UNIVERSITY OF CALIFORNIA

LICK OBSERVATORY TECHNICAL REPORTS

NO. 21

32K FOCAL USER'S GUIDE

Robert Kibrick

Santa Cruz, California

February 1977
Amended June 1977

ALPHABETICAL INDEX OF 32K FUNCTIONS

NOTES: * IN NAME FIELD INDICATES COMMAND IS IN AN OVERLAY
*1 SEE DOCUMENTATION FOR 3-PHOTOMULTIPLIER COUDE DEVICE
*2 SEE DOCUMENTATION FOR 32K DATA TAKING SYSTEMS

ALPHABETICAL INDEX OF 32K FUNCTIONS (CONT.)

NOTES: * IN NAME FIELD INDICATES COMMAND IS IN AN OVERLAY
*1 SEE DOCUMENTATION FOR 3-PHOTOMULTIPLIER COUDE DEVICE
*2 SEE DOCUMENTATION FOR 32K DATA TAKING SYSTEMS

ALPHABETICAL INDEX OF 32K FUNCTIONS (CONT.)

NOTES: * IN NAME FIELD INDICATES COMMAND IS IN AN OVERLAY
*1 SEE DOCUMENTATION FOR 3-PHOTOMULTIPLIER COUDE DEVICE
*2 SEE DOCUMENTATION FOR 32K DATA TAKING SYSTEMS

Acknowledgments

In addition to Jack Baldwin, who did the bulk of the original system design, I would like to thank the following people:

Sandy Faber and Heywood Sobel, who, as the first users of the system, helped to locate a number of weak points in the system that have now (hopefully)been eliminated. Dave Burstein, Steve Grandi, and Ed Kemper, for their advice on deadtime corrections, peak finding, and IBM tapes, respectively. Ted Cantrall, for his suggestions on having 32K Focal adapt itself to various size machines. Alan Koski, who suggested that 32K Focal should be able to handle scans longer than 2048 channels. And finally, Lloyd Robinson, whose suggestions are too numerous to elaborate, and who always managed to find something else new to add to the system.

ABSTRACT

This report is designed as a User's Guide to the Lick 32K Focal System. It assumes some familiarity with Digital Equipment Corporation's Focal 1969 language. Although this report makes numerous comparisons with the Lick 8K Focal System (which is described in previous Lick Technical Reports), an understanding of Lick 8K Focal is not a prerequisite.

The first part of this report describes the similarities and differences between the Lick 8K Focal System and the Lick 32K Focal System. It also describes some of the new features and concepts one needs to understand to efficiently use the system. The second part describes the operation of specific groups of 32K Focal functions, and serves as an extended summary of all the commands that are a resident part of 32K Focal. The third part describes additional commands that are available as overlays. ** Appendix A gives a series of indices and cross-references of Focal function names and related codes. Appendix B gives a complete list of error codes and their associated meanings.

It is probable that updated versions (or additional pages for this version) will appear from time to time.

** Appendix A was deleted when this report was amended in June 1977.

INDEX

APPENDICES

32K FOCAL USER'S GUIDE

I. Introduction

a) Background

To assist in data acquisition and data reduction, three PDP 8I computers were installed at Lick Observatory in the early 1970's. Since these computers were to be programmed by astronomers and researchers and not by programming specialists, a programming language was needed that was:

1. Easy to learn
2. Simple to use
3. Able to perform specific operations on large arrays of numbers at high speed
4. Able to interact with a wide variety of specialized electronic hardware.

Digital Equipment Corporation, the manufacturer of the PDP-8I, supplied an interactive programming language called Focal 1969, which met the first two criteria. Lick then made extensive modifications to Focal so that it satisfied the latter two. The end product of those modifications, Lick 8K Focal, is extensively documented in previous Lick Technical Reports.

Although Lick 8K Focal has evolved considerably since its inception, its growth in certain directions has been constrained by a lack of core memory. Despite such useful features as the extended disk buffer, many operations involved in processing scanner data remain cumbersome and awkward to program. Further, as the volume of data to process has grown, the PDP 8I's have become increasingly backlogged. But without additional core memory, little could be done to improve Focal.

When less expensive add-on core memories for the PDP-8I became available in the mid-1970's, the idea of developing a more efficient Focal became feasible. The decision was made to upgrade the PDP-8I's to their maximum memory capacity of 32K of core, and to expand Focal to utilize this additional memory. Work on 32K Focal began in 1975 by Jack Baldwin, and continued through May 1976 at which point it was taken over by Bob Kibrick. This report describes the result.

b) Major differences between Lick 8K Focal and Lick 32K Focal

1. Where 8K Focal had 2 core data buffers of 512 channels each, 32K Focal has 5 core data buffers of 2048 channels each. As a result, it is much simpler to manipulate 2048 channel scans in 32K focal.

2. In the 8K Focal System only a few Focal functions could be in core memory at any one time. Because of this space limitaton, the Focal functions were stored on the disk. If a Focal function was not already in core at the time it was needed, a copy of it was read from disk to core. Read time ≈ 100 msec.

In the 32K Focal system, all of the Focal functions (excepting those brought in with the XNAME command) are core resident for faster access.

3. The text area for Focal programs has been expanded so one can write larger programs and thus not have to do as much chaining between programs.

4. The DF 32 disk is no longer used. Data can now be transferred directly from buffers to IBM tape without having to be written onto the disk.

5. The Lick Focal functions are now faster and simpler to use.

6. XNAME overlays written for the 8K system cannot be used on the 32K system, and vice versa.

7. Focal programs written for one system generally will not work the same on the other system.

8. Two XNAME overlays can be in the core memory at the same time.

9. The 8K Focal and 32K Focal systems record scans on Dectape in different formats, although they both use the same amount of space for each scan.

10. A total of 36 scans of 2048 channels each could be stored on a Dectape under the 8K system; 46 scans can be stored on a Dectape using the 32K system. As a result, the scans are numbered differently in the 2 systems. 8K scan 0 starts in block 500 octal (320 decimal) while 32K scan 0 starts in block 0. 8K scan #N is located on the same place on the Dectape as 32K scan #N+10.

11. In the 8K Focal System, if one erased the CRT and then immediately tried to write on the screen, the resulting image would be too faint. As a result, one had to insert Focal delay loops to wait 0.5 seconds between CRT erase and CRT write commands. In 32K Focal, all commands which write on the CRT check a hardware flag and wait until the CRT has finished erasing before beginning to write on the screen.

12. In the 8K Focal System, if a Lick Focal function was called with invalid arguments or asked to operate on invalid data, the function would often proceed with its calculations and return a result as if nothing were wrong. This would make program debugging difficult. In the 32K Focal system, wherever possible, if a Lick Focal function detects invalid arguments or data, it will print a specific error message and stop execution.

c) Similarities Between Lick 8K Focal and Lick 32K Focal

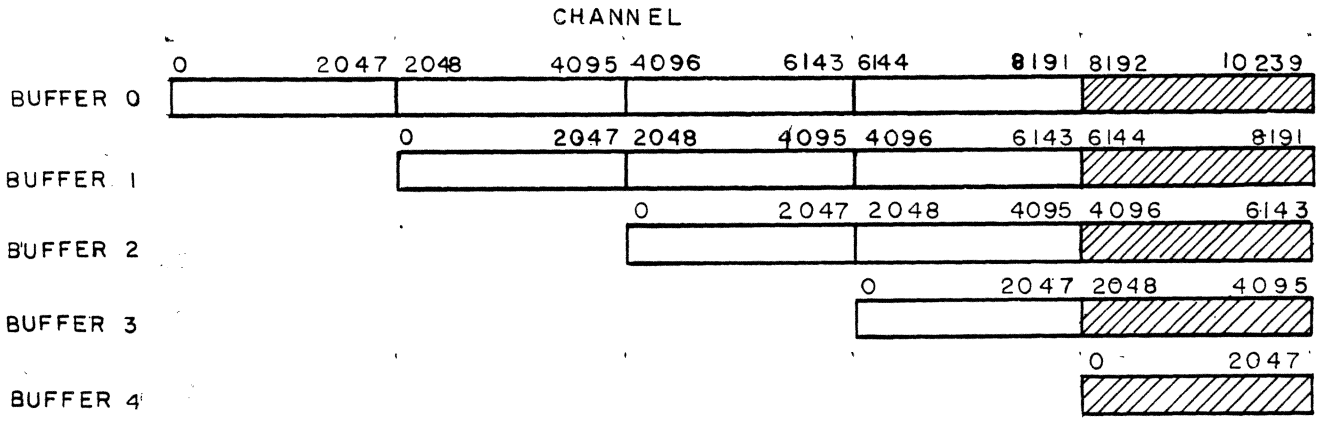
1. The layout of the program Dectape is very much the same for both systems. X NAME Overlay 1 begins on block 134 (octal); Program 0 begins in block 160 (octal). Thus, the Focal tape copiers can still be used on 32K Focal tapes.

2. The 32K Focal system and 8K Focal systems record scans on tape in different formats, although they both use the same amount of space for each scan. However, the 32K system can read scans written in either format and will automatically perform any needed conversions. The 32K system can also write scans in either format. At present, (Jan/77) the 8K system can only read and write scans in 8K format.

3. The bootstrap procedure for 32K Focal is the same as that for 8K Focal.

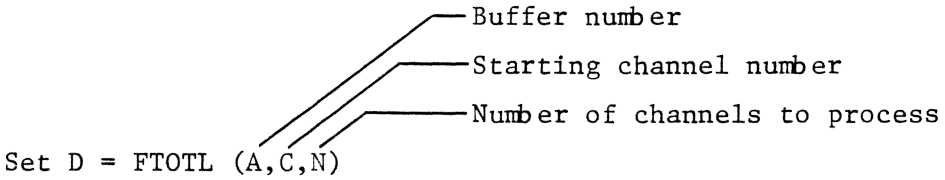
d) The 5 Core Data Buffers

The object of 32K Focal is to make processing scans of 2048 channels both faster and easier. Five core data buffers are available, each with a capacity of 2048 channels. Two different modes of addressing allow one to treat these buffers as 5 separate 2048-channel buffers, or to treat groups of these buffers as if they were a single, longer buffer. For example, 2 of the buffers can be treated as if they were a single 4096-channel buffer; or 5 of the buffers could be treated as if they were a single 10,240 channel buffer.



The above diagram illustrates how the same area of memory can be addressed in a variety of different ways. Specifying channels 0-2047 of buffer 4 (hashed in area) is equivalent to specifying channels 6144-8191 of buffer 1, or channels 8192-10239 of buffer 0. The different addressing modes are best illustrated by example.

FTOTL is a Focal function which sums up the counts in a specified number of channels. The form of the FTOTL command is:



To get the sum of the counts in channels 1-20 of buffer 0, one could code Set D = FTOTL (0,1, 20). To get the sum of channels 2051-2055 of buffer 0 one could code Set D = FTOTL (1,3,5), since channel 2051 of buffer 0 is equivalent to channel 3 of buffer 1.

It is often the case that one wishes to treat the buffer area as 5 distinct buffers of 2048 channels each, and to have a command operate on all 2048 channels of each such buffer. To simplify coding in these cases, if the starting channel number (C) and the number of channels to process (N) are both left to default to zero, then the first 2048 channels of a buffer are processed; that is

Set D = FTOTL (A) is equivalent to Set D = FTOTL (A, 0 , 2048). In the case where N is zero but C is not, the processing begins in the channel specified by C and continues through channel 2047 of that buffer; that is

> Set D = FTOTL (A,C) is equivalent to Set D = FTOTL (A,C, 2048-C) (to apply this rule in cases where C > 2047, adjust A to bring C into the range 0 <= C <= 2047, i.e., Set D = FTOTL (0,2051) is equivalent to Set D = FTOTL(1,3), which is equivalent to Set D = FTOTL (1,3,2045).

On the other hand, one may often want to operate on scans longer than 2048 channels, and that is the reason for the other mode of addressing. To get the sum of channels 0-4095 of buffer 0, one would code Set D = FTOTL (0,0,4096). To sum channels 4096-10239 of buffer 0 one would code Set D = FTOTL (0, 4096, 6144) or equivalently Set D = FTOTL (2,0,6144). In this last case, one is

effectively treating the buffer area as if it were only 2 long buffers -- buffer 0 which has 4096 channels and buffer 2 which has 6144 channels. Note: operations need not begin or end on channel numbers which are even multiples of 2048. To sum up channels 2001-10000 of buffer 0, one would code Set D = FTOTL (0, 2001, 8000).

A few words of caution are in order on this point. The dual mode of addressing allows one to concisely refer to scans of exactly 2048 channels while at the same time providing the flexibility to manipulate in a single command scans in excess of 2048 channels. However, the ability to refer to the same area of memory in more than one way also makes it easier to get confused and to try to use the same area of storage at the same time for mutually exclusive purposes.

For example, if one wants to maintain a long scan of 6144 channels in channels 0-6143 of buffer 0, he must remember that in essence he is also occupying channels 0-2047 of buffers 1 and 2. Buffers 3 and 4 are still safe to use in this case as they do not overlap with the area in which the long scan is being stored.

If one is only working with scans of 2048 channels and as long as there is no need to specify a non-zero number of channels (N), then different buffer numbers will never refer to overlapping areas of memory. That is, Set D = FTOTL (A,C) and Set D = FTOTL (B,C) will always reference mutually exclusive areas of memory (provided $A \neq B$, of course!).

It is also possible to specify combinations of buffer numbers (A), starting channel numbers (C), and numbers of channels to process (N), which are plainly illegal, and some which perhaps should be legal but are not. Here are some examples of INVALID commands:

Set D = FTOTL (5) - Buffer 4 is highest legal buffer number (on a machine with 32K memory - see next section).

Set D = FTOTL (3,5100) - The highest legal channel in buffer 3 is 4095.

Set D = FTOTL(3,-6144,10240) - Nice try but it won't work. Although channel 6144 buffer 0 is equivalent to channel 0 buffer 3, channel -6144 buffer 3 is not equivalent to channel 0 buffer 0. Negative channel numbers are completely taboo!

Set D = FTOTL (0,2047,-2048) - Nice try again; no you can't start at the righthand side and count backwards to the left. Negative numbers of channels to process are also taboo

If you use any of these illegal varieties of buffer specifications, Focal will print an error message and refuse to execute the command. (Error messages are listed in Appendix B.)

It is also possible to come up with specifications of A,C, and N which, although not really proper, will be executed without giving an error message. As an example

Set D = FTOTL (3,0, 8192) is not really correct in that there are only 4096 channels available if one starts at channel 0 of buffer 3. No error message is given and the command is treated as if it had been Set D = FTOTL (3,0, 4096).

As a final note on the data buffers, it should be noted that although Lick 32K Focal is designed to operate on a PDP-8 with 32K of core memory, it can still be used on PDP-8's with less than 32K of core. If less core is available, fewer buffers are available. The following table relates the number of available buffers to the amount of core memory available.

Amount of Memory	Available Buffer Numbers
32K	0, 1, 2, 3, 4
28K	0, 1, 2, 3
24K	0, 1, 2
20K	0, 1
16K	0
12K	No buffers available Extended instructions not usuable
8K	No buffers available Extended instructions not available Only 1 X NAME buffer available
4K	Nothing available.

When Lick 32K Focal is bootstrapped, it automatically determines the amount of core memory available on the machine, and adjusts itself accordingly. Attempts to use buffers that are not available will be diagnosed by Focal as errors. Thus, the command:

Set D = FTOTL (2)

will work correctly on machines with 24K or more of core, but will cause an error message to be printed if used on a machine with less than 24K of core.

II. 32K FOCAL Functions

a) Equivalent Functions

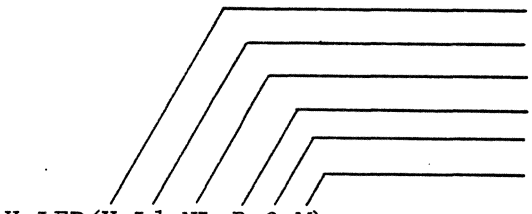
The following Lick FOCAL functions work the same in both 8K and 32K FOCAL and are always available:

X CPEN(P,T) — P = 0: Pen Up. P = 1: Pen Down. Pause for a time about 10*T msec. Pen motion needs about 100 msec, which can be used for computation, or by the pause.

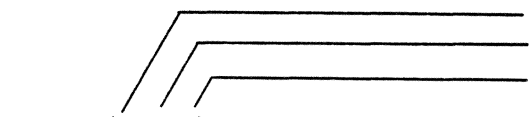
X DIS(X,Y) — Store a dot on the CRT at location (X,Y). Full scale is 1023.

X GO(S,L)
X DO(S,L) — Like ordinary GO, DO but with computed arguments. (Subroutine S, line L.)

X END(0) — Return to calling program; next line



X LED(X,L1,NL,B,O,M) — Display numbers on LED digits on switch panel



X STAT(X,Y,S) — Direct all future printing to the CRT. Redirect printing to teletype if X = -1 or for CTRL-C, or for any error diagnostic.

X SWIT(-1) — Erase CRT (Can be followed immediately by CRT write).

X SWIT(0,L) — Load lamps L. Lamps are coded 1,2,4,8,16,32.

SET D = FSWIT(N,S,0,0,M) — Read switch N,S to D. Set M=4095 to read all of group N at once (Use 4094 to read all of group 3). M=129 would read switch 1 & 8, weighted. M = 0 to read only switch N,S.

SET D = FSWIT(3,11,X,Y) — Display joystick marker on CRT at (X,Y). When switch 3,11 is pushed, return 1024*X1 + Y1 where (X1,Y1) is final marker location on CRT.

SET D = FVAR(0) — Set D to end point of FOCAL variable list.

X VAR(D) — Erase all variables defined beyond point D in list.

X WHAT(N,M) — Type the names of M user generated overlay programs as found on program DECtape, starting at overlay #N.

b) Program/NAME Functions

The following Lick Focal Commands are also always available, but are slightly different than their 8K counterparts

X NAME(N,F) - Reads into name buffer #F and makes available for use the commands from user generated machine language overlay #N. There are 2 name buffers (numbered 1 and 2) on machines with 12K or more of core; only name buffer 1 is available on machines with less than 12K of core.

Note: A) F $\neq$ 2 is treated as if it were F = 1

B) X NAME(0) is treated the same as X NAME(1), except that X NAME (0) will always cause a fresh copy of overlay #1 to be read in from tape. The commands in NAME overlay 1 are read into name buffer 1 at bootstrap time.

X CALL(N,S) - Call Program N from Dectape. If S > 0, starts program N at subroutine S. (Code S*128 + L to start program N at line L of subroutine S). Use X END(0) to return to the line following the X CALL command in the calling program.

(For greater versatility in calling, see the X CADO command, pg. U7)

Important Notes:

A) It is illegal to use either X CALL or X END from inside a DO group or FOR loop.

B) If FOCAL's text area contains new or modified text that has not yet been filed (See X FILE command below), then execution of X CALL or X END will cause FOCAL to type "REALLY?" on the teletype, and then to wait for a reply. Any reply other than "Y" will cause the X CALL or X END command to be supressed, and the contents of the text area to be preserved. A reply of "Y" will cause the command to proceed and the contents of the text area to be lost.

C) If X CALL is used to address a program area on the Dectape that does not contain a FOCAL program, the following action is taken:

1) If the program area addressed by X CALL is not completely empty, the error message

BAD PROGRAM nnnn

is printed, where nnnn is an octal number indicating the number of non-empty (i.e., non-zero) locations in the program area. Focal's text area is erased, and is marked as not containing any program.

2) If the program area addressed by X CALL is completely empty, no error message is printed. Focal's text area is erased, and marked as containing the requested program.

X FILE(N) File program N on Dectape. If program N was not first called in using X CALL (N), the message

OK?

will be printed. If the previous program (or perhaps old data on the tape) is to be overwritten, type "Y." To avoid writing, type "N" or CTRL-C. The program is still intact in core after being filed (or not filed) on tape. X FILE also prints last address used for text;highest address=12567.

A note on adding new programs:

Before one starts typing in a new program, one should decide which program number to try to file it under. Next, one should check to see if that program number is in use. To check if the area on the tape for program N is in use, type X CALL(N). If this area on the tape contains data or an overlay, one will get the message

BAD PROGRAM nnnn

If this happens, another program number should be used (unless one doesn't care about losing the data or overlay). If one doesn't get the message

BAD PROGRAM nnnn

then the area of tape referenced by X CALL(N) either contains a valid Focal program or is empty. Type W to list the program. Continue hunting for an available program number until you find an empty portion of tape or until you find a program that you no longer need.

c) Buffer Independent DECtape I/o Functions

X PUT (W,B,U,I)	- Store integer I in word W of Dectape block B on Dectape unit U.
SET I = FTAK (W,B,U)	- Return single precision value of word W in Dectape block B from Dectape unit U (Note integers have values of 0 <=I<=4095)
X STOR(W,B,U;V)	- Store variable V starting at word W of Dectape block B on Dectape unit U. (Note Semicolon!)
SET V = FASK(W,B,U)	- Return 10-digit floating point format variable starting at word W in Dectape block B from Dectape unit U (Note - each floating point variable occupies 4 words)

Notes on the use of W,B, and U in PUT, TAK, ASK, STOR

1. There are 129 words in each block numbered 0-128
2. There are 1474 blocks on a Dectape, numbered 0-1473
3. If one counts total words on a Dectape, starting at word 0 block 0, there are 190146 words, numbered 0-190145. The location of a given word on the Dectape can be specified in a number of different ways. For example, word 128 block 1473 is equivalent to word 190145 block 0; word 0 block 1 is equivalent to word 129 block 0.
4. If B=W=0, the previously used Dectape location will be incremented (by 1 for PUT and FTAK; by 4 for STOR and FASK) and taken as the Dectape location to be used for the current command. However, if the unit specified (U)is not the same as the immediately previously used unit, an error message will be given.
5. One block of the Dectape is usually kept in core. If a negative unit number is specified (use -8 for -0), and the desired block is already in core, the Dectape will not be re-read or rewritten. A block in core will only be saved in this case when a different block needs to be brought in. This process is called data chaining and saves time since the Dectape doesn't have to be read and written as often.

If a positive unit number is used, the specified block is always re-read from the Dectape, regardless of whether it is already in core. In the case of PUT and STOR using positive unit numbers, the desired block is read into core, modified, and then immediately written back out to tape.
Although data chaining is faster, it is more dangerous and should be used with caution. Since a block that has been brought into core and modified is not written back out to tape until a different block is read in, one could lose information and also unintentionally write on the wrong tape if one switches tapes while data chaining is in progress. To avoid this problem, and to also provide other useful information, a companion command to PUT/TAK/ASK/STOR has been implemented, called CLOS.

X CLOS(0)	- Closes the Dectape buffer. That is, if the core buffer contains a chained tape block that has been modified and needs to be written to tape, this will cause it to be written. IF DATA CHAINING HAS BEEN USED, THIS COMMAND SHOULD BE GIVEN BEFORE SWITCHING TAPES OR UNITS.
X CLOS(-1)	- Purges the Dectape buffer. That is, if the core buffer contains a chained tape block that needs to be written to tape, X CLOS(-1) will turn off the data chaining flag and make it appear as if the block had been written to tape, without actually writing it. The contents of the Dectape buffer remains the same. X CLOS(-1) should not normally need to be used unless you get into trouble.
SET D = FCLOS(1)	- D is set to the absolute word address of the next available word on the Dectape. X PUT(127, 1473) followed by SET D = FCLOS(1) would cause D to be set to 190145.

It should be noted that certain other commands use the Dectape buffer for scratch space - in particular X WHAT uses the buffer. These commands effectively execute an X CLOS(0) before they begin execution.

A Note on Write Protection

Lick 32K Focal has a write protection feature that prevents commands like X PUT and X STOR from writing over the Focal system, the user generated machine language overlays, or the user's Focal programs up to program #65. Only write operations to unit 0 are protected -- writes to any other unit can reference any block.

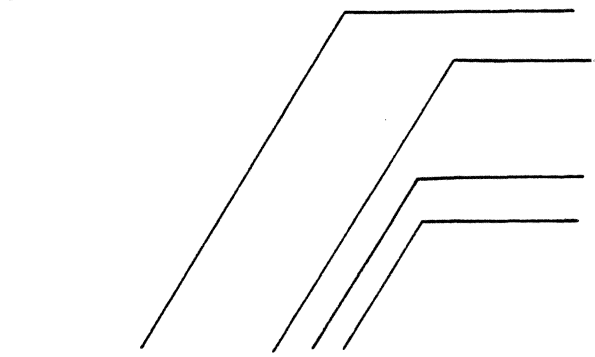
Different Focal tapes may have different numbers of blocks protected. Currently, Jan./77, (and this is likely to change) write attempts below block 640 decimal (block 1200octal or start of program 66 or start of scan 20) are prevented. Please note that the physical reel of tape itself is not protected -- it is only protected when it is mounted on unit 0; the same physical reel of tape is unprotected if it is mounted on a unit other than 0.

d) Single Buffer Functions

The following instructions are available only on machines with 16K or more of core. Although these commands operate on only one buffer, this buffer can be up to 10,240 channels long. Remember- for functions of the form X func(A,...,C,N), unless otherwise noted, the following compact forms apply:

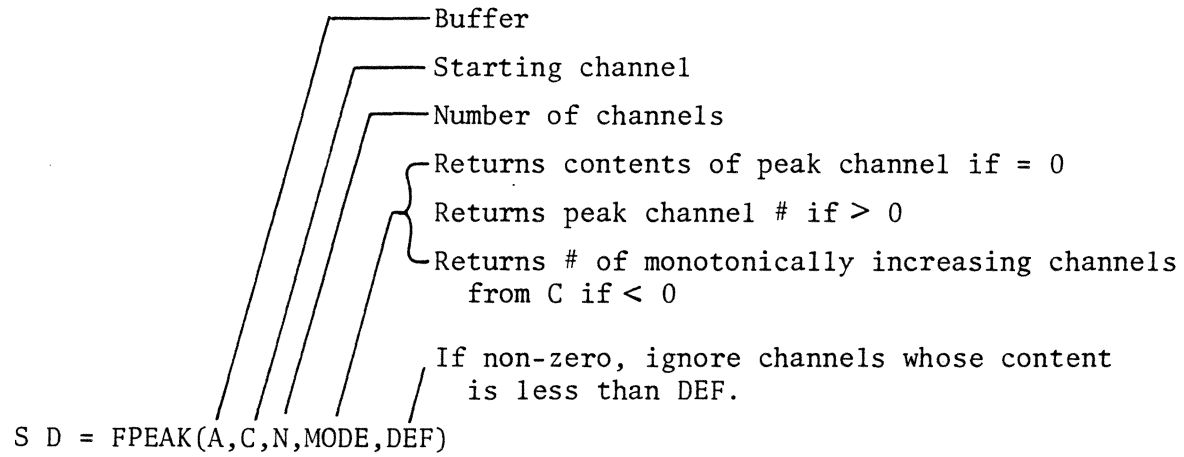
X func(A) is equivalent to X func(A,...,0,2048)
X func(A,...,C) is equivalent to X func(A,...,C,2048-C)

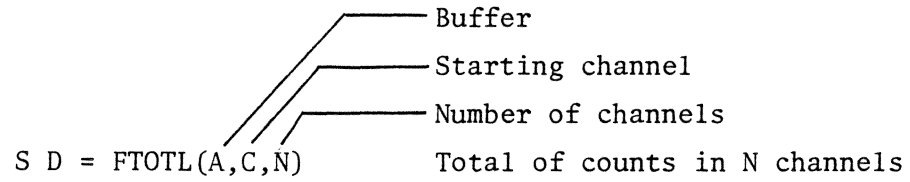
(See section I-d, "The Five Core Data Buffers", on page 7)

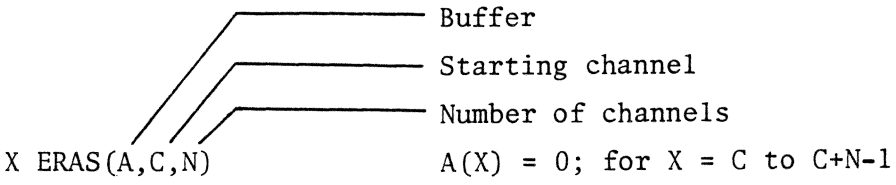


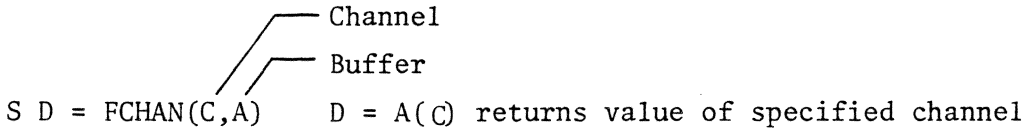
X SHOV (A, 4096*D,C,N)

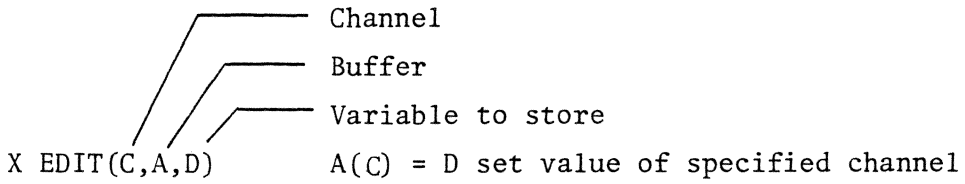
A(x) = A(x+D), where A(x) means channel x of buffer A. Shifts scan by both integral and fractional amounts. If N=0, channels shifted past channel 0 or 2047 are lost. If N≠0, channels can be shifted past buffer boundaries. For example, X SHOV(0,8192,20) will shift channels 20 to 2045 of buffer 0 into channels 22 to 2047; the original contents of channels 2046 & 2047 are lost. X SHOV(0,8192,20, 2048) will shift channels 20 to 2067 into channels 22 to 2069.

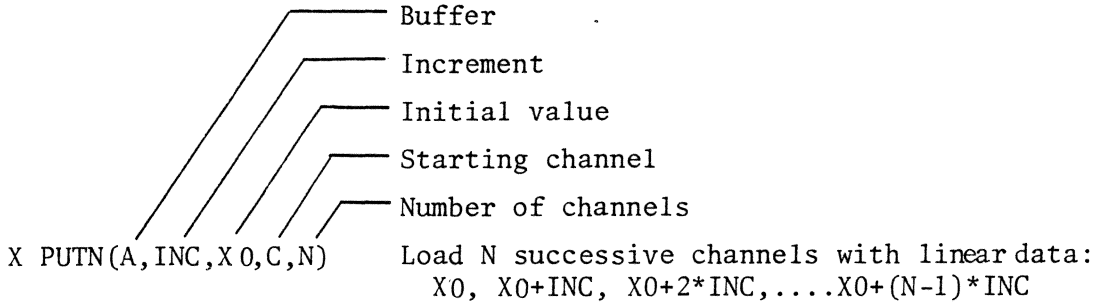

Buffer
Starting channel
Number of channels
Returns contents of peak channel if = 0
Returns peak channel # if > 0
Returns # of monotonically increasing channels from C if < 0
If non-zero, ignore channels whose content is less than DEF.
S D = FPEAK(A,C,N,MODE,DEF)


Buffer
Starting channel
Number of channels
S D = FTOTL(A,C,N)
Total of counts in N channels


Buffer
Starting channel
Number of channels
X ERAS(A,C,N)
A(X) = 0; for X = C to C+N-1

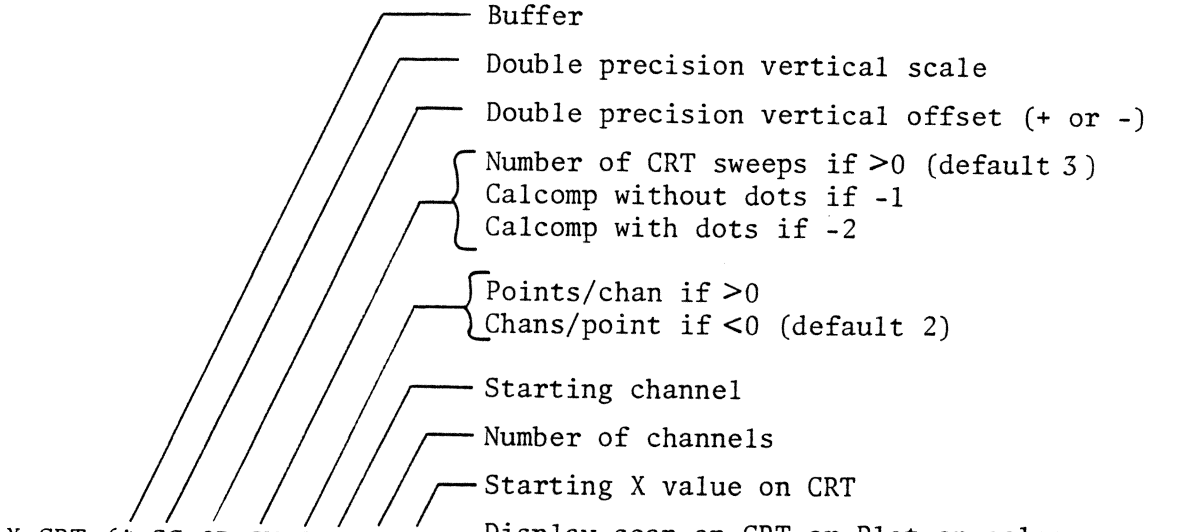

Channel
Buffer
S D = FCHAN(C,A)
D = A(C) returns value of specified channel

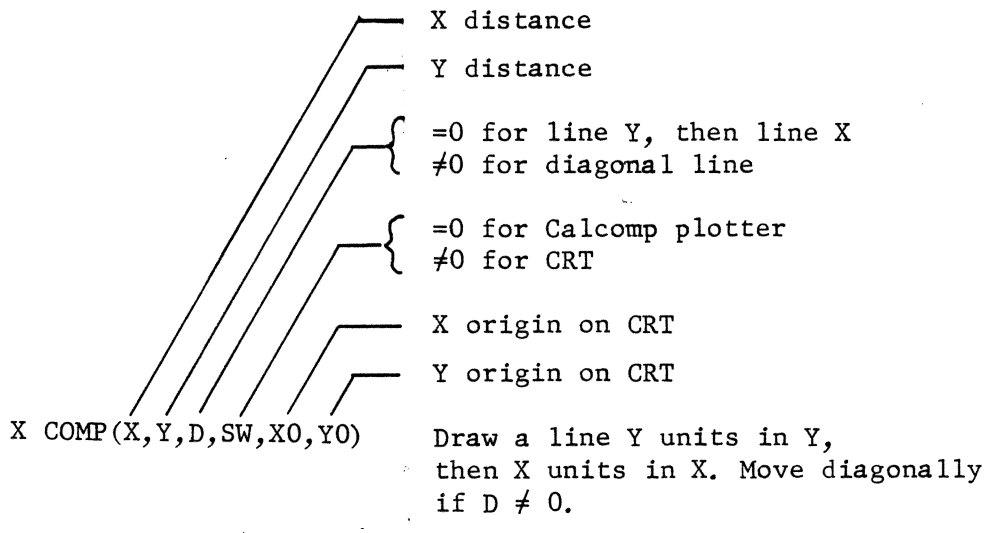

Channel
Buffer
Variable to store
X EDIT(C,A,D)
A(C) = D set value of specified channel


Buffer
Increment
Initial value
Starting channel
Number of channels
X PUTN(A,INC,X0,C,N)
Load N successive channels with linear data:
X0, X0+INC, X0+2*INC,....X0+(N-1)*INC

S D = FZCOM(R)

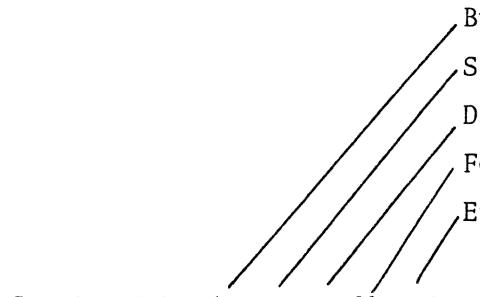
Resets plotting origin to R, returns previous plotting origin. Used with X CRT to plot scans. Plotting origin is set to 4095 bootstrap.



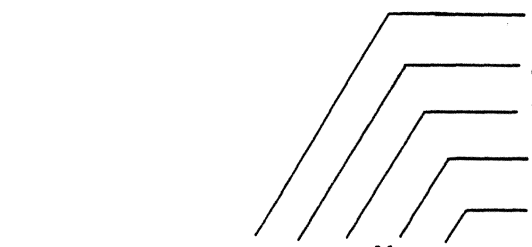


IMPORTANT NOTE ON CALCOMP PLOTTING

In order to speed up data taking operations, the X CRT and X COMP commands have been modified so that they can be interrupted by the scanner memory (see Appendix C).



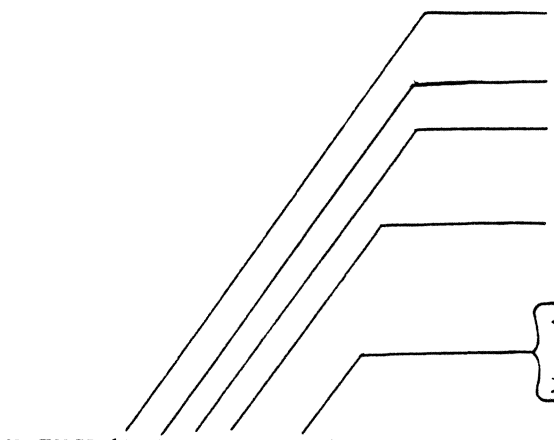
Set D= FMSAV(A, S, U, M, E) Saves 2048 channel buffer A and its 31 word ID area as scan S on Dectape unit U.* Translates to 8K scan format if M ≠ 0; otherwise writes without translation. If E ≠ 0 returns control to focal with a value of -1 if tape error occurs; otherwise types TAPE? TAPE? ETC. (Note: unit 0 is write protected against MSAV - see PUT and STOR) also note: if M≠ 0 the buffer is left in 8K scan format after the write has completed. To translate buffer back to 32K format, use X REFM command.



Set D = FMGET(A, S, U, M, E) Reads scans from dectape unit U into 2048-channel buffer A and its 31 word I.D. area.* If M = 0, performs format conversion automatically if needed. If M ≠ 0, forces 8K to 32K translation to be done (even if data is already 32K). Error control same as in MSAV.

X REFM(A, MODE) Translates Buffer A and its associated I.D. area according to MODE*

*Each of the five 2048-channel data buffers has a 31 word I.D. area associated with it. (See page 22)



General Purpose Exchange Routine performs exchange on an area L words (not channels) long beginning at word (not channel) S in buffer A. Mode specifies type of exchange. If mode = 0, exchange is a circular shift of D places starting at word S in buffer A. (D > 0 to shift right, < 0 to shift left).

If mode > 0, takes L/2 low order words followed by L/2 high order words, starting at word S in buffer A, and pairs them in order HI LO HI LO...HI LO. (Similar to X pair in 8K Focal)

If mode < 0, takes L/2 pairs of words (HI,LO) starting at word S in buffer A, and shuffles them into L/2 low order words followed by L/2 high order words. Note: If mode ≠ 0, L must be an even number.

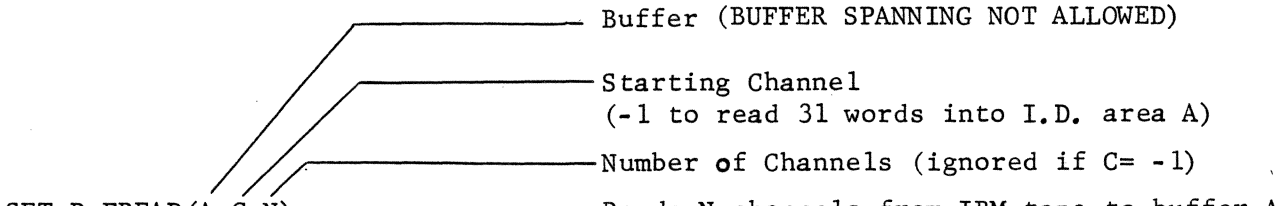
e) IBM Tape Functions

The 9-track, IBM compatible tape transports connected to the PDP 8I computers can be controlled using commands in the Lick 32K FOCAL language. These commands allow reading or writing of single records of up to 2048 channels of data. The PDP 8I's use two 12-bit words per IBM 32-bit word, ignoring the most significant 8 bits of each IBM word. Positive and negative numbers of magnitude less than $2^{22}-1$ are handled correctly.

Data is written as "records" or "blocks" at a density of 800 8-bit bytes per inch, up to 8200 bytes per record (8192 bytes data, 8 bytes control information). Records are separated by a record gap of 0.6" or larger of erased tape. "Tape Marks" or "End of File" marks are followed by about 3.5" of erased tape. In order to locate a particular record on tape, one must count records from the beginning of the tape, or one may count "End of File" marks and then count records within a particular file.

When writing tape, a "read after write" error test is made. If an error is detected, the tape will backspace over the bad record, erase 5" of tape, and try again on the next segment of tape. The IBM format allows gaps of any length greater than 0.6", so that large segments of bad tape can be safely skipped over by the writing program.

The following IBM tape commands are available:

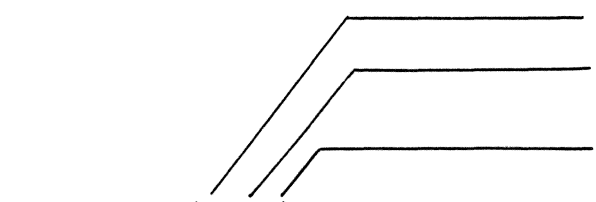


Reads N channels from IBM tape to buffer A starting at channel C. Reads to end of buffer if N=0.
D=4096 for normal read.
D=0 for end of file. Tape left positioned past end of file.
D is negative if read parity error occurs more than 3 times; reads anyway.

Notes:

1. If the number of channels on tape record exceeds N, excess channels are ignored, and no error indication is given. These excess channels are skipped completely; they will not be read by the next READ.

2. If the number of channels on tape record is less than N, only available channels are read. Remaining channels in core are not modified, and no error indication is given. See LREC command for READ length.



SET D=FWRIT(A,C,N)

Writes N channels to IBM tape from buffer A starting in channel C. Writes to end of buffer if N=0.
D=0 if write successful on first try.
D>0 gives count of number of segments of tape that had to be erased before write was successful.
D<0 if write fails after 3 attempts. Tape is left positioned prior to start of last bad write attempt.

Notes:

1. If C= -1, the 31 word I.D. area is written. This I.D. area will appear on tape as a 16 channel record.
2. In order to detect tape slippage, FWRIT will time out if it does not find a record gap pulse within 0.1 seconds of finishing a write. If slippage is detected, FOCAL quits with a ?2-5151?00.00 error message. If this happens, try cleaning tape capstan.
3. When writing many records, or whenever you are writing to a fairly full tape, the tape status should be checked after every write (See FIBM command below). When "end of tape" is sensed after a write, two end of file marks should be written (See X EOF command below). Then the tape should be rewound, unloaded, and a new tape mounted in its place. If one fails to check for end of tape, in the course of a long night of data taking, one might fill a tape and end up winding the tape off the end of the reel, thus losing data and time.

X IBME(0) — Erases 4 feet of tape.

X EOF(0) — Writes an end of file mark. Tape is left positioned past file mark.

Notes:

1. In order to do any operation that writes on the tape, a WRITE RING must be inserted into the back side of the tape reel hub. If one uses any of the FOCAL commands which write on the tape (WRIT, EOF, or IBME) on a tape without a WRITE RING, FOCAL will start printing PROTECT every 2 seconds. If writing is actually desired, and if the writing is to commence at the start of the tape, then one should:
   a) Switch tape drive to OFF LINE. (FOCAL now starts printing OFF LINE instead of WRITE RING.)
   b) Rewind and unload tape.
   c) Insert WRITE RING into back of tape reel hub.
   d) Remount tape and position at load point.
   e) Switch tape drive to ON LINE. FOCAL will stop printing OFF LINE and will write the desired record at start of tape.
2. One can check for a WRITE RING before attempting to write by using the FIBM command to check the tape status.

SET D=FIBM(M) ——— D is set equal to the tape status, masked by M unless M=0.

Status bit values:

2048 - END OF TAPE
(Use this when writing)

1024 - TAPE READY
(Tape unit power is on, and tape is tensioned, positioned on or past load point)

512 - FILE PROTECT
(No write ring is in tape reel hub - i.e., tape cannot be written on)

256 - REWINDING
(Tape drive is busy rewinding)

128 - BEGINNING OF TAPE
(Tape is positioned at beginning, i.e., load point or B.O.T.)

8 - ON LINE
(Tape transport is switched on line. This is like the REMOTE position on DECtape drives; OFFLINE is like LOCAL)

X RWND(0) ——— Rewind up to beginning of tape (BOT).

X HUNT(0) ——— Used to locate end of data on tape (not physical end of tape). Moves forward till double end of file (EOF) is found and stops following the double EOF. The teletype will type "EOF" each time a file mark passes the read head, so that one can use X HUNT to count the number of files on a tape.

SET D=FADV(N,J) ——— Advances N records.
If J=0, stops past EOF if EOF found.
If J≠0, continues advancing past EOF.
D is set= 4096*(# of file marks passed)
+ # of records with parity errors

SET D=FBAK(N,J) ——— Backspaces N records.
If J=0, stops past EOF if EOF found.
Note- Two "EOF" messages are printed: 1 for backspacing over the EOF, and 1 for advancing back over it. D is set to 4096 plus # of records with parity errors.
If J≠0, continues backspacing past EOF.
D is set = 4096 * (# of file marks passed)
+ # of records with parity errors.

=0 for record length
=-1 for record number

SET D=FLREC(MODE) If MODE=0, D is set to the length (in channels) of the last record read or written on IBM tape. If MODE= -1, D is set to the record number (modulo 4096) of the last record read or written to IBM tape. The record number is a count of the number of records (including EOFs) from the start of the tape.

Notes

1. The IBM tape record number counter is set to 0 by X RWND(0)
2. File marks are counted the same as data records.
3. The IBM tape record number counter is displayed in the MQ register while the tape is in motion during X BAK or X ADV.
4. File marks do not have a defined length, i.e., if FLREC(0) is used immediately after reading or writing a file mark, the length returned is unpredictable.
5. Remember that the IBM tape record counter counts modulo 4096; thus FLREC(-1)=0 does not necessarily mean one is at the beginning of tape.

A FINAL NOTE ON IBM TAPE COMMANDS

By adding 4096 to the first argument of the following commands, the printing of the "EOF" messages on the teletype is suppressed:

READ HUNT ADV BAK

f) I.D. Functions

Each of the 5 2048-channel data buffers has a 31 word ID area associated with it. All 5 of these I.D. areas are available as long as the computer being used has at least 12K of core memory. In addition to these 5 I.D. areas (numbered 0-4), there is an additional 31 word I.D. area which can be referenced by using a buffer number of -1. All of these I.D. areas are safe over bootstrapping. (NOTE - Words 0 to 3 of I.D. buffer -1 are used by the P/S monitor; see App. C)

The MSAV and MGET commands (see page 19) transfer both the I.D. buffer and its corresponding data buffer to and from the DECtape. However, as there is no data buffer -1, I.D. buffer -1 cannot be transferred to or from DECtape using MSAV or MGET. Neither can the IBM tape commands READ or WRIT be used with I.D. buffer -1.

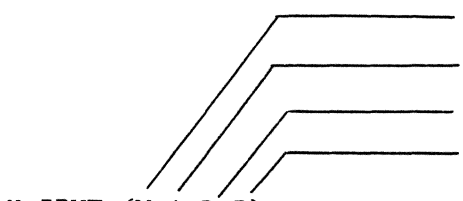
The conversion table below shows the correspondence between the location of the I.D. words on an 8K format DECtape scan, and the 32K FOCAL I.D. buffer locations that these I.D. words will occupy. Note that this table applies to 2048 channel scans; the I.D. information from a 4096 channel scan will occupy two separate I.D. buffers.

CONVERSION TABLE - LOCATION OF I.D. WORDS - 8K .VS. 32K FORMAT

	8K WORD #							
8K BLOCK #	121	122	123	124	125	126	127	128
7	0	1	2	3	4	5	6	7
15	8	9	10	11	12	13	14	15
23	16	17	18	19	20	21	22	23
31	24	25	26	27	28	29	30	**

32K FOCAL I.D. Buffer Word Numbers are shown in corresponding box

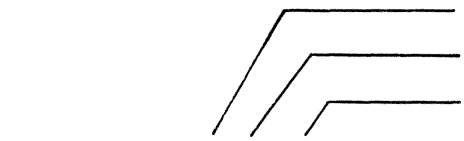
** Reserved for use by 32K FOCAL



Word (0-30)
Buffer
Variable to Store
P ≠ 0 for double precision

X IPUT (W,A,D,P)

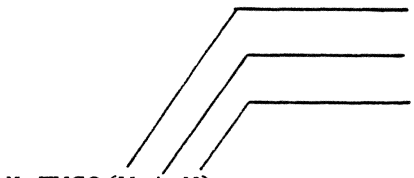
Store single (or double) precision variable D into word(s) W (and W+1) of I.D. area A.



Word (0-30)
Buffer
P ≠ 0 for double precision

Set D=FITAK(W,A,0,P)

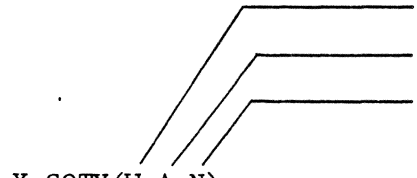
Read single precision variable from word W, I.D. area A (or double precision variable from words W, W+1)



1st Word number (0-30)
Buffer
No. of characters (1-62) (rounded up to next even number)

X TYCO(W,A,N)

Accept up to a maximum of N characters from the teletype and pack them 2 characters per word starting at word W in I.D. area A. If used in the function form (i.e., S D= FTYCO (...)) returns the number of characters accepted, or -1 if ESCAPE or ALTMOD was hit.



1st word number (0-30)
Buffer
No. of characters (1-62) (rounded up to next even number)

X COTY(W,A,N)

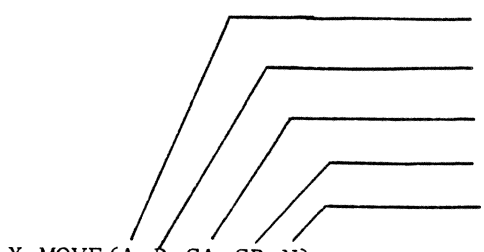
Starting at word W in I.D. area A, unpack and print I.D. information for up to a maximum of N characters. Characters are assumed to be in packed format, 2 characters per word. Printing stops if a null character is encountered. PRINTS ON EITHER TELETYPE OR CRT. If used in the function form (i.e., S D=FCOTY(...)) returns the number of characters printed.

g) Scanner Memory Control Functions

X MEME(0)	Erase scanner memory
Set D = FMEMC(N)	Set counting time N cycles ($\leq 2^{22}-1$). Returns remaining counting time. Stops if N=0. Doesn't set time if already counting.
X MEMW(W,N, OR)	Write N channels starting at Channel 0 (N<=2048) Buffer 0 of core memory to scanner memory, starting at scanner channel W. Writes Low 12 bit part if OR=0, otherwise writes High 12 bit part.
Set D = FMEMR (A,B,H)	Read scanner memory to core. First 2048 channels read to buffer A (not read if A = -1) second 2048 channels read to buffer B (not read if B = -1) reads all 24 bits of each channel if H = 0; reads only Low 12 bits if H $\neq$ 0 (much faster). Sets D if counting (S D = F MEMR(...))
X PAUS(N)	Stop counting if N = 0, continue if N = 1. No change in actual live time.

h) Double Buffer Instructions

(require at least 20K of core)

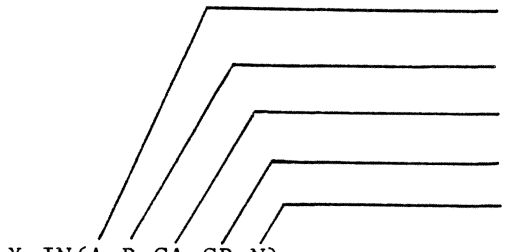


Moves N channels, starting at channel CB in buffer B, to buffer A, starting at channel CA. Contents of buffer B unchanged

$$A(x) = B(y)$$

For X = CA to CA+N-1

Y = CB to CB+N-1

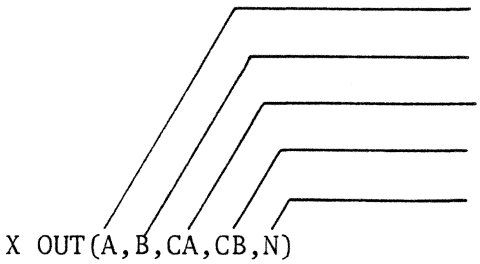


Adds N channels, starting with channel CB in buffer B, to buffer A, starting at channel CA. Contents of buffer B unchanged

$$A(x) = A(x) + B(y)$$

For X = CA to CA+N-1

Y = CB to CB+N-1

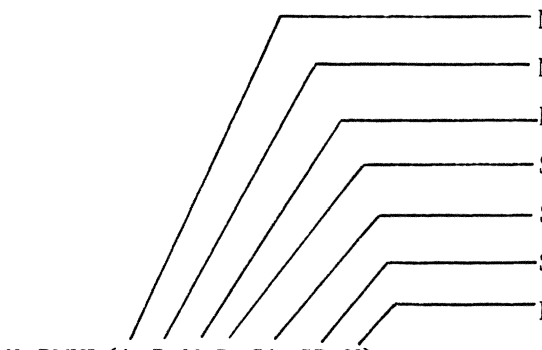


Subtracts N channels, starting with channel CB in buffer B, from buffer A, starting at channel CA. Contents of B unchanged

For X = CA to CA+N-1

$$A(x) = A(x) - B(y)$$

Y = CB to CB+N-1

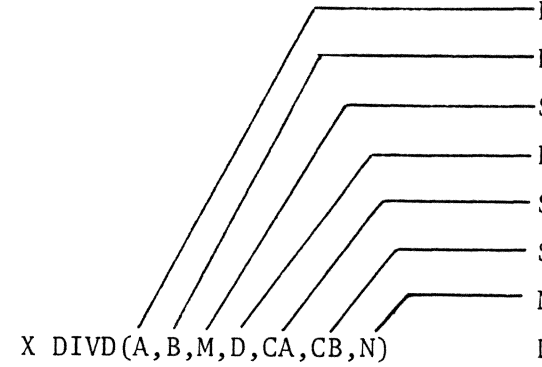


X DMUL(A,B,M,D,CA,CB,N)

Does Double precision multiply of N channels starting at channel CA in buffer A and CB in buffer B. Result goes to buffer A; Buffer B is unchanged. Result is divided by single precision constant D, which defaults to 1.

A(X) = (A(x) * B(y)) / D for x=CA to CA+N-1
for y=CB to CB+n-1

†If M ≠ 0, Buffer A is multiplied by constant M, i.e., A(x) = [A(x)*M)/D] for x = CA to CA+N-1. (B and CB are ignored in this case)



X DIVD(A,B,M,D,CA,CB,N)

Does double precision divide of N channels in buffer A by N channels in buffer B starting at channel CA in Buffer A and in channel CB in buffer B. Result goes to buffer A; buffer B is unchanged. Result is multiplied by single precision constant M, which defaults to 1.

A(x) = (A(x) * M) / B(y) for x=CA to CA+N-1
for y=CB to CB+N-1

*If D ≠ 0, Buffer A is divided by constant D, i.e.,

A(x) = [A(x) * M]/D for x=CA to CA+N-1
(B and CB are ignored in this case)

Note: Division by zero results in a value of zero.

i) Notes

A NOTE ON ARITHMETIC FUNCTIONS

DMUL
DIVD
LOGB
PUTN
IN
OUT
POLY
PUTN

If these commands are used in the function form (i.e., S D = FDMUL (...etc.)), D is set equal to the number of channels in which arithmetic overflow occurred. Negative overflows are set to $-(2^{23} -1)$; positive overflows to $+2^{23} -1$.

Notes on Double Buffer Instructions

1. If N=0, the instruction terminates as soon as channel 2047 of either buffer has been processed.

2. Channels are processed one by one, from left to right.

3. If N≠0 and N exceeds the number of channels remaining in either buffer, the instruction terminates as soon as the last available channels in either buffer has been processed: NO ERROR MESSAGE IS GIVEN IN THIS CASE.

4. IF THE AREAS OF MEMORY IMPLIED BY (A, CA, N) AND (B, CB, N) OVERLAP, RESULTS MAY NOT BE THOSE DESIRED: NO ERROR MESSAGE IS GIVEN IF OVERLAP OCCURS:
   Overlap cannot occur if N=0 and A≠B.
   Overlap is sometimes desirable; for example, a quick way to multiply a buffer by 2 is to add it to itself, i.e., X IN(A,A). Care should be exercised when specifying arguments that cause overlap to occur.

Miscellaneous commands

SET D=FLO(X)	Return as an unsigned 12-bit integer the low-order 12 bits of 24-bit signed integer X.
SET D=FHI(X)	Return as an unsigned 12-bit integer the high-order 12 bits of 24-bit signed integer X.
SET D=FAND(X,Y)	Returns as a signed 24-bit integer the logical AND of the two 24-bit signed integers X and Y.
SET D=FBIG(X)	Return as a 24-bit signed integer the 12-bit signed integer in X. (The 12-bit number X is copied into the low order 12 bits of D; then the sign bit of X is extended into the high order 12-bits of D.)

III. 32K FOCAL OVERLAY FUNCTIONS

a) Description

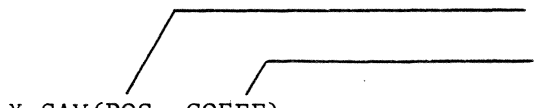
The commands that have been described to this point are the ones most frequently used and as such have been made a permanent part of 32K FOCAL. Additional commands that are used less frequently can be added to the program DECtape as "overlays" using the system program TOVR (see LOTR 32K Focal System Description). The commands in these overlays can then be made available to a Focal program by using the X NAME command (see pg. 12).

An "overlay" can contain from 1 to 10 Focal functions, depending on the length of each function. All ofthe functions residing in the same overlay become available whenever that overlay is loaded into either of the 2 NAME buffers. All of the overlays described on the following pages can be used in either of the 2 NAME buffers.

However overlays operating on the data buffers will run slightly faster in NAME buffer 2, and overlays using the Serial Data Multiplexer (see LOTR 16) will run slightly faster in NAME buffer 1.

In the future, additional overlays are likely to be developed and described in other technical reports.

b) Polynomial Overlay

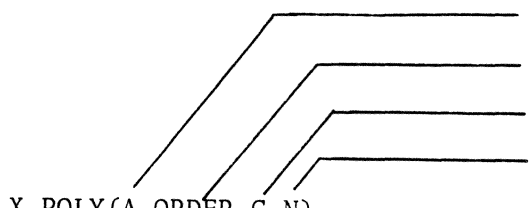


Position of coefficient (0<= POS<= 10)

Coefficient

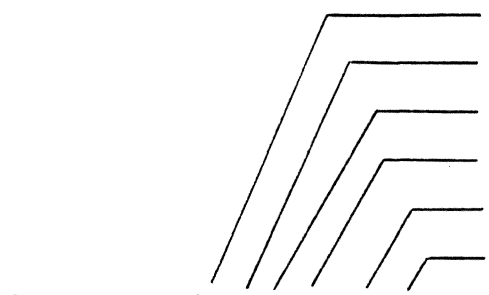
Sets coefficients for X POLY.

Note: No X NAME commands should be given in between setting coefficients with X SAV and using them with X POLY because the coefficients are stored in the overlay area with X SAV and POLY! NOTE SEMICOLON BEFORE COEFF.



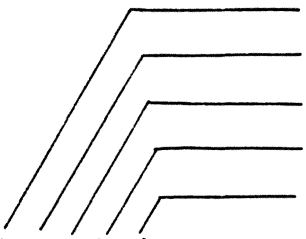
Generates a polynomial of order "ORDER" on contents of buffer A, starting at channel C and processing N channels. Coefficients are set using the X SAV command. Note: no X NAME commands should be given and in between setting coefficients with X SAV and using them with X POLY, because the coefficients are stored in the overlay area with X SAV and X POLY!

c) Dead Time, Extinction, Logarithm Overlay



SET D=FTINC(A,B,Z,CA,CB,N)

Extinction correction. Expects a table of $4096*K_\lambda$ in buffer B (K_λ = extinction coefficient for each channel). Data in buffer A is multiplied by $\exp(K_\lambda Z)$.
D is set to number of times overflow occurred.

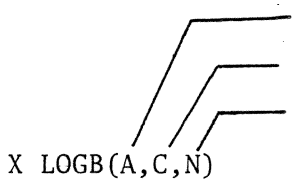


SET Q=FTIM(A,T,D,C,N)

Deadtime correction finds number of real counts from observed. For each channel, computes

$$N_R = \text{counts}* \left(1 + \frac{x}{1-x}\right)$$

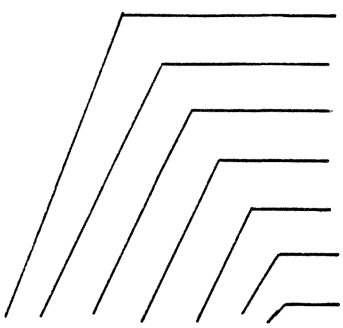
where x = (counts*4096/T)*D
Q is set to number of times overflow occurred.



X LOGB(A,C,N)

$$A(X) = 10000*\text{LOG } 10(A(X))$$

d) Peak Finding and Linearization Overlay




```
SET D1 = FIND(A,XP,CT,CD,MW,C,N)
SET D2 = FSIG(0) - Dummy argument
```

1st moment peak finder. This routine finds an accurate position for a peak whose approximate position is already known (see LOTR #12 pg. 14).

Returns:

$$D2 = \sum_{J=0}^{N} FCHAN(J+C)$$

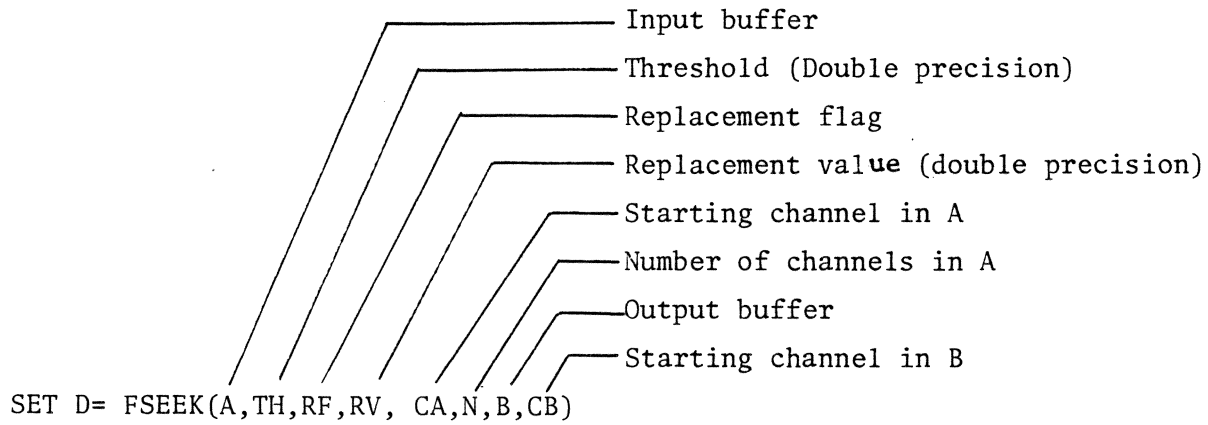
$$D1 = \sum_{J=0}^{N} J^{*}\ FCHAN(J+C)$$

or D1 < 0 if error occurs (overflow or peak too narrow)

FOCAL Routine to Use FIND/FSIG

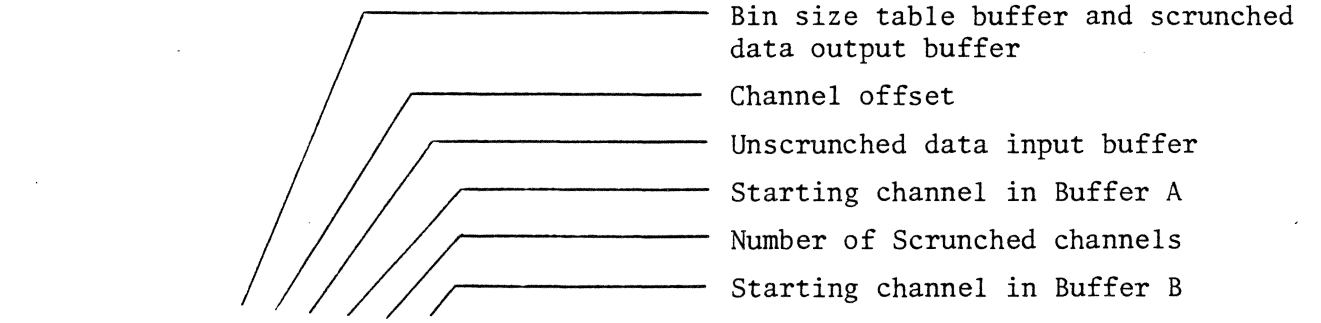
```
1.10 SET PK=FIND(A,...,C,N); IF (PK)1.2; SET PK=(PK/FSIG(0))+C; GO 1.3
1.20 C - error recovery
1.30 C - continue
```

Note that $\frac{\text{FIND}}{\text{FSIG}}$ returns number of peak position relative to channel C.



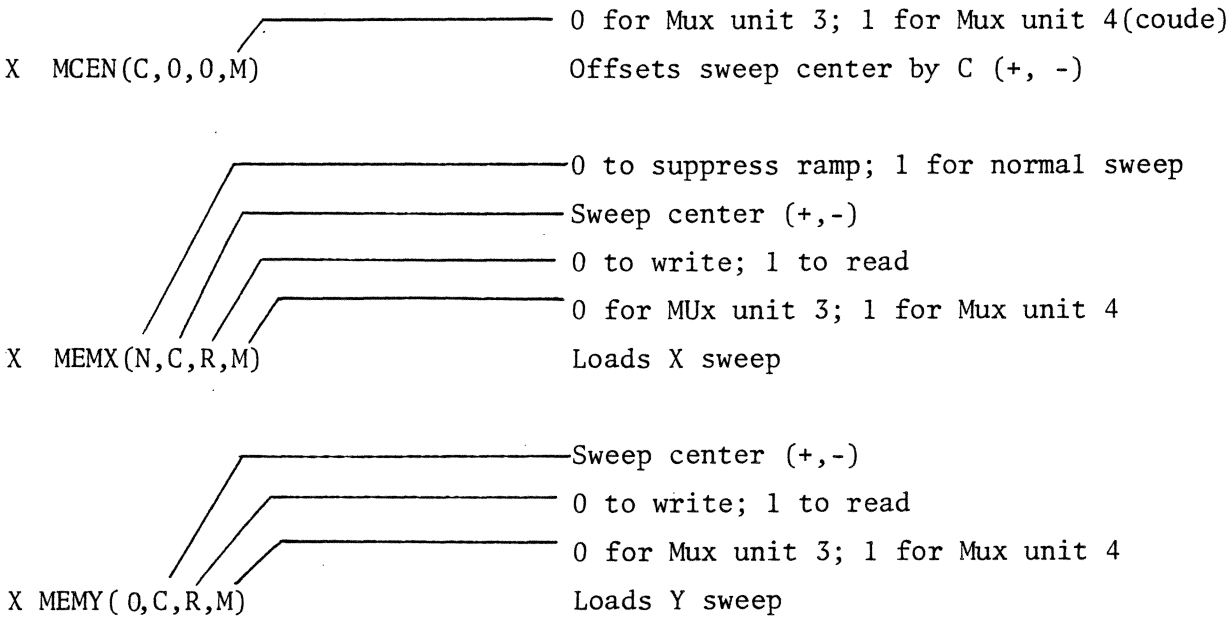
Course peak finder and wild point destroyer searches for peaks in buffer A starting at channel CA and searching N channels. (A peak is a series of consecutive channels all of whose values are >= to the threshold TH). The mean position of each peak (truncated to the nearest integer channel number) is stored in buffer B. The position of the Nth peak is stored in channel CB+N of buffer B. The position recorded for the peak is a point halfway between where the peak sticks up through the threshold and where it falls back below it again.

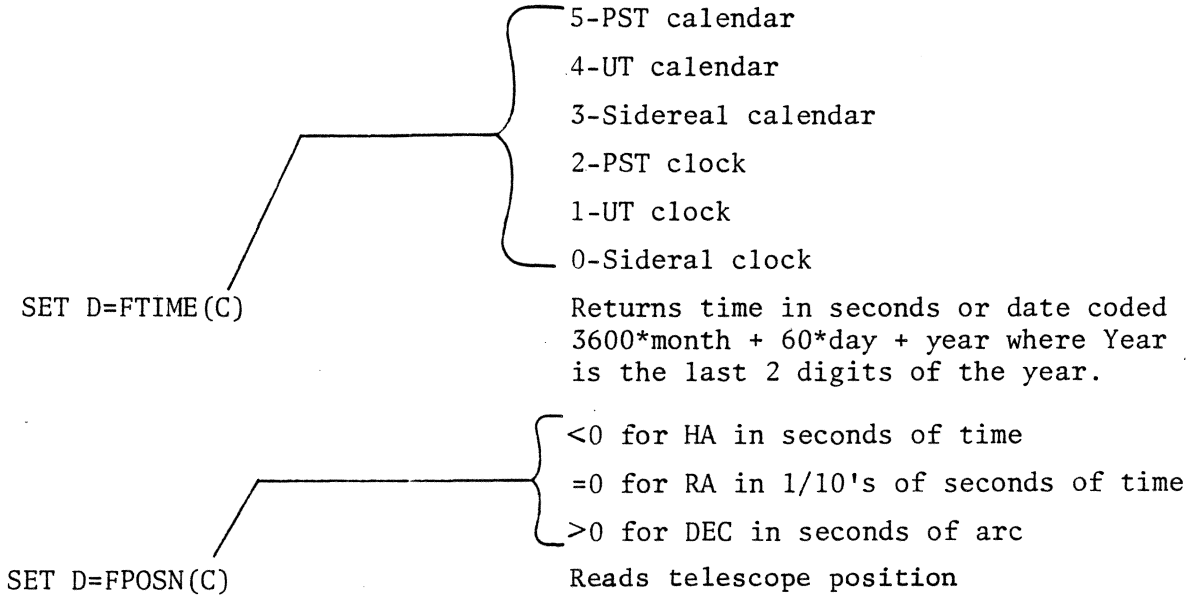
The number of peaks found is returned as D and is also stored in channel CB of Buffer B. If RF≠0, any point >TH is replaced with RV_1.



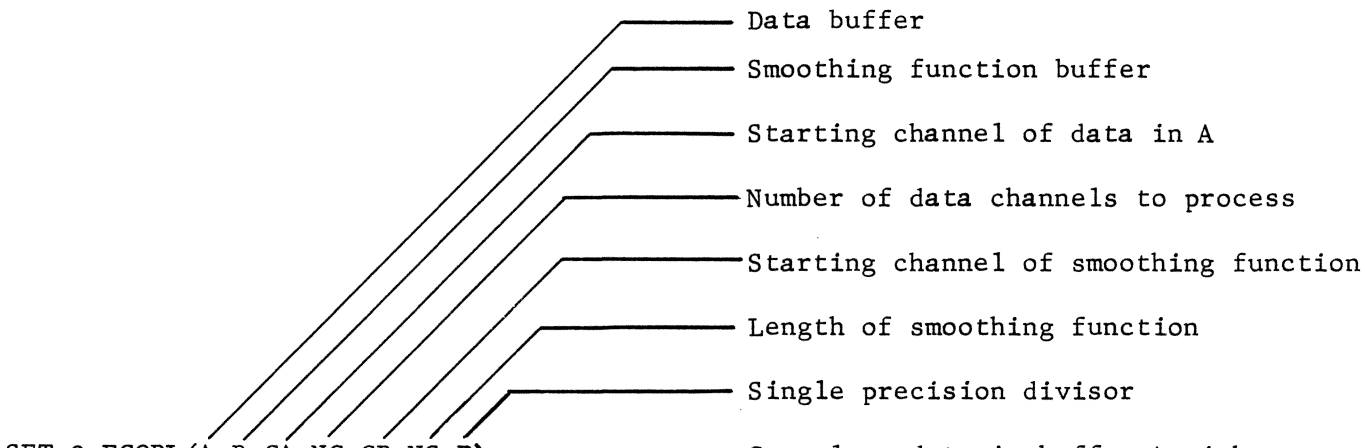
Scrunches data from buffer B to a table of bin sizes in Buffer A. W specifies the position in the unscrunched data of the left edge of bin 0, times 4096. For instance, the middle of channel 1 has W = 6144 (1.5*4096). The bin size table contains the number of unscrunched channels, times 4096, in each scrunched channel. Bin sizes must be greater than or equal to 0.

e) Sweep,Time, Position Overlay (Uses Serial Multiplexer)





f) Cross correlation overlay

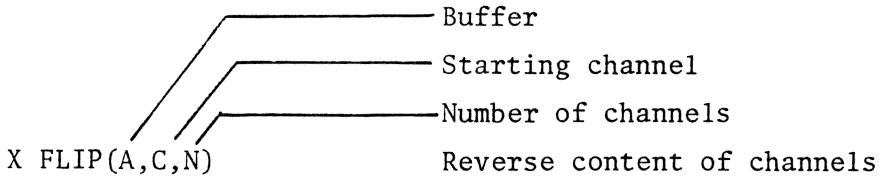


Computes:

$$A(J)=(A(J)*B(CB)+A(J+1)*B(CB+1)+\ldots$$
$$\ldots+A(J+NS-1)*B(CB+NS1))/D;\ \text{for } J=CA,CA+NC-NS$$
$$A(J)=0\ ;\ \text{for } J=CA+NC-NS+1,NC$$

Notes

1. Number of channels to process (NC) MUST BE NON-ZERO. NC=0 will give an error message. Use NC=2048 to process a full 2048 channel buffer.
2. Length of smoothing function (NS) MUST BE NON-ZERO. NS=0 will give an error message.
3. NS must be less than or equal to NC.
4. Single precision divisor D defaults to 1.
5. Result Q is set to number of times overflow occurred.

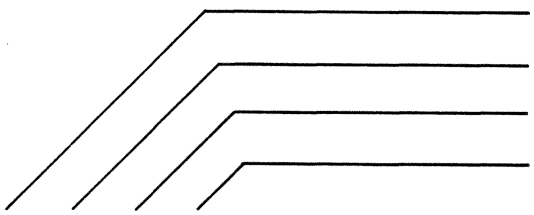


A(C) = A(C+N-1)

A(C+1) = A(C+N-2)

A(C+N-1) = A(C)

g) Debugging utility overlay



SET D=FZAP(FLD,LOC,VER,REP)

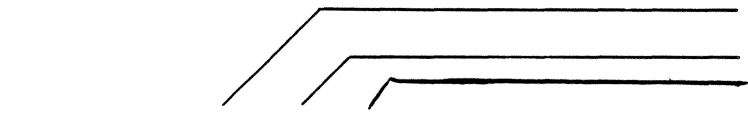
Inspects, verifies, and replaces specified location in core memory. Returns in decimal-coded octal the contents of location LOC in field FLD. FLD, LOC, VER, and REP are all specified in decimal-coded octal.
If either VER or REP is non-zero, replaces the contents of LOC in field FLD with the value REP, provided that the value of VER identically matches the current contents of LOC. If VER does not match, location LOC is not changed, and an error message is printed.
If FLD=LOC=VER=REP=0, returns in decimal-coded octal the contents of the next sequential location. This feature is useful for core dumps, and can spill across field boundaries.

SET D=FADDR(0) — Returns in decimal-coded octal the core address of the last location referenced with ZAP.

SET D=FIELD(0) — Returns in decimal-coded octal the memory field of the last location referenced with ZAP.

SET D=FDEC(X) — Returns decimal value of decimal-coded octal number X. TYPE FDEC(7777) would print 4095 on teletype. Only accepts numbers from 0-7777.

SET X=FOCT(D) — Returns in decimal-coded octal the value of decimal number D. TYPE FOCT(409 would print 7777 on teletype. Only accepts numbers from 0-4095.



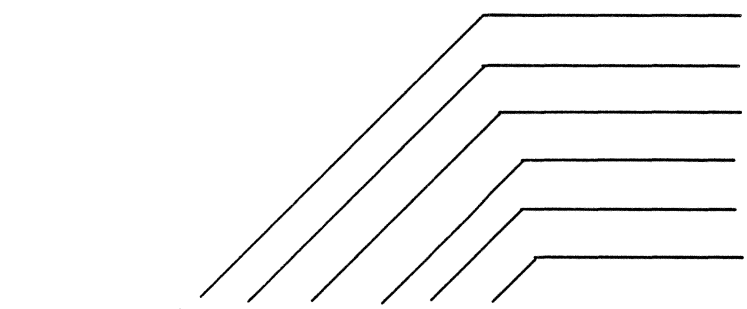
SET D=FMPX(FUNC,DATA ,BUS)

Transmits function word FUNC and data word DATA to serial multiplexer. FUNC and DATA are both specified in decimal-coded octal. Returns data received in decimal-coded octal .

SET D=FMPX(650) would read data from Cable 8, Unit 5.

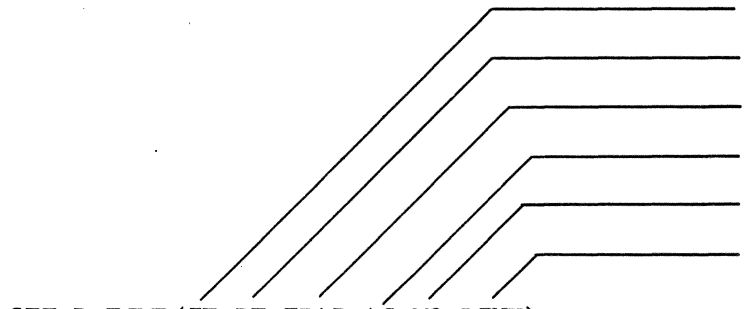
X MPX(4320,20) would send Pulse 1 to Cables 8 and 18, Unit 3.

See LOTR #16, page 5.



SET D=FJMS(IF,DF,EPAD,AC,MQ,LINK)

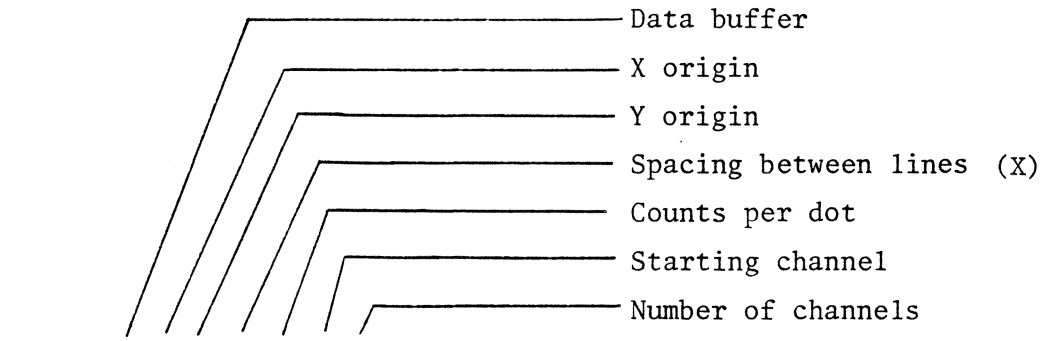
Does a PDP-8I JMS machine command to location EPAD in field IF. Immediately prior to executing the JMS the data field is set to DF, the AC to AC, the MQ to MQ, and the link to LINK. IF and DF must be in the range 0 to 7. EPAD, AC, and MQ are all specified in decimal-coded octal. LINK can only be 0 or 1. Result D is set to 24-bit contents of AC and MQ upon eventual return (if ever) to FOCAL.



SET D=FJMP(IF,DF,EPAD,AC,MQ,LINK)

Does a PDP-8I JMP machine command to location EPAD in field IF. DF, AC, MQ, and LINK are set as described in FJMS instruction above. It is unlikely that control will ever return to this command.

h) Intensity map overlay



Draw an intensity map. Uses low 12 bits of channel, only. Plots N vertical lines of varying density, starting at X,Y, with X spacing ST, intensity scale factor SC, starting at channel C in buffer A.

IV. 32K FOCAL UPDATE - June 20, 1977

Significant changes have been made to the 32K Focal System. The current version of 32K FOCAL (version 77B) is described in Lick Observatory Technical Report #21, "32K FOCAL User's Guide" (Feb. '77). Effective June 20, 1977, version 77B of 32K FOCAL was replaced by version 77C*. This update describes the differences between the two versions, and the updates to be made to LOTR #21 so that it will correspond to version 77C.

The differences between version 77B and 77C fall into the following areas:

I. Modifications/corrections of resident functions

II. Corrections of overlay functions

III. Addition of new resident functions

IV. Addition of new overlay functions

V. Relocation of resident functions

VI. Correction of inherited bugs.

Robert Kibrick

* The version number of any 32K FOCAL System DECtape or floppy disk appears in the title line that is printed whenever a FOCAL WRITE command is executed.

I. MODIFICATIONS/CORRECTIONS OF RESIDENT FUNCTIONS

A. Corrections

1. SET D = FMEMR (A,B,H) (See p. 23, LOTR #21)

   The "read only low 12 bits" option (H ≠ 0) now works correctly. In version 77B, use of this option would give correct results, but would actually run slower than if all 24 bits were read.

2. SET D = FCHEK (0) (See p. C-3, LOTR #21)

   Counting time display now counts down correctly to 0. In version 77B, counting time display would count down past 0 and leave a final display of 4095. Counting time now counts down in minutes/seconds; in version 77B countdown was in seconds only.

B. Modifications

1. X CALL (N,S,Q) (See pgs. 12-13, LOTR #21)

   a) Calls can no longer be nested to 10 levels as could be done in version 77B. Attempts to use the nesting option (Q ≠ 0) now generate a ?5422?00.00 error message. Nested calls can now be done using the new CALL/DO function (See X CADO command, pg. U7 of this update)

   Changes to text of LOTR #21

   1) p. 12, delete these 2 sentences under X CALL
      "If Q = 1, calls can be nested to 10 levels.
      Nesting cleared for Q = 0."

   2) p. B2, insert error code ?5422?00.00
      "5422 X CALL NESTING NO LONGER SUPPORTED. USE X CADO INSTEAD."

   3) p. B1, delete error code ?22.24.

b) Using the X CALL (or X END) command from inside a DO group or FOR loop has always been illegal (see LOTR #1, pg. Z9, paragraph C). However, such illegal uses were never detected in either 8K FOCAL or 32K FOCAL version 77B, and would cause unpredictable results. In version 77C, such illegal uses of X CALL or X END are detected, and a ?5440?00.00 error message is printed.

Changes to text of LOTR #21

1) p. B2, insert error code ?5440?00.00:

"5440 X CALL or X END FROM WITHIN A DO GROUP OR FOR LOOP"

c) In both 8K and 32K FOCAL version 77B, one could easily wipe out many minutes worth of keypunching FOCAL text if one accidentally started a new (or modified) program that contained an X CALL (or X END) command, before one had filed the new program on DECtape or floppy disk. In version 77C:

If FOCAL's text area contains new or modified text that has not yet been filed, then execution of X CALL (or X END) will cause FOCAL to type

REALLY?

on the teletype, and then to wait for a reply. Any reply other than 'Y' will cause the X CALL (or X END) command to be supressed, and the contents of the text area to be preserved. A reply of 'Y' will cause the X CALL (or X END) to proceed, and the contents of the text area to be lost.

Changes to text of LOTR #21:

1) Cut off bottom half of this page; paste on pg. 12, LOTR #21 over X CALL (N,S,Q)

2. X END (O) (See pg. 11, LOTR #21)

Modifications b) and c) as described under X CALL above apply also to X END.

3. X FILE (N) (See pg. 13, LOTR #21)

The contents of FOCAL's text area is written to DECtape or floppy disk as program N; then the program N just written out is read back into FOCAL's text area, as if an X CALL (N) had been used. In version 77B, the text area is written out but not read back. The user should not notice any real difference between the two versions, except that the version 77C X FILE command will take slightly longer.

Place over bottom half of page 12, LOTR #21

Cut here

X CALL(N,S)

(For greater versatility in calling, see the X CADO command, pg. U7)

- Call Program N from Dectape. If S > 0, starts program N at subroutine S. (Code S*128 + L to start program N at line L of subroutine S). Use X END(0) to return to the line following the X CALL command in the calling program.

Important Notes:

A) It is illegal to use either X CALL or X END from inside a DO group or FOR loop.

B) If FOCAL's text area contains new or modified text that has not yet been filed (See X FILE command below), then execution of X CALL or X END will cause FOCAL to type "REALLY?" on the teletype, and then to wait for a reply. Any reply other than "Y" will cause the X CALL or X END command to be supressed, and the contents of the text area to be preserved. A reply of "Y" will cause the command to proceed and the contents of the text area to be lost.

C) If X CALL is used to address a program area on the Dectape that does not contain a FOCAL program, the following action is taken:

II. CORRECTIONS TO OVERLAY FUNCTIONS

A. Sweep, Time, Position Overlay (See p. 32, LOTR #21)

The commands in this overlay have been changed so that they work correctly in either NAME buffer. This new overlay (dated 5/16/77) will not work if used with version 77B FOCAL. The old Sweep, Time, Position overlay (dated 2/14/77) will not work with version 77C FOCAL. The old Sweep, Time, Position overlay worked correctly if used in NAME buffer 1, but caused occasional MUX errors if used in NAME buffer 2. When copying version 77C to your FOCAL tape, be sure to copy an updated version of this overlay.

Changes to text of LOTR #21

p. B3: Change error code 6171 to 6165, 6174 to 6170.

B. DebuggingUtility Overlay (See p. 34, LOTR #21)

1. The X MPX command in this overlay has been changed so that it works correctly in either NAME buffer. (The X MPX command had the same problem as the commands in the Sweep, Time, Position overlay, and the same considerations apply. The new Debugging Utility Overlay is dated 5/4/77; the old one was dated 2/23/77).
2. The X MPX command no longer hangs up the spectrograph control panel at the 120" readout room; the previous version did.

III. ADDITION OF NEW RESIDENT FUNCTIONS

A. X BRK(0)

One of the problems with FOCAL is that:

> "If a GOTO or IF command that is inside a DO group transfers control to a line outside the DO group, that line is executed and control then returns to the command following the DO." (see DEC. FOCAL manual)

A similar problem exists with GOTO or IF statements that are used inside of FOR loops. This feature of FOCAL makes it awkward to bail out of a DO group or FOR loop, as is illustrated by the following example program.

```
*01.10 FOR J=1,3;DO 2
*01.20 TYPE !"L 1.2"
*01.30 TYPE !"L 1.3";QUIT

*02.10 TYPE %2,J;DO 3
*02.20 TYPE !"L 2.2"

*03.10 IF (J-2)3.3,3.2,3.3
*03.20 TYPE !"L 3.2"
*03.25 GO 1.2
*03.30 TYPE !"L 3.3"
*GO
  1
L 3.3
L 2.2   2
L 3.2
L 1.2
L 2.2   3
L 3.3
L 2.2
L 1.2
L 1.3*
```

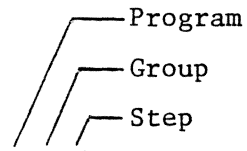
The X BRK command solves this problem by purging FOCAL's pushdown list, which is where FOCAL stores information relating to the nesting of DO groups and the control of FOR loops.

The effect of the X BRK command is to break one out of the inside of any DO groups or FOR loops, as illustrated by the example below. Note that any commands on the same line following an X BRK command are ignored, and execution continues with the next sequential line.

```
*01.10 FOR J=1,3;DO 2
*01.20 TYPE !"L 1.2"
*01.30 TYPE !"L 1.3";QUIT

*02.10 TYPE %2,J;DO 3
*02.20 TYPE !"L 2.2"

*03.10 IF (J-2)3.3,3.2,3.3
*03.20 TYPE !"L 3.2";X BRK(0);TYPE "ABC"
*03.25 TYPE !"L 3.25";GO 1.2
*03.30 TYPE !"L 3.3"
*GO
  1
L 3.3
L 2.2   2
L 3.2
L 3.25
L 1.2
L 1.3*
```



B. X CADO (P,G,S)

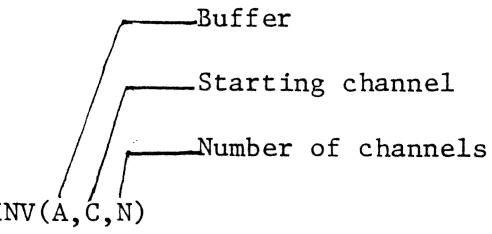
This command functions exactly like a FOCAL DO command, except it allows one to "DO" text that is in another FOCAL program. Program P is called, then FOCAL 'DO'es group G, step S, as if it were part of the calling program. If S = 0, all of group G is "done." If G = S = 0, all of program P is "done". When the specified text has been "done," control returns to the statement following X CADO in the calling program. The X CADO command can be used anywhere a FOCAL DO command can be used, and X CADO's can be nested. The degree of nesting allowed depends upon space available in the pushdown list; X CADO requires 1 more word of pushdown list space than does a regular FOCAL DO statement.

Note - If FOCAL's text area contains new or modified text that has not yet been filed, then execution of an X CADO command will result in a ?3336?00.00 error.

Changes to text of LOTR #21

p. B2, insert error code 3336

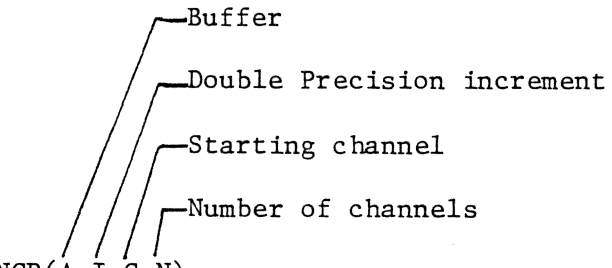
"3336 X CADO EXECUTED FROM MODIFIED TEXT BUFFER. USE X FILE FIRST."



C. X INV(A,C,N)

Negative of buffer

A(X) = -A(X); for X = C to C + N - 1

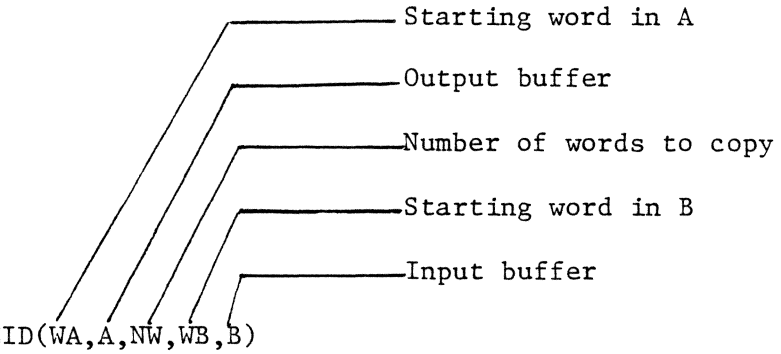


D. X INCR(A,I,C,N)

Increment a buffer by a constant

A(X) = A(X) + I; for X = C to C + N - 1

FINCR(...) returns overflow count.

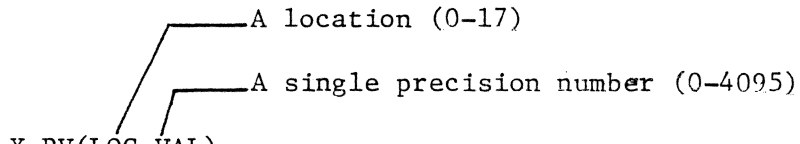


E. X CID(WA,A,NW,WB,B)

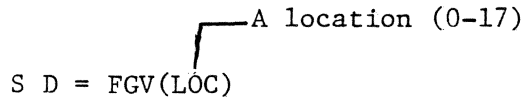
Copies NW words of I.D. information from I.D. buffer B starting at word WB to I.D. buffer A starting at word WA.

F. 2 new functions, X PV and GV allow one to store 18 single precision numbers into an area of memory that is safe over both power failures

and re-bootstrapping on any of our PDP-8s.



Stores value into indicated location.



Sets D equal to value of location LOC.

G. Occasionally, one needs to execute a sequence of FOCAL commands without interruption. The X IOF command allows one to prevent FOCAL from responding to the CONTROL/C key, while leaving the computer's interrupt enabled.

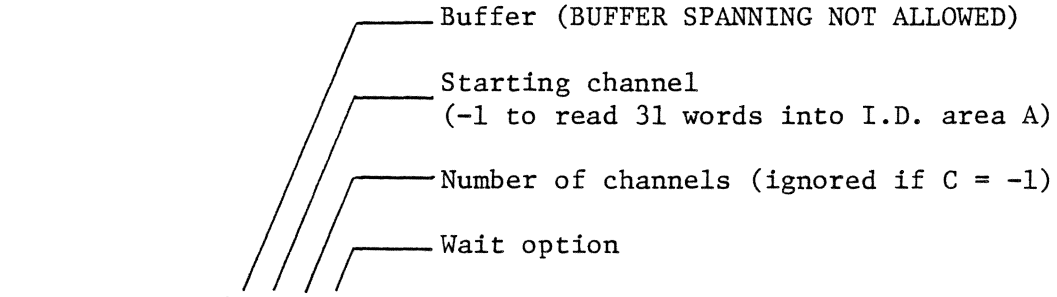
X IOF(0) Causes FOCAL to ignore any subsequent CONTROL/C's until the next X IOF (-1) command.

X IOF(-1) Causes FOCAL to respond to any subsequent CONTROL/C's until the next X IOF(0) command.

If one has mistakenly set X IOF(0) and needs to abort the FOCAL program, hit the STOP button on the CPU. Then set switch registers to octal 0200, hit load address, and START. This implicitly executes an X IOF(-1) and executes the same restart sequence as if FOCAL had been aborted with CONTROL/C. One need not re-bootstrap.

IV. ADDITION OF NEW OVERLAY FUNCTIONS

A. Quick Data Overlay



Starts reading N channels from IBM tape to buffer A. Reads to end of buffer if N = 0. Works like READ command except:

1) Control returns to FOCAL as soon as the IBM tape starts moving. Data transfer can go on in parallel with computing. (One must be careful not to use the portion of buffer A into which the data is being transferred until the transfer is completed! Use DONE * command to check for completion.)

2) D returns the status of any previous IBM read or write operation, not the status of this command. See DONE command for description of status value returned.

3) If the tape drive is still busy processing a previous RDQ or WRQ command when the current command is executed, three options are available

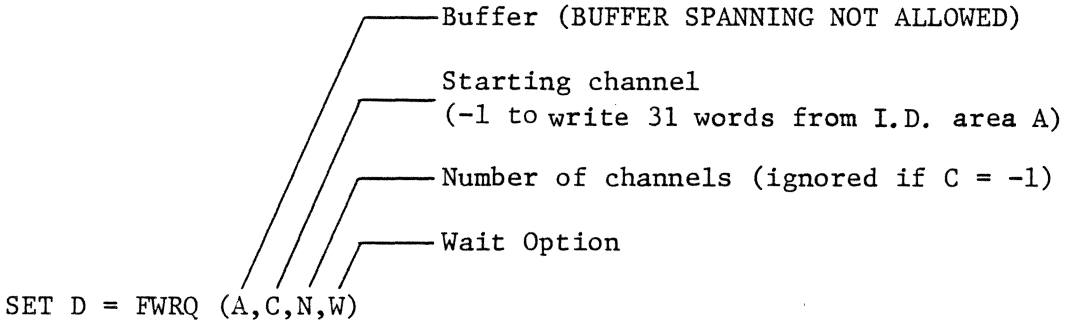
   W = 0 A ?6062?00.00 error is printed and the FOCAL program is terminated

   W < 0 The current command is ignored and control returns immediately to the FOCAL program. D is set -1

*The DONE command is described on page U10

W > 0 The current command waits for the previous IBM tape operation to complete. Then the current command is carried out.

4) If the IBM tape is at END OF TAPE (EOT), the tape drive will appear "busy". The same actions are possible as described in 3) above, except that in the case of W < 0, D will be set to -2.



Starts writing N channels to IBM tape from buffer A, starting in channel C. Writes to end of buffer if N = 0. Works like WRIT command except for the same 4 differences described under the RDQ command above. Also, if a WRQ command is directed to a write protected IBM tape, a ?6273?00.00 error is given.

Notes: The RDQ and WRQ commands can be used interchangeably with READ and WRIT, and are fully compatible with all other resident IBM tape commands, with the exception that one should allow approximately a 20 millisec delay after using X BAK before using X RDQ or X WRQ.

Changes to text of LOTR #21:

p. B3 - add error codes 6062 and 6273

"6062 RDQ or WRQ USING 'NO WAIT' OPTION FOUND IBM TAPE BUSY
6273 WRQ FOUND IBM TAPE WITHOUT WRITE RING"

SET D = FDONE(0)

Returns status of last READ or WRITE operation.

D = 0 if last RDQ or WRQ operation completed successfully

D < 0 if last RDQ or WRQ operation has not yet completed

D > 0 if tape error on last RDQ or WRQ, or if last record was a file mark.

The low 12 bits of D contain the remaining word count

The high 12 bits of D contain:

BIT 11 is 1 if last record was file mark (4096 bit)

BIT 10 is 1 is parity error on last record (8192 bit)

BITS 1-9 unused

BIT 0 is 0

SET D = FADDQ (A) ← Number to be queued. ($0 \leq A \leq 4095$)

Appends A to the end of a 5 element queue.*

D = 0 if successful.

If queue is already full, D is set < 0 and queue remains unchanged.

SET D = FDLTQ (0)

Sets D to the value of the front element of a 5 element queue*, then removes this element from the queue.

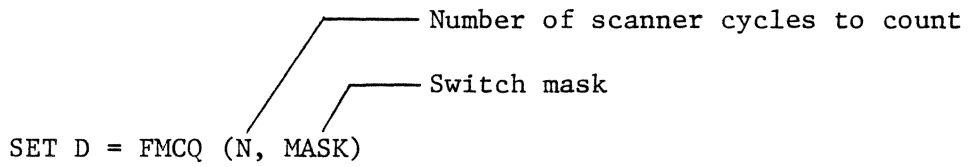
Sets D < 0 if queue already empty.

Notes: 1. ADDQ and DLTQ are useful for maintaining a queue of buffers to be read or written using the RDQ and WRQ commands. By queueing data, one can handle instantaneous data rates

* A queue works on the basis of first in, first out.

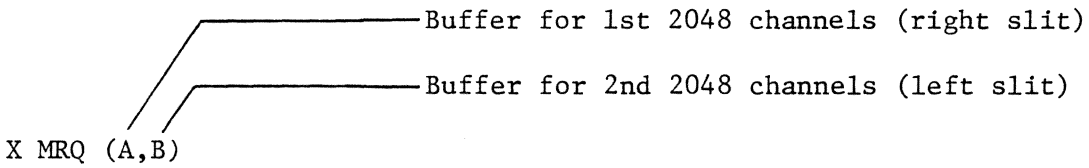
that exceed that average data transfer rate of the IBM tape drive. Such situations occur in high speed data taking applications.

2. The storage area used for the queue is contained within the overlay. The queue is initially empty when the overlay is brought in with X NAME. No X NAME commands should address the buffer containing this overlay in between saving queue elements with ADDQ and retrieving them with DLTQ.



If scanner is already counting or if N < 0, returns remaining counting time, and leaves scanner unchanged. If scanner is not counting and N $\geq$ 0, then

1) Waits for a pulse to be detected on the switch(es) in group 3 selected by MASK. (External clock pulses can be brought in across switches 3.8, 3.9, and 3.10. Pulse width should be at least 30 μsec)
2) Waits to synchronize with scanner memory (max. 4.4 millisec). D returns time spent waiting to synchronize with memory in units of 13.25 μsec.
3) Once synchronized, sets counting time of N scanner cycles and starts counting. 1 scanner cycle is $\approx$ 4.4 μsec. (N = 0 gives a counting time of $\approx$ 10 hours).

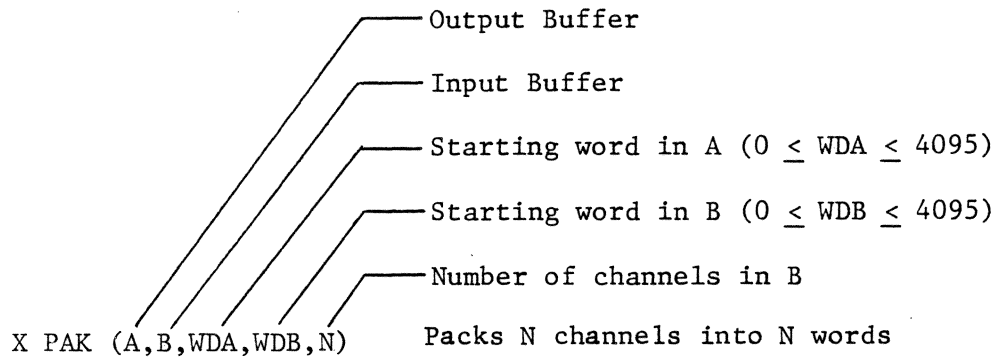


Wait for current scan to complete, then read scanner memory to core.

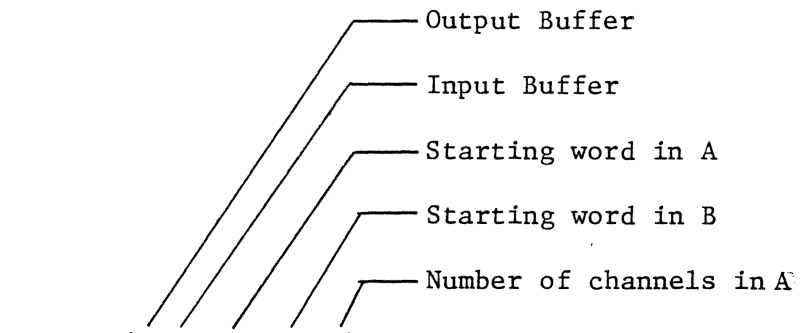
First 2048 channels read to buffer A, (starting in channel 0) and erased from scanner memory as they are read (Not read or erased if A = -1). Similarly for 2nd 2048 channels and buffer B. Reads all 24 bits normally.

To read only the low 12 bits of each scanner channel, and to pack 2 of these 12 bit scanner channels per 24 bit buffer channel, add 4096 to the buffer number.

B. Data Compaction Overlay



Each channel in B consists of 2 12-bit words. Starting at WDB in buffer B and processing N such channels, the low order word of each channel is stored into consecutive words in buffer A, starting at WDA. The high order words of each channel in B are ignored.



Reverse of PAK.

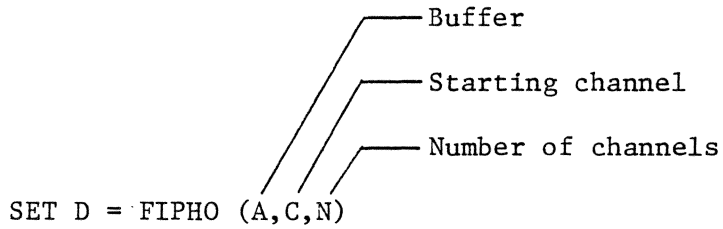
Starting at WDB in buffer B and processing N words, appends a high order word of 0 to each word and stores into successive channels in buffer A, starting at WDA.

Changes to text of LOTR #21:

p. B3 - add error code 6255

"6255 starting word(s) <0 or >4095 in PAK or UNPK"

C. Isophote Display Overlay (Insert as pg. 38 of LOTR #21)



Takes data values in the range 0, 1, or 2 and displays them as PIXELS of 3 different densities on CRT.

0 displays as a blank PIXEL.

1 displays as a lined PIXEL.

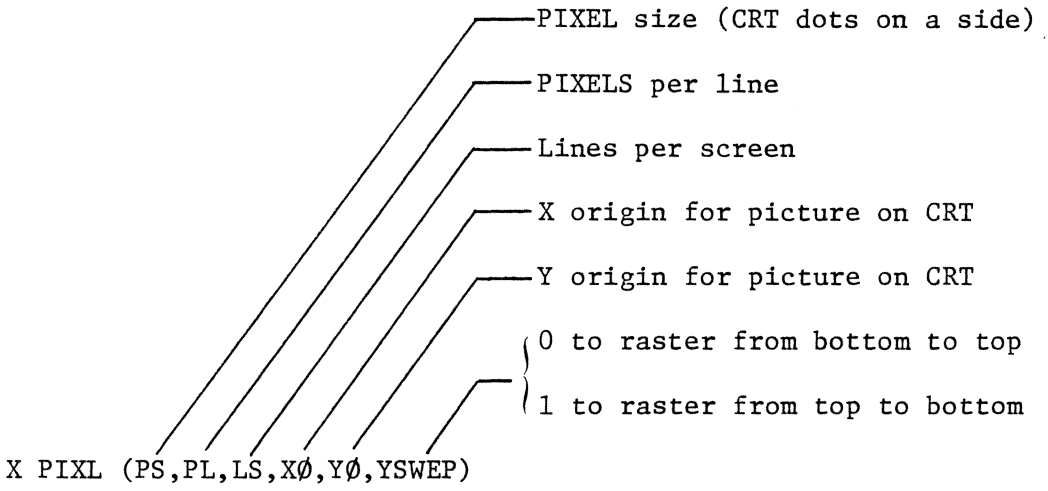
2 displays as a solid PIXEL.

Display consumes 1 channel of data per PIXEL, sweeping the CRT in a raster pattern. PIXEL size and CRT screen parameters are set using the X PIXL command. D is set to number of PIXELS displayed by this command.

Changes to text of LOTR #21:

p. B3 insert error code 6344.

"6344 screen full in XIPHO or XPIXL not called first."



Defines PIXEL size and screen parameters used by IPHO command.

The input arguments must pass the following tests:

1) PS must be 2 or a multiple of 4

2) PL, LS, XØ, YØ must all be $\geq$ 0

3) PL*PS+XØ must be $\leq$ 1024

4) LS*PS+YØ must be $\leq$ 1024

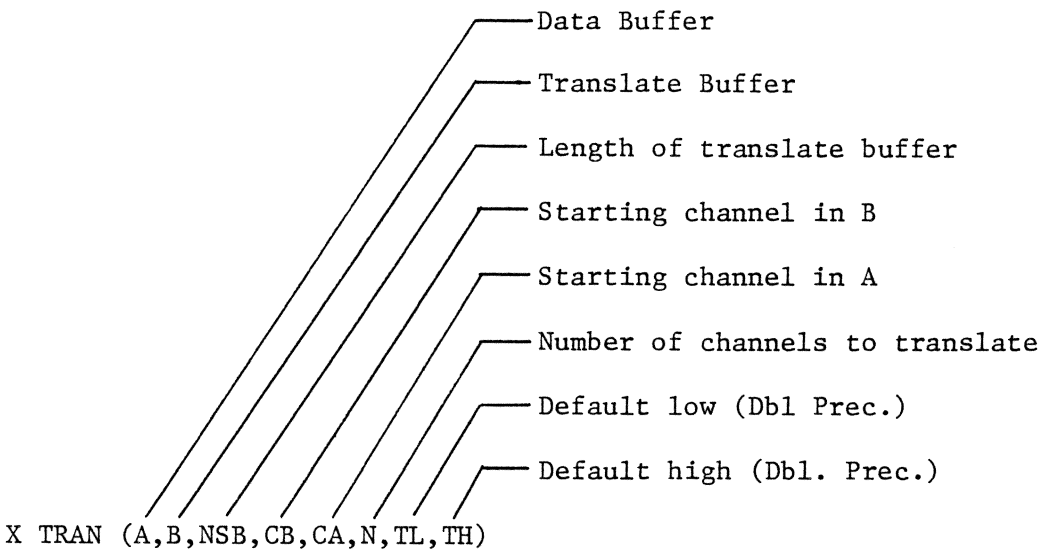
Changes to LOTR #21:

p. B3 insert error codes 6301, 6302, 6303

"6301 Bad PIXEL size in X PIXL

6302 Bad combination of PL, PS, XØ in X PIXL

6303 Bad combination of LS, PS, YØ in X PIXL"



This command allows one to apply an arbitrarily defined function to a buffer of data. The function values are assumed to have been previously computed and stored into the translate buffer. The function value stored into a given channel of the translate buffer is simply the value obtained by applying the function to the number of the channel. Specifically, the TRAN command operates as follows:

Translates buffer A, starting at channel CA and translating N channels.

Buffer B contains the translate table, starting in channel CB and running NSB channels long. The following operation is performed on each indicated channel:

The value of the channel in buffer A is used as an index into the translate table in buffer B. If this index value is >0 and <NSB, the value of the indexed channel in buffer B is used to replace the value of the indexing channel in buffer A. If the value of the indexing channel in buffer A is <0, then it is replaced with the double precision value TL. If the value is > = NSB, then it is replaced with the double precision value TH.

V. RELOCATION OF RESIDENT FUNCTIONS

The starting addresses of the following resident functions have been moved. In some cases, this has caused a corresponding change in the error messages produced by these functions. Appendices A1-A3, and B2 should be updated accordingly.

NAME	OLD ADDR.	NEW ADDR.	OLD ERROR	NEW ERROR
ADV	5257	5256	--	--
AND	3730	4770	--	--
CHEK	3722	3732	--	--
END	5537	5550	--	--
FILE	5473	5501	--	--
PAUS	3630	3627	--	--
RWND	5121	5120	--	--
VAR	5260	5255	5273	5270
			5277	5274

VI. CORRECTION OF INHERITED BUGS

A. If one used a DO statement to invoke a single line that contained an error, FOCAL diagnosed the error as occuring on the line containing the DO, and not on the line in which the error actually occurred.

Example

```
*E  A
*1.2   DO 2.1
*1.3   C
*2.1   TYPE A/Ø
*G
 ?28.73 @ 1.20  error should be diagnosed on line 2.1
```

B. The Lick FOCAL functions X DO and X GO did not behave the same as the standard FOCAL DO and GO commands. In particular, these commands had the same effect as the new X BRK command. That is, X DO and X GO would purge FOCAL's pushdown list and cause the program to break out of any DO groups or FOR loops in which it was nested. Also, any text following X DO on the same line was ignored. In 32K FOCAL version 77C, X DO and X GO now work exactly like FOCAL's DO and GO commands.

C. If the X EOF(0) command was immediately followed by an X BAK(1,1) command, one would fail to backspace behind the file mark. Now fixed.

D. If an IBM tape were backspaced to beginning of tape (BOT) using the X BAK command, or rewound using the X RWND command, the IBM tape drive was erroneously left in the "start" state. If another IBM

tape command immediately followed, it could possibly start too soon and cause I/O errors. This problem fixed in version 77C.

APPENDIX B1 - ERROR DIAGNOSTICS OF FOCAL 1969

ADDR	CODE	MEANING
	?00.00	CONTROL-C FROM KEYBOARD OR MANUAL START
0250	?01.40	ILLEGAL STEP OR LINE NUMBER USED
0316	?01.78	GROUP NUMBER IS TOO LARGE
0340	?01.96	DOUBLE PERIODS FOUND IN A LINE NUMBER
0351	?01.↕5	LINE NUMBER IS TOO LARGE
0362	?01.↓4	GROUP ZERO IS AN ILLEGAL LINE NUMBER
0440	?02.32	NONEXISTANT GROUP REFERENCED BY 'DO'
0464	?02.52	NONEXISTANT LINE REFERENCED BY 'DO'
0517	?02.79	STORAGE WAS FILLED BY PUSH-DOWN LIST
0605	?03.05	NONEXISTANT LINE USED AFTER 'GOTO' OR 'IF'
0634	?03.28	ILLEGAL COMMAND USED
1047	?04.39	LEFT OF "=" IN ERROR IN 'FOR' OR 'SET'
1064	?04.52	EXCESS RIGHT TERMINATORS ENCOUNTERED
1074	?04.60	ILLEGAL TERMINATOR IN 'FOR' COMMAND
1147	?04.↕3	LINE TOO LONG
1345	?05.↕1	PROGRAM TOO LONG
1260	?05.48	BAD ARGUMENT TO 'MODIFY'
1406	?06.06	ILLEGAL USE OF FUNCTION OR NUMBER
1466	?06.54	STORAGE IS FILLED BY VARIABLES
1626	?07.22	OPERATOR MISSING IN EXPRESSION OR DOUBLE 'E'
1646	?07.38	NO OPERATOR USED BEFORE PARENTHESIS
1753	?07.↕7 (?07.↕9)	NO ARGUMENT GIVEN AFTER FUNCTION CALL
1764	?07.↓6	ILLEGAL FUNCTION NAME OR DOUBLE OPERATORS USED
2057	?08.47	PARENTHESIS DO NOT MATCH
2213	?09.11	BAD ARGUMENT IN 'ERASE'
2551	?10.↕5	STORAGE WAS FILLED BY TEXT
5042	?20.34	LOGARITHM OF ZERO REQUESTED
5175	?20.<5	'IF' ARGUMENT NOT ENCLOSED IN (), [], OR <>
5644	?23.36	LITERAL NUMBER IS TOO LARGE
6543	?26.99	^ POWER IS TOO LARGE OR NEGATIVE
7111	?28.73	DIVISION BY ZERO REQUESTED
7405	?30.05	IMAGINARY SQUARE ROOTS REQUIRED
7541	?30.97	EXCESSIVE RIGHT TERMINATORS IN 'TYPE'
	?31.<7	ILLEGAL CHARACTER, UNAVAILABLE COMMAND, OR UNAVAILABLE FUNCTION USE.
7626	?31.22	ATTEMPT TO USE FOCAL 'L' COMMAND (TAPE COPIER) AFTER TOO MUCH TEXT ENTERED

IF PUSH DOWN LIST OVERFLOWS, TRY 'TYPE $' TO SEE IF UNWANTED VARIABLES ARE PRESENT. USE 'ERASE', TO REMOVE VARIABLES, BUT RETAIN PROGRAM.

*

APPENDIX B2 - 32K FOCAL RESIDENT FUNCTION ERROR DIAGNOSTICS

THE FOLLOWING ERROR DIAGNOSTICS PRINT OUT AS ?NNNN?00.00

CODE	MEANING
0317	ILLEGAL RECURSIZE USE OF A LICK FOCAL FUNCTION
0455	ATTEMPT TO PASS A NUMBER > IN MAGNITUDE THAN 2^23-1 (>8388607)
2002	ATTEMPT TO USE TAPE COPIER OPTION 1 ON PDP8 WITH <12K OF CORE
3336	X CADO EXECUTED FROM MODIFIED TEXT BUFFER. USE X FILE FIRST
4360	FLOATING POINT ARGUMENT TO LICK FOCAL FUNCTION NOT LAST IN LIS
4411	SEMICOLON MISSING IN X STOR(B,W,U,D). SHOULD BE X STOR(B,W,U;D
4560	PUT, STOR, OR MSAV TRIED TO WRITE TO PROTECTED AREA ON TAPE #0
4567	DECTAPE/FLOPPY BLOCK NUMBER IS <0 OR >1481
4573	DECTAPE I/O ATTEMPTED BEFORE FLOPPY I/O DONE OR ACKNOWLEDGED
4615	ILLEGAL UNIT NUMBER SPECIFIED IN PUT, TAK, ASK, OR STOR
4623	BLOCK NUMBER SPECIFIED IN PUT, TAK, ASK, OR STOR IS <0 OR >147
4642	UNIT # CHANGED DURING USE OF PUT, TAK, ASK, OR STOR WITH B=W=0
4647	WORD NUMBER < 0 SPECIFIED IN PUT, TAK, ASK, OR STOR.
4655	WORD NUMBER > 190145 SPECIFIED IN PUT, TAK, ASK, OR STOR
4660	COMBINATION OF WORD AND BLOCK #S TOO BIG (PUT,TAK,ASK, OR STOR
5046	NEGATIVE CHANNEL NUMBER SPECIFIED
5071	NEGATIVE BUFFER NUMBER SPECIFIED
5103	COMBINATION OF BUFFER & STARTING CHANNEL > AVAILABLE MEMORY
5105	NUMBER OF CHANNELS TO PROCESS > TOTAL AVAILABLE CHANNELS
5110	SAME AS EITHER 5103 OR 5105
5216	ATTEMPT TO USE X NAME BUFFER 2 ON PDP8 WITH < 12K OF CORE
5225	NEGATIVE OVERLAY NUMBER REQUESTED
5260	LOCATION < 03200 (OCTAL) SPECIFIED IN X VAR
5263	LOCATION > 04600 (OCTAL) SPECIFIED IN X VAR
5406	NEGATIVE PROGRAM # SPECIFIED IN X CALL, X FILE, OR X CADO
5424	X CALL NESTING NO LONGER SUPPORTED. USE X CADO INSTEAD.
5442	X CALL OR X END USED WITHIN A DO GROUP OR FOR LOOP
7027	ATTEMPT TO USE X NAME BUFFER 2 WHEN THIS FEATURE IS DISABLED
7050	BAD OVERLAY FOUND ON TAPE - OR OVERLAY FOUND IS NOT 32K FOCAL
7253	NON-EXISTENT PROGRAM CALLED WITH X CALL OR X CADO
7267	TOO MANY DECTAPE/FLOPPY DISC ERRORS
7410	ATTEMPT TO USE FIELD 2 ROUTINES ON PDP8 WITH < 12K OF CORE

THE FOLLOWING ERROR CODES PRINT OUT A ?2-NNNN?00.00

2-1006	SINGLE PRECISION DIVISOR IN DMUL IS <0 OR >4095
2-1260	MICROPHOTOMETER OVERLAY FUNCTION USED FROM SCANNER FOCAL
2-2013	WORD COUNT > 2048 SPECIFIED IN X EXCR
2-2120	DISASTER IN 32K TO 8K TAPE FORMAT TRANSLATOR. REPORT THIS
2-2511	ILLEGAL UNIT NUMBER SPECIFIED IN MGET OR MSAV
2-2542	NON-EVEN WORD COUNT SPECIFIED IN X EXCR AND MODE NOT 0
2-5151	GAP NOT FOUND WHEN WRITING IBM TAPE. CHECK FOR DIRTY CAPSTAN
2-7003	NUMBER OF WORDS <0 IN X CID
2-7011	# OF CHARACTERS >62 IN TYCO OR COTY, OR # OF WORDS >31 IN CID
2-7032	# OF CHARACTERS <0 IN TYCO OR COTY
2-7363	NEGATIVE BUFFER NUMBER SPECIFIED
2-7370	BUFFER SPECIFIED NOT AVAILABLE ON THIS SIZE MACHINE
2-7432	ITAK OR IPUT TRIED TO READ OR WRITE 24-BIT # OFF ID BUFFER END
2-7441	ID BUFFER WORD # < 0 IN IPUT, ITAK, TYCO, COTY, OR CID
2-7444	ID BUFFER WORD # > 30 IN IPUT, ITAK, TYCO, COTY, OR CID
2-7450	BUFFER < -1 SPECIFIED IN IPUT, ITAK, TYCO, COTY, OR CID
2-7453	BUFFER > 4 SPECIFIED IN IPUT, ITAK, TYCO, COTY, OR CID
2-7474	LOCATION < 0 OR > 17 SPECIFIED IN X PV OR GV

APPENDIX B3 - 32K FOCAL OVERLAY FUNCTION ERROR DIAGNOSTICS

THE FOLLOWING ERROR DIAGNOSTICS PRINT OUT EITHER AS
?NNNN?00.00 OR AS ?2-NNNN?00.00

CODE	MEANING
6045	AREA SCANNER NOT ATTACHED OR NOT TURNED ON
6046	NUMBER OF CHANNELS < 0 OR > 4095 IN FIND
6051	COEFFICIENT POSITION > 10 IN X SAV OR X COEF
6053	SINGLE PRECISION DIVISOR IN X CORL IS < 0 OR > 4095
6055	MINIMUM PEAK WIDTH < 0 OR > 4095 IN FIND
6056	COEFFICIENT IN X SAV OR X COEF NOT PRECEDED BY ;
6060	VALUE IS < 0 OR > 4095 IN X PUTW
6062	X RDQ OR X WRQ WITH 'NO WAIT' OPTION FOUND IBM TAPE BUSY
6102	STARTING WORD (W) < 0 IN X GETW, PUTW, COPY, RBLK OR WBLK
6105	NEGATIVE DATA FOUND BY X LOOK
6106	SMOOTHING FUNCTION OF LENGTH 0 SPECIFIED IN X CORL
6113	PEAK CHANNEL > 4095 CHANNELS AWAY FROM STARTING CHANNEL IN FIN
6120	INVALID UNIT IN RIT, LFT, UP, DN
6124	# OF CHANNELS TO PROCESS < SMOOTHING FUNCTION LENGTH IN X CORL
6134	X POSF TRIED TO POSITION TO PROTECTED AREA OF FLOPPY ON UNIT 0
6137	X BIN OUT OF ROOM FOR OUTPUT DATA
6140	X POSF TRIED TO POSITION TO FLOPPY BLOCK <1
6153	INTERNAL MICROPHOTOMETER SYSTEM ERROR. REPORT THIS.
6155	GAP NOT FOUND IN X RJPL. CHECK FOR DIRTY IBM TAPE CAPSTAN.
6160	ZERO CHANNELS TO PROCESS SPECIFIED IN X CORL
6162	ORDER OF POLYNOMIAL > 9 SPECIFIED IN X POLN
6165	CLOCK/CALENDAR SELECT CODE <0 SPECIFIED IN FTIME
6166	ORDER OF POLYNOMIAL > 10 SPECIFIED IN X POLY
6170	CLOCK/CALENDAR SELECT CODE >5 SPECIFIED IN FTIME
6174	INVALID AREA SCANNER ADDRESS IN X ASCR OR ASCW
6204	AIR MASS*256 (Z) IS < 0 OR > 4095 IN X TINC
6205	CHANNEL OFFSET (W) IS < 0 IN X SCRN
6206	FIELD IS < 0 OR > 4095 IN FZAP, FJMP, OR FJMS
6210	ANY OF THE ARGUMENTS IS < 0 OR > 4095 IN X SIZE
6216	BUFFER 0 NOT ALLOWED IN X SET
6222	PX, PY, PPL, OR LPS IS =0 IN X SIZE
6231	# OF BITS TO SHIFT > 11 IN X COMB
6234	X BIN RAN OUT OF UNBINNED INPUT DATA
6244	A ZERO BIN SIZE IN BIN
6250	PX*PPL*MU+X0 IS > 1024 IN X SIZE
6251	X SCRN RAN OUT OF UNSCRUNCHED DATA
6255	STARTING WORD(S) <0 OR >4095 IN X PAK OR X UNPK
6270	STARTING WORD (W) < 0 IN X RJPL
6273	NO WRITE RING IN IBM TAPE WHEN USING X WRQ
6274	NEGATIVE BINSIZE FOUND BY X SCRN
6301	BAD PIXEL SIZE IN X PIXL OR X SIZE
6302	BAD COMBINATION OF PL, PS, X0 IN X PIXL
6303	BAD COMBINATION OF LS, PS, Y0 IN X PIXL
6311	X SET NOT CALLED BEFORE X CALB
6313	VERIFY FAILED IN FZAP
6317	INVALID MAILBOX MEMORY LOCATION SPECIFIED
6331	INVALID FLOATING POINT BUFFER LOCATION
6332	SEMICOLON MISSING IN X PUTV(S,B,V). SHOULD BE X PUTV(S,B;V)
6333	# OF WORDS MODULO(# OF CHANNELS TO COMBINE) NOT 0 IN X COMB
6335	INVALID OCTAL NUMBER IN FZAP, FDEC, FJMP, FJMS, OR FMPX
6340	READ WILL SPILL OFF END OF MEMORY IN X RJPL
6343	INVALID STARTING WORD (WA) SPECIFIED IN X ASTV
6344	SCREEN FULL IN X IPHO OR X PIXL NOT CALLED FIRST

6362 # OF WORDS TO READ (NW) <= 0 IN X RJPL
6363 ATTEMPT TO TAKE MICROPHOTOMETER DATA DIRECTLY TO DECTAPE
6364 INVALID X COORDINATE IN X TVJY
6365 INVALID Y COORDINATE IN X TVJY
6366 INVALID MULTIPLIER FACTOR (MU) IN X TVJY
6373 PY*LPS*MU+YO IS > 1024 IN X SIZE
6412 DECTAPE I/O PENDING OR NOT ACKNOWLEDGED IN X RJPL
6422 SCREEN FULL IN X PICT OR X SIZE NOT CALLED FIRST
6434 INVALID STARTING BLOCK (SB) IN X RBLK OR WBLK
6442 PATTERN BUFFER LENGTH (NSB) TOO BIG FOR PIXEL SIZE IN X PICT
6445 ILLEGAL UNIT NUMBER SPECIFIED IN X RBLK OR WBLK
6462 INVALID NUMBER OF BLOCKS (NB) IN X RBLK OR WBLK
6472 DECIMAL NUMBER < 0 OR > 4095 IN FOCT
6475 COMBINATION OF W AND NB SPILLS ACROSS FIELD IN X RBLK OR WBLK
6520 X BIN OR INT CALLED WITHOUT CALLING X TABE FIRST
6521 X HDTS OVERLAY REUSED, OR QUICK IBM OVERLAY NOT LOADED
6537 EXTERNAL GATE IN ONE STATE > 55 SECONDS IN X ASCT
6542 INTERNAL MICROPHOTOMETER SYSTEM ERROR
6545 INVALID STARTING ROW (SR) IN X ASTV OR TVOF
6546 INVALID NUMBER OF ROWS (NR) IN X ASTV OR TVOF
6547 INVALID NUMBER OF COLUMNS (NC) IN X ASTV OR TVOF
6550 INVALID MULTIPLIER FACTOR IN X ASTV OR TVOF
6556 # OF BITS TO SHIFT >= IN MAGNITUDE TO 24 IN X SHFT
6571 FLOPPY I/O ERROR WHILE TAKING MICROPHOTOMETER DATA
6601 AREA SCANNER MODE SET INCORRECTLY IN X ASCT
6603 STARTING DATA WORD (WA) < 0 IN X HIST
6604 DWELL TIME < 0 OR > 4095 IN X DTIM
6607 DEADTIME FACTOR (D) IS < 0 OR > 4095 IN X DTIM
6615 BUFFER 0 USED IN X SRCH
6626 ILLEGAL MODE PARAMETER IN X ASCT
6627 LOW OR HIGH BOUNDARY (LB OR HB) IS < 0 OR > 4095 IN X HIST
6644 HIGH BOUNDARY (HB) < LOW BOUNDARY (LB) IN X HIST
6661 ODD NUMBER OF WORDS PER ROW (NC) IN X TVOF. MUST BE EVEN
6665 NUMBER OF DATA WORDS (NW) < 0 IN X HIST
6670 MICROPHOTOMETER DATA RATE EXCEEDS FLOPPY DATA RATE
6703 PEAK WITH A WIDTH > 4095 CHANNELS FOUND BY FSEEK
6706 SPILLED BACKWARDS OFF END OF MEMORY IN FLIP. REPORT THIS.
6711 BIT NUMBER OF LENGTH <0 OR >24 IN X BIT
6713 INVALID UNIT IN X RDF OR WRF
6714 END OF BUFFER SPACE IN X SRCH
6715 X INT RAN OUT OF UNINTERPOLATED INPUT DATA
6725 UNIT SPECIFIED IN X RDF OR WRF NOT POSITIONED WITH X POSF
6735 # OF BLOCKS <0 OR >31 SPECIFIED IN X RDF OR WRF
6742 FSEEK RAN OUT OF OUTPUT BUFFER SPACE
6757 X RDF OR WRF WITH 'NO WAIT' OPTION FOUND FLOPPY BUSY

DECTAPE ERROR MESSAGE DECODING

MESSAGE	MEANING
TAPE?U---60--	MARK TRACK ERROR
TAPE?U---50--	END OF TAPE ERROR
TAPE?U---44--	SELECT ERROR
TAPE?U---42--	PARITY ERROR
TAPE?U---41--	TIMING ERROR

NOTES:

1. "U" INDICATES THE UNIT ON WHICH THE ERROR OCCURRED; DECTAPE UNIT 8 IS INDICATED AS UNIT 0. "-" INDICATES ANY DIGIT BETWEEN 0 AND 7; THESE DIGITS ARE NOT SIGNIFICANT IN DECODING THE MESSAGE.

2. SELECT ERRORS ARE THE MOST COMMON AND INDICATE A USER ERROR. SELECT ERRORS ARE CAUSED IF THE DESIRED UNIT IS NOT SELECTED, OR IF ONE TRIES TO WRITE TO A UNIT WHICH IS SWITCHED TO READ ONLY OR WRITE LOCK.

3. AN END OF TAPE ERROR IS ALSO PROBABLY A USER ERROR; IT MEANS YOU TRIED TO ACCESS INFORMATION OFF THE END OF THE TAPE.

4. TIMING ERRORS INDICATE A HARDWARE PROBLEM AND SHOULD BE REPORTED IMMEDIATELY.

5. MARK TRACK AND PARITY ERRORS MAY INDICATE A MARGINAL DISK; IF THEY OCCUR REPEATEDLY, THE DISK OR TAPE IS PROBABLY GOING BAD.

*

MULTIPLEXER ERROR MESSAGE DECODING

MESSAGE	MEANING
MUXffff2650	PRIVATE MULTIPLEXER UNIT NOT RESPONDING
MUXffff2657	PRIVATE MULTIPLEXER CHECK BITS TEST FAILED
MUXffff2664	PRIVATE MULTIPLEXER FUNCTION ECHO FAILED
MUXffff2665	PUBLIC MULTIPLEXER UNIT NOT RESPONDING
MUXffff2674	PUBLIC MULTIPLEXER CHECK BITS TEST FAILED
MUXffff2701	PUBLIC MULTIPLEXER FUNCTION ECHO FAILED
MUXffff5305	PUBLIC MULTIPLEXER BUSY BUS BUSY > 95 MILLISEC.
MUXffff5322	SAME AS MUXffff5305

NOTES:

1. "ffff" IS THE OCTAL REPRESENTATION OF THE FUNCTION WORD BEING TRANSMITTED WHEN THE ERROR OCCURRED. THIS FUNCTION WORD INDICATES WHICH MULTIPLEXER UNIT WAS BEING ADDRESSED, AND WHAT OPERATION WAS BEING PERFORMED. TO DECODE THE FUNCTION WORD, SEE PAGE 4 OF LICK OBSERVATORY TECHNICAL REPORT #16, "SERIAL DATA MULTIPLEXER - EL-331", BY T.P. RICKETTS AND L.B. ROBINSON, FEBRUARY 1976.

2. NOTE THAT MULTIPLEXER UNIT 3 CAN ONLY BE ADDRESSED BY THE READ OUT ROOM COMPUTER, AND CANNOT BE ADDRESSED BY THE ECHELLE COMPUTER. SIMILARLY, MULTIPLEXER UNIT 4 CAN ONLY BE ADDRESSED BY THE ECHELLE COMPUTER AND CANNOT BE ADDRESSED BY THE READ OUT ROOM COMPUTER. MULTIPLEXER UNITS 1 AND 2 CAN BE ADDRESSED BY EITHER COMPUTER.

3. TURNING EITHER COMPUTER ON OR OFF CAN CAUSE THE OTHER COMPUTER TO GET A FEW MULTIPLEXER ERRORS. WHEN EITHER COMPUTER IS BEING USED FOR OBSERVING, ANY UNNECESSARY POWERING UP OR DOWN OF EITHER COMPUTER SHOULD BE AVOIDED.

4. PLEASE REPORT ANY OF THE ABOVE LISTED MULTIPLEXER ERRORS TO BOB KIBRICK, TERRY RICKETTS, OR LLOYD ROBINSON. PLEASE RECORD WHICH COMPUTER OR COMPUTERS WERE IN OPERATION AT THE TIME THE ERROR OCCURRED. ALSO RECORD WHICH DATA TAKING SYSTEM OR SYSTEMS WERE IN USE.