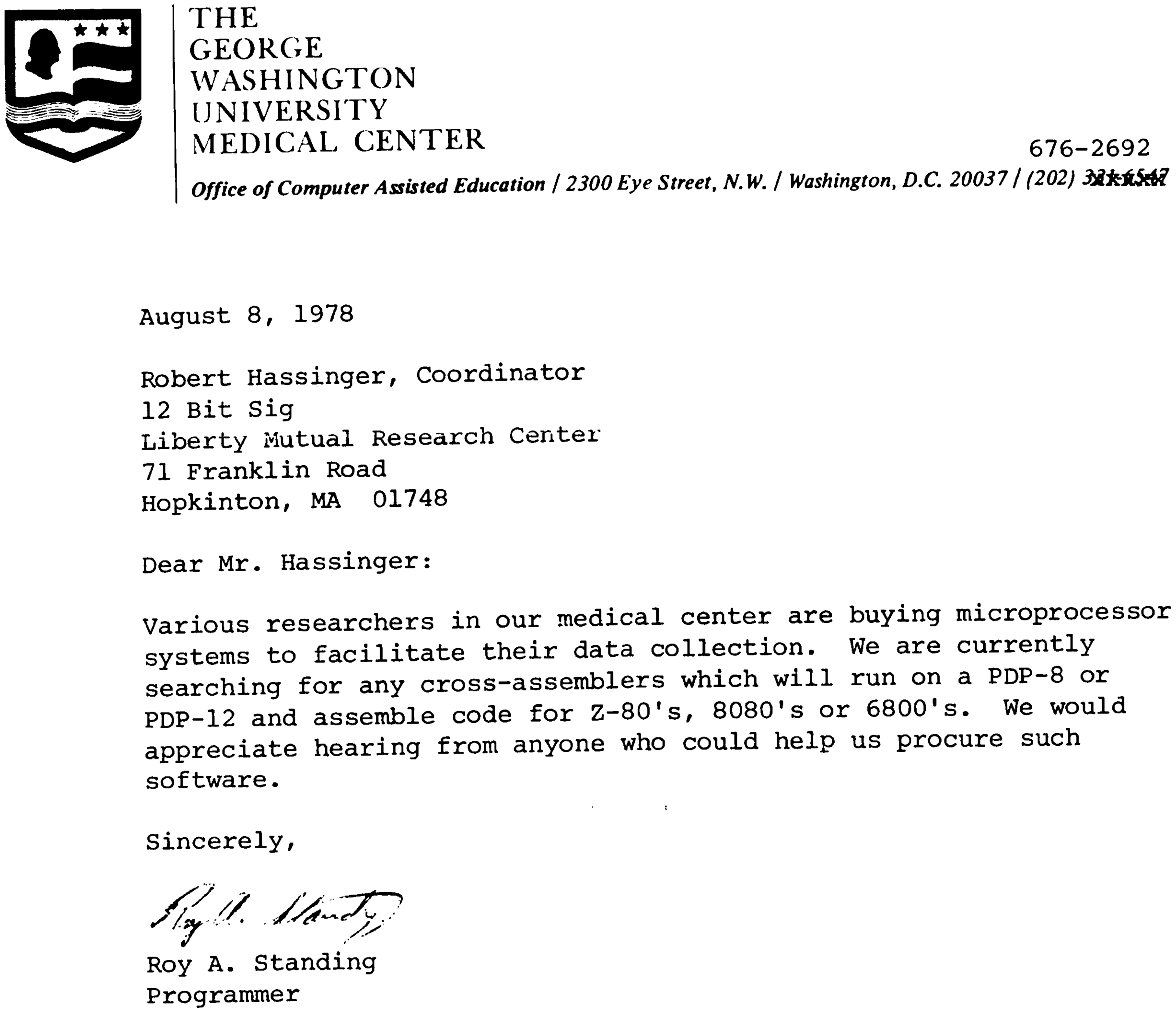
> "=" can be used in CCL commands in addition to "<" and "_". For example, the command "DIR TEST.XX=DTA1:/E=5" is now valid. So is "COPY FILE1=FILE2". This feature improves compatibility with the DECsystem 10 moniter TOPS 10.

The "F(UTIL)" command (see newsletter #28) is fully implemented.

If you are interested and do not want to wait for Don to release this version of DECsystem 8, you should contact Lyle at 47 Ivy Mills Rd., Glen Mills, PA 19342.

NOTE FROM ERIC WOGSBERG

Eric wrote to say that he is selling a PDP-8A based system and some extra option boards. Anyone interested can contact him at Computer Technology, 6043 Lawton Ave, Oakland, CA. 94618 (415) 653-4844.



THE
GEORGE
WASHINGTON
UNIVERSITY
MEDICAL CENTER

676-2692

Office of Computer Assisted Education / 2300 Eye Street, N.W. / Washington, D.C. 20037 / (202) ~~331-6547~~

August 8, 1978

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

Dear Mr. Hassinger:

Various researchers in our medical center are buying microprocessor systems to facilitate their data collection. We are currently searching for any cross-assemblers which will run on a PDP-8 or PDP-12 and assemble code for Z-80's, 8080's or 6800's. We would appreciate hearing from anyone who could help us procure such software.

Sincerely,

Roy A. Standing

Roy A. Standing
Programmer

DECUS 12 BIT SPECIAL INTEREST GROUP NEWSLETTER
NUMBER 30 SEPTEMBER 1978

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

NOTES FROM MICRO-8 WORKING GROUP

By Jonathan Lockwood Phone: (305) 724-7542

FALL-78 SYMPOSIUM

This year's symposium is expected to be a good one for microprocessor users. There are four application papers and one poster paper planned that will provide a foundation for discussion. There will also be a Hardware/Software Workshop for any mini-papers. The schedule for MICRO-8 applications on Monday Nov. 27th is as follows:

DECstation-78 Product Panel — 4:15 - 6:15 pm
By Gary Cole of DEC

Design of an Ocean Based Seismometer System — 8:00 - 8:30 pm
By Bob Moore of Scripps Institution of Oceanography
An all CMOS, 16k word system based on the HM-6100 which store hydrophone data on a 4 track digital cassette. Used for seismograhic research and exploration.

The GDP-12 Geophysical Data Acquisition System — 8:30 - 9:00 pm
By Bob Staley of Zonge Engineering & Research Org.
An all CMOS, battery operated, dual processor system used by the mining and oil industry. The 20k word system has interfaces for two high speed A/D converters.

An all CMOS MICRO-8 Development System — 9:00 - 9:30 pm
By William Beals and Henry Smith of Criterion Logic Corp.
A hardware development and debug system with a 40-pin in-circuit emulator that interfaces to a DECstation-78. Features include backtrace, memory overlay, hardware break point, and a trigger for a logic analyzer.

PDP-8 Development System for a Bit Slice Microprocessor — 9:30 - 10:00 pm
By Doug Gluntz of Harris Govt. Electronic Systems Div.
PDP-8/E used for source entry, linkage to a functional simulator, and downloading to a hardware protyping system. Functional simulation done in CDL (Computer Design Language) on a UNIVAC 1108 computer.

Hardware/Software Workshop — 10:00 - ??? pm
Forum for users to share ideas on various hardware/software hints, kluges, and maybe even solutions. Ten minute mini-papers accepted up to day of session; just call me for a time slot, no abstracts required.

Data Acquisition System for Offshore Oil Rigs — Poster Paper
By John Kracik of Interstate Electronics Corp.
Bouy based system to monitor temperature, pressure, strain, and acceleration. Utilizes an RF link to communicate to oil rig.

SPRING-78 SYMPOSIUM

The Spring-78 Symposium that was held in Chicago was quite succesful and interesting for the PDP-8 user. A summary of several RTS-8 sessions was presented in the May newsletter (#28). A summary of some of the other sessions follows.

OS/78-V2

Ron Jansen presented an overview of the new Commercial Basic and Symbiont features for the DECstation-78. Commercial Basic is DEC's answer to the proliferation of business Basic packages for INTEL 8080 based systems. Some of the important features include:

A better editor with more commands: now can change a string in line without retyping the whole line; automatic sequence generation of line numbers (similar to feature in DIBOL).

Full 8-bit ASCII support, ie. can now use both upper and lower case as well as all the control chars to drive the VT-78 terminal.

15 place string arithmetic that supports +, -, *, and / and which uses decimal arithmetic instead of binary arithmetic.

PRINT USING statement that allow extremely versatile formatting of displayed, printed, or stored (file) output. It provides for readable and customized printing of numbers and dollar amounts, especially in columns. It is similar to FORMAT statement in FORTRAN.

Unit Record I/O that allows access to individual records within a file. A record may be any length up to 4093 characters and all records in the file must be the same length.

Special commands to position the cursor on the VT-78 terminal; used to generate a form entry package.

Symbiont is the word used to describe a new, long overdue multi-tasking function that allows OS/78 to run simultaneously with RTS-8 on a DECstation-78. This allows OS/78 to run concurrently with say a line printer spooling routine. What DEC did was to set the software core size to 12k words and then use the last field to run an RTS-8 task. OS/78 has been patched to allow running with interrupts on. The following convention is used:

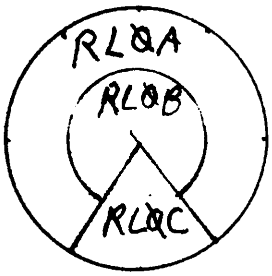
```
FIELD 0
*0001
CIF 30
JMP .
```

JMP .-1 ??(RH)

The one requirement of the Symbiont is that no DMA device load into page zero of Field zero. Since the DECstation-78 does not used DMA transfers this is not a problem. Several new CCL commands have been added to support this function.

OS/8 WORKSHOP

Jim Mechtel, the new OS/8 Project Leader, provided information concerning RL01 support, 128k word support, and gave some hints at future enhancements. The RL01 is a new "5-megabyte" capacity, top loading cartridge disk drive. Since it is organized in 8-bit words, it looks like an overgrown, high speed floppy disk. By tne time OS/8 uses the "funny" 12-bit packing, the capacity has shrunk to 2.6 million 12-bit words. Because of directory size limitations, the disk is chopped up into three logical units as follows:



RL0A: 4081 Blocks;
Tracks 0 - 177;
16 blocks per track
RL0B: 4081 Blocks;
Tracks 200 - 377
16 blocks per track
RL0C: 2025 Blocks;
Tracks 1 - 377
4 blocks per track

You will need two seperate handlers, 2 pages each, to talk to these devices. The handlers will read/write 1-32_{10} pages per call.

According to Jim there will be limited 128k word support in the next release of OS/8. Only the following was promised: PAL8 will write a 128k pgm; ABSLDR will load all 128k words; and SAVE, GET, RUN, and ODT will support 128k words. The following limitations WILL REMAIN: CUSP's like PIP will work only with 32K; USR calls also limited to 32K; and FORTRAN II, FORTRAN IV, and BASIC will not support 128k words.

Looking toward the future, Jim said that they are considering putting the Symbiont features on PDP-8/A's. Also, they are looking at some SORT/MERGE packages and other user generated programs like the enhanced version of DIRECT. However, no definite promises were made as to when these features might happen. If you have any specific features that you would like added you can write to:

The PDP-8 Suggestion Box
Digital Equiptment Corp.
PK3-1/M34
Maynard, Ma. 01754

DECSTATION-78

During the session on the DECstation-78, Gary Cole mentioned that currently there are about 150 OEMs selling various configurations that range from low end minicomputers to full blown business systems. He said that Word Processing usuage is three times larger this year than last. For users needing communications, he mentioned that a MODEM made by Racal-Vadic (Sunnyvale, Ca.) would accept an escape sequence to automatically dial a phone number. (Hum, that would be easy with the new Commercial Basic software, J.L.)

MICRO-12

A new, all CMOS, single board computer system - The MICRO-12 (HB-61000) - was described by Bill Bennett of Harris Semiconductor. This product is similar only in size and concept to the Intercept Jr. that is made by Intersil. A preprogramed ROM in Control Panel provides: a system monitor, keyboard and display utilities, and system diagnostic capabilities. The MICRO-12 includes an 8 digit LED display and 16 key keyboard which allows direct progam insertion, execution, and examination.

The system monitor allows the user to enter his program either manually through the keyboard or a TTY, or automatically from a Kansas City Standard tape cassette (300 Baud) or a DECstation-78 (9600 Baud) using the Binary Loader feature. The system monitor also provides the user with four (4) independent breakpoints for program debug. A 64 X 12 RAM for Control Panel page zero is used so that the monitor does not need any of the user program memory.

A special function key allows the user program to be listed in octal on either an external TTY or CRT. Another special function key allows the user to punch a program tape on either a TTY or a cassette.

The MICRO-12 can be used as an evaluation board when you are trying to learn about the 6100 microprocessor. It can also be used as a system board for low volumne applications. The board can be configured with up to 1k words of memory, a serial interface (either RS-232 or 20mA current loop), and 12 bit parallel interface. In addition there is a large wire-wrap area for custom I/O requirements. A dual 22-pin connector lets you connect the MICRO-12 to your system and to support boards like a 4k RAM board (available late Oct.)

For more information on this system, such as how to order one, call either me or your local Harris rep.

HARDWARE/SOFTWARE WORKSHOP

Earl Ellis presented the first three papers at the Hardware/Software workshop. His first paper described a system that used a 12k word PCM-12 and a TV camera to measure the speed of a Coast Guard Icebreaker. The system used a Colorado Video Instruments compressor to interface the TV camera with the PCM-12. The compressed raster was then processed and the ship's speed displayed on a 3 digit BCD display. The speed could also be printed on a TI-743 terminal. The operating system was U/W-FOCAL (V3S).

Earl's second paper described two Scientific Information Processing Systems (SIPS - large pdp-8/e systems). The first system is primarily intended to be a real-time multi-purpose minicomputer system. It is used to develop software for microprocessor based instrument processors, to generate report quality graphs, to provide media and code conversions, and to provide real-time data collection. The 32k word system supports dual dectape, 3 RK05 cartridge disk drives, dual floppy disk drives, a 9 track 800 BPI tape drive, and various other peripherals. Quite a variety of languages are used on this system including ALGOL-60. The second system is primarily intended for the collection and analysis of oil and hazardous chemical information. It is the system that is used to determine the source of oil spills around the country. The primary operating system is a time share system called ETOS.

A Spectrofluorometer was the topic of Earl's last paper. This instrument performs the actual analysis of the oil. It is connected to a 4k PCM-12 system via an A/D converter. The complete system is then connected to the SIPS II via a 20ma current loop (see newsletter #27 pgs 45 - 46.) The language used for this configuration is FOCAL-69, which Earl says is the best one for 4k systems. The PCM-12 boots to Field 7, location 7777 and then transfers the program to Field 0. During periods of change Field 7 is CMOS RAM with battery backup, later the program will be blown into PROMS. (This is similar to using the Electronic Program Injection module on the DECstation-78, J.L.)

For more information on these papers or if you like to talk about FOCAL, contact:

MSTC Earl T. Ellis Jr. [Phone: (203)-445-8501 Ext. 296]
USCG Research & Development Center
Avery Point
Groton, Conn. 06340

During the next session Richard Karhuse described an 8 bit handler for OS/8. Normally OS/8 writes 96 bytes out of 128 bytes per sector and uses 4 sectors per OS/8 Block, thus wasting 30% of the diskette. With his handler, which has both drives co-residenent with SYS:, he writes the full 128 bytes and uses only 3 sectors per OS/8 Block. He also described how to add Write Protect to an RX01 drive. Since the electonics already exist in the drive, all he needed to add was a lighted toggle switch and a LED inteceptor for the ASCII standard write protect hole (not on DEC diskettes, J.L.) Finally, he mentioned that the layout for the timing capacitor for a one-shot on the M1705 Dual Output module can cause false triggering of the one-shot. For more information on these topics contact:

Richard A. Karhuse [Phone: (312)-492-5248]
Northwestern University
Computer Sciences Lab
TECH B626
2145 Sheridan Road
Evanston, Il. 60201

Footnote: There was a very interesting field trip to Northwestern during the Symposium. They have a PDP-8/E that the provides management of various disks and tape drives for a laboratory computer network. This network utilizes a high speed (56k Baud) serial data link to talk to other PDP-8's, some microprocessor development systems, and to a CDC 6600. It is a worthwhile side trip whenever you happen to be going to Chicago.

A mini paper about OMNILINK, a DMA processor link for OMNIBUS devices, was presented by Ernst Lopes. This interface allows high speed transfers (150k words per second) between to PDP-8's which may be separated by several hundred meters. The transfers occur on a cable consisting of 16 twisted pairs. The device is implemented on Quad wirewrap board. For more information contact:

Earnst Lopes Cardozo [Phone: 030-882221]
European 12-Bit SIG Steering Comm.
Vondellaan 24
Utrecht, Holland

PDP-8 MANUALS

There are a variety of software manuals that exist for the PDP-8 user. There have been several updates made to these manuals which you may not be aware of, especially concerning the OS/8 Handbook. These manuals can be purchased separately and used to evaluate new features before you purchase new software. They may be ordered either through your local sales office or directly from the Tecnical Documentation Center serving your region. The order numbers below were obtained from the Technical Documentation Catalog Spring 1978 (EA 09342 86/78 030 3805.)

The OS/8 Handbook has been revised steadily since it was printed in 1974 and needs to be rewritten now. The most complete and well formated documentation can be found in the OS/78 User's Manual which comes with the DECstation-78. This contains information about the latest OS/8 revision, ie. V3-D, and some information that was formerly in the OS/8 Software Support Manual. However, since OS/78 is a subset of OS/8, it does not list all the feature of OS/8-V3/D. The order numbers for these and other OS/8 related manuals follow:

NAME	ORDER NUMBER
OS/8 Handbook	AA-4637A-TA
OS/8 Handbook Update	AD-4637A-T5
OS/8 V3D Release Notes	AA-4645B-TA
OS/8 V3C Software Support Manual	AA-4646A-TA
OS/78 User's Manual	AA-5748A-TA
OS/78 Command Summary	AV-5582A-TA
OS/8 F4 Software Support Manual	AA-4532A-TA
OS/8 Macrel/Linker User's Guide	AA-5664A-TC
OS/8 Macrel/Linker Release Notes	AA-5663A-TC
RTS/8 User's Manual	AA-0724C-TA
RTS/8 Release Notes	AA-5158A-BA
RTS/8 DECNET/8 User's Guide	AA-5184A-TA
RTS/8 DECNET/8 Release Notes	AA-5747A-TA
PDP-8 Programing Manual	AA-0586A-TA
PDP-8 Pocket Reference Guide	EH-01805-77
PDP-8 Family Commonly Used Utility Routines	AA-4338A-TA
PAL 8 Assembler Document	AA-0615A-TA
FOCAL-8 Document	AA-0627A-TA
FORTRAN/SABR Document	AA-0632A-TA

MICRO-8 COMPUTERS

I have recived information about another Micro-Computer system based on the 6100 microprocessor, this time from Europe. This 32k word system has several interesting interface cards available including:

- Floppy disk controller for either DSD 310 drives or Shugart SA400 mini drives.
- IEEE 488 Instrument Interface bus; either a software driven version or a hardware driven version.
- 512 X 8 Video RAM including modulator for VHF.
- A/D multiplexed subsystem.

The system uses an advanced control panel, completely implemented in software to perform all the required utilities necessary for efficient software development/ debugging. The control panel software allows for:

- Loading, inspection, and modification of binary programs
- Start, suspension either by hardware or software and continuation of programs.
- Trace functions giving all information on actual progam execution.

Several ROM based application modules are available including:

- A 1k word floating point package providing add, subtract, multiply, and division. Also converting and formating for display.
- A 1k word Real-Time Monitor for parallel processes and utilities for interactive communications.

For further information contact:

J. Molgaard or Chris Bagge Telephone: (02) 86 77 22
ELEKTRONIKCENTRALEN
Venlighedsvej 4
DK 2970 Horsholm
Denmark

SHORT NOTES

In the last newsletter (#29 page 14) Michael Mazzoni "discovered" that system CUSP's could not be called via the .RUN command under OS/78. This is easily solved by the following CCL command: .SET SYS OS8 and restored by the CCL command: .SET SYS OS78. For more information on this and other SET commands see Appendix J of the OS/8 HANDBOOK UPDATE.

Also from the last newsletter (page 17), Dave Kocsis mentioned that Intersil was looking for computer programers. Well Harris is also looking for some PDP-8 computer programers. If you are a professor at some college, then you could suggest to your students that they look at some of the companies that are using the 6100 microprocessor for posible job oportunities.

Harris Semiconductor has just finished a 200 page Systems Design Manual for the HM-6100 microprocessor. This manual covers techniques for hardware interfacing of a variety of systems from the bare bones, minimum controller to a full blown 32k word computer. For more information on how to purchase this and other manuals just call me or your local Harris rep.

EYE RESEARCH INSTITUTE
OF RETINA FOUNDATION
20 Staniford Street
Boston, Massachusetts 02114
19 August 1978

(617) 742-3140

Changing Terminal Device Codes on a PDP-12:

In the May newsletter I described our continuing effortto patch OS/8 for a console device code of 40/41. Since then, I have received a deluge of letters (three) on the general theme of "do it in hardware, dummy!"

These notes all refer to PDP-12's, but are quite likely applicable to PDP-8I's.

Fred Brandt has noted (Newsletter 29:12) a solution by Dave Talkin, which modifies the existing teletype interface. The final result is a single interface, with device code 03/04, with output switchable between (1) 300 baud, current loop and (2) 1200 baud, RS232.

Joe Madden sent me full details, including a wirelist, which I append. In the normal configuration, 03/04 controls a 110-baud, current-loop teletype, and 40/41 controls a 1200-baud, RS232 video terminal. In the switched configuration, 03/04 is the terminal and 40/41 is the teletype. To switch configurations requires changing four jumpers and swapping two cables that plug into the backplane, so it is inconvenient in a situation where frequent switching must be done by nontechnical people. His solution has the very great virtue, however, that the "normal" configuration is absolutely identical to that supplied by DEC.

Dwight Smith describes a modification designed by Bruce Robins of DEC Field Service in Maine (207 797-9220), which 1) changes the DP12 device codes from 40/41 to 43/44, 2) installs a DPDT switch on the frame of the backplane, wired to 3) swap bit 6 of the TTY and DP12 device codes, thus:

	TTY	DP12
normal	03/04	43/44
switched	43/44	03/04

"The total bill for the hardware modifications was under $100," according to Dwight Smith. We are going to have this modification made on our -12, and our field service reps have been talking to Maine, but all they will promise is to do it on an hourly-rate basis; they estimate it will cost $250. I'll note the eventual outcome. This is the most convenient solution for us, and note that it is the only one of the three which works if it is necessary to output properly to a teletype which requires two stop bits and to input block transfers from a terminal which generates only one stop bit. However, it produces a non-standard configuration and could conceivably cause trouble someday if we ever wanted to install a PT08 . . .

A UWF/v4 trick:

Now that we have within-group group-independent addressing (via "group 0"), you and I and everybody else want group MOVE's. While we're waiting for Jim van Zee to figure out a clever way to implement this in less than one more word, here's a way to pass the time.

UWF detects the entry of an indirect line as a line starting with a digit. However, the line number is evaluated in the normal way, and can be any general expression, so long as it begins with a digit. I write my subroutines as single groups, using exclusively group-zero addressing. Let's say it's group 9. I output the group as a .DA file (O O SUB;W 9;O C). I then edit the file to change all the line numbers from 9.XX to 0.XX+#. I have a (sorry, MOBY MUNGERS) FOCAL program to do this automatically. To insert this into a new program as, say, group 18, I just SET #=18;O I SUB

Switchable Asynchronous Ports

The PDP-12 has a pre-wired Dataphone interface that uses device codes 40, 41.

Unlike the console teletype, the Dataphone port allows a choice of baud rates by crystal and/or jumper selection.

In order to switch the clocks such that the 1200 baud Ann Arbor terminal can be used as the console device, the following modification has been installed:

Delete	Add
N25E1 - N11J1	N25E1 - N11J1
N25E1 - N12P2	
N10P2 - N11E1	N10P2 - N11E1
N12V2 - N11E1	
N08J2 - N08P2	
N08K2 - N11D1	

To switch ports such that the console device codes 03, 04 control the E1A RS232 1200 baud terminal and device codes 40, 41 control the teletype, install four jumpers as shown below. Change the corresponding cable connections to the backplane.

To restore the original configuration, change the jumpers and the cables.

N11	Normal	Switched
D1	N08K2	N12V2
E1	N12V2	N08K2
H1	N08J2	N12P2
J1	N12P2	N08J2

Cables

Console	N2	N3
Remote	N3	N2

Above is from Joseph A. Madden, Veterans' Administration Hospital, 13000 North 30th Street, Tampa, Florida 33612; (813) 971-4500 x 301.

Notes: (by DPBS) The effect of the modification is to provide a jumper-switchable interchange of the baud rates of the two interfaces. As supplied by DEC, the teletype interface is 110 baud, RC-controlled, and the DP12 is customer-selected (1200 baud in Joe Madden's case), crystal-controlled.

The DP12B is normally supplied with a BC01A-25 cable assembly, with an RS232 connector on one end and a PC board on the other; the teletype cable terminates in a PC board. The slots on the teletype and DP12B interfaces are functionally equivalent, according to Joe, the necessary level conversions, etc. being done on the PC boards. So it is possible to simply switch the cables, and all that remains to be done is to set up the proper baud rates. (Note, however, the stop bit question--as wired, both TTY and DP12 send/receive 2 stop bits, which is OK for most applications. If the DP12 has been modified for 1 stop bit, presumably it will not operate the TTY properly even if the baud rate is correct).

GALLAUDET COLLEGE

KENDALL GREEN, WASHINGTON, D.C. 20002

SENSORY COMMUNICATION
RESEARCH LABORATORY
HEARING AND SPEECH CENTER

August 2,1978

Mr. Bon Hassinger
12 bit Sig Coordinator
Liberty Mutual Research Center
71 Frankland Road
Hopington, Massachusetts 01748

Dear Bob and Fellow decus Members:

In the May 1978 issue (#28) on page 39, there is a letter from Dan Smith reguarding OS/8 and device codes. We had a similar problem and here is how we (David Talkin and myself) solved the problem.

If you take a look at the PDP-12 prints you will indeed see that the TTY interface and the DB-12-B are in fact similar. The primary difference being as Dan noted the device codes (03 & 04 for TTY and 40 & 41 for the Dataphone). You will also see that the speed for the TTY is set by a RC clock (M 452). The speed for the Dataphone is set by crystal clock (M405 in slot N 11). Also the Dataphone requires a BC01A cable. It is this cable that does the level conversion from RS232 to the TTL level required by the M706 & M707). A note on the prints for the DAtaphone states that the clock rate must be 128 times the Baud rate for speeds up to 10000 Baud.Apparently speeds are available up to 100,000 baud with a slight additional change.

From this it can be seen that to change the console terminal to something other than a tty at 110 Baud all one has to do is change the clock speed if going to 300 Baud or plug in a new clock card and the BC01A cable for the RS232.
On our PDP-12 we installed a switch so that we could use the Decwriter at 300 Baud or a CRT in an adjucent office at 1200 Baud.

There has been over the past year or so ocassional references to the Bat Handler and how to use it in the OS/8 system. We use Fortran IV for signal generation and data analysis on both ourPDP8 and 12 systems as well as the Dec System 10 at the computer center. On the 10 system it is possible to have the batch processor intercept all input and output to you TTY. Unfortnately the Batch processor for the OS/8 system is not as intellignet. But by use of the Bat handler you can read in data from the batch stream. The one necessary piece of information not given in an obvious place is how to terminate the data input to the batch handler. I submitted an SPR on this and received a call from the Maintainer. He had checked various things including the coding and found that the data stream needs to be terminated by a '$'. This is shown in the code for the batch handler but nowhere in the documentation for batch itself. It is mentioned in a section called Advanced Features in MS BAtch. There is for MS Batch a fourth type of instruction '$EOD' or I assume 'End Of Data'. This bears out my experience with the Bat handler. I found that if there was nothing past the data except the '$EOF' then all seemed well. If however there were additional commands following the data the results were unpredictable. The enclosed TTY printout will illustrate the use of the Bat Handler in a Fortran IV system.

The restriction that there must be an extra 4 K of memory still holds. Our systems are all 32K. I have a special version(Frts without the patch to run in 32 K) of FRTS to be used with the Bat handler. Batch has been patched to run in the full 32k while FRTS only knows about 28K.

This could be a problem for people with less than 32K. I tried to use the version of FRTS set up to 32K and then the first command after begining the job to lower the core size. This does not work! Maybe some day these restrictions will be removed.

Sincerely,

Fred D. Brandt

THE FOLLOWING IS THE LISTING OF THE JOB SUBMITTED TO THE BATCH STREAM...

THE 'BAT' HANDLER WILL READ THE DATA FROM THE BATCH JOB BUT NOT TYPE IT OUT INTO THE LOG.

```
$JOB TEST FORTRAN 4 BATCH READING
.COM F4TST.,TTY:<F4TST.FT
.LOAD F4TST.RL
.R FRTS1
*F4TST
*BAT:/8$
23
34
45
56
$
.RES
.MEM
$END JOB

.SU F4TST.BI
```

```
$JOB TEST FORTRAN 4 BATCH READING


.COM F4TST.,TTY:<F4TST,FT
        FORTRAN IV  4BAAA    7-AUG-78               PAGE  ONE

        C        FORTRAN IV TEST                   8-2-78
        C
0002             READ (8,100)A
0003             READ (8,100)B
0004             READ (8,100)C
0005             READ (8,100)D
0006    100      FORMAT (I6)
0007             WRITE (4,105)A,B,C,D
0010    105      FORMAT(' A= ',I6,'   B= ',I6,'   C= ',I6,'   D= ',I6)
0011             STOP
0012             END
.LOAD F4TST.RL
.R FRTS1
*F4TST
*BAT:/8$

A=      23   B=      34   C=      45   D=      56

.RES
SYS,DSK,RKB0,RKA1,RKB1,NULL,DTA0,LPT,DUMP,BAT,TTY

.MEM
32K MEMORY!
$END JOB
```

NOTE FROM BILL HAYGOOD ON MULTI-USER OS/8

I would like to let the 12-bit community know of the completion (at long last!) of the MULTOS-8 project (Multi-user OS/8). It took quite a while longer to get out from under the Postal Service CAI project than I had anticipated. But MULTOS-8 is now up and running and ready for delivery. MULTOS-8 presently supports the following hardware:

* 16 - 32K memory (with 20K minimum recommended)
  Each user has a virtual 32K memory regardless of physical amount of memory.
* PDP-8/A,e,f,m
  PDP-8/I (code not completely debugged for 8/I)
* Up to 4 TTYs/terminals/CRTs each with own apparent copy of OS/8, peripherals, etc.
* KE8E EAE (supports both 8/e and 8/I type EAEs)
* RK8E RK05 disk drive 0 used as swapping and SYS disk
* Up to 3 additional RK05 disk drives (RKA1-3 and RKB1-3)
* PC8E Paper Tape Reader and/or Punch
* RX01 Dual Drive Floppy in both 12-bit (RXA0-1) and 8-bit (R0-1) modes (handlers for both modes included)
* Up to 8 TM8E Magtape transports
* Up to 6 TD8e DECtape transports (Yes, TD8E -- no hardware mods)
* Real Time clocks: DK8EA, DK8EC, DK8EP and PDP-8/A DKC8AA
* Any Line Printer using device code 66

Other features include:

* A Print Spooler which can be called with a keyboard command or under program control. Up to 32 files of any length can be spooled. As soon as the command is issued, control returns to the OS/8 Monitor so that other processing may be done while the printer is outputting the desired files.
* Full Batch processing from any and/or terminals.
* Automatic device assignment/de-assignment.
* Files can be transferred between terminals and/or any devices on the system.
* Separate passwords for each terminal may be used, if desired (nice for modem applications).
* Surprisingly low priced.

Future enhancements include support for 8 TC08 DECtape transports, RL01 disk and KL8A terminal interface.

For more information, please call or write me at 801-942-2300, ComServ (Computer Services) Enterprises, 7822 Oakledge Road, Salt Lake City, UT 84121 USA.

NORTHWESTERN UNIVERSITY
EVANSTON, ILLINOIS 60201

DEPARTMENT OF ELECTRICAL ENGINEERING
AND COMPUTER SCIENCE

THE TECHNOLOGICAL INSTITUTE

August 14, 1978

Mr. Robert Hassinger
Co-ordinator, 12 Bit Sig
Liberty Mutual Research Center
71 Frankland Road
Hopkinton, MA 01748

Dear Bob,

As a frequent RTS/8 user-programmer, I would like to make some responses to Lee Nichols letter in the July '78 Newsletter (Number 29, pages 2-5).

First, let me thank Lee for taking the time to hhlp improve RTS/8. He was an excellent choice for the head of th ⁞ RTS/8 working group.

Second, I would like to make one big suggestion to anyone interested in changes to RTS/8. This is "Look at RSX-11". RSX-11 is a real-time, multi-tasking system for the PDP 11. It is probably thh closest thing to RTS/11. RT-11, the other PDP 11 system, is real-time, but not truly multi-tasking. Thus, many of the problems we are encountering with changing RTS/8 have been seen by the developers of RSX-11. I will grant that RSX-11 is more sophisticated than we may want RTS/8 to be, but many of their features can solve our problems.

The main reason for an executive in a real time system is to provide services which are inherent in the hardware, but because of multiprogramming and real time constraints, must be disbursed on a controlled basis. An obvious example of this is I/O. However, most executives also augment these "hardware" services. It is th augmentation which we discuss here.

For example, there is nothing inherent in the hardware about getting a command line. However, almost every task must do it, so RSX provides an executive call to retrieve a command line which has been entered to MCR. Thus, most RSX tasks are entered via the MCR command:

```
nam ...command line...
```

where "nam" is the task name. MCR automatically does a "RUN nam". The task then issues an exec request to fetch the command line, and processes it. By convention, any task entered in this way terminates after processing the single command line. If multiple command lines are to be entered, the task is invoked with the MCR command:

```
nam
```

again, where "nam" is the task name. MCR merely does a "RUN nam", and the task comes up. It does a "get command line", finds none, so prompts for one and continues prompting for one until the user specifically exits the task (by entering CTRL/Z). There is a certain logical consistency here. In the first case, the user is requesting the task to do a specific function. It does it and exits. In the second case, the user is requesting the task, but not specifying the function. Thus the task assumes that there is more than one function to be done.

Another thing which I can't stress enough is "USE MACROS". We have a nice macro assembler now and making common code macros is _so_ nice. This way your code is not scattered with

```
CAL
SENDW
TTY
TTMSG
```

but instead

```
SEND.W    TTY,TTMSG
```

Not only is this easier to read, but it avoids the problems with future changes in the format of executive requests. You just change the RTS/8 macro library, and everyone has made the change. The advantage of this method is that those poor souls who cannot use macros (because of limited space on assembly) can continue to write it all out. The real solution would be to have MACREL use secondary storage (e.g. disk) for extended symbol and macro table storage if there isn't enough core. MACRO-11 does this. Stan Rabinowitz, are you listening??

RSX-11 also provides "informational" directives. These allow a task to obtain information about its enviornment, including "task parameters" (name, number, priority, etc.), "partition parameters" (partition name, length, etc.). These are not immediately useful in RTS/8, since tasks usually know these things. However one directive which would be useful is "get time parameters". This returns the date and time in an 8 word buffer.

Something which the DECnet authors found lacking in RTS/8 are AST's. For reasons I don't understand, they implemented them directly in NSP rather than in RTS/8.

An AST (Asynchronous System Trap) is a task interrupt intiated by the executive to allow servicing of contingencies including signalling events, such as the completion of a previous I/O request. The executive keeps track of all AST's, queues them (FIFO), and is aware when a task is servicing an AST. Upon exiting an AST service routine (which is similar to an interrupt service routine in structure), control is returned to one of three places.

1. Another (queued) AST;

2. The task, or

3. Another task (e.g., the corresponding task was in a wait or suspend state prior to the execution of the AST).

Some examples of AST's include:

RECIEVE MESSAGE AST which is invoked when a message is queued for the task.

POWER RECOVERY AST which is invoked during the power recovery procedure.

I/O COMPLETION AST which is invoked upon completion of an I/O request.

MARK TIME AST which is invoked upon completion of a mark time request.

Two directives, DISABLE AST RECOGNITION and ENABLE AST RECOGNITION, allow AST's to be queued during critical sections of code that access data bases that are also accessed by AST service routines. If AST's occur while AST recognition is disabled, they are queued and processed when AST recognition is enabled.

A method of waiting for the logical 'OR' of event flags must be provided. The method described in the RTS/8 User Manual is crude at best. RSX provides a directive which does this.

As for an RTS/8 User Command Language (UCL), the method of invoking tasks described above is often enough for experienced users. However, for those times when it isn't, MCR *should* have a user interface. This probably shouldn't be done dynamically at run-time via messages. Few applications require such flexibility. Instead, the command language should be fully defined at assembly time. There should be a "User Command Table", which would be a global symbol, thus any tasks can access it via .EXTERNAL. The table will be fully defined in the PARAM file. I can't think of any way of including code in tasks to create this table. How could we keep track of the next free table entry?M@ L.MAR +5

Since the language must be totally general (from MCR's point of view), it is difficult to have MCR (or USERCD) do much besides invoke a task and pass the command line to it. However, some initial parsing could be done by MCR, particularly if we define our language rather well. The user would define each argument to the command, whether it is optional or required, and what type of argument it is. The types would include octal number, decimal number, memory location, task name, and character string. In the first four cases, MCR could do conversion and print errors (perhaps more descriptive than "BAD NUMBER"), thus providing a high level parser for the user tasks. This parsing is both modular and will save space, since only one copy of each conversion routine is needed (in the high level parser), rather than a seperate copy in each task. The character string type is a "catch-all", which prevents MCR from parsing this arg -- it is passed directly to the user task.

The entire command table would be constructed using macros, of course. The macro CMD would start a command table entry. It defines the command, possibly including required and optional characters, the task to be unblocked, and which task status bits should be unblocked. The command arguments are defined by successive calls to ARG, which defines each argument, types it, classifies it as optional or required, and specifies a user buffer to contain the parsed argument. Finally, a call to ENDE ends this table entry.

Thus, a sample command table entry would be coded as:

```
CMD  DUMP,DMP,RUNWT
     ARG  TSKNAM,OPT,DMPNUM
     ARG  MEMLOC,REQ,DMPST
     ARG  OCTNUM,OPT,DMPLEN
ENDE
```

This defines a command, "DUMP", which will cause the task DMP to be removed from RUN wait. The command has three arguments. The first is a task name or number (TSKNAM) which is optional (OPT) and the parser will place the task number in DMPNUM. The second arg is a memory location, which is required. The parser will create a two word block at DMPST which contains a CDF to the field specified as the first word, and the absolute memory location as the second word. The third argument is an octal number (OCTNUM), which is optional. If it is present, it will be converted and placed at DMPLEN. The ENDE call will end the entry (nominally place a zero in the table).

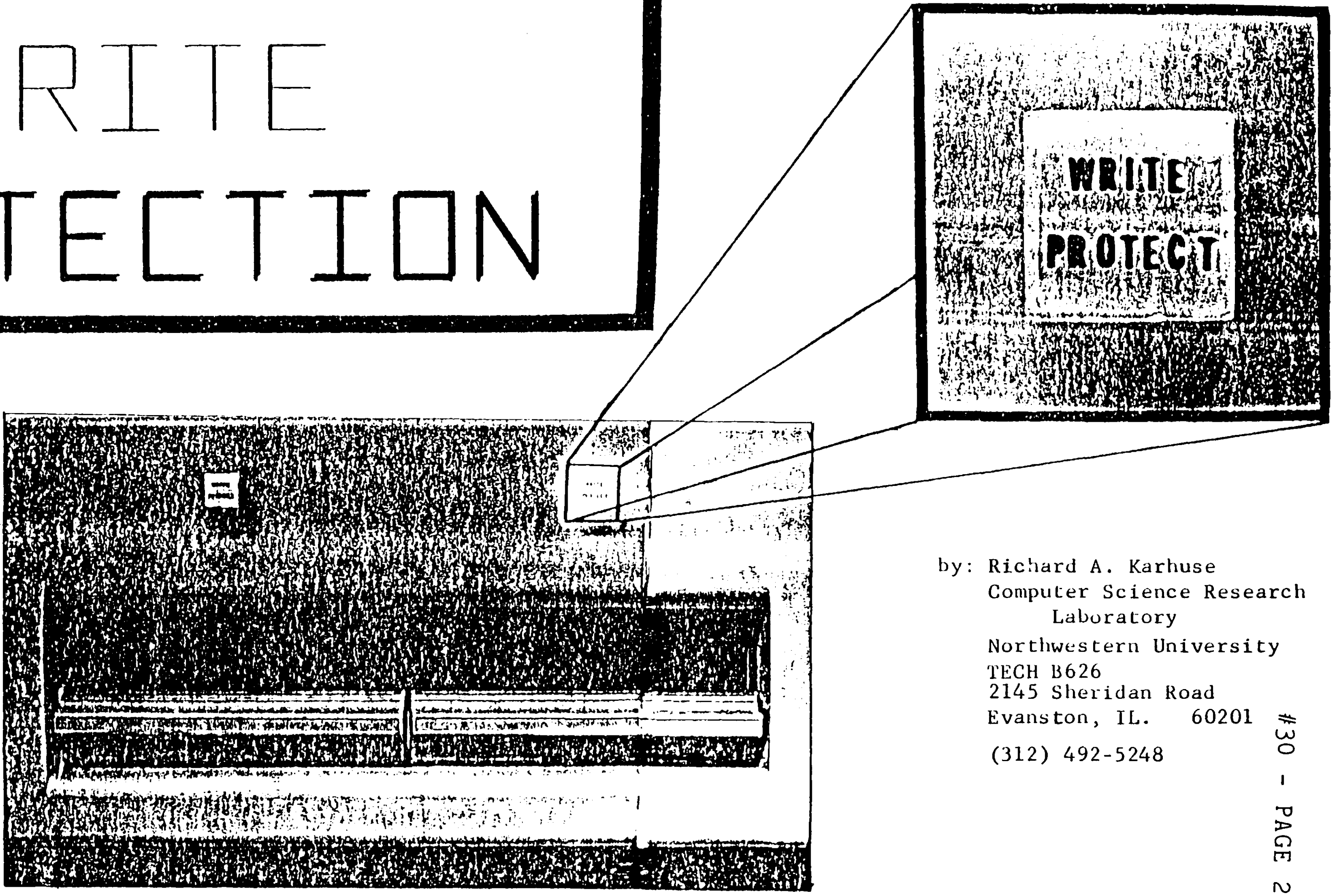
I would be interested in reaction to this proposal.

Sincerely yours,

Michael Kelly

Michael J. Kelly

RX01 WRITE PROTECTION



by: Richard A. Karhuse
Computer Science Research Laboratory
Northwestern University
TECH B626
2145 Sheridan Road
Evanston, IL. 60201

(312) 492-5248

The Computer Sciences Research Laboratory of Northwestern University purchased one of the first DEC RX01 dual floppy disk drives. Shortly after the warranty period expired, so did the floppy disks. DEC in their infinite wisdom wanted a flat $300.00 + travel + etc. to fix it. The laboratory felt this to be exorbitant and typically has done all of its own maintenance in-house. In the process of diagnosing the disk drive, we uncovered what appeared to be a drive write-protection circuitry.

Installing write protection into the the RX01 floppy disk drive turn out to be simple. DEC has included all the necessary logic and microprogramming for it, but documents it nowhere, except marginally in the prints. DEC has brought all the necessary signals out to a Berg-type connector. It appears DEC had planned on using a floppy drive which would sense a hole in the corner of the floppy diskette. This is the way IBM defines write protect on floppy disks much like the tab on cassettes. Apparently the drives that Digital is using does not have this capability (although we have never verified this fact).

To write protect floppies on the RX01, all that is needed is a switch, a little cable, and a Berg connector. Power is provided on the connector if a lighted switch is desired. In this case an additional resistor must be added to the driver board so that sufficient current is supplied to light the lamp.

When the switch is closed (see circuit diagram), DRV WRT PRCT is generated. The microcontroller senses this signal whenever a write operation is issued by the host computer, for a given drive (0 or 1). If the drive is protected, the controller aborts the operation and sets the done, and error flags. The RXES is set to a value of 410 (octal) and the ERROR register has a value of 210 (octal). All system software tested (OS/8, RTS-8 RT-11) did zero or more retries and then terminated the operation unsucessfully.

Installation Hints

The simpiliest way to write protect a floppy disk is to install a jumper or switch between pins #1 and #3 on the appropriate Berg pins of the M7727 drive electronics board. Any sort of toggle switch will do. We opted for a more esthetically pleasing back-illuminated, push-button switch. In addition to jumpering the above two pins, a six volt lamp and switch is placed between pins #6 and #8 to indicated when the drive is write protected. To obtain the half ampere needed to drive the lamp a 12 ohm resistor was tacked on top of R128 (for drive 0) or R111 (for drive 1).

The labeling "WRITE PROTECT" was obtained rather ingeniously. First, the logo was generated with 18 pt.

rub-on letters to a piece of paper. This art-work was reduced several times with a Zerox machine until the correct size was obtained. The logo was then transfered to clear plastic by using a Thermo-Fax machine—the device used to create overhead tranparencies. The plastic logo was then mounted to the button using Scotch PHOTO MOUNT Spray Adhesive (cat. no. 6094). The plastic is cut to the size of the button.

The switches are then mounted on the RX01 face plate. This plate is heavy cast aluminium. Thus a drill press should be used to punch the holes. Masking tape should be place on the outside of the face plate to prevent marring and the hole should be punched from the backside. The hole should be located such that the backside of the switch is just above the back retention plate.

Conclusion

The write protect switches have been use in the laboratory for over six months with no problems and prevented some good floppies from being accidentaly creamed. If there is sufficient (but not overwhelming) demand, the laboratory could provide the write protect switches built and tested for a nominal charge for installations which do not have the specialized equipment nor personnel to build the switches. These users would still have punch the holes in the aluminium and install the switches.

To conclude with a continuation on the history of our drive, we have developed a RX01 micro-controller diagnostic. This diagnostic allows us to single step the RX01 controller and examine various internal signals. More importantly, it allows us to "dump" the micro-controller ROM's. This dump then can be compared (via source compare programs) with a good floppy's ROM to determine which location if any have changed--we have encountered several floppies with this problem. In our case, an "unused-bit" changed state and caused our RX01 to jump internally to non-existant memory. We fixed our problem with about $0.10 of wire by totally disabling this unused bit.

The diagnostic will only be useful for installations which extensively do their own maintenance. It requires 27 bits of parallel input interfaces from the micro-controller to the diagnosing computer. M1703's or DR11's will suffice to bring this data in. However, a cable interface must be built on a flip-chip module. The diagnostic is written in OS/8 FORTRAN II with one SABR level subroutine to read the M1703. It should be fairly easy to transport this diagnostic to a PDP-11.

The diagnostic is not very well documented currently. But, I am willing to work with anyone who really wants to use it. It goes without saying that any installation who wants to use it must have some other mass storage device other than the floppies because you cannot run the diagnostic off a sick floppy. Our laboratory currently supports: DECtapes, DOS format 9-track magtapes, RK08 disk packs, RK05 disk packs, and paper tape.

PARTS LIST

(per drive)

Lighted Push-Button Switch

Push-on, push-off 5 volt switch

(Cutler-Hammer S31DE191-1 or equiv. available from Newark)

Berg Header/Pins

Header 65043-033 2x4 header, 0.10 spacing with four 47712 pins

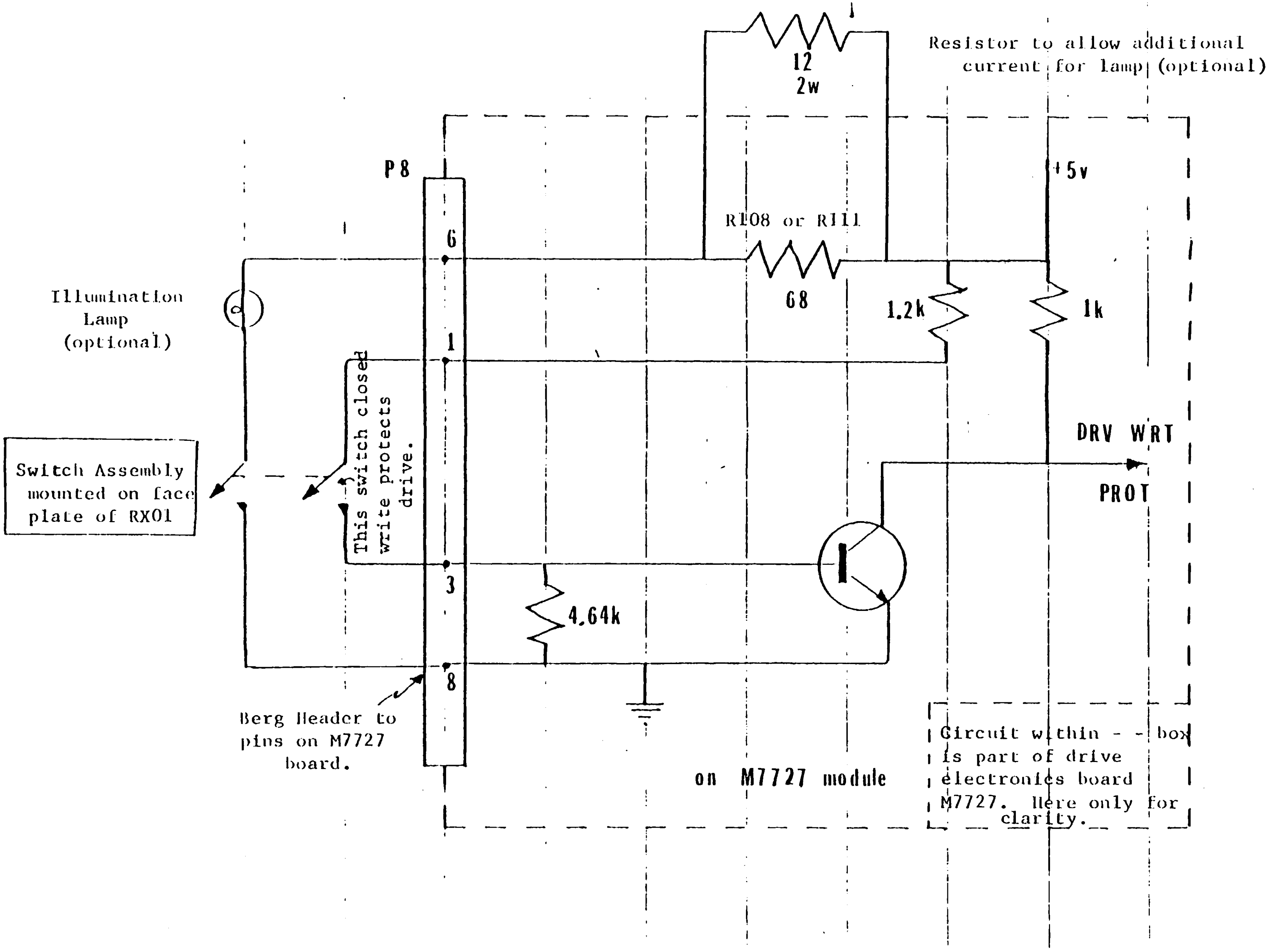
Cable

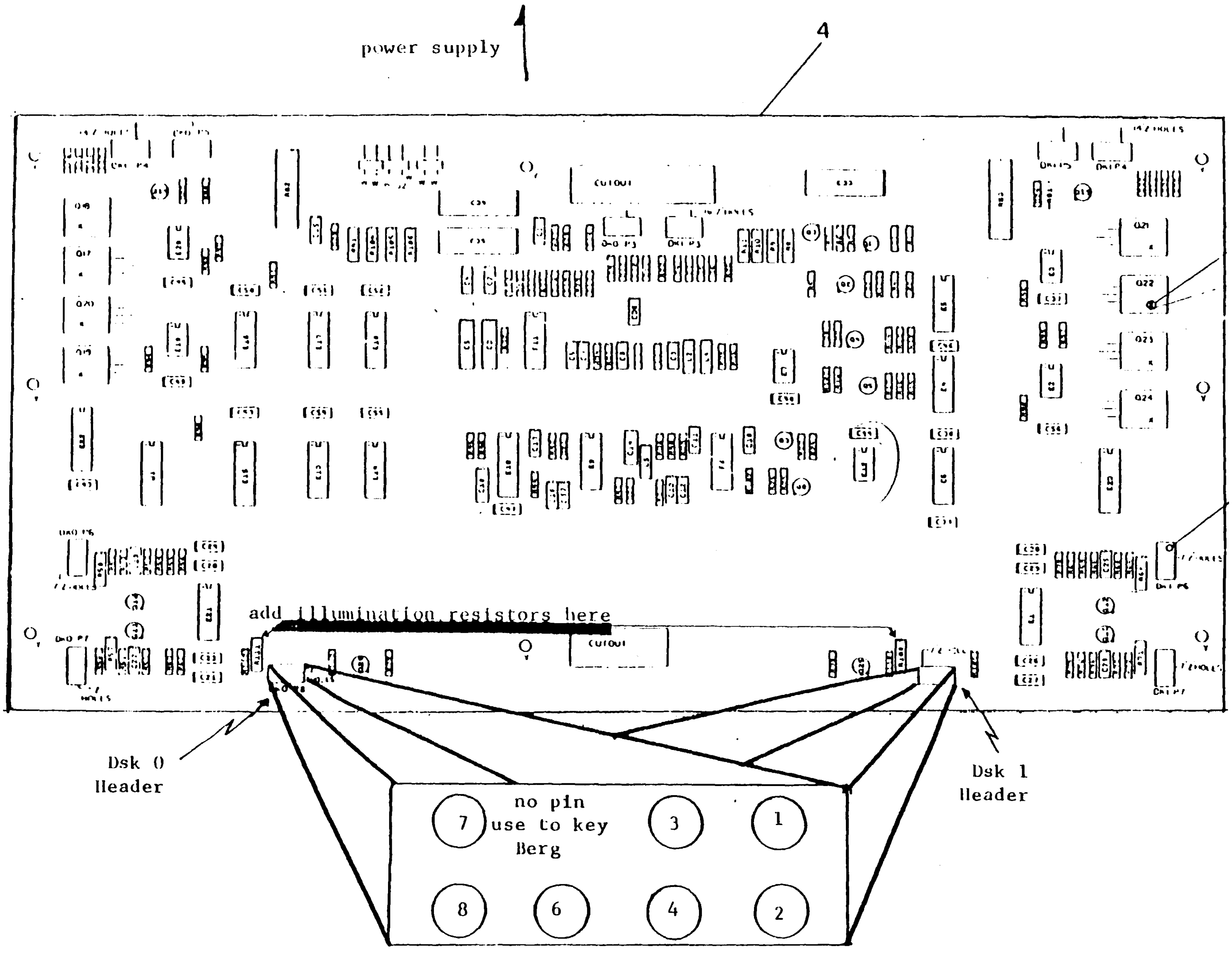
Four conductor approx. 1 foot

Resistor

12 ohm, 2 watt resistor

RX01 WRITE PROT. CIRCT.


12
2w
Resistor to allow additional current for lamp (optional)
P8
+5v
R108 or R111
6
68
1.2k
1k
Illumination Lamp (optional)
1
This switch closed write protects drive.
DRV WRT PROT
Switch Assembly mounted on face plate of RX01
3
4.64k
8
Berg Header to pins on M7727 board.
on M7727 module
Circuit within - - box is part of drive electronics board M7727. Here only for clarity.



Berg Header Pin numbering

M7727 Component Layout

SUMMER INSTITUTE OF LINGUISTICS, INC.

INTERNATIONAL LINGUISTICS CENTER • 7500 W. Camp Wisdom Road, Dallas, Texas 75211 • (214) 298-3331

July 5, 1978

Mr. Robert Hassinger
c/o DECUS
129 Parker Street, PK-3/E55
Maynard, MA 01754

Dear Mr. Hassinger,

Seeing the CPU Hints and Kinks published in the 12 bit #27 encouraged me in include a change we made several years ago to our Straight-8 to overcome its inability to combine the Group 1 Operate Instruction INC and the various ROTATES. The problem is the set up time required by the DCD gates used for the rotates. By adding R302 delays in the various Rotate signals the gates can re-setup with the incremented value in the AC before the rotates. We used PF11 and PF12,two slots that were originally wired for the A-D converter. Others may need to find a free space to add the two cards. Don't be tempted to just delay the PQP1signal, the MB control bits change too soon for that. I hope this will be helpful to some of those who are still using the original 8's or someone who is thinking of buying one from the surplus market.

We would appreciate it if you could send us a copy of the back issues of 12 bit on microfiche if there are any left. Sent

We would be glad to duplicate paper tape copies of DIRECT V5 if that is still a live option. If not, how can we get a copy of the latest version? It sounds like something we'll need.

Sincerely yours,

Dick Bronson

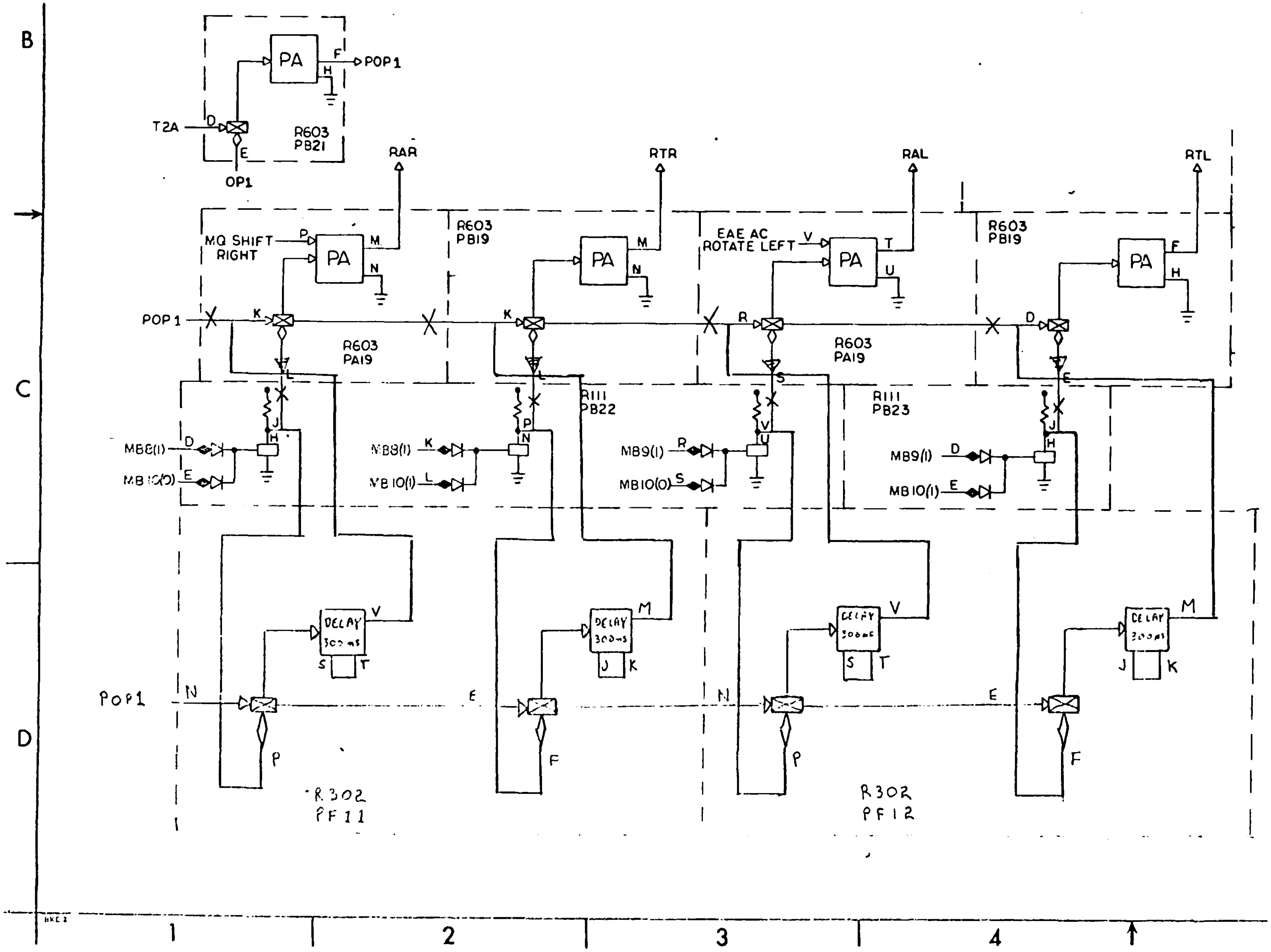
Dick Bronson

DB:ns

#30 - PAGE 34

NO.

TITLE COMBINE INC + ROTATE



IOTA SYSTEM SUMMER INSTITUTE OF LINGUISTICS

TITLE COMBINE INC + ROTATE

DRAWN BY

DATE

NO.

C

DP-102-8

USCG Research & Development Center
Avery Point, Groton, CT 06340

Mr. C.J. Thompson
Montreal Neurological Hospital and Institute
McGill University
3801 University Street
Montreal, Canada H3A 2B4

Dear Mr. Thompson,

Single to Double Precision in a PDP-12 or PDP-8

I have read your article in the 12-Bit SIG Newsletter and am writing in responce to your letter which was printed on page 16 of SIG #29. I hope that the code you show is not running, as a large falt is present, which other sections of your code may correct. The error is that the command SNL CLL will never clear the link. You have combined a Group I with a Group 2 Operate instruction. SNL CLL will be assembled by PAL8 into a 7520 (7420 is SNL, 7100 is CLL; these are Inclusive ORed into the 7520). When the PDP8 executed the 7520, It will decode it as a SMA SNL ! and the LINK is still set............

I had this problem and solved it as follows. It does not use any Group Two. It happens to be a little cleaner inthat it only uses 6 locations.

```
TAD I 12        / number from data array
TAD I 13        / add to low order word
DCA I 14        / save results, overflow in the link
RAL             / Link to AC-11 and AC-0 to Link (CLL)
TAD I 13        / add 0 or 1 to High Order
DCA I 14        / Finsihed, and Link = 0 for NEXT'TAD I 12'
```

I had made the same error in a A/D Averager I developed for FOCAL. It uses the call FADC(C1,T1) , where C1 is the Channel, and T1 is the number of times to read it, defalt = 1. Since we use a 12-Bit unsigned A to D, overflow will occur. The program which calls FADC does the division, why duplicate code. This runs on a 6100 chip, and can convert 1000 readings to double precision in less than .05 Sec, which is the time between events. When the event time gets much less than that, doing single to double this way is not effective, and a DMA (FPP-12) method has to be used.

Earl T. Ellis Jr.
System Programer

Copy to:
Bob Hassinger, 12 bit SIG

Dear Bob,

I have just read #29, and I would like to comment on the quanity of RX01 handlers which use over 600 blocks. I have used Mr. Dewar's RX handler for non-systems and gets 650 OS/8 blocks from two devices. I have also used the DSD RX handler which gets 658 blocks on only one device. Dr. Lynch of Xerox has submitted a handler to DECUS which gets 666 blocks.

As a member of the Software Exchange Committee, I would like to see DEC adopt one of these (or write their own) so that users can take advantage of the increased storage from the RX01. I understand that soon????? DEC will offer the RX02 with increased density. This will not eliminate the problem. I have tried to read a Dewar's RX with the DSD handler and the reverse. To date these have not been successful.

I would also like to comment on some Interrupt Code which we recently 'found' in DEC literature. When using the KL8 type interface, the instruction "KIE" (6035) can be used to enable/disable interrupts from the interface. (the KL8-E is enabled by the CAF instruction?) The need to didable interrupts can accur when more than one KL8 is being used (two or more serial devices) or if you want to service the Clock, and disable the TTY for a while, an easy way to ignore Control-C. FOCAL, INBASIC, F-IV,and RTS-8 are all examples.

The problem arrises in the interrupt service or skip-chain. It is normally coded as:

```
            TSF             /DID TTY OUTPUT INTERUPT
            SKP             /NO
            JMP SERTSF      /YES, SERVICE IT
            KSF             /TEST TTY INPUT
            SKP             /NOT THAT EITHER
            JMP SERKSF      /YES, SERVICE
            CLSK            /TEST THE CLOCK
            ....
```

If the 'KIE' instruction turned off TTY interrupts, the Clock interrupt will cause the TSF to be tested, and posiblly skip, and also the KSF if a key is struck on the TTY. However, the KL8 provides a way around this. It is the "SPI" or "TSK" (6045) instruction. It will skip if and only if the interrupt is enabled and a reader/punch flag is up. This is coded up as:

```
            TSK             /DID TTY INTERRUPT?
            JMP CLØCK       /NO, NEXT TEST CLOCK
            TSF             /DID PUNCH INTERRUPT?
            JMP SERKSF      /NO, MUST BE READER!!
            ....            /SERVICE PUNCH HERE
            ....
            KSF             /BEST TO EXIT TESTING READER
            SKP             /WASN'T READER TOO
            JMP SERKSF      /WOW!  READER UP TOO
   CLØCK,   CLSK            /TEST THE CLOCK
            ....
```

This enables the KIE instruction to control the TTY I/O to an interrupt driven program. I would like to hear from others who are working on the same problem.

EARL T. ELIAS Jr., USCGR&DC, Avery Point, Groton, CT 06340

DIGITAL EQUIPMENT GMBH Wallensteinplatz 2 8000 München 40

digital

DIGITAL EQUIPMENT GMBH

Mr. Robert Hassinger
c/o DECUS
129 Parker Street, PK-3/E55
Maynard, MA 01754

Wallensteinplatz 2
8000 München 40
Telefon (089) 35031
Telex 05215780

Dept. R605

Ihre Zeichen	Ihre Nachricht vom	Unsere Zeichen	Durchwahl	Datum
			3503-	

Dear Bob:

Reading Jon von Zelowitz 'self erasing core zeroer' in the 12 bit SIG newsletter No. 28 reminded me of the times when we had the PDP-8 sense switch, the deposit HALT in all of memory and the zero memory contests (among others). This was in those old days when the DECUSCOPE used to be a colorful marketplace for PDP-5/8 users, around 1967.
May I recall some of these ventures for the delight of todays users?

1. A refined German-Wolfberg technique:

```
ØØØØ  3ØØØ
2772  SKP
2773  DCA .+3
2774  ISZ Ø     /start, AC=Ø
2775  DCA I Ø
2776  JMP .-4
2777  DCA .-3
3ØØØ  DCA I Ø
```

2. This speed record can be beaten , however, by a basically different technique which is more than twice as fast:

```
7771  DCA I .+5  /start, AC=Ø
7772  ISZ 11
7773  ISZ .+3
7774  JMP .-3
7775  DCA .-1
7776  ØØØ1
7777  JMS I 11
ØØØØ  DCA I 11
```

3. Another one with less than 8 instructions:

```
TAD 5      / start at Loc ØØØ4
DCA I 1Ø
JMP 4
JMP I 4
11
ISZ 1Ø
```

Sincerely yours,

Rudi Stange

Royal Melbourne Institute of Technology LIMITED

124 LA TROBE STREET, MELBOURNE, VIC. 3000
BOX No. 2476V G.P.O. MELBOURNE, VIC. 3001
Telegraphic Address: "Meltech" Melbourne.
Telephone: 347 7611 Extension No. 274
In reply please quote

Letters should be addressed to the Principal.

R.M.I.T. Technical College,
Applied Science & Mathematics Division,
80-92 Victoria Street,
SOUTH CARLTON, 3053.

Robert Hassinger,
Co-ordinator - 12 BIT S.I.G.,
C/o. D.E.C.U.S.
129 Parker Street, PK-3/E55,
MAYNARD, MA 01754
U.S.A.

27th July, 1978.

Dear Bob,

The Applied Science and Mathematics Division of R.M.I.T. is currently using EDUSYSTEM 25 Version 2, to teach it's students BASIC in an interactive environment.

At the moment we cannot upgrade our software to Version 3 (which would allow us to use our mark-sense card reader) because D.E. (Aust.) tell us that D.E.C. have withdrawn all EDUSYSTEMS from sale.

We think that this is in a poor state of affairs and wonder if you or your members know whether D.E.C. is intending to upgrade these software items or replace them with any similar timesharing systems (preferably device-configuration independent)?

Failing this, is it known whether these or similar products are sold under license by O.E.M's or software houses?

Finally, we would be very interested in corresponding with other 12 BIT educational timesharing users, on matters of common interest.

Yours sincerely,

A. McClaren.

(A. McCLAREN).

SPERRY UNIVAC

P.O. BOX 12056
10 PARK PLAZA
RESEARCH TRIANGLE PARK, NORTH CAROLINA 27709

August 10, 1978

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

Dear Bob,

Here is a concise summary of what I've learned working with the fixed-point mode of the FPP-12, along with some illustrations and a listing of still another test program. As you can see, the test program is a bit specialized, but demonstrates the real problems involved and how to control them. I also discovered a special averaging situation where floating-point mode loses precision merely due to normalization problems in addition. I doubt hardware FPPs, at least the cheaper ones, could be made intelligent enough to address the problem, but anyone designing a software floating-point package could easily code a solution that would probably be faster than the software equivalent of the hardware technique and more precise to boot.

Since there is little terribly new here, please edit out what won't fit, since it will use up a lot of room and there are few FPP users out there.

I would also like to make a couple of retractions (both #28, p. 35)

1.) ALN does work correctly if given 2's complement left shift values; it requires 2's complement values for a correct left shift.

2.) In fixed-point FPP division for averaging, the divisor is not put into the MSW of the FAC fraction, but put there and shifted right one bit. This will then result in a numerically correct 2's complement 12-bit quotient in the MSW. The reason it must be done this way is that the binary point is not at the edge of the fraction, but shifted right one bit.

Finally, a question. I always avoid SQUISHing SYS under BATCH either from CCL or PIP. If BATCH is running with input from a file on SYS, is it possible to SQUISH SYS? I seem to remember horror stories about SQUISHing SYS and dead indexes and exploding OS/8. What is the current legality (and functionality) of a BATCHed SQUISH of SYS?

Sincerely,

Brian C. Converse - SLR

Brian C. Converse
Associate Programmer Scientific

BCC/slr
Enclosures

FIXED-POINT CALCULATIONS ON THE FPP-12

Arithmetic operations that are straightforward in floating-point mode (such as divide) now require some thought. In the FPP-12 User's Manual, DEC points out that it is harder to maintain precision since no normalization is performed in fixed-point mode (the user is presumably aware that fixed-point mode has a limited dynamic range compared to floating-point). Certainly it is frustrating and (happily) hard to justify complicated jumbles of 4-function arithmetic in fixed-point. However, fixed-point mode uses 1/3 less core and runs marginally faster; it is an attractive alternative for applications such as histograms, pulse counting, and signal averaging. In either mode, the FPP-12 takes care of rounding (it would be nice to be able to turn that OFF at times!) and signs, items that eat up time in double-precision software routines with or without EAE.

In fixed-point mode, all values are treated as fractions, so any arithmetic operation generating a non-fractional result causes the FPP to exit. Also, in fixed-point mode, the instructions will not bother the exponent. If one must float a fixed-point number, this situation can be used to advantage. (See below)

Converting single-precision signed integers to double-precision signed fractions may be done via FLDA and ALN. Each FLDA will load two integers into the FAC fraction. Using ALN, the more significant 12 bits of the FAC fraction can then be moved into the least significant 12 bits, leaving an extended sign in the most significant 12 bits. The preferred ALN index register value is 14 (decimal 12). PDP-12 users note that one's complement integers, such as ADC values, must be converted to two's complement before they are subjected to any FPP arithmetic.

Once a number is available as a fixed-point fraction, it may be converted to a floating-point value. XTA is the fastest, most straightforward way to do this when starting from single-precision integers (again, check one's complement data), but the following technique may be used to float a fixed-point sum or histogram bin or pulse counter after it has been used for awhile to allow subsequent, more complicated calculations to be done in floating-point mode. The prime accessory required for the conversion is a dummy floating-point constant. Its exponent must be 27 (23 decimal); the fractional portion may be anything, including any "illegal" value, but zero is preferred. The program should switch to floating-point mode and load this constant, via FLDA, into the FAC. The program then returns to fixed-point mode and loads, via FLDA, the value to be converted. In fixed-point mode, the exponent will not be disturbed and remains 27. The program then switches back to floating-point mode and executes an FNORM to produce a floating-point value. This value will be "real" if the fixed-point fraction was derived as explained in the previous paragraph or was produced by combining fractions so derived in a correct manner.

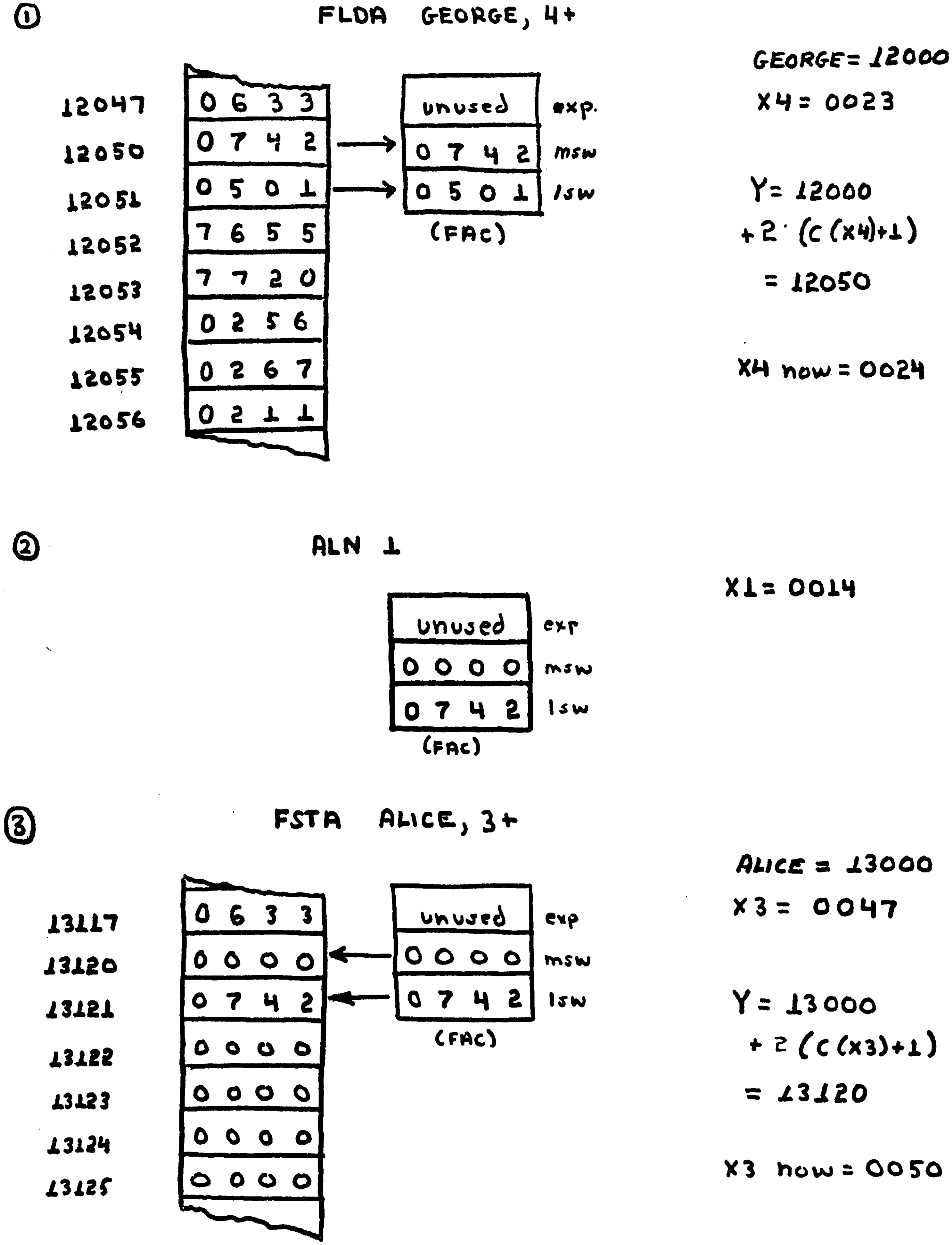
A test program was developed to help understand the use of the FPP-12 for signal averaging purposes. The OS/12 (OS/8) core image is built from a LINC mode segment and a FPP segment. The LINC code simply generates some "fake" data, starts the FPP, and throws some status information into the AC and MQ. A PDP-8 user with FPP-12 (and probable an 8A user with FPP) could write duplicate PDP-8 code in about thirty minutes; the FPP code should be universal.

The test program starts at 4020 and halts. The user sets an "N" value in the left switch register of the PDP-12 and some signed constant in the right switch register. The program uses four buffers of 512 words; a data buffer beginning at loc.0, a sum buffer beginning at loc. 10000, an average buffer beginning at 12000, and a baseline-corrected average buffer beginning at 14000. When the user hits CONTINUE, the LINC code portion of the program eats the two switch register values, stores "N" in a location known to both program segments (setting up such locations is a good way to develop an interest in MACREL or RALF!), and fills the data buffer with the signed constant. Continuing, the LINC code clears the sum, average, and corrected average buffers and starts the FPP-12 with the CPU locked out. Once the FPP is done, the LINC code puts the FPP status in the MQ and the FPC in the AC.

The FPP code sums the data buffer into the sum buffer N times, then divides the final sum by N and puts the result into the average buffer. This result should be the same signed constant as entered in the MSW of the fraction (the switch register input is assumed to be 1's complement while the FPP values are 2's complement). Next, an average of the 512 sums is computed. This calculation can involve a sum which overflows in fixed point, so it is done in floating point. This avoids the overflow problem, but in turn presents a significance problem. The *FPP-12 User's Manual* (Section 3.8.7) states, " In order to add or subtract two-floating-point numbers, the exponents must be aligned; that is, the fractional part of the number with the smallest exponent must be shifted right and the exponent incremented until the two exponents are equal." If a program is averaging numbers such that the sum gets very large, then even with four extra bits of precision, the smaller values being summed get scaled away. Eventually, the sum divided by N is not quite what it would be if infinite precision were available. [As R. K. Richards puts it in *Digital Design*, "...the problem of determining the number of digits that are truly significant in the final results, is not solved at all through the use of normalized floating-point notation." (p. 370) Software and firmware floating-point packages, since they cannot match the raw speed of a hardware FPP, might adopt a more intelligent approach to addition and subtraction: de-normalizing the larger value until it contains no *trailing* zeroes.] The "mean sum" so calculated should equal the value of the individual sum values, since the program is working with constants, but for moderate to large constants and N's, it does not due to the aforementioned precision problem. The average is calculated again, using the sum minus the "mean sum" and placed in the corrected average buffer.

The results are: the signed constant in each location of the data buffer, N times the signed constant in each location of the sum buffer (fixed-point "fraction"), the signed constant in the even locations of the average buffer, and zero or -1 in the even locations of the corrected average (and zero or garbage in the odd locations depending upon whether or not there was a precision problem). Any precision problems can be verified by comparing the fixed-point mean sum, MEANLC, with any sum buffer value.

FIG. 1: CONVERTING SINGLE-PRECISION -> D.P. FIXED-PT.



GEORGE = 12000
X4 = 0023

Y = 12000
+2 (C(X4)+1)
= 12050

X4 now = 0024

X1 = 0014

ALICE = 13000
X3 = 0047

Y = 13000
+ 2 (C(X3)+1)
= 13120

X3 now = 0050

FLDA GEORGE+1,4

12047	0633			
12050	0742		unused	exp
12051	0501	→	0501	msw
12052	7655	→	7655	lsw
12053	7720		(FAC)	
12054	0256			
12055	0267			
12056	0211			

GEORGE=12000

X4= 0024

Y= 12001

+ 2 (c(x4))

= 12051

ALN 1

X1= 0014

unused	exp
0000	msw
0501	lsw
(FAC)	

FSTA ALICE, 3+

13117	0633			
13120	0000			
13121	0742		unused	exp
13122	0000	←	0000	msw
13123	0501	←	0501	lsw
13124	0000		(FAC)	
13125	0000			

ALICE = 13000

X3= 0050

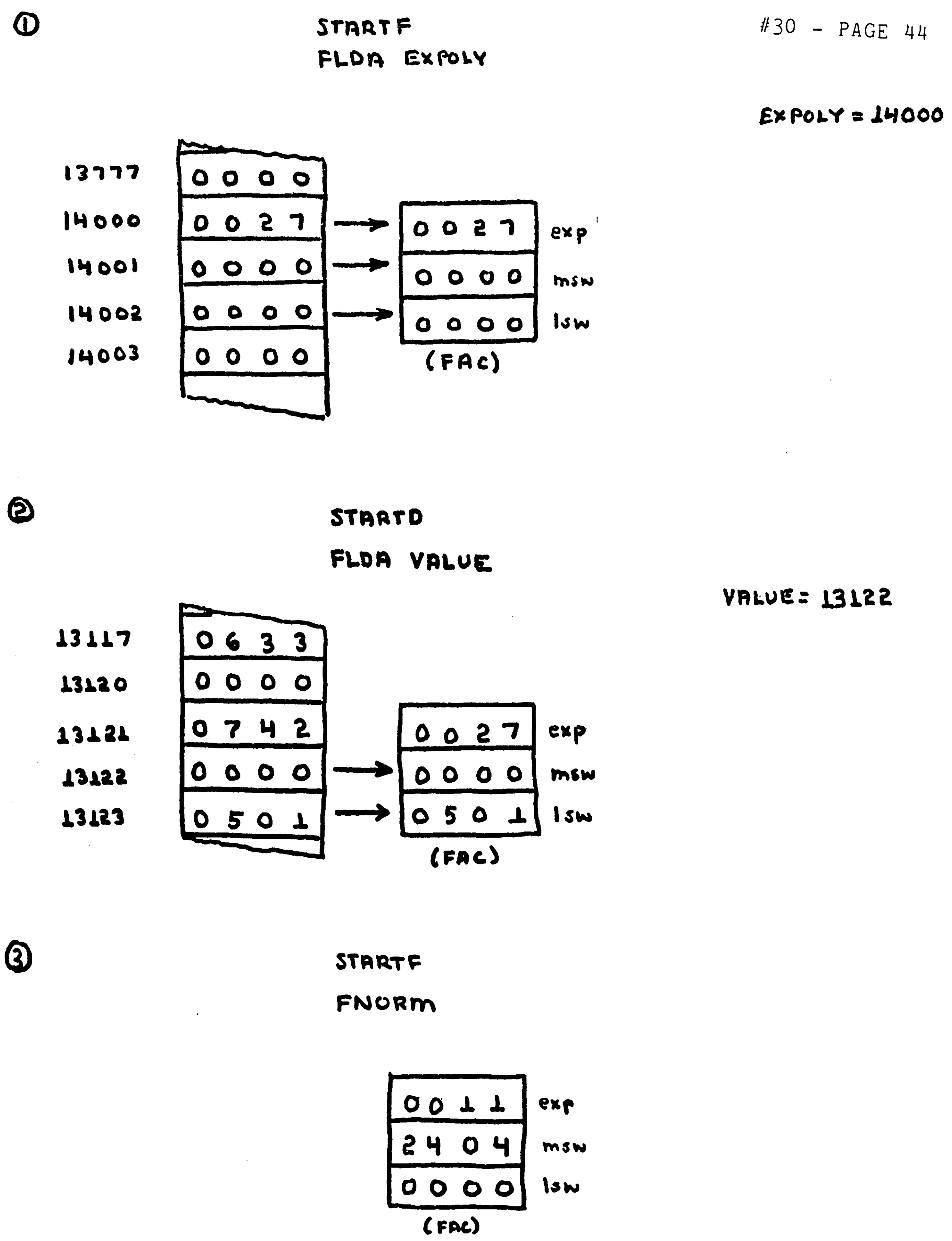
Y= 13000

+2 (c (x3)+1)

= 13122

X3 now = 0051

FIG. 2: CONVERTING DOUBLE-PRECISION FIXED-POINT TO FLOATING POINT



①

```
STARTF
FLDA EXPOLY
```

EXPOLY = 14000

13777	0000
14000	0027
14001	0000
14002	0000
14003	0000

0027	exp
0000	msw
0000	lsw

(FAC)

②

```
STARTD
FLDA VALUE
```

VALUE = 13122

13117	0633
13120	0000
13121	0742
13122	0000
13123	0501

0027	exp
0000	msw
0501	lsw

(FAC)

③

```
STARTF
FNORM
```

0011	exp
2404	msw
0000	lsw

(FAC)

```
                /TEST PROGRAM FOR FPP ALGORITHMS
                /TO CHECK THE PROPERNESS OF THE RESULTS WITH
                /FIXED POINT MODE ON FPP
                /
                /THIS PROGRAM DOES THE FOLLOWING:
                /    1. SETS UP A 512-WORD FIXED SIZE FAKE DATA BUFFER
                /    2. STARTS THE FLOATING-POINT SUMMATION/AVERAGE
                /    3. STOPS SO USER CAN EXAMINE RESULTS AND/OR RESTART
                /
                /PROGRAM STARTS AT LOC 4020- START BY I/O PRESET IN
                /    LINC MODE; START 20...OR LOAD FROM OS/12 (STARTS
                /    AT 4017 FOR YOU)...PROGRAM WILL HALT
                /USER ENTERS DESIRED VALUES INTO LSW AND RSW AFTER HALT.
                /LSW- NUMBER OF SUMMATIONS DESIRED BEFORE AVERAGING
                /RSW- VALUE TO BE PUT IN EACH POINT OF RAW DATA BUFFER
                /
                /HALTS WHEN DONE WITH FPP STATUS IN MQ AND FPC IN AC
                /
04017           *4017                       /FOR OS/12 ENTRY
04017   6141            LINC;LMODE
04020   0000            HLT                 /FOR USER START FROM CONSOLE
04021   0517            LSW                 /GET # TIMES TO SUM FAKE DATA
04022   6200            JMP NTRIAL-1        /LEAVE SPACE SO NTRIAL WILL BE
04023   0016            NOP                 /FIXED EVEN THOUGH PGM IS ADDED
04024   0016            NOP                 /TO IN HERE
04200           *4200
04200   1060            STA I               /SAVE FOR FPP USE
04201   0000    NTRIAL, 0

04202   0071            SET I 11            /COUNT POINTS
04203   6777            -1000
04204   0070            SET I 10            /START OF BUFFER
04205   3777            3777
04206   0640            LDF 0               /BUFFER BANK
04207   0516            RSW                 /GET USER'S DESIRED VALUE

04210   0441    LOOP,   SNS 1               /IF THIS SW UP, ALLOW A SQ.
04211   6216            JMP .+5             /WAVE TO BE BUILT USING + AND
04212   1520            SRO I               /- VALUES OF THE DATA EVERY 6
04213   3737            3737                /POINTS. FLIPS POLARITY OF
04214   0467            SKP                 /FAKE DATA EVERY 6 POINTS...
04215   0017            COM
04216   1070            STA I 10            /WRITE INTO BUFFER
04217   0231            XSK I 11            /COUNT POINTS
04220   6210            JMP LOOP            /NOT DONE


04221   0644            LDF 4
04222   6275            JMP CLRBUF
04223   0645            LDF 5
04224   6275            JMP CLRBUF
04225   0646            LDF 6
04226   6275            JMP CLRBUF
```

```
04227  1020            LDA I           /DONE, START FPP
04230  0020            20
04231  0004            ESF             /DO AN I/O PRESET

04232  1020            LDA I           /NOW FIX UP APTLOC
04233  5000            5000
04234  4501            STC APTLOC+1
04235  1020            LDA I;PMODE
04236  4510            FPINDX;LMODE
04237  4502            STC APTLOC+2
04240  4500            STC APTLOC
04241  4503            STC APTLOC+3
04242  4504            STC APTLOC+4
04243  4505            STC APTLOC+5
04244  4506            STC APTLOC+6
04245  4507            STC APTLOC+7

04246  1020            LDA I           /SET UP COMMAND REG
04247  4010            4010
04250  0500            IOB
04251  6553            FPCOM
04252  1020            LDA I;PMODE
04253  4500            APTLOC;LMODE
04254  0500            IOB             /START FPP; PDP-12 IS LOCKED OUT
04255  6555            FPST
04256  6272            JMP PROBLM      /THIS IS IF THERE'S A PROBLEM
04257  0500            IOB
04260  6557            FPIST
04261  6257            JMP .-2
04262  0354            SCR 14          /->MQ
04263  1000            LDA             /NORMAL RETURN
04264  0501            APTLOC+1
04265  0000            HLT
04266  0000            HLT
04267  0000            HLT
04270  6270            JMP .
04271  0000            HLT
04272  0011    PROBLM, CLR
04273  0354            SCR 14
04274  6265            JMP .-7


04275  0070    CLRBUF, SET I 10        /LITTLE ROUTINE TO CLEAR BUFFERS
04276  3777            3777
04277  0071            SET I 11
04300  6000            -1777
04301  0011            CLR
04302  2000            ADD 0
04303  4307            STC CLREXT
04304  1070            STA I 10
04305  0231            XSK I 11
04306  6304            JMP .-2
04307  6307    CLREXT, JMP .
```

```
/TEST PROGRAM FOR FPP ALGORITHM  PAL12-V5  8/3/70     PAGE 1-2


04500            *4500
04500  0000  APTLOC, 0
04501  5000          5000
04502  0510          FPINDX
04503  0000          0
04504  0000          0
04505  0000          0
04506  0000          0
04507  0000          0

04510  0000  FPINDX, 0
04511  0000          0
04512  0000          0
04513  0000          0
04514  0000          0
04515  0000          0
04516  0000          0
04517  0000          0

       6553          FPCOM=6553
       6555          FPST=6555
       6557          FPIST=6557
                     $
```

```
/TEST PROGRAM FOR FPP ALGORITHM  PAL12-V5  8/3/70     PAGE 1-3


APTLOC 4500
CLRBUF 4275
CLREXT 4307
FPCOM  6553
FPINDX 4510
FPIST  6557
FPST   6555
LOOP   4210
NTRIAL 4201
PROBLM 4272
```

```
                /FPP-12 PART OF TEST PROGRAM FOR FPP ALGORITTHM
                /STARTS AT LOC 5000
                /DOES N SUMS OF FAKE DATA BUFFER INTO BUFFER
                /STARTING @ LOC 10000; 512 POINTS.  USES VALUE
                /FOUND IN NTRIAL FOR # SUMS TO DO. USES VALUE
                /IN NDXSET TO DECIDE WHAT TO SET INDEX REG 2 TO.
                /DOES MEAN ON SUM B4 AVERAGE; STORES MEAN IN
                /MEANLC; WRITES RAW AVERAGE INTO BUFFER STARTING
                /@12000; WRITES AVERAGE WITH SUBTRACTED MEAN INTO
                /BUFFER STARTING @14000.
                /
                /
                        ORG 5000
05000 0006      START,  STARTD
05001 1100              SETX FPINDX
05002 4510
05003 1120              JSA COMNDX
05004 5202


05005 0400              FLDA NTRIAL     /GET # FAKE TRIALS TO SUM
05006 4201
05007 1040              JNE CONTNU      /IF ZERO, GET OUT!!!!
05010 5012
05011 0000              FEXIT
05012 0011      CONTNU, ALN 1           /GET # IN LSW OF FAC
05013 6400              FSTA INPUTN     /SAVE AS N VALUE FOR CALAVG
05014 5245
05015 0003              FNEG            /MAKE IT -
05016 6400              FSTA NEGTRL     /SAVE IT FOR LATER
05017 5227
05020 0024              ATX 4           /4 COUNTS SUMMATIONS
05021 0400              FLDA INPUTN     /NEED N IN MSW TO DO DIVISION
05022 5245
05023 0015              ALN 5           /FOR AVERAGE
05024 6400              FSTA AVGNVL
05025 5243


05026 1120              JSA COMNDX
05027 5202
05030 0530      FIRSUM, FLDA DATBUF,3+  /GET A PAIR
05031 0000
05032 0011              ALN 1           /STRIP OUT LSW; MSW->LSW
05033 1120              JSA KOREKT      /1'S->2'S COMPLEMENT CONV.
05034 5170
05035 6521              FSTA SUMBUF,2+  /SUM VALUE INTO SUM BUFFER
05036 0000
05037 0430              FLDA DATBUF+1,3 /GET OFFSET PAIR
05040 0001
05041 0011              ALN 1
05042 1120              JSA KOREKT
05043 5170
05044 6521              FSTA SUMBUF,2+
05045 0000
05046 2100              JXN FIRSUM,0+
05047 5030
05050 2140          JXN LOPSET,4+
05051 5054
05052 1030          JA MENCAL        /HERE ONLY IF # SUMS=1!!!
05053 5100


              /NOW DO A LOOP TO DO THE REMAINING (IF ANY) SUMMATIONS
              /OF THE DATA BUFFER
              /
05054 1120    LOPSET, JSA COMNDX
05055 5202
05056 0530    SUMLOP, FLDA DATBUF,3+  /NOW LOOP FOR REMAINING TRIALS TO
05057 0000
05060 0011          ALN 1            /BE SUMMED; USE FADDM STATT FLDA!
05061 1120          JSA KOREKT       /1'S->2'S COMPLEMENT CONV.
05062 5170
05063 5521          FADDM SUMBUF,2+ /SUM VALUE INTO SUM BUFFER
05064 0000
05065 0430          FLDA DATBUF+1,3 /GET OFFSET PAIR
05066 0001
05067 0011          ALN 1
05070 1120          JSA KOREKT
05071 5170
05072 5521          FADDM SUMBUF,2+
05073 0000
05074 2100          JXN SUMLOP,0+
05075 5056
05076 2140          JXN LOPSET,4+
05077 5054


05100 1120    MENCAL, JSA COMNDX       /CALCULATE THE MEAN SUM VALUE
05101 5202
05102 0100          LDX -777,0
05103 7001
05104 0005          STARTF           /->FL. PT. MODE
05105 0400          FLDA EXPOLY      /GET 27 OCTAL IN EXPONENT OF AC
05106 5236
05107 0006          STARTD           /BAK TO FIXED PT
05110 0521          FLDA SUMBUF,2+
05111 0000
05112 0005          STARTF           /FL. PT. MODE
05113 0004          FNORM            /MAKE IT NORMAL!
05114 6400          FSTA MEANSM      /WHY NO ABNORMALIZE INSTRUCTION?
05115 5224
05116 0400          FLDA EXPOLY      /GET EXPONENT BACK
05117 5236
05Y20 0006    MEANLP, STARTD           /...BECAUSE PGMS. SEEM TO DO THAT AUTOMATIC
05121 0521          FLDA SUMBUF,2+   /LOOP TO DO THE REMAINDER NOW THAT
05122 0000
05123 0005          STARTF           /ANY RESIDUE IN MEANSM OVERWRITTEN
05124 0004          FNORM            /BOY, DOES THIS BURN UP TIME!
05125 5400          FADDM MEANSM     /NOW ACCUMULATE SUM
05126 5224
05127 0400.         FLDA EXPOLY
```

```
05130 5236
05131 2100          JXN MEANLP,0+
05132 5120
05133 0400          FLDA MEANSM     /SAVE SUM FOR INSPECTION
05134 5224
05135 3400          FDIV CF1000     /DIVIDE BY 1000
05136 5233
05137 0010          ALN 0           /FIX IT BACK TO DP
05140 0006          STARTD          /...AND GO BAK TO DP
05141 6400          FSTA MEANLC     /SAVE FOR USE IN MEAN SUBTRACTION
05142 5222


05143 1120          JSA COMNDX
05144 5202
05145 0100          LDX -1000,0     /COUNT FOR AVERAGE
05146 7000
05147 0521  AVGLOP, FLDA SUMBUF,2+  /GET A SUM
05150 0000
05151 3400          FDIV AVGNVL     /DIVIDE BY N
05152 5243
05153 6531          FSTA AVGBUF,3+  /SAVE IN AVERAGE BUFFER
05154 2000
05155 0421          FLDA SUMBUF,2   /GET THAT ONE AGAIN
05156 0000
05157 2400          FSUB MEANLC     /NOW SUBTRACT MEAN
05160 5222
05161 3400          FDIV AVGNVL     /DIVDE BY N
05162 5243
05163 6431          FSTA CORAVG,3   /SAVE IN "CORRECTED" AVERAGE BUFFER
05164 4000
05165 2100          JXN AVGLOP,0+   /REPEAT FOR ALL POINTS
05166 5147
05167 0000  FINISH, FEXIT


05170 0000  KOREKT, 0;0
05171 0000
05172 1050          JLT BIASNG
05173 5176
05174 1030          JA KOREKT       /# IS +
05175 5170
05176 1400  BIASNG, FADD SPRONE     /# IS -; ADD 1
05177 5241
05200 1030          JA KOREKT
05201 5170


05202 0000  COMNDX, 0;0
05203 0000
05204 0100          LDX -400,0      /THIS IS MOST-USED VALUE FOUND
05205 7400
05206 0101          LDX 14,1        /ALIGNMENT CONSTANT
05207 0014
05210 0102          LDX -1,2
05211 7777
```

```
FLAP V  50  AUG  3, 70 PAGE 1-3

05212 0103          LDX -1,3
05213 7777
               /NOTE NOTE NOTE DO NOT SET 4 INSIDE THIS SUBROUTINE!!!!!!!
05214 0105          LDX -13,5        /USE THIS ONLY FOR DIVISORS!
05215 7765
05216 0107          LDX -1,7
05217 7777
05220 1030          JA COMNDX
05221 5202


05222 0000     MEANLC, 0;0                 /MEAN SUM
05223 0000
05224 0000     MEANSM, 0;0;0               /SUM OF THE 512 SUMS- FLOATING POINT
05225 0000
05226 0000
05227 0000     NEGTRL, 0;0                 /-N FOR COUNTING SUMMATIONS
05230 0000
               /
               /***************************NOTE!***************************
05231 0400     C1000,  400;0               /OCTAL 1000=DEC. 512 IN MSW FOR DIVISIONS
05232 0000
                                           /NOTE IT IS ROTATED RIGHT IN MSW 1 BIT!!
05233 0012     CF1000, F 512.0             /FOR FLOATING POINT USE
05234 2000
05235 0000
05236 0027     EXPOLY, 27;0;0              /JUST A NICE EXPONENT FOR FL. PT. ADJUSTS.
05237 0000
05240 0000
05241 0000     SPRONE, 0;1                 /SINGLE PRECISION "1" FOR CONVERSIONS
05242 0001
05243 0000     AVGNVL, 0;0                 /N VALUE IN MSW
05244 0000
05245 0000     INPUTN, 0;0                 /N VALUE IN LSW
05246 0000


                    DATBUF=0
                    SUMBUF=10000
                    AVGBUF=12000
                    CORAVG=14000
                    NTRIAL=4201
                    FPINDX=4510


FLAP V  50  AUG  3, 70 PAGE 1-4

NO ERRORS
27 SYMBOLS, NO LINKS

 AVGBUF 12000    AVGLOP 05147    AVGNVL 05243    BIASNG 05176
 CF1000 05233    COMNDX 05202    CONTNU 05012    CORAVG 14000
 C1000  05231    DATBUF 00000    EXPOLY 05236    FINISH 05167
 FIRSUM 05030    FPINDX 04510    INPUTN 05245    KOREKT 05170
 LOPSET 05054    MEANLC 05222    MEANLP 05120    MEANSM 05224
 ↑C
.R PAL12
PAL12-VØ6
*TESTLN,TESTLN<TESTLN.12
```

/. 12 extention eases editing via SCROLL

```
.R FLAP
*TESTFP,TESTFP<TESTFP.12


.LO TESTLN,TESTFP=4017
```

/∴ HLTs when loaded by OS/8 or after I/O Preset, START 20 on PDP-12 (the only reason things start @ 4020 is restart convenience on PDP-12)

```
.SAV SYS TESTLN

.
```

```
.SUB GLOPTER/T

$JOB ERASE AFTER 8/3


.R FLAP
*FPAVRG,FPAVRG<FPAVRG.12

.R FLAP
*,TESTFP<LTAØ:TESTFP.12

.R PAL12
PAL12-VØ6
*ONLDGR,ONLDGR<WORKD/S
ASSEMBLY OF ON-LINE DIGITIZING PROGRAM>
ASSEMBLY OF ON-LINE DIGITIZING PROGRAM>
.R PAL12
PAL12-VØ6
*,TESTLN<LTAØ:TESTLN.12
.R CREF
*ONLDGR
.R PIP
*LPT:<FPAVRG.LS/A
$MSG TURN PAPER AROUND!
.R PIP
*LPT:<TESTLN.LS/A
*LPT:<TESTFP.LS/A
*FPAVRG.LS,ONLDGR.LS,TESTLN.LS/D<
*TESTFP.LS/D<
.R DIRECT
*TTY:<SYS:/E/B/=2
#ILLEGAL SYNTAX
$END

#END BATCH
```

← DOSEN'T WORK

Bob - I can't find any reference to this DIRECT problem in the newsletter - has it been reported? Am I doing something dumb?

Brian Converse

BATCH JOB

```
.R DIRECT
*TTY:<SYS:/E/B=2

 Ø3-AUG-78

ABSLDR.SV↑C
```

← WORKS...

THE ROCKEFELLER UNIVERSITY

1230 YORK AVENUE · NEW YORK, NEW YORK 10021

June 29, 1978

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

Dear Mr. Palmer:

I am writing to you about DECUS program number 8-690. As you may recall, it is a random number generator which can be called from OS/8 FORTRAN IV. I cannot get it to work correctly in a standard configuration, and I thought that you might have some idea why it is giving trouble.

Briefly, the subroutine produces non-random numbers when used with an Extended Arithmetic Element (KE8-E), but seems to produce a nice flat random distribution when the EAE is disabled. If you have come across this problem, or can think of a reason for it, I would very much appreciate hearing from you. If not, perhaps you would be good enough to forward this letter along to the ~~No. 12~~ 12-BIT SIG Newsletter in order to inform others of the apparent problem.

To be more specific, I have a PDP-8/F with 24k words of core memory and an EAE. Other FORTRAN IV programs seem to run perfectly well whether or not the EAE is disabled as described in the FORTRAN IV software support manual. But when a test program for subroutine RANDU runs with EAE, it produces no numbers in the range 0.02 to 0.33, about twice as many numbers as it should in the range between 0.34 and 0.65, and approximately the correct frequency of numbers in the range 0.66 to 1.00. Attached is my little test program of RANDU and some output from it with the EAE in and out of the system.

The version of FRTS which I am using is version 4C, with a patch included to run the Phelps USR routines.

I should mention that the EAE diagnostics seem to run perfectly well on my machine.

I hope that someone with more knowledge than I have concerning the Floating Point Processor conventions and the FORTRAN IV Run Time System can find the solution to this problem and allow me to run your extremely valuable random number generator.

Sincerely yours,

Ron Lark

Ronald P. Larkin
Assistant Professor

P.S. My RALF listing is identical to the one in the writeup to 8-690.

RP?/rbk

Hässle

AB Hässle Subsidiary of Astra Pharmaceuticals AB

Date	Our reference
Your date	Your reference

Ronald Larkin
Rockefeller university
1230 York ave
New York
NY 10021
USA

Dear sir

I have looked a bit at your problem. I ran it in the FPP configuration and got exactly the same result as you did in non EAE systems.I do not have an EAE so I cannot test that but I think that what we have here is one further case of the now too frequen errors in FRTS where the FPP emulator does not behave the same way as the hardware.

I do not heve the time to try to trace the error by reading the FRTS list but I suspect there is something wrong in the usage of the EAE (in association with the TRAP ?).

My attitude is that the hardware (FPP12 or 8A)should be the reference and any divergences in FRTS suftware rutines(I know of at least two one which mede the early veriosns of R.Phelps USR rutine misbehave in an fpp configutations)be regarded as errors.
I´m sorry I cannot help you more then this,I shall forward the material to Bob Hassinger and hope that someone can help you.

Lars Palmer

Hässle

AB Hässle Subsidiary of Astra Pharmaceuticals AB

Date

Our reference

Your date

Your reference

Dan smith
Eye research centre
20 stanifor street
Boston mass 02114
USA

re EXPIP

Expip should handle all dates correctly but due to a programming bug it doe not.Patch the following and it will:

locations 47 54 61 70 and 126 are cma(7040) should be cia(7041)
location 63 7450/7540
location 72 7550/7510

I will look at the C problem. It is really 2 parts to it : one that Expip resets the date to get the correct date on the new file and that I think is solvable, the other is where EXPIP does a FOTP like transfer and writes a dictionary before transfer(/W option). this is more difficult so solve and I probably will not have the time to do so

I use EXPIP mainly with the /L option to find the most recent file whe using slo media(dectape) as FOTP transfers are much faster(even htan the /W form) Transferring between disks there is no need for the faster form and the problem does not occur on ordinary transfers.

I find that the options that I use EXPIP for now (that FOTP has come) are:
1)/L comparisitions to see were the most resent file is
2)transfer and delete after transfer
3)merge files (is much faster to construct a 20 rutine Macrel library
than using PIP 3 times)
4)recovering lost blocks or rather files as a compliment to the STECO
and pip /I methods.

The real diadvantage with EXPIP as I see it compared to FOTP is the weaker decoding of wildcards and That I have no time to fix.

I hope this helps you a bit

/lars Palmer/

cc to bob hassinger

digital **SOFTWARE PERFORMANCE REPORT**

FIELD NO.:	SPR NO.:	151934
SOFTWARE SERVICES NO.:		PAGE 1 OF 1

PERATING SYSTEM	VERSION	SYSTEM PROGRAM OR DOCUMENT TITLE	VERSION OR DOCUMENT PART NO.	DATE
OS/8	V3D	EDIT	V12B	1-JUN-75

EE EXAMPLE IN INSTRUCTIONS)

AME: ALISTAIR WINDRAM

RM: THE GRASSLANDS RESEARCH INSTITUTE

DDRESS: HURLEY, MAIDENHEAD, BERKS.
U.K.
ZIP:

UBMITTED BY: A. WINDRAM
PHONE:

DEC OFFICE: READING

DO YOU HAVE SOURCES? HAVE GOT ACCESS YES □ NO ☑

REPORT TYPE
☑ SOFTWARE ERROR
□ DOCUMENTATION ERROR
□ INQUIRY
□ FOR YOUR INFORMATION/SUGGESTION

PRIORITY
□ LOW
□ STANDARD
☑ HIGH

CAN THE PROBLEM BE REPRODUCED AT WILL? YES ☑ NO □

ATTACHMENTS
AG TAPE □ FLOPPY DISKS □ LISTING □
DECTAPE □ OTHER

COULD THIS SPR HAVE BEEN PREVENTED BY BETTER OR MORE DOCUMENTATION? PLEASE EXPLAIN IN PROVIDED SPACE BELOW. YES □ NO ☑

PU TYPE	SERIAL NO.	MEMORY SIZE	DISTRIBUTION MEDIUM	SYSTEM DEVICE	DO NOT PUBLISH
8E	2048	16K	DECTAPE	RK8E	□

The patch to EDIT V12A to cure lost Tabs brought to light a problem in the N command handler. Location COM1, -1 is altered by the N and Y commands, and is initially JMP PUNCH+3. None of this matters until you start jumping in to the middle of the N handler. Then :-

1) If you have never used N or Y, but have used something which left OUTDEV set to console, and you say Q, the output buffer gets sent to the console instead of to the output file !

```
e.g.    #R
        #/=
        #Q
```

This could be fixed by changing JMP PUNCH+3 to JMP PUNCH

```
        1300/5304   5301
        (N or Y)
```

2) If the last command before Q was Y, then when you type Q you just kill the buffer instead of outputting it. Thus you lose the last page! This is not so easy to fix.

I think the V12B patch needs a re-think ! One way is :-

```
        2014/1301   2774
        2774/xxxx   3112;5776;1301
```

The simple source change is to relocate the DCA TABIND from COM1, -2 to PUNC, -1

SOFTWARE PERFORMANCE REPORT

FIELD NO.:

SPR NO.:

SOFTWARE SERVICES NO.:

151935

PAGE 1 OF 1

OPERATING SYSTEM	VERSION	SYSTEM PROGRAM OR DOCUMENT TITLE	VERSION OR DOCUMENT PART NO.	DATE
OS/8	V3D	PIP	V11A	6-JUN-79

(SEE EXAMPLE IN INSTRUCTIONS)

NAME: ALISTAIR WINDRAM

FIRM: GRASSLAND RESEARCH INSTITUTE

ADDRESS: HURLEY, MAIDENHEAD, BERKS
U.K.
ZIP:

SUBMITTED BY: A. WINDRAM

PHONE: LITTLEWICK GREEN 3631

DEC OFFICE

DO YOU HAVE SOURCES? HAVE ACCESS YES □ NO ☑

REPORT TYPE

☑ SOFTWARE ERROR

□ DOCUMENTATION ERROR

□ INQUIRY

□ FOR YOUR INFORMATION/SUGGESTION

PRIORITY

□ LOW

□ STANDARD

☑ HIGH

CAN THE PROBLEM BE REPRODUCED AT WILL? YES ☑ NO □

ATTACHMENTS

MAG TAPE □ FLOPPY DISKS □ LISTING □ DECTAPE □ OTHER ✓

COULD THIS SPR HAVE BEEN PREVENTED BY BETTER OR MORE DOCUMENTATION? PLEASE EXPLAIN IN PROVIDED SPACE BELOW. YES □ NO ☑

CPU TYPE	SERIAL NO.	MEMORY SIZE	DISTRIBUTION MEDIUM	SYSTEM DEVICE	DO NOT PUBLISH
8E	2048	16K	DECTAPE	RK8E	□

```
.R PIP
*output file < input file /Y
```

always gives "BAD SYSTEM HEAD" if input file is on a device not containing a system.

The reason is that the file to file /Y code checks block 7 of the device instead of block 2 of the file.

One way of fixing the problem is to use the device < file /Y code to do the checking.

The source change is detailed on the enclosed sheet.

A possible patch is :-

```
1601/3243  3240
16 --
```

6

SOURCE CHANGES TO PIP V11A TO CURE *FILE<FILE/Y BUG

```
AT  K200, +2     CHANGE  JMP YTSOUT  TO  JMP YNOOUT
AT  YOUSYS, -4   RELOCATE  TAD I (7601; SZA CLA; JMP I (IMGTST
                 TO JUST BEFORE  YINREC, +6  (TAD YINREC)
AT  ERR11, +2 LINES     INSERT  IMGTST, TAD I (7620; SZA CLA; JMP IMGTS2
AT  IMGTST, -2   CHANGE  TAD K7  TO  TAD (7
AT  IMGTST,      CHANGE  IMGTST,  TO  IMGTS2,
AT  IMGTST, +3   DELETE 7 LINES  (CIF 0  TO  JMS I (TSTHED)
```

SOFTWARE
PERFORMANCE
REPORT

FIELD NO.:	SPR NO.:

SOFTWARE SERVICES NO.:

154131

PAGE 1 OF 1

PERATING SYSTEM	VERSION	SYSTEM PROGRAM OR DOCUMENT TITLE	VERSION OR DOCUMENT PART NO.	DATE
OS/8	V3D	PIP	V11A	9-JUN-78

E EXAMPLE IN INSTRUCTIONS)

AME: Alistair Windram
IRM: The Grassland Research Institute

DDRESS: Hurley, Maidenhead, Berks, U K

DEC OFFICE
Reading

DO YOU HAVE SOURCES? YES □
Have Access NO ☒

REPORT TYPE/PRIORITY

☒ PROBLEM ERROR
□ SUGGESTED ENHANCEMENT
□ OTHER

□ 5.
□ 4.
□ 3.
□ 2.
☒ 1.

UBMITTED BY: A.Windram
PHONE: Littlewick Green 3631

ATTACHMENTS
AG TAPE □ FLOPPY DISKS □ LISTING □
DECTAPE □ OTHER ____

CAN THE PROBLEM BE REPRODUCED AT WILL? YES ☒ NO □

COULD THIS SPR HAVE BEEN PREVENTED BY BETTER OR MORE DOCUMENTATION? PLEASE EXPLAIN IN PROVIDED SPACE BELOW. YES □ NO ☒

PU TYPE	SERIAL NO.	MEMORY SIZE	DISTRIBUTION MEDIUM	SYSTEM DEVICE	DO NOT PUBLISH □
8E	2048	16K	Dectape	RK8E	Please Publish

The modifications to the /Y code have introduced a serious bug.

The problem is that the /Y code now reads 14 blocks at a time instead of 13, into 00000 - 06777.
This overlays the input device handler which loads into 06600 - 07177.
Thus /Y transfers with non-resident input handlers blow up !!

06600 - 07577 is all needed for device handlers in case of *dev1<dev2/Y
where dev1 and dev2 have different non-resident 2-page handlers.

You must go back to 13 blocks per transfer as before, and find some other way to fix the problem of system-head files which are right at the end of a device.

Since there is another /Y bug, and the source fix for this actually frees some space, there may be room to combine the fixes in the area currently available.

I think a patch may be slightly more difficult !

215-985-7733

BICKLEY LABORATORIES
COMPUTER APPLICATIONS

47 Ivy Mills Road
RD2
Glen Mills, PA 19342

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

Dear Bob:

Here are a couple of patches for BASIC and one for CCL which I've submitted to DEC. Knowing the incredible delays that occur from the time an SPR is reperted to publication, I'm forwarding these to you so 12 Bit SIG members will have advantage of them much sooner.

By the way, Bob, you are doing a great job! Thanks ever so much.

Sincerely,

Lyle P. Bickley

57950

digital SOFTWARE PERFORMANCE REPORT

FIELD #: SPR #:

FOR DEC USE ONLY

Page 1 of 3

SYSTEM PROGRAM AND VERSION (OR DOCUMENT): CCL.PA

MONITOR AND VERSION: OS/8 V3D

DATE: 6-AUG-78

NAME: Lyle P. Bickley
FIRM: Bickley Labs

ADDRESS: 47 Ivy Mills Road
Glen Mills
PA
ZIP 19342

DEC OFFICE: Blue Bell, PA

REPORT TYPE
☒ LOGIC/CODING ERROR
☐ DOCUMENTATION ERROR
☐ SUGGESTION
☐ INQUIRY
☐ FOR YOUR INFORMATION

PRIORITY
☐ LOW
☐ STANDARD
☒ HIGH

SUBMITTED BY: Lyle P. Bickley PHONE: 215-985-7733

LIST ATTACHMENTS
(1) Source compare (SRCCOM) of changes

CAN THE PROBLEM BE REPRODUCED AT WILL? ☒ YES ☐ NO

CPU TYPE	SERIAL NO.	SYSTEM DEVICE	MEMORY SIZE	DISTRIBUTION MEDIUM
PDP-12	561	RF08	16K	DECTAPE

PROBLEM:
When an @file.CM is used with the semi-colon feature of CCL, unpredictable results and system crashes can occur.

DIAGNOSIS:
When @file is used and the command string is relatively long, the buffer used to contain the concatination of the @file with other command data can overlap the GLINE routine in the keyboard monitor. This created no problems with earlier releases of CCL because the GLINE routine was never again refenenced by CCL. With the advent of the semicolon option, however, this was changed. The semicolon routine stores a 7600 in GLINE and in the event of a trailing ';' in a command file, will actually referenee and use the GLINE routine. Both of these events cause errors to occur: in the first, because the data in the buffer is clobbered with a 7600; in the second, because GLINE may have become destroyed.

CURE:
These changes insure that GLINE is not modified by a 7600 store if GLINE has been overwritten by the @ buffer. In addition, if GLINE has been overwritten and a trailing ';' occurs in an @file, the error message "?I/O OR ';' ERROR" is displayed on the console and a return is made to the keyboard monitor.

```
SRCCOM V4A

1)      /CCL FOR DECSYSTEM 8    V3D
2)      /CCL FOR DECSYSTEM 8    V3D

1)003   /            JUL 29, 1978  FIX SEMICOLON PROCESSING ROUTINE
1)004   /         FORMAT OF CCL TABLE
****
2)004   /         FORMAT OF CCL TABLE

********

1)101   SEMSG1, TEXT    /?ENTER ERROR/
1)      SEMSG2, TEXT    \?I/O OR ';' ERROR\
1)      SEMSG3, TEXT    /?DEVICE FULL/
****
2)101   SEMSG1, TEXT    /? ENTER ERROR/
2)      SEMSG2, TEXT    \?I/O ERROR\
2)      SEMSG3, TEXT    /?DEVICE FULL/

********

1)108   BATAIL, ".;"R;240;"F;"O;"T;"P;215;212
1)              "*;"S;"Y;"S;":;"C;"C;"B;"T;"C;"H;".;"T;"M;"/;"D;215;212
1)              "$;"E;"N;"D
1)      BACKLF, 215;212;0
1)      TEMNAM, FILENAME CCBTCH.TM
****
2)108   BATAIL, ".;"R;240;"F;"O;"T;"P;215;212
2)              "*;"S;"Y;"S;":;"C;"C;"B;"T;"C;"H;".;"T;"M;"/;"D;"$;215;212
2)              "$;"E;"N;"D;215;212;32;0
2)      TEMNAM, FILENAME CCBTCH.TM

********

1)110           TAD I (GLINE+1
1)              TAD (-1163      /MAKE SURE GLINE NOT CLOBBERED
1)              SNA CLA         /BY '@' CODE
1)              JMP .+4         /ALL OK
1)              TAD BATBLK+1    /ERROR JUMP (IF @ AND TRAILING ;)
1)              DCA NEWLN-1     /INSTEAD OF CALL!
1)              JMP .+3
1)              TAD (7600
1)              DCA I (CTRLCK   /FORCE ^C IN GLINE TO GO TO 7600
1)              JMS BATLST
1)              BATHED
1)              JMP NEWLN       /INIT XR
1)      S2,     TAD I XR
****
2)110           TAD (BEGLN-1
2)              DCA XR
2)              TAD (7600
2)              DCA I (CTRLCK   /FORCE ^C TO GLINE TO GO TO 7600
2)              JMS BATLST
2)              BATHED
2)      S2,     TAD I XR

********

1)111   /****PREVIOUS LINE MAY BE JMP SEMER2 TO PREVENT USE OF
1)      /     'GLINE' AFTER IT HAS BEEN CLOBBERED BY AN @FILENAME
1)      /**** WHAT IF WE'RE RUNNING UNDER BATCH ****
1)      NEWLN,  TAD (BEGLN-1
1)              DCA XR
```

```
2)111   /**** WHAT IF WE'RE RUNNING UNDER BATCH ****
2)              TAD (BEGLN-1
2)              DCA XR
********
1)111           JMS BATLST
1)              BACRLF
1)              JMP I KRLF
****
2)111           TAD (215
2)              JMS BATPUT
2)              TAD (212
2)              JMS BATPUT
2)              JMP I KRLF

********
1)113   SEMER2, IAC     / I/O ERROR OR INVALID TRAILING ;
1)      SEMER1, IAC     /ENTER ERROR
****
2)113   SEMER2, IAC     / I/O ERROR
2)      SEMER1, IAC     /ENTER ERROR

********
```

57949

digital **SOFTWARE PERFORMANCE REPORT**

FIELD #:	SPR #:
FOR DEC USE ONLY	

Page 1 of 4

SYSTEM PROGRAM AND VERSION (OR DOCUMENT) BASIC.FF	MONITOR AND VERSION OS/8 Extension Kit	DATE 6-AUG-78

NAME: Lyle P. Bickley
FIRM: Bickley Labs

ADDRESS: 47 Ivy Mills Road
RD2
Glen Mills, PA
ZIP 19342

DEC OFFICE
Blue Bell, PA

SUBMITTED BY: Lyle P. Bickley
PHONE: 215-985-7733

LIST ATTACHMENTS
(1)patch, (2)sample program

REPORT TYPE
☒ LOGIC/CODING ERROR
□ DOCUMENTATION ERROR
□ SUGGESTION
□ INQUIRY
□ FOR YOUR INFORMATION

PRIORITY
□ LOW
□ STANDARD
☒ HIGH

CAN THE PROBLEM BE REPRODUCED AT WILL?
☒ YES □ NO

CPU TYPE	SERIAL NO.	SYSTEM DEVICE	MEMORY SIZE	DISTRIBUTION MEDIUM
PDP-12	561	RF08	16K	DECTAPE

PROBLEM:
Invalid "FE" errors at run time when OPENING a file after a previous CLOSE. Under some conditions the system crashes and the system must be rebooted.

DIAGNOSIS:
BASIC.FF de-allocates driver space but does not communicate this information to OS/8's USR by modifying the device residency table or performing a RESET. In other words, BASIC thinks it can load another device handler in previosly used (but now assumed free) handler space. Unfortunately, USR does not understand what has happened(it only can assume the device residency table is correct)! Therefore, if a previosly opened ~~xxxx~~ and closed file is referenced afer BASIC has loaded another handler in its slot, very serious problems can occur (destroying the system device's data, etc.).

In addition, due to another bug in the same routine in OPEN, the OPEN routine sometimes thinks there is inadequate driver space available when such space should be available after a CLOSE (and the above bug is fixed!).

CURE:
This patch corrects both of the above problems. BASIC.FF does not attempt to free driver space until it is possible for it to issue a RESET to USR. In this way, both USR and BASIC agree as to driver residency. A sample program is included to demonstrate the fix. For those interested, the mneumonics for the patch are shown below:

```
        *13455
        TAD MM4
        DCA TEMP3
        TAD TADINS
        DCA CHECK
        DCA I WORD4   /RESET FILE OPEN
        TAD W4PTR
        DCA TEMP1
        TAD I TEMP1
        SZA CLA
        JMP CRETN     /ALL FILES NOT CLOSED, CAN'T RESET
        ISZ CHECK
        ISZ TEMP3
        JMP CHECK

                (continued)
                CIF 10        /ALL FILES CLOSED
                JMS I USR
                13            /PERFORM RESET
                CLA
                DCA DMAP      /FREE ALL DEVICE HANDLER
        CRETN,  JMS I P1SWAP                    SPACE
                JMP I ILOOPL
       TADINS,  TAD W4PTR
```

SOFTWARE COMMUNICATIONS USE ONLY

DO NOT PUBLISH □

DATE RECEIVED	BACK FROM MAINTAINER	LOGGED ON
TO MAINTAINER	DATE CLOSED	LOGGED OFF

#57949 BASIC.FF V3D PAGE 2 of 4

```
.GE SYS BASIC.FF
.OD
13455/1466 1335;3042;1301;3262;3472;1331;3040;1440
13465/1042 7640;5277;2262;2042;5262;6212;4522;0013
13475/7650 7200;3035;4573;5570;1331
13531/6714 6720;6735;6752;6767
^C
.SA SYS BASIC.FF
```

#57949 BASIC.FF V3D PAGE 3 of 4

THIS IS A TEST PROGRAM TO TEST AND ILLUSTRATE
THE PATCH DESCRIBED ABOVE. THE PATCHED BASIC
IS SAVED UNDER THE NAME 'NBASIC.FF'.

```
.BASIC
NEW OR OLD--OLD IOTEST

READY
LIST

IOTEST  BA    5A    24-JUL-78

1 DIM I$(20),O$(20),X$(80)
10 PRINT\PRINT "FROM";\INPUT I$
20 PRINT "TO";\INPUT O$
30 FILE #1:I$
40 FILEV #2:O$
50 INPUT #1:X$
60 IF END #1 THEN 90
70 PRINT #2:X$
80 GO TO 50
90 CLOSE #1
100 CLOSE #2
110 GO TO 10
32767 END

READY
RUNNH

FROM?DTA0:TDUMP.EU
TO?LPT:

FROM?DTA0:TDUMP.EU
TO?TTY:

FE AT LINE 00040

READY
```

```
.
NOTE: THE ABOVE 'FE' ERROR IS INCORRECT!  THERE SHOULD
BE PLENTY OF DRIVER SPACE AVAILABLE AFTER BOTH CLOSES.
```

```
.COPY BASIC.FF<NBASIC.FF
FILES COPIED:
NBASIC.FF
```

#57949 BASIC.FF V3D PAGE 4 of 4

```
.BASIC
NEW OR OLD--OLD IOTEST

READY
RUN

IOTEST  BA    5A    24-JUL-78

FROM?DTA0:TDUMP.EU
TO?LPT:

FROM?DTA0:TDUMP.EU
TO?TTY:
[
'NEW' L1,L2,DUMP,TEXT,CRLF,A,B,C,D,E,F,G;

DUMP:='CODE' DUMP;
TEXT:='CODE ^C
READY

.
```

digital

SOFTWARE
PERFORMANCE
REPORT

FIELD #:

FOR DEC USE ONLY

57948

Page 1 of 3

SYSTEM PROGRAM AND VERSION (OR DOCUMENT) BASIC.FF	MONITOR AND VERSION OS/8 Extension Kit V3D	DATE 6-AUG-78

NAME Lyle P. Bickley
FIRM: Bickley Labs

ADDRESS: 47 Ivy Mills Road
RD2
Glen Mills, PA
ZIP 19342

DEC OFFICE
Blue Bell, PA

REPORT TYPE
☒ LOGIC/CODING ERROR
□ DOCUMENTATION ERROR
□ SUGGESTION
□ INQUIRY
□ FOR YOUR INFORMATION

PRIORITY
□ LOW
□ STANDARD
☒ HIGH

SUBMITTED BY: Lyle P. Bickley
PHONE: 215-985-7733

CAN THE PROBLEM BE REPRODUCED AT WILL?
☒ YES □ NO

LIST ATTACHMENTS
(1)patch, (2)sample test program

CPU TYPE	SERIAL NO.	SYSTEM DEVICE	MEMORY SIZE	DISTRIBUTION MEDIUM
PDP-12	561	RF08	16K	DECTAPE

PROBLEM:
BASIC.FF generates invalid error messages when file #0 is opened (see example).

DIAGNOSIS:
BASIC.FF's OPEN routine modifies the file table before determining if the file is file "0" (the system console). OPEN should not modify the table at all for file "0" (i.e. it should be a NOP as per CLOSE).

CURE:
This patch causes OPEN to test for file "0" before attempting to modify the file table. If the file is #0, a logical NOP occurs and an immeditae return is made from open. (see example program for effect of fix!)

#57948 BASIC.FF V3D PAGE 2 of 3

```
.GE SYS BASIC.FF

.OD

14004/3466 3227
14010/1472 5363
14163/5227 1227;3466;1472;5211
^C
.SA SYS BASIC.FF

.
```

SOFTWARE COMMUNICATIONS USE ONLY

DO NOT PUBLISH □	DATE RECEIVED	BACK FROM MAINTAINER	LOGGED ON
	TO MAINTAINER	DATE CLOSED	LOGGED OFF

#57948 BASIC.FF V3D PAGE 3 of 3

THIS PROGRAM DEMONSTRATES AND TESTS THE ABOVE PATCH TO 'BASIC.FF'.
THE PATCHED BASIC IS SAVED UNDER THE NAME 'NBASIC.FF'.

```
.BASIC
NEW OR OLD--OLD OPTEST

READY
LI

OPTEST  BA    5A    24-JUL-78

1 DIM X$(80)
10 N=0
20 FILEV #N:"WHEN N=0 THIS IS NEVER LOOKED AT BY OPEN!"
30 INPUT X$
40 PRINT #N:X$
50 IF X$<>"STOP" THEN 30
60 CLOSE #N
32767 END

READY
RUNNH
?NOW IS THE TIME

VR AT LINE 00030

READY
```

NOTE: THE 'VR' ERROR ABOVE (ATTEMPT TO READ VARIABLE LENGTH FILE)
IS OBVIOSLY INCORRECT!

```
.COPY BASIC.FF<NBASIC.FF
FILES COPIED:
NBASIC.FF

.BASIC
NEW OR OLD--OLD OPTEST

READY
RUNNH
?NOW IS THE TIME
NOW IS THE TIME
?FOR ALL GOOD
FOR ALL GOOD
?MEN TO COME
MEN TO COME
?AND
AND
?STOP
STOP

READY
```

PROCESS CONTROL SYSTEMS, INC.

18130 S. Thornapple Lane • New Berlin, Wisconsin 53151 • (414) 782-3945

HARDWARE CONSULTATION
SOFTWARE DESIGN
PROCESS CONTROL SYSTEM DESIGN

August 22, 1978

Mr. Robert Hassinger
Liberty Mutual Research Center
71 Frankland Road
Hopkinton, MA 01748

Dear Bob:

These are date patches to OS8BOL, The BOOL-143 Control Equation Translator for the OS/8 Industiral 14 software package. The year on the page heading will be correct instead of in the range 1970-1977.

```
 For 1978-1979:                  For 1980-1985:
    .GET SYS OS8BOL                 .GET SYS OS8BOL
    .ODT                            .ODT
    2066/1362      1357             2157/XXXX      0016
    2157/XXXX      0030             2164/0027      0030
     C                               C
    .SAVE SYS OS8BOL                .SAVE SYS OS8BOL
```

These are date patches to OS8PAL, the PAL-143 Symbolic Program Assembler for the OS/8 Industrial 14 software package. The year on the page heading will be correct instead of in the range 1970-1977.

```
 For 1978-1979:                  For 1980-1985:
    .GET SYS OS8PAL                 .GET SYS OS8PAL
    .ODT                            .ODT
    5446/1371      1367             5567/XXXX      0016
    5567/XXXX      0030             5573/0027      0030
     C                               C
    .SAVE SYS OS8PAL                .SAVE SYS OS8PAL
```

Yours truly,

Michael E. Mazzoni
President

MEM:blm

DECUS HAS MOVED!!

As of August 14, 1978, the DECUS International Headquarters and DECUS U.S. Chapter offices will be located at Digital Equipment Corporation in Marlboro. Our new address is:

DECUS
MR2-3/E55
One Iron Way
Marlboro, Massachusetts 01752

Marlboro is not on Centrex, a direct access telephone system, so all calls will come thru the switchboard at (617) 481-9511.

DECUS extensions are as follows:

Central Number	4100
Executive Director	4120
Admin/Finance	4122
Order Processing	4135
Accounting	4136
Membership	4166
Publications	4131
Library	4177
U.S. Chapter	4141

DECUS

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

MOVING OR REPLACING A DELEGATE?

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

( ) Change of Address
( ) Delegate Replacement

DECUS Membership No.: ____________________

Name: ______________________________

Company: ____________________________

Address: ____________________________

____

State/Country: ________________________

Zip/Postal Code: ______________________

Mail to: DECUS - ATT: Membership
One Iron Way, MR2-3
Marlboro, Massachusetts 01752 USA

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