Disclaimer

This note has not been internally reviewed by the DØ Collaboration. Results or plots contained in this note were only intended for internal documentation by the authors of the note and they are not approved as scientific results by either the authors or the DØ Collaboration. All approved scientific results of the DØ Collaboration have been published as internally reviewed Conference Notes or in peer reviewed journals.

DØ Note 1259

D0 Clock Control and Test Program V3.02
4/30/91
R. Angstadt
(Original Version 1.03 by Bill Graves 1/31/90)

Please refer to the file "CLOCK.VER" for details of version differences, enhancements, and bug fixes.

Background

This is an upgrade built on the previous version of the clock software, "CT", originally written by Bill Graves. The re-write was driven by the need to address more Selector Fanout Modules than the 25 the original program allowed. Also the original version was restricted to only one VME crate by the Bit3 Interface. The new version features a "device independent interface" programming style that allows essentially the same program to be run either over the Token Ring Network to communicate with any VME node on the Network or to be run over the Bit 3 in the lab for testing. See "IMPLEMENTATIONS" below.

Main Enhancements

Most of the original notes still apply to the actual routines being called by the sub-menu picks but some of the sub-menus have had considerable re-working or are all new. The user interface has been moved more in the direction of "PCPLOT" wherever it was quick and easy to do so for quicker response to the user and for maintainability by me. In places where the former interface was clearer and the code was more difficult to understand the former interface was left intact. This also helps provide continuity.

Generally all of the old *.txt file formats of this version of "CT" are not the same the old version and are incompatible with it. An exception is the Static and Dynamic data files used to download (or upload) from the Sequencer which have not been changed.

The old *.txt file formats were changed because they did not have enough information in it's format, i.e., no VME Node numbers. Also this versions file format is much more readable with a comment string following the value so that file can be easily edited by any standard program text editor. Another reason no backward compatibility was maintained is because almost all of the file handling software of the original version has been replaced by more robust routines so that most run time errors that used to occur when accessing non-existent files should be eliminated. Also the addresses are now 32 bits and are written by the program as hexadecimal values as opposed to the high byte of a 24 bit address (in decimal). The user may still input decimal addresses to the program by not

adding a "$" in front of the value.

A batch file "clock" has been created and put in the path so that command will set the current default to the correct sub-directory and start the "CT" program. Please see the Versions section for more detail.

Another significant change is in the design philosophy of this program. In general, this version has been changed so that the menu picks do not read addresses from a default file and then blast out to the Clock hardware the values in the file. What happens is that the program will still try to read the default files and get addresses, but then it will read and display the values currently in the hardware. Additional Menu Options have been provided for manually saving and restoring old values from files. This should allow this program to somewhat better co-exist with other processes running on other computers if desired.

CT has been compiled using Turbo Pascal Version 6.0. The Bit 3 requires a 16 bit AT slot although IBM's Token Ring board only requires an eight bit slot. (The Token Ring Version Probably could be made to work on an XT if one really had too.)

The Bit 3 is a commercially made interface board set by (Bit3 Computer Corp.) that serves as a bridge between the AT bus and the VME bus in which the FADC boards reside. The present set-up of the Bit3 model #403 board set is that it provides a means of mapping a 64Kb AT address software controlled movable window of address space into the 0-16Mb (24bit) address space of the VME based FADC crate. (The newer model #406 adapter allows 32 bit addressing for $1,365 for a new VME card.) This program will operate with either model as long as the program is set to match the hardware with F7. The PC side of both models is "the same" only the VME board is different. (The #406 has LED's on it and the #403 does not.) Remember to use General Address Modifiers to get over the Vertical Interconnect when using 32 bit addressing. (General Address Modifiers have a 0 in the upper nibble: e.g., $39 changes to $09.)

IMPLEMENTATIONS

CT comes in three flavors depending on how the two compile switches are thrown in "IO.INC". If the compiler switch "IO_VME" is true, "DEFINED", then code that uses the Bit3 board will be compiled. If "IO_VME" in set false, "UNDEF", then the Token Ring code will be used. Another compiler switch, "BIT3POLL", controls how the Bit3 board will be utilized when "IO_VME is set true. If "BIT3POLL" is "DEFINED"=true then the Bit3 polling version will be used.

For now I am adhering to the following naming conventions:

non-polling Bit3 version "CTNP.EXE" and "CTNP.OVR"

polling Bit3 version "CTP .EXE" and "CTP .OVR

(From Version 3.01 and above the polling Version is not supported.
Odd byte transfers as bytes do not work.)

Token Ring version "CTT .EXE" and
"CTT .OVR"

Copy whichever version you wish to run to "CT.EXE" and "CT.OVR"
and the "clock" batch file will then run the one you want
automatically. It is now necessary to copy the corresponding
overlay file as well.
To use the Bit3 non-polling version at the DOS prompt type:
"copy ctnp.exe ct.exe" (return) and then type
"copy ctnp.ovr ct.ovr" (return)
Then type "ct" and the Bit3 non-polling version will load.
To switch to the Token ring version at the DOS prompt type:
"copy ctt.exe ct.exe" (return) and then type
"copy ctt.ovr ct.ovr" (return)
Then when "ct" is typed the Token ring version will load.
In the lab on the fourteenth floor they would probably want
to run the non-polling Bit3 version for higher bandwidth in
testing. To actually run the Detector one will probably want
to run the Token Ring Version from the Control Room although
they can run either version from a P.C. on the 2nd floor
counting house if desired. Further details follow.

Bit3 Non-Polling Implementation

Be sure to set the program to use either 32 bit or
24 bit addressing with "F7" depending on whether you
are using the model #403 (24bit) or model #406 (32bit) Bit3
adapter. The only difference is the VME card: the model
#406 has three LED's on the front panel and the #403 has none.
Remember to use General Address Modifiers to get over the
Vertical Interconnect when using 32 bit addressing. (General
Address Modifiers have a 0 in the upper nibble: e.g.,
$39 changes to $09.)

The first Non-polling version is most similar to the previous
CT version. It uses the Bit3 board set to address the
D0 Clock in VME address space. At run-time this version may
be distinguished by the phrase "non-polling via Bit3" on the
second line of the main menu.

It has a limitation in that the VME bus signal, "DTACK",
must be returned in ~15 micro-seconds or an error, "*VME
TIMEOUT*" will occur. This is a limitation of the Bit3
board set in conjunction with the PC bus. This error is misleading
since VME bus arbitration can take as long as ~120 micro-seconds.
(Most Motorola processors that serve as the bus arbiters
used here time out in the ~50-80 micro-second range depending
on which processor is being used.) Although the clock hardware
is much faster than this if the Control Crate is very busy then
this situation could still occur in theory.

Both Bit3 versions detect VME bus errors by specifically
checking Bit3 status registers after every read or write via

the Bit3. (They do not use interrupts for error handling.) A small delay (this delay may have to be increased for machines faster than a 20Mhz AST Premium/386C) has been added to all writes on this version so that write errors may be properly reported on the non-polling version when the Bit3 VME board jumpers are not configured as the VME Bus Arbitrator. This delay is to overcome pipelined writes and prevent only reporting errors on every other write.

Be sure to set make

Bit3 Polling Implementation

(This flavor is no longer supported for Versions greater than or equal to 3.01 due to the inability to accessing odd byte addresses in this mode.)

Be sure to set the program to use either 32 bit or 24 bit addressing with "F7" depending on whether you are using the model #403 (24bit) or model #406 (32bit) Bit3 adapter. The only difference is the VME card: the model #406 has three LED's on the front panel and the #403 has none. Remember to use General Address Modifiers to get over the Vertical Interconnect when using 32 bit addressing. (General Address Modifiers have a 0 in the upper nibble: e.g., $39 changes to $09.)

The second Polling version uses the Bit3 board set in a "polling" mode to address the FADC cards in VME address space. At run-time this version may be distinguished by the phrase "this version for old and new cards via (polling) Bit3" on the second line of the main menu.

To overcome the first version's premature timeout problems, a second, Polling version was developed. In this version a VME "*BUS ERROR*" will occur when DTACK is NOT returned from a VME bus I/O operation as determined by the VME bus arbitrator. For a MV133 card this is in the ~55-80 micro-second range. This version operates the Bit3 board set in a mode where the PC polls the Bit3 until VME DTACK is returned or has timed out. It then checks the other Bit3 status bits and reports any other errors before going on in the program. This method will allow the Bit3 board to be placed in the D0 control's crate and access different Selector Fanout crates over different Vertical Interconnects without physically moving the VME Bit3 board. If the VME Bus arbitrator hangs and does not return a "*BUS ERROR*" signal the program will give up polling the Bit3 and generate "*POLL TIMEOUT*" The operator may set this program polling timeout as they desire on F6.

Token Ring Implementation

This flavor is always using 32 bit addressing and the address modifiers displayed on the menus are ignored. Address modifiers are taken care of automatically by the Remote 68K processing the message.

The Token Ring network for all I/O to the D0 Clock. The Bit3 board set is not used. It sends and receives D0 control

messages (Bob Goodwin's S1 messages) over the Token Ring. At run-time this version may be distinguished by the phrase "this version for old and new cards via Token Ring" on the second line of the main menu.

The advantage of this version is that it may talk to any VME (clock or not) crate at FERMILAB that is connected to the Token Ring via a D0 control crate and Vertical Interconnect.

If the program reads a non-existent location via the Token Ring "0" will be returned and if error checking is turned on "*NT RD ERR*" will be displayed or something similar. Downloads are automatically and/or manually verified by reading back what was written. (The Bit3 version will typically return $FF18 in an AST premium running at 10Mhz from a location where no hardware is responding. IBM's specification for the AT bus is that the AT should return $FFFF.)

When talking to hardware you know exists, if a screen full of "*NET ERROR*" occurs, this is an indication that the node you are talking to is not responding. Give it a minute to re-boot itself and attempt to come back on the air. In some case a manual (done locally) re-boot may be necessary. If there is a bridge between your location and the node that could be a problem as well. If the bridge is down it may have to be powered down locally and re-started.

A "*NET ERROR*" may also occur if the message was somehow garbled while traversing the Token Ring. This may happen when a Token Ring node is taken off or inserted on to the Ring. It means that the message protocol was violated in some manner and could not be correctly decoded. It could have been a checksum error or an invalid message type.

ADDRESSING

This latest release supports both 32 bit and 24 bit addressing using the Bit3 #406 adapter (32 bit addressing) or the Bit3 #403 adapter. The previous "CT" program only supported 24 bit addressing and could not easily be used over the Vertical Interconnect via a Bit3. This version can but be sure to set the software switch on F7 accordingly and to use the correct address modifier as discussed in the "INITIALIZATION" section.

All versions now display the full 32 bit address wherever a base address is called for. In most cases the old version of "CT" only called for the high byte of a 24 bit address. This byte was then internally shifted left 16 bits. This is no longer the case. When converting the old files to the new files format keep this in mind.

PROGRAM INITIALIZATION

The program first looks for the file "Defaults.txt" which contains defaults used for controlling the Sequencer module on F1. If it does not find this file it will let you know that it failed to find the file in the current sub-directory and then say that it is using some internal compiled defaults.

If you are running the Token Ring version of CT then the next file it looks for is "CT.INI". This is a human readable ASCII file that contains the last word of the address the PC will use as it's Local Area Network, LAN, address. It must be unique for each PC or the program will not start. This mechanism is used as a simple lockout mechanism to prevent two users in different locations from inadvertently trying to access the same hardware. If "CT.INI" is not found the user will be prompted for a new filespec. The user may input a new filespec or hit "ESC" to get out of this loop and exit CT.

If either of the Bit3 versions is being run the program will initialize the Bit3 board. The program always sets itself to use 24 bit addressing mode with the Bit3 model #403 VME adaptor. If you want to run over the Vertical Interconnect using the 32 bit addressing Bit3 model #406 you must set the program to that mode with F7 first. This software switch must agree with the hardware you are using or the program will not work as intended.

Also remember when using the 32 bit #406 over the Vertical Interconnect that the high nibble of the address modifier generally gets set to 0. For example if you would normally use $39 for the 24 bit non-Vertical Interconnect configuration then one would use $09 if going over the Vertical Interconnect. When using the Token Ring Version the Address Modifier is ignored. Likewise if using the Bit3 version then the VME node is ignored. Both are required in the data base so that only one set of data files are required.

At this time the main menu is brought up and the user is now in control. Be sure to manually set F7 to the correct addressing mode. To this point nothing has been written to VME hardware.

General Notes:

After CT has initialized itself and the main menu appears the user may activate any of the sub-functions or sub-menus

CT then displays the Main Menu and awaits the next request. If "Esc" is pressed while in any of the sub-menus the program redisplays the Main Menu. The function keys and the "Esc" are "hot", i.e., they take (relatively) immediate action when a key is pressed without necessitating going back up to the main menu. If running CT on a color monitor the fact that a menu pick is "hot" is signified with the Capital key represented in yellow.

Interactive help is available by pressing the "Alt" key and then the appropriate function key. Additionally, on most sub-menus the "H" key provides help specific to that sub-page. On the VME command line interpreter available on F1,F2, and F3 typing "Help" followed by the "Enter" key will provide help for valid commands. Most of the sub-menu help files may be customized by the user as desired by editing the appropriate file with any ASCII text editor. The name of the file assigned to a "Alt-Fn" sequence is internal to the program but is displayed at the end. Each display is 24 lines long.

On several sub-menu pages where files are used, there is an option to do a directory of files without exiting CT. At program initialization the masks are set to what one is probably interested in but the opportunity exits to change the mask when it is displayed. If a function key or "Esc" is pressed then CT will do whatever was pressed without changing the mask. If the "Enter" key is hit after the mask is displayed then the file directory using that mask will be used. If the "Backspace" key is hit then the last character will be deleted; then if another character key is pressed it will be appended to the remaining mask. The first time the mask is displayed if any key other than the "Backspace" or "Enter" is hit then the whole string will be deleted and whatever key was first hit will be the first letter of the new mask. The arrow keys and other IBM editing keys are not implemented as editing keys, they delete the old mask and display whichever key was pressed. See the DOS manual "dir" command for further valid filespec information.

The user is free to use their own file names and extensions. The ones used internally by the program are suggestions. In general the user is not restricted by the program to use a specific filename or extension except at program load time. Most file types are determined by a string in the first line of the file and checked internally by the program to avoid run-time errors while reading the file. (For example a sequencer, "*.seq", save/restore file has different format than a selector fanout, "*.sf", save/restore file.

MAIN MENU

"F1" Sub-Menu: "sequencer"

This is mostly the original CT main menu pick 1. It provides options for controlling, saving, restoring, downloading and plotting the Sequencer Clock module memory or files.

"F2" Sub-Menu: "selector"

This is mostly the original CT main menu pick 2. It allows reading and setting of the Selector Fanout modules. This menu provides a method of reading the Selector Fanout modules, adding, deleting, and modifying their addresses and data, saving and restoring modules individually. A sub-menu "Global save/restore" provides a means of saving and restoring user created arbitrary lists of Selector Fanout modules (for the whole detector or any part of it)! The only limits on the current scheme is how much disk space is available to DOS (as opposed to the old scheme of 25 Selector Fanout Modules.)

"F3" Sub-menu: "pcc"

This is an expanded and enhanced version of the original CT main menu pick 3. It allows reading, setting, saving and restoring of the PCC (Phase Coherent Clock) module.

"F4" "auto save on exit from above three"

Allows the user to turn on or off (manually) whether they

wish CT to always save their last settings to files on the disk when they exit sub-menus. (If one is browsing one may not want to be constantly over-write (good) data?) "SF_DEF.TXT" is always automatically updated and (only) contains another device (file) name which was the last Selector Fanout the user was reading.

"F6" { polling Bit3 version only }
"software limit on how many times to poll for vme dtack?"
Determines the upper limit on how many times the program will check to see if VME "DTACK" has been returned before it will give up. (In some places where error reporting is turned on to the screen then "*POLL TIMEOUT*" should appear. A good value for this for now seems to be about 100 tries. If set to 1 then it will generally almost always time out before the hardware responds.

"F7" { all Bit3 versions}
"use 32 bit mode?"
Set this software switch to agree with which Bit3 VME version is hooked to your P.C. The 32 bit #406 is easy to tell from the 24 bit addressing #403 because the #406 has three LEDs on the front of it.

or

"F7" { Token Ring Version }
"number of times to retry token ring: 'n'"
If this is set > 1 and a network error is detected then the error will not be reported to the user until after the program has retried it the number 'n' is set too. If success is achieved before 'n' is exceeded then no network error will be reported to the user. This should not be set excessively high or the program will appear to "hang" while it tries to read some (sequence of) non-existent addresse(s). There is a software clamp on this to not let the user go < 1 or > 100. This is a useful mechanism of overcoming a "noisy" network while downloading/uploading memory in a straightforward manner. (A noisy network may be caused by a lot of nodes physically coming on and off the network or a node with a loose cable.) However setting this > 1 can result in non-deterministic results when trying to access custom deterministic hardware that requires specific sequences of known quantities of reads and writes in a particular order. Unless you know the hardware intimately, 1 is the recommended setting.
It has no effect on the Block Network Transfers used when accessing Sequencer memory during plotting, saving and restoring.

"F8" "peek/poke vme."
This is new to the CT program. It has been brought in almost unchanged for little effort from "PCPLOT". It has been very useful in working with FADC's and VBD's. Besides being useful to myself it may be useful to

others in working with the Clock hardware.
NOTE:
About data value byte order. When addressing clock hardware words from this page the bytes will be swapped. This is because of the clock hardware and not this software.
For F8 only:
when reading standard VME memory 16 bit words that represent 16 bit quantities as words, the bytes will be correctly displayed by this page else if the data at that address was byte data then the bytes will be swapped.
For the rest of "CT" (exclusive of F8):
(The rest of Version 2.0 of "ct" program is always swapping bytes to the hardware but displaying them normally to the user. Most save/restore files are this way except for the sequencer download files.)

"F10" Immediately exits the program.
The Token Ring version the PC removes itself from the Network. The polling version does nothing special at program termination. On a normal exit all versions restore the old CRT text image.
Abnormal exits includes ^C exits and infamous 201 errors. (201 errors are unexpected "out of range errors" that the program has failed to trap internally. They may be due to unexpectedly large values input by the user or generated by an unexpected value read from hardware.) Recording and reporting the run time error and the address following it will help in making the program more robust than it is.

Welcome to D0 Clock Test V3.02
token ring version

Help (in general).
F1 sequencer (time line generator).
F2 selector fanout.
F3 pcc (oscillator) control .
F4 auto save on exit from above three (toggle) current: .. OFF

F7 number of times to retry token ring: 1

F8 peek/poke vme.

F10 eXit program.

Welcome to D0 Clock Test V3.02
(non-polling) bit3 version

Help (in general).
F1 sequencer (time line generator).
F2 selector fanout.
F3 pcc (oscillator) control .
F4 auto save on exit from above three (toggle) current: .. OFF

F7 use 32 bit mode? on=32bit=#406 off=24bit=#403. current: .. OFF

F8 peek/poke vme.

F10 eXit program.

token ring. d0 clock sequencer test/control or eXit, Quit or ESC

1 save/restore file (device): MAIN.SEQ

CSR0 = $0523

3 vme node: $073A CSR1 = $0004

4 module address: $10600000

5 error file: SCREEN 7 trigger bucket: 0

6 error mask: $01FF 8 address modifier: $09

status: BC Phase

---

main menu

toggle	simple options	sub-menu options
Clock is on/off	I clear errors	Download & file plot mer
A group is on/off	J set errors	Plot timeline memory mer
B group is on/off	K reset clock	Test clock menu
Group select A/B	Save configuration	N step through menu
Free run/sync	Restore configuration	Vme diagnostics
Enable outputs on/off	List of files	
Mode (step) is on/off	Open dos door	

F1 Sub-Menu: D0 Sequencer Help Notes
12/16/90

This page is similar in function as the original Version 1.03 main menu pick one except that the user interface has been slightly changed and that some menu picks have different letters and have been re-located.

This page no longer reads the last selected address and Downloads the Control Status Register automatically although the user is allowed to do this by pressing the "R" key for "Restore". The page reads "MAIN.SEQ" to get the address and then reads the hardware and displays it's current settings. Nor are the last downloaded settings automatically saved to the file unless "auto save on exit is enabled" on the main menu and the user type 'y' or 'Y' to the question asking if he wants to over-write the existing file. Also the user may at any time manually over-write the existing file with the "Save" option below.

"1 save/restore file"
Allows the user to select which file to use. If the file is found then the program reads in the file's addresses and filenames to read the Sequencer without changing the values in the Sequencer control registers from the file as does "Restore" in the menu pick below. (One can go to DOS and type out the file to view it's format.) Also it may be edited by hand. Note that this format is not the same as for Version 1.03. Using this information the program then continuously does byte reads of the (byte) registers and display the results until a key is pressed. This button calls the same routine that is called at program load time. If the file is not found a warning message is displayed on the screen.

When typing in the filename the following rules apply. If the "Backspace" key is hit first then the string may be deleted and changed one character at a time; else if a normal ASCII character is struck the whole string will be deleted and the user may input a new string. The arrow keys are not implemented in editing this field. If the "ESC" key is pressed the value will not be changed. Also if the "Enter" key is pressed with no previous input the old value will remain.

"3 vme node:"
This is a hex (byte) address of which D0 Control Crate the Sequencer is tied to through the Vertical Interconnect (VI).

"4 module address"
This is the full 32 bit VME base address the Sequencer card is set to. For both Bit3 versions note to check that if one is using an address that is greater than 24 bits one must be using a Bit3 406 adapter and the main menu pick "F7 use thirty-two bit mode" must be set to "ON". When using the 24

bit addressing #403 Bit3 adapter then this must be set to "OFF". If the #403 is being used the high byte of the 32 bit address will be ignored but the program must still be set to the correct state.

"5 error file:"
Controls where some output for some of the tests will be sent. This may be any legal DOS disk and filespec (up to 79 character). If this is set to "SCREEN" then output will come to the screen. Some output will always come to the screen as well as the file.

"6 error mask"
Allows the user to ignore errors coming back from the Sequencer error register. Entering 0 will ignore all bits and there will be no errors. $01FF is a good value to display the normal bits you are interested in. You can set this in hex by preceding your input with a "$" sign or else decimal is assumed. If the "ESC" key is pressed the value will not be changed. Also if the "Enter" key is pressed with no previous input the old value will remain.

"7 trigger bucket:"
User input is clamped to the valid range of 0..1113. Input is in decimal or use a "$" sign for hex. If the "ESC" key is pressed the value will not be changed. Also if the "Enter" key is pressed with no previous input the old value will remain.

"8 address modifier"
This is the address modifier that the Bit3 will use to talk to the VME crate. It is always in the file of data but is not used if this program is doing I/O over the Token Ring. That way only one set of Device files need be maintained no matter which interface is being used.

A word about address modifiers is in order here. This version has been modified so that it is always doing BYTE reads of the SF byte registers so one needs to put in a valid BYTE VME address modifier. Further if the program is being used over a Vertical Interconnect then one needs to change the high nibble to $0. For example if the hardware normally responds to $3D then when using the VI one would use $0D.

"Clock is on/off"
Sets or clears bit 1 in CSR0 which is the Clock_on bit.

"A group is on/off"
Sets or clears bit 8 in CSR0 which is the group_A_on bit.

"B group is on/off"
Sets or clears bit 9 in CSR0 which is the group_B_on bit.

"Group select A/B"
Sets or clears bit 10 in CSR0 which is the Sel_gr_A/B bit.

"Free run/sync"
Sets or clears bit 4 in CSR0 which is the Free_run/Sync bit.

"Enable outputs on/off"
Sets or clears bit 2 in CSR0 which is the Output/Enable bit.

"Mode (step) is on/off"
Sets or clears bit 3 in CSR0 which is the Step_mode bit.

"I clear errors"
Sets bit 6 in CSR0, the Clear_errors bit, then clears the same bit. The message "Errors Cleared." is then printed below option "Restore configuration" on this menu.

"J set errors"
Sets bit 0 in CSR0, the set_errors bit, then clears the same bit. Before clearing the bit, CSR0 is read, and if bit 0 is set, the message "Errors Set." is then printed below option "Restore configuration" on this menu. If bit 0 is not set then the message "Could not set bit 0 in csr0" is printed instead of "Errors Set."

"K reset clock"
Sets bit 7 in CSR0, the Reset bit, then clears the same bit. The message "Clock reset." is then printed below "Restore configuration" on this menu.

"Save"
Reads the current values and writes them to the "1 save/restore file".

"Restore"
Reads the current "1 save/restore file" and writes the values to hardware.

"List of files"
Allows the user to do a directory of files on the disk. The current mask is displayed which the user may change by typing in a new one: if the "Backspace" key is hit first then the string may be deleted and changed one character at a time; else if a normal ASCII character is struck the whole string will be deleted and the user may input a new string. The arrow keys are not implemented in editing this field. When the user hits "Enter" then the program will shell out to DOS and do a directory if there is memory available in the PC.

"Open a door to dos"
Allows the user to shell out to DOS and do what ever they can with the remaining memory available including editing any files. To return to the program hit "Enter" with out typing in a string.

"Download & file plot menu"
Allows the static and dynamic Sequencer timeline RAM to

be either saved to a file or restored from a file.
Also new code has been written that allow the files to be plotted in graphic or text modes. Some of the plotting control options are set by the next menu pick "Plot timeline memory menu" below. This sub-menu has its own help notes under "H".

"Plot timeline memory menu"
Allows plotting the timelines currently in Sequencer RAM. Note that in order to read Sequencer RAM the clock must be off for the Static RAM and that the Dynamic RAM can only be read from the inactive bank. This page makes checks and asks questions similar to the "Download..." menu-pick above. It will not automatically change the state of the clock. It has no picks for changing the state of the clocks as does the "Download..." menu above so one should put the clock in the correct state before plotting. This sub-menu has it's own help notes under "H".

"Test clock menu"
Is essentially the same sub-menu as it was previously under "M: Clock Tests". Some bugs have been fixed. Specifically it should say "test failed" when they fail now. Not all sections are applicable in all versions: i.e., "A" "random addressing test" is meant to be run with the Sequencer card in an otherwise empty crate; "D" "address modifier test" is meaningless over the Token Ring and it also will not give the same results when working over the Vertical Interconnect. The "Test clock menu" has it's own help notes.

"N step through"
From section 4.4.3:
"Upon selecting option N, the users enters the step through menu. The current state of the Sequencer (CSR0) is saved and the Sequencer is: put in step mode, turned on, outputs are enabled, and put in free run mode. When one exits Step Through, the previous state of the Sequencer is restored. Step Through is a way of verifying the contents of Sequencer RAM. It allows one to examine the potential state of the timing lines at any desired bucket. Step through will only work if a file has been loaded into the Sequencer. The Line Status column of the Step Through menu shows the RAM contents of the current bucket..." This sub-menu has it's own detailed help notes.

"VME diagnostics"
Is also the same as it was before under 4.4.4 "O: VME Diagnostics".
Upon selecting this option, the user enters the VME Diagnostics. VME Diagnostics is a simple BASIC - like language intended for the debugging of VME modules. VME Diagnostics has no immediate mode: program lines are preceded by a line number, and programs are executed when a run command is entered. The VME Diagnostics prompt is >. At the prompt, a user may type in a valid command, or a line number followed by a valid VME Diagnostics instruction. If one responds to the > prompt with and "h", the help screen for VME Diagnostics is printed:

commands:
ERASE filename: erases file <filename>
NEW: clears program currently in memory
TYPE filename: types file <filename>
LIST: types program currently in memory
LOAD filename: loads file <filename> into memory
DIR directory: gives directory of <directory>
SD: shows the current path
CD directory: changes current directory to <directory>
HELP
SAVE: saves the program currently in memory to disk
RUN: runs the program currently in memory
I: initializes bit3 outside of help. in help shows more instructions.
X: exit bit3 basic

If you want to see the set of instructions, type i.
If you want to return to the VME test cursor, type r.

If one responds with i, one sees the list of valid instructions:

instructions:
REM remark
PRINT var,"string"
GOTO line number
VRD (vme address, address modifier, variable to read data into)
  Vme Read, Double byte
VWD (vme address, address modifier, variable or constant to write)
  Vme Write, Double byte
VRB (vme address, address modifier, variable to read data into)
  Vme Read, Byte
VWB (vme address, address modifier, variable or constant to write)
  Vme Write, Byte
BERDIS disables printing of vme errors
IF var or const, = < >, variable or const THEN line number
 ELSE line number
variable name = variable name or expression + variable name or const
END halts program execution at this point

** hexadecimal numbers should be entered preceded by a '$' **
** do not do double byte operations (vwd, vrd) with odd addresses **

To execute a command you don't have to type in the whole command, only the bold letters. The following sections elaborate on the above help screens.

4.4.4.1. Syntax

The only parameters which the VME Diagnostics command set uses are <filename> and <path>. These parameters have exactly the same syntax in VME Diagnostics that they have in DOS.

The instruction set of VME Diagnostics takes as parameters variable names, constants, line numbers, and print strings.

A variable name consists of a letter A-Z followed by up to 13 letters or numbers in any order. A variable name may not be a

reserved word. Reserved words are:

4.4.4.2. Erase

Erase <filename> attempts to erase the file <filename> from the current path. If the file does not exist in the current path an error message is printed.

4.4.4.3. New

New clears the VME Diagnostics memory of any program. If a user has a 100 line VME Diagnostics program typed in, and then types new, the program will be gone forever unless it has been previously saved to disk.

4.4.4.4. Type

Type <filename> attempts to type the <filename> in the current path. If the file does not exist in the current path, an error message is printed.

4.4.4.5. List

List outputs the program which is currently residing in VME Diagnostics memory to the screen.

4.4.4.6. Load

Load <filename> loads the file <filename> from the current path into VME Diagnostics memory. If the file does not exist in a current path, an error message is printed.

4.4.4.7. Dir

Dir <path> outputs the directory of path to the screen. If <path> is a non-existent path, an error message is printed.

4.4.4.8. Sd

Sd outputs the current path to the screen.

4.4.4.9. Cd

Cd <path> changes the current directory to <path>. If <path> is and invalid path, an error message is printed.

4.4.4.10. Help

Outputs a help screen to the CRT.

4.4.4.11. Save

Save <filename> saves the program currently residing in VME Diagnostics memory to disk in the current path. If <filename> is in invalid filename, an error message will be printed.

4.4.4.12. Run

The Run command compiles and runs the program which is currently residing in VME Diagnostics memory. If there are syntax errors, and error message will be printed, and the program will not be executed.

4.4.4.13 I

I initializes the Bit 3 interface and returns a status message.

(On the Token Ring Version it does nothing.)

4.4.4.14 X

X exits VME Diagnostics and returns the user to the calling menu.

4.4.4.15. Rem

Rem is the VME Diagnostics remark delimiter. The line is ignored by the compiler.

4.4.4.16. Print

Print allows the user to print the values of variables and print strings to the screen. The syntax of PRINT is as follows: print may be followed by any combination of variables names and print strings separated by commas.

4.4.4.17. Goto

Goto <linenumber> transfers program control to the statement at line number. If the line number is invalid, a syntax error is generated.

4.4.4.18. Vrd

Vrd (address, address modifier, variable name) does a read on VME bus in A24:D16 mode of address <address>, with address modifier <address modifier>, and stores the data in <variable name>. Both address and address modifier can be constants - for example, the following programs perform the same function:

```
10 ADDR = $802000
20 AM = $3D
30 Vrd(ADDR,AM,RDATA)

10 Vrd($802000,$3D,RDATA)
```

If the VME cycle does not terminate normally (with DTACK), and the printing of bus errors is not disabled, one or more of the following messages will be printed when the read occurs.

For the bit 3 version:

```
'*BIT3 TIMEOUT*'
'*PARITY ERROR*'
'*VME BUS ERROR*'
'*POLL TIMEOUT*'
'* VME crate disconnected or not powered! *';
```

For the Token Ring version:

```
'*NET ERROR*'  { generic read or write net error }
'*NET RD ERR*' { is short for "network read error." }
'*NET WT ERR*' { is short for "network write error." }
```

In the case of the bus error message, no module responded with DTACK to the read cycle, so the system controller asserted the BERR line. In the case of the time out message, no module has returned DTACK, and the system controller hasn't asserted BERR with in the Bit3 timeout period of

~15 micro-seconds for the non-polling version.
(Note: for the polling version the user may set how many times the software will poll for DTACK with F6.
For the Token Ring Version if no DTACK occurs the error is not always returned to the caller. For downloading the "Verify" operation overcomes any limitation that might occur. If the user reads a location where there is nothing there to pull no DTACK, 0 is returned. So for the Network version right now 0 means 0 or no DTACK. Another possibility is that the D0 VME control node was not there and did not respond or that somehow the message was garbled and/or the message checksum did not agree. Garbled messages can occur when there is electrical noise on the network for some reason. This program has an automatic re-try feature that is user changeable at run time on F7. When this is set greater with n > 1 then no errors will be reported to the user unless it failed all n times. This automatic re-try feature is not presently supported on Sequencer block memory accesses.)

4.4.4.19. Vrb

(Note: "Vrb" accessing the byte in question by word read and the correct byte is returned. Word access Address Modifiers should be used.)

Vrb (address, address modifier, variable name) does a read on VME bus in A24:D8 mode of address <address>, with address modifier <address modifier>, and stores the data in <variable name>. Both address and address modifier can be constants - for example, the following programs perform the same function:

```
10 ADDR = $802000
20 AM = $3D
30 Vrb(ADDR,AM,RDATA)

10 Vrb($802000,$3D,RDATA)
```

Error messages are the same as "Vrd"

4.4.4.20. Vwd

Vwd (address, address modifier, variable name) does a write on VME bus in A24:D16 mode of address <address>, with address modifier <address modifier>, and stores the data in <variable name>. Both address and address modifier can be constants - for example, the following programs perform the same function:

```
10 ADDR = $802000
20 AM = $3D
30 WDAT = $1234
40 Vwd(ADDR,AM,WDAT)

10 Vwd($802000,$3D,$1234)
```

Error messages are the same as "Vrd"

4.4.4.21. Vwb

(Note: "Vwb" really accesses the VME by word. This means the Bit 3 flavors should use word Address Modifiers. On write operations the word is first read into P.C. memory, the byte is changed and then the whole word is written back to VME memory. This is not an indivisible read-modify-write VME cycle but completely separate VME bus operations. On the Network version there could be as much as 12 milli-seconds between the read and write cycles. Word access Address Modifiers should be used.)

Vwb (address, address modifier, variable name) does a write on VME bus in A24:D8 mode of address <address>, with address modifier <address modifier>, and stores the data in <variable name>. Both address and address modifier can be constants - for example, the following two programs are the same:

```
10 ADDR = $802000
20 AM = $3D
30 WDAT=$12
40 Vwb(ADDR,AM,WDAT)

10 Vwd($802000,$3D,$12)
```

Error messages are the same as "Vrd"

4.4.4.22. Berdis

Berdis controls the printing of VME bus operation status errors. VME Diagnostics defaults to printing the errors, but to scope loop one will want to shut off the printing.

4.4.4.23. If Then Else

The if then else structure is provided to allow program control based on data values. The only tests available are equal and not equal. There are four forms:

```
IF xxxx=yyyy THEN linenumber 1
IF xxxx< >yyyy THEN linenumber 2
IF xxxx=yyyy THEN linenumber1 ELSE linenumber2
IF xxxx< >yyyy THEN linenumber1 ELSE linenumber2
```

In the IF THEN structure, if the branch condition is not true, program execution continues with the next line. The symbols xxxx and yyyy may be variable names of constants, hence the following two programs perform the same function:

```
10 VAR1 = 1
20 VAR1 = 2
30 IF VAR1 < > VAR2 then 40 ELSE 100
40 PRINT "VAR1 AND VAR2 WERE NOT EQUAL", VAR1," ",VAR2
50 END
100 PRINT "VAR1 AND VAR2 WERE EQUAL"

10 IF 1 < >2 THEN 40 ELSE 100
```

```
40 PRINT "ONE IS NOT EQUAL TWO"
50 END
100 PRINT "ONE EQUALS TWO"
```

4.4.4.24. End

End returns the user back to the VME Diagnostics prompt, >.

```
  download timeline menu or eXit, Quit or ESC
Help
List of *.dat
Display timeline graph of current files to screen
Make a timeline text file to file.tl
looK (browse) file.tl

Choose static filename.      current: STATBOUT.DAT
Browse static file.
Save sequencer static memory to       STATBOUT.DAT
Restore & verify static memory from   STATBOUT.DAT
Verify sequencer static memory to     STATBOUT.DAT

Pick dynamic filename.       current: DYNBOUT.DAT
broWse  dynamic file.
sAve sequencer dynamic memory to      DYNBOUT.DAT
rEstore & verify dynamic memory from DYNBOUT.DAT
verIfy sequencer dynamic memory to    DYNBOUT.DAT

Turn clock off.
turn Unselected group off.
turn selected Group off.
Open a door to dos.
```

download timeline menu or eXit, Quit or ESC

"Help"
Displays this file one screen at a time. This page allows the user to both "Save" Sequencer memory to a file on the P.C.'s disk and also to "Restore" files to the Sequencer's timeline memory. It also provides the ability to plot the timelines from the memory files.

"List of .."
Provides an easy way to see what files already exist. The user is prompted to edit the mask or if "Enter" is pressed the current mask will be used.

"Display timeline graph of current files to screen"
It reads the files selected below and plots the timelines. This is the same as "F1", "Plot.." , "Graphic display to screen of timelines" except the data is read from a file instead of Sequencer memory. Most of the plotting options are controlled from this other page including setting the value of the "Increment..", what kind of printer is attached to the P.C., and control of whether the static address lines come out on the bottom or top of the plot. There was not room to bring these features to this page.

Files are selected with the "Choose static filename" and "Pick dynamic filename" options below.

For help on using the plot and its associated menu see the help on "F1", "Plot...".

"Make a timeline text file to FILE.TL"
is the same as "F1", "Plot.." , "Save (write) current timelines to" except that the user has no option to change the output file. The user may however "Open a door to dos" and use the DOS "copy" or "rename" commands to manipulate files. The format of the output of this file is the exact same as on the other menu. Also this file comes in two flavors depending on where the user wants the address lines. This is again set from "F1", "Plot.." menu with the "Display static address lines (toggle)" (TOP or BOTTOM). This controls where on the plot and for the timeline files the Static Address lines will be positioned. When set to "TOP" the normal timelines will be displayed first, 1 to 24, and then the sixteen address lines, 25 to 40. When this is set to "BOTTOM" the sixteen address lines will be on the bottom followed by the 24 timelines which will be numbered 25-40. This flag also controls the plot on the "download timeline menu". This switch controls both the "Save (write) current timelines" and the "Graphic display to screen of timelines" menu picks on this sub-menu as well as the "Make a timeline text file..." and "Display a timeline graph of current files" on the "Download" sub-menu pick of "F1".

file "DZERO SEQUENCER HARDWARE DESCRIPTION"
mnemonic description by M. Fachin, C. Rotolo 1/18/90

"upa" = upper parity = "HPARITY" bit D11 of dynamic memory.
"syn" = synch = "SYNC_REF" bit D12 of static address memory.
"bc" = beam crossing = "BC_REF" bit D12 of static address memory.
"lpa" = lower parity = "LPARITY" bit D14 of static address memory.
"bct" = bc trigger = "BC_TRIG" bit D15 of static address memory.

"looK (browse) FILE.TL"
Allows the user to look at the data without going to DOS. The "Browse" utility must be available in the current path for this to work.

"Choose static filename."
Allows the user to select which static file to save or restore from.
If the "Backspace" key is hit first then the string may be deleted and changed one character at a time; else if a normal ASCII character is struck the whole string will be deleted and the user may input a new string. The arrow keys are not implemented in editing this field.

"Browse static file."
Allows the user to look at the static file selected above without going to DOS. The "Browse" utility must be available in the current path for this to work.

"Save sequencer static memory to..."
Reads CSR0 to make sure the Sequencer is off as this memory cannot be validly read while the Sequencer is running. If it is not off the user is informed and the operation is aborted. The user may then decide if he really wants to "Turn clock off" below (this is the same control as "Clock..." on F1.)
If the clock is off then the P.C. reads the Sequencer static memory locations as defined by the base address and other parameters on "F1" into the file "..." as selected by " Choose". If the file already exists the user is asked if he wants to over-write it. Only a "y" or "Y" will over-write the file. It is not a good idea to overwrite original source files using this method as various read errors may occur and then corrupt the files. It is a good idea to change this to a new file before saving. After the file is saved the memory is again read and compared to what was just written to the file as in "Verify.." below. This file is in the same format as files that may later be downloaded.

"Restore & verify static memory from ..."
Reads CSR0 to make sure the Sequencer is off as this memory cannot be validly accessed from the VME bus while the Sequencer is using it to generate timelines (on). If the Sequencer is not off the user is informed and the operation is aborted.

The user may then decide if he really wants to "Turn clock off" below (this is the same control as "Clock is on/off" on F1.)

If the clock is off then it reads the file specified under "Choose static filename" and loads it into Sequencer Static memory locations as defined by the base address and other parameters on "F1". The program then reads back the just loaded Sequencer memory and compares what was read back to the file reporting errors to output as specified on "F1", "5 error file". (If "SCREEN" is entered here then error output will be to the screen. A summation is always reported to the screen.) In this version all of the errors are reported. Since this can go on for a long time if the address are not set up right on "F1" the user may use the "ESC" key to abort the list.

The following is a more detailed description from the original Bill Graves document:

"The message 'reading from file' will be printed in the cleared window. The static file is opened and read into an array - the first 1113 double byte data words are read from the file, converted and stored in the first 1113 locations in the array. The next 2982 locations are set to 0. At this point, half of the array is filled, and this half will eventually be loaded into the Next Address portion of Sequencer RAM. Because the locations past address 1112 are 0, if for some reason the Sequencer strays from its normal 0 to 1112 address range, it will return to address 0. Now the next 1113 data words are read from the file, converted and stored in the array starting at index 4096. The remaining 2982 array locations are set to 0.

If during the course of loading the array from the file, the program runs out of data (the file is to small) or there is data left at the end of the load (the file is too large), the message 'file size mismatch in load' will be printed.

Before attempting to write the data to the Sequencer, the Sequencer's CSR0 is read to make sure the Sequencer is off (attempting to access the static portions of Sequencer memory with the Sequencer on will result in Bus Errors). If the sequencer is on, the message 'please turn clock off' will be printed, a low pitched beep will be heard, then the message 'hit a key to return' will appear in the window. Upon hitting a key the user will be returned to the download main menu where the Sequencer may be shut off.

If the Sequencer is off, the data is written to the Sequencer and the verify begins. Once again, the file is read and loaded into an array. The static part of the Sequencer RAM is read into another array, and the two arrays are compared. If the arrays are not the same, the message 'verify failed at addr=xxxxxxxx wrote yyyy read zzzz' is printed..."

"If the file sizes were incorrect, the message 'file size mismatch during verify' will be printed. If the verify was ok, then the message 'download and verify ok' will be printed, and a high pitched beep will be heard. If something went wrong, a low pitched beep will be heard.

The prompt 'hit a key to return' will then be printed. Upon hitting a key, the user will be returned to the download

main menu."

"Verify sequencer static memory to ..."
The following is a more detailed description from the original Bill Graves document:
"The static file is read and loaded into an array. Before attempting to read the data from the Sequencer, the Sequencer's CSR0 is read to make sure it is off (attempting to access the static portions of Sequencer memory with the Sequencer on will result in Bus Errors). If the Sequencer is on, the message 'please turn clock off' will be printed, a low pitched beep will be heard, then the message 'hit a key to return' will appear in the window. Upon hitting a key the user will be returned to the download main menu where the Sequencer may be shut off.

The static part of the Sequencer RAM is read into another array, and the two arrays are compared. If the arrays are not the same, the message 'verify failed at addr=xxxxxxxx wrote yyyy read zzzz' is printed..."

"If the file sizes were incorrect, the message 'file size mismatch during verify' will be printed. If the verify was ok, then the message 'download and verify ok' will be printed, and a high pitched beep will be heard. If something went wrong, a low pitched beep will be heard.

The prompt 'hit a key to return' will then be printed. Upon hitting a key, the user will be returned to the download main menu."

"Pick dynamic filename."
Allows the user to select which dynamic file to save or restore from.
If the "Backspace" key is hit first then the string may be deleted and changed one character at a time; else if a normal ASCII character is struck the whole string will be deleted and the user may input a new string. The arrow keys are not implemented in editing this field.

"broWse dynamic file."
Allows the user to look at the dynamic file selected above without going to DOS. The "Browse" utility must be available in the current path for this to work.

"sAve sequencer dynamic memory to..."
Reads the Sequencer dynamic memory locations as defined by the base address and other parameters on "F1" into the file "..." as selected by "Pick". If the file already exists the user is asked if he wants to over-write it. Only a "y" or "Y" will over-write the file. It is not a good idea to overwrite original source files using this method as various read errors may occur and then corrupt the files. It is a good idea to change this to a new file before saving. After the file is saved the memory is again read and compared to what was just written to the file as in "verIfy.." below. The ability to truely read the memory is conditional on that bank of memory being

off. The program makes this check and does not read the memory and write to the file unless the user selected bank is off.

"rEstore & verify dynamic memory from ..."
Reads the file specified under "Pick dynamic filename" and loads it into Sequencer dynamic memory locations as defined by the base address and other parameters on "F1" if the bank chosen is off. If it was off then the program reads back Sequencer memory and compares it to the file reporting errors to output as specified on "F1", "5 error file". (If "SCREEN" is entered here then error output will be to the screen. A summation is always reported to the screen.)

The following is a more detailed description of this menu pick which corresponds to the original Bill Graves document:

"...loads the data from the dynamic file into the unselected dynamic RAM area (based on the state of the Sel_gr_A/B bit in CSR0), reads the unselected dynamic RAM area back and compares the data. In order to download dynamic data the unselected area of dynamic RAM must be turned off.

"... The message 'reading from file' will be printed in the cleared window. The dynamic file is opened and read into an array - the first 1113 double byte data words are read from the file, converted and stored in the first 1113 locations in the array. The next 2982 array locations are set to $0800. Because the locations past address 1112 contain $0800, if for some reason the Sequencer strays from its normal 0 to 1112 address range, a parity error will occur.

If during the course of loading the array from the file, the program runs out of data (the file is to small) or there is data left at the end of the load (the file is too large), the message 'file size mismatch in load' will be printed.

Before attempting to write the data to the Sequencer, CSR0 is read to make sure the unselected bank of dynamic RAM is off (attempting to write data to the dynamic portions of Sequencer memory while they are on will result in Bus Errors). If the unselected bank is on, the message 'please turn off the unselected group' will be printed, a low pitched beep will be heard, then the message 'hit a key to return' will appear in the window. Upon hitting a key the user will be returned to the download menu where the unselected bank may be shut off.

If the unselected bank is off, the data is written to the Sequencer, and the verify begins. Once again, the file is read and loaded into an array. The unselected dynamic bank of Sequencer RAM is read into another array, and the two arrays are compared. If the arrays are not the same, the message 'verify failed at addr=xxxxxxxx wrote yyyy read zzzz' is printed..."

"If the file sizes were incorrect, the message 'file size mismatch during verify' will be printed. If the verify was ok, then the message 'download and verify ok' will be printed, and a high pitched beep will be heard. If something went wrong, a low pitched beep will be heard.

The prompt 'hit a key to return' will then be printed. Upon hitting a key, the user will be returned to the download

main menu."

"verIfy sequencer dynamic memory to ..."
Reads the file specified under "Pick" and compares it to what is in the Sequencer dynamic memory reporting any differences as above.

The following is a more detailed description of this menu pick which corresponds to the original Bill Graves document:

"The Verify Dynamic differs from the static verify in that the user is prompted for the dynamic group (A or B) to verify. This group may be selected or unselected and on or off.

Once again, the dynamic file is read and loaded into an array. Before attempting to read the data from the Sequencer, CSR0 is read to see which dynamic group is currently selected. If group A is selected, the following message appears:

'group A is currently selected
verify to group A or B?'

If group B is currently selected, the message states 'group B is currently selected'. The user will be prompted until an A or B (in upper or lower case) is entered. If the desired group is off, the actual verify happens. If the desired group is on, one of two messages will occur:

'please turn off the unselected group'

or

'please turn off the selected group'

depending on whether the group is selected or unselected. A low beep will be heard, and the message ' hit a key to return' is printed. The user is then returned to the main download menu where the group may be turned off.

"Turn clock off"
Is repeated here from "F1" for the users convenience. It turns off the Sequencer by clearing the Clock_on bit in CSR0.

"turn Unselected group off."
Is repeated here from "F1" for the users convenience. Shuts off the unselected dynamic group by clearing either the Group_A_on bit or the Group_B_on bit in CSR0.

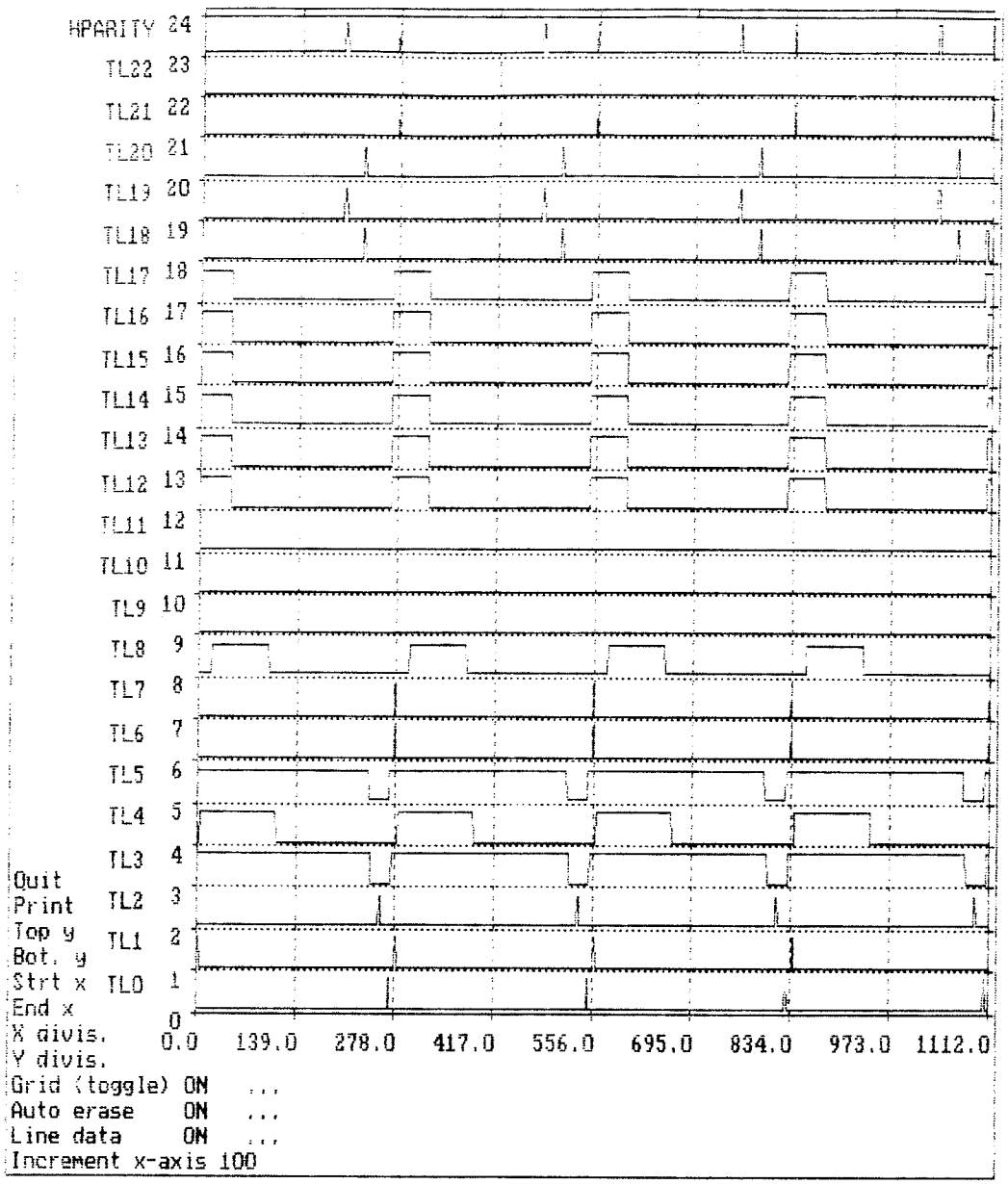
"turn selected Group off"
Is repeated here from "F1" for the users convenience. Shuts off the selected dynamic group by clearing either the Group_A_on bit or the Group_B_on bit in CSR0.

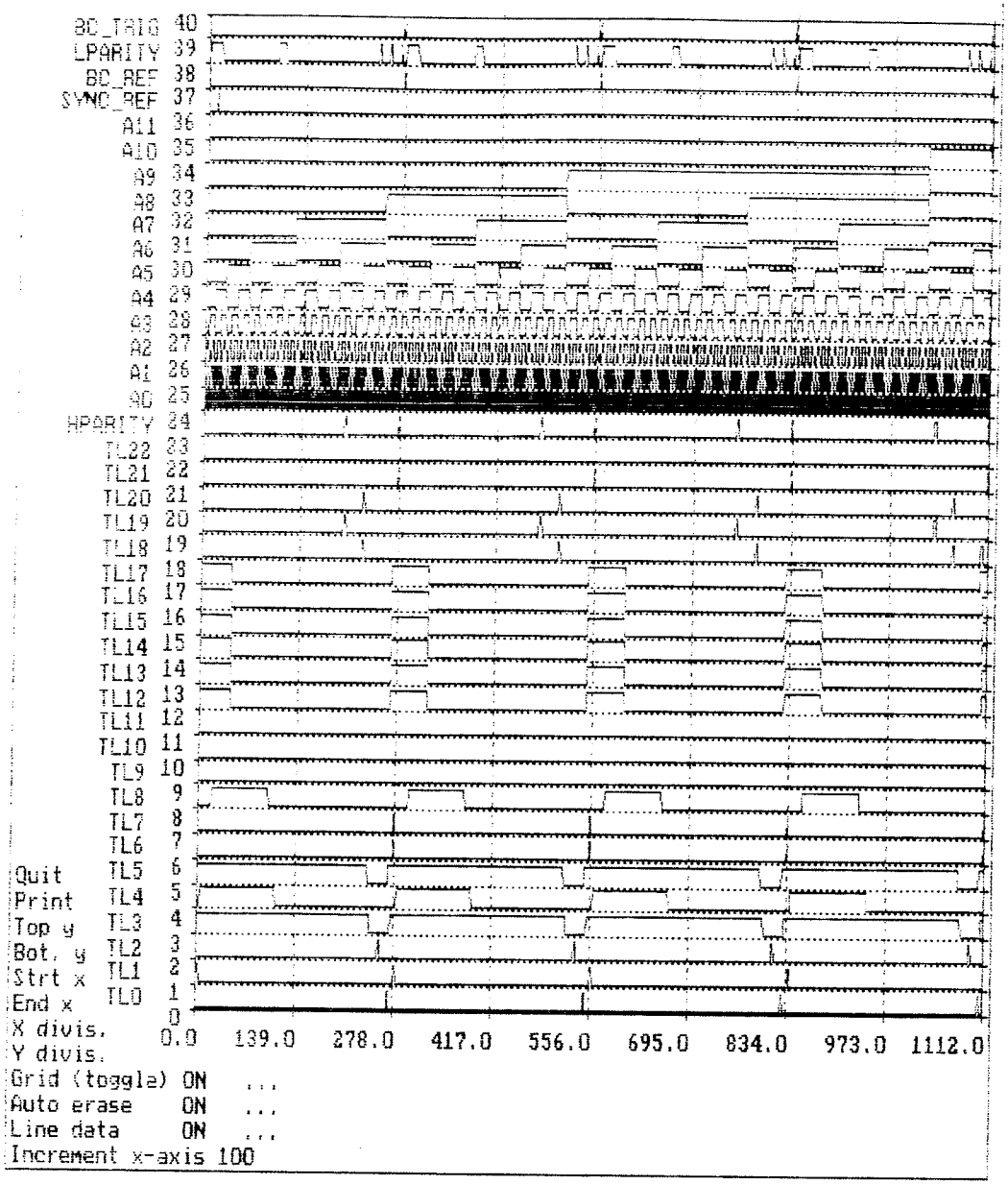
"Open a door to DOS"
Passes whatever the user type's to DOS without leaving the program until the user hits the "Enter" key. Then it comes back into the program. "CT" is still using up considerable memory of DOS's precious 640k bytes so there may not be enough to do what you want.

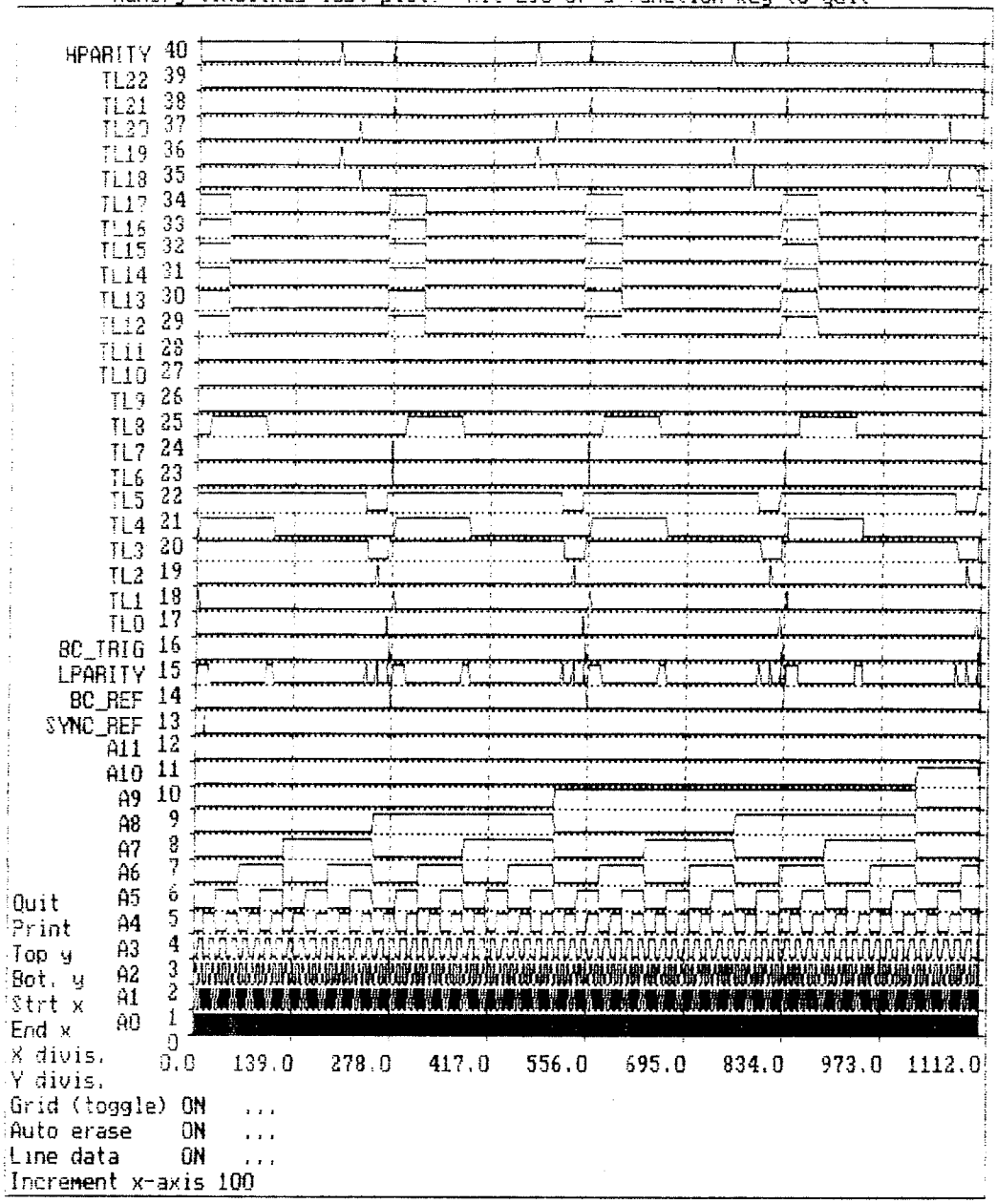
memory plot menu or eXit, Quit or ESC

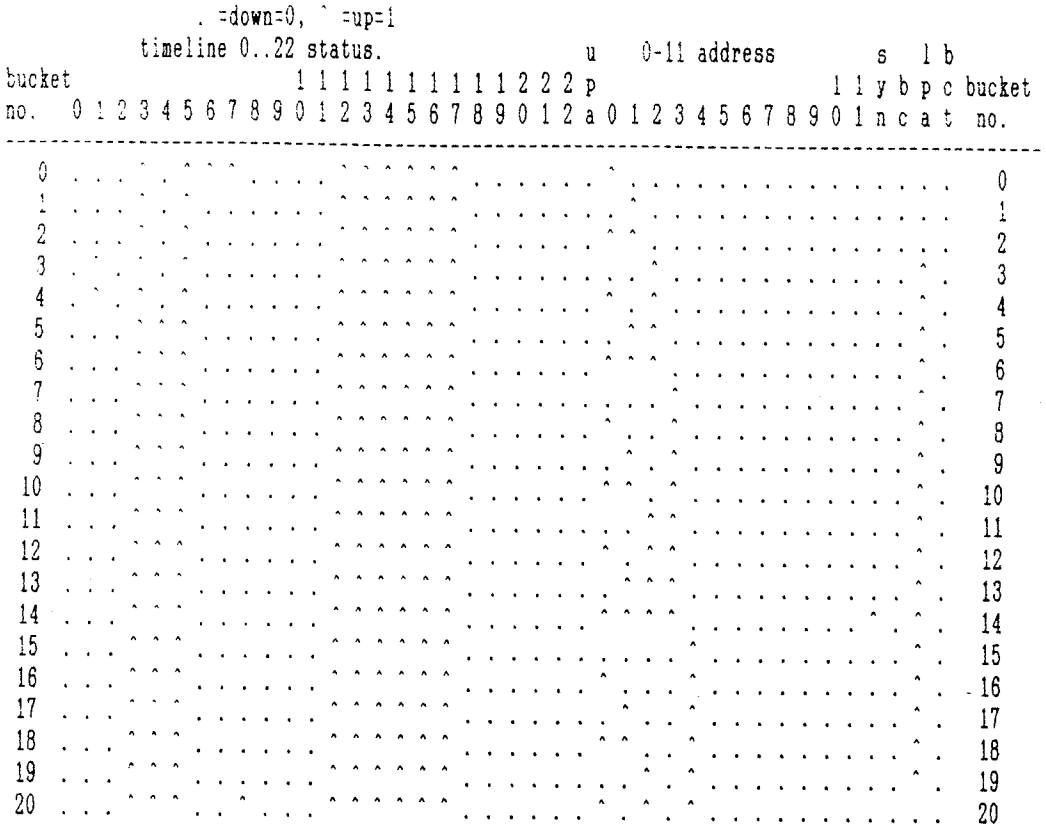
```
Help
Minimum (first) bin to start from.        current:  0
Ending bin.                               current:  1112
0 (down) is represented by                current:  .
1 (up) is represented by                  current:  ^
Text display to screen of timelines

Display static address lines (toggle)     current:  TOP ....
Auto increment on x-axis by               current:  100
Ibm pro/graph printers=on.epson=off.      current:  ..   OFF
Graphic display to screen of timelines

List of output files.          current mask: *.tl
File to act on.                     current: read.tl
Save  (write)  current  timelines  to file: read.tl

Browse file = read.tl
Print  file = read.tl  (use "print" in DOS first!)
Copy   file = read.tl to "lpt1".
Open a door to DOS
```

memory timelines first plot. hit ESC or a function key to Quit




memory timelines first plot. hit ESC or a function key to Quit
BC_TRIG 40
LPARITY 39
BC_REF 38
SYNC_REF 37
A11 36
A10 35
A9 34
A8 33
A7 32
A6 31
A5 30
A4 29
A3 28
A2 27
A1 26
A0 25
HPARITY 24
TL22 23
TL21 22
TL20 21
TL19 20
TL18 19
TL17 18
TL16 17
TL15 16
TL14 15
TL13 14
TL12 13
TL11 12
TL10 11
TL9 10
TL8 9
TL7 8
TL6 7
TL5 6
TL4 5
TL3 4
TL2 3
TL1 2
TL0 1
0
Quit
Print
Top y
Bot. y
Strt x
End x
X divis. 0.0 139.0 278.0 417.0 556.0 695.0 834.0 973.0 1112.0
Y divis.
Grid (toggle) ON ...
Auto erase ON ...
Line data ON ...
Increment x-axis 100


memory timelines last plot. hit ESC or a function key to Quit
HPARITY 40
TL22 39
TL21 38
TL20 37
TL19 36
TL18 35
TL17 34
TL16 33
TL15 32
TL14 31
TL13 30
TL12 29
TL11 28
TL10 27
TL9 26
TL8 25
TL7 24
TL6 23
TL5 22
TL4 21
TL3 20
TL2 19
TL1 18
TL0 17
BC_TRIG 16
LPARITY 15
BC_REF 14
SYNC_REF 13
A11 12
A10 11
A9 10
A8 9
A7 8
A6 7
A5 6
A4 5
A3 4
A2 3
A1 2
A0 1
0
Quit
Print
Top y
Bot. y
Strt x
End x
X divis.
Y divis.
0.0 139.0 278.0 417.0 556.0 695.0 834.0 973.0 1112.0
Grid (toggle) ON ...
Auto erase ON ...
Line data ON ...
Increment x-axis 100


. =down=0, ^ =up=1
timeline 0..22 status.
u
0-11 address
s
l
b
bucket
no.
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 a 0 1 2 3 4 5 6 7 8 9 0 1
y
n
b
c
a
p
t
bucket
no.
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20

1113

1113

Memory Timeline Display/Plot Sub-Menu Detailed Help Notes

"Help."
Displays this file one screen at a time.
This page reads the Sequencer hardware selected on "F1"
in the same way that the "F1" "N step through menu"
uses the hardware. It interferes with the normal operation
of the clock. Please refer to the "Fermilab Users Guide
D0 Clock Control and Test Program V1.03" by Bill Graves
1/31/90 section 4.4.3 through 4.4.3.4.

"Minimum (first) bin to start from."
Determine the first bin to start displaying to the screen.
Valid bin numbers are 0 to 1112. Input is clamped inside
of the valid range.

"Ending bin."
Determine the last bin to start displaying to the screen.
Valid bin numbers are 0 to 1112. Input is clamped inside
of the valid range.

"0 (down) is represented by"
This allows the user to choose the ASCII character which
will represent the binary value of 0.

"1 (up) is represented by"
This allows the user to choose the ASCII character which
will represent the binary value of 1.

"Text display of screen of timelines"
Reads the sequencer memory by "stepping" through it
and displays the timelines in a character based mode
modulo 24 of the bucket number.

"Display static address lines (toggle)" (TOP or BOTTOM)
Controls where on the plot and for the timeline files the
Static Address lines will be positioned. When set to "TOP"
the normal timelines will be displayed first, 1 to 24, and
then the sixteen address lines, 25 to 40. When this is set
to set to "BOTTOM" the sixteen address lines will be on the
bottom followed by the 24 timelines which will be numbered
25-40. This flag also controls the plot on the "download
timeline menu". This switch controls both the "Save (write)
current timelines" and the "Graphic display to screen of
timelines" menu picks on this sub-men as well as the "Make a
timeline text file..." and "Display a timeline graph of
current files" on the "Download" sub-men pick of "F1".

file "DZERO SEQUENCER HARDWARE DESCRIPTION"
mnemonic description by M. Fachin, C. Rotolo 1/18/90

"upa" = upper parity ="HPARITY" bit D11 of dynamic memory.
"syn" = synch ="SYNC_REF" bit D12 of static address memory.

"bc" = beam crossing="BC_REF" bit D12 of static address memory.
"lpa" = lower parity ="LPARITY" bit D14 of static address memory.
"bct" = bc trigger ="BC_TRIG" bit D15 of static address memory.

"Auto increment on x-axis by"
Pertains to the next menu pick. This allows the user to change the size of how many bins to jump when incrementing the plot from the Graphic sub-menu. This was put here instead of the graphic due to space limitations.

"Ibm pro/graph printers=on.epson=off."
If "current= ... OFF" then this program is set to work with an Epson FX (9 pin) dot matrix printer when the Epson DIP switch set to Epson mode. (See your printer manual as they vary.)
If current= ...ON" then this program is set to work with the IBM Professional and Graphics printers and compatible (including Epson set to IBM compatible mode). (If your plot comes out with funny vertical separations in it then change this and try it again!)

"Graphic display to screen of timelines"
Reads the Sequencer memory into the P.C. by "stepping" through Sequencer memory. It then plots the timelines. This has it's own menus associated with it allowing the user to expand the plot or look at a particular section of the timelines in detail in a graphical manner. It only reads the Sequencer once and keeps reploting what it read until the user presses the "Escape" key or another function key. To make use of the "Increment.." feature set the " End x" to the same as what "Auto increment" is set to first and then press "I" for "Increment..". The x-axis maximum is clamped and there is auto wrapping logic in the code on the x-axis minimum. These clamps and wrapping are only valid on the "Increment.." feature so they may be manually over-ridden.
All 1113 buckets of all 40 timelines are always being plotted. If the scaling is changed during the plot it is possible to not see anything happening and not to see a plot. It may be necessary to strike a key (the spacebar is usually safe) to get the screen to refresh after changing one or more of the following:
"Quit" goes all the way out to the main menu.
"Print" tries to open "LPT1" as a file and then write an image of the screen to it.
"Top y" meaningful values here are 2-40. Can be used with below to select a range of timelines. Changing this will automatically adjust the number of y divisions to be the number of timelines you are plotting but you may over-ride this with the "Y divis." pick yourself.
"Bot y" meaningful values here are 1-39. Can be used with above to select a range of timelines. Changing this will automatically adjust the

number of y divisions to be the number of timelines you are plotting but you may over-ride this with the "Y divis." pick yourself.

"Strt x"Determines which R.F. bucket (0-1112) will be lined up at the left plot axis.

"End x"Determines which R.F. bucket (0-1112) will be lined up at the right plot axis.

"X divis." How many x-axis divisions will be on the plot.

"Y divis." How many y-axis divisions will be on the plot.

"Grid" Turn on and off the dashed lines on the inside of the plot.

"Auto erase" Is probably not very useful here but it can be used to stop erasing the screen after the user changes one of these menu picks.

"Line data" Turns on and off connecting the buckets with lines. It can be hard to see a single pixel but one might only be interested in them. Note that x-axis plot for an EGA and VGA adapter is 512 x-axis pixels between the margins. It is possible to have this off and still look (mostly) like lines when packing 512 or more buckets into this space.

"Increment x-axis..." Adds the value displayed and set on the calling menu to both the "Strt x" and "End x" values to save time. This has some auto-wrap logic.

"List of output files."
Provides an easy way to see what files already exist. The user is prompted to edit the mask or if "Enter" is pressed the current mask will be used.

"File to act on."
This is the file name that this page will use to write to, or print from.

"Save (write) current timelines to file: ..."
Reads the Sequencer memory by "stepping" it as above into P.C. memory and then writes this information to the "File to act on."

WARNING:
THERE MUST BE ENOUGH AVAILABLE MEMORY IN YOUR MACHINE FOR THE FOLLOWING TO WORK. (If you are calling this program from another there may not be enough left for these to dynamically use the extra memory that they need.)

"Browse file"
Allows the user to look at the data without going to DOS. The "Browse" utility must be available in the current path for this to work.

"Print file"

Passes the name of the file to the DOS "print" queue buffer.
WARNING: this can have unwanted side effects!
If the DOS "print" queue buffer was not previously run or set up before this program was loaded then DOS sets up the "print" queue buffers in the memory after the memory this program is using. Then when this program is exited DOS will not recover the program memory because the "print" buffers were loaded after it. The only (normal) way to recover this memory is to reboot as DOS cannot recover fragmented memory.

(This problem may be avoided by allocating memory at boot time by placing the following two lines in your autoexec.bat file:

"mode LPT1:
print/d:LPT1"

The exact form of this may vary slightly from clone to clone so see the DOS "mode" and "print" commands for your machine.)

WARNING:
It is possible to exit or "Open a door to dos" and delete the file you sent to the printer before the it has finished printing. If this happens then only part of the file will be printed.

If the file is a longer file then the DOS "print" buffers and the printer's buffers then the file will be deleted before it gets to the printer. (This is true unless you are using a more sophisticated "spooling" print queue utility than the standard simple DOS one.)

"Copy file..."
Uses the DOS "copy" command to copy the file to the DOS device "lpt1" which is normally the printer. Use this instead of "Print" to avoid the memory buffer problems mentioned above. (A possible penalty here is that it may tie up the machine longer while the file is being printed. This depends on the file length and the length of the printer's hardware buffer.)

"Open a door to DOS"
Passes whatever the user type's to DOS without leaving the program until the user hits the "Enter" key. Then it comes back into the program. "CT" is still using up considerable memory of DOS's precious 640k bytes so there may not be enough to do what you want.

ram/register/address tests or eXit, Quit, or ESC

A address switch test.
B random read addressing test. (for "bit3" in empty crate.)
C csr test
D address modifier test (for "bit3" only).
E static "RAM" test
F dynamic bank "A RAM" test
G dynamic bank "B RAM" test
H next sequencer address "RAM" test
I loop any option "B-H" until key pressed
J loop all options "B-H" until key pressed
K error message destination. current: SCREEN
L additional information current: .. OFF
M how many of the first ram failures to display. current: 17

N help.

your choice ?

D0 Sequencer Clock Test Sub-Sub-Menu
4/11/91

This page is similar in function as the original Version 1.03 main menu pick one except that the menu picks "K" through "N" have been added.

A lot of the following is mostly quoted from section 4.4.2.1 through 4.4.2.12 of the original Version 1.03 Users Guide D0 Control and Test Program by Bill Graves. My changes and comments are in parenthesis preceded by the word "Note:".

4.4.2.1. Suggested Test Procedure
(Note: this suggestion was originally intended for the original version which ran only in the lab. Tests 'B' and 'D' are meaningless at D0 or over the Token Ring.)

It is recommended that the tests be run in the order in which they appear in the menu. Once tests B-H work, one should select option J and let the tests run continuously for 5 minutes or so. If a test fails, read the test description below, and enter the VME Diagnostics (option V in the main menu) and write a small program which will allow you to look with an oscilloscope and see what is wrong with the Sequencer.

4.4.2.2. Test descriptions

The following sections explain what is being tested in each test, and how the tests work. Upon receiving an error message while testing the Sequencer, a technician should be able to read the appropriate section and know where to look on the Sequencer (and how to write a smaller test in the VME Diagnostics) in order to fix the problem.

"A"
4.4.2.3. Address Switch Test

Upon selecting A in response to the Clock Tests menu, the user begins the address switch test. The screen will clear and the following message will appear:

Now commencing address switch test

Set D0 Clock address switch to $80.
After each beep, switch in the next switch position, until they are all set.
Hit a key to begin

The address switch test works by telling the technician to set the Sequencer address switch to $80, then doing a read of the Sequencer control/status register. After each read, the computer beeps, telling the technician to switch the next Sequencer address switch until all the switches have been switched, resulting in an address of $FF. If the returned VMEbus status from each read was not DTACK, the following error message is printed:

Did not respond to VMEbus address xxxx

(Note: currently the Token Ring Version does not report "no DTACK". This test will not correctly work over the Token Ring on this version.)

In this error message, xxxx is the value to which the Sequencer address switch should be set. The Sequencer has not responded to a read of its control/status register. If the Sequencer has passed this test, the next line to appear on the screen will be:

test passed
set your switch back to xxx
Type any key to exit

"B"

4.4.2.4 (random read) address test.

(Note: This test is obviously not to be run in place at D0. It should only be run in the lab in an empty crate via the Bit 3. and not over the Token Ring.)

Upon selecting B from the Clock Tests menu, the user begins the address test. The screen will clear, and the following message will appear:

Now commencing address test

The address test performs its function by generating a random VMEbus address, and doing a read on VMEbus using this address. This procedure is repeated 1000 times. If the Sequencer responds to and invalid address, the following message will appear:

Addr test: Received dtack to address of xxxxxxxx

In this error message, xxxxxxxx is the VMEbus address which was being read from when the DTACK was received. It may be that the DTACK came from some other board int the VME crate(s)... non-essential VMEbus slaves should be removed from the crate before testing the Sequencer. If the Sequencer passes this test, the next line to appear on the screen will be:

test passed
Type any key to exit

In order to be a valid Sequencer address, the upper 8 bits of an (upper 8 bits of a 24 bit) address must correspond to the address switch setting, and the low 16 bits must be $2000, $2001, $4000, $40001 or in the range $8000-$FFFF.

"C"

4.4.2.5 CSR Test

Upon selecting C from the Clock Tests menu, the user begins the control/status register test. The screen will clear, and the following message will appear:

Now commencing CSR test

The CSR test performs its function by setting and clearing

bits in the CSRs and reading the CSRs back to see if they contain the correct data. Pseudocode for the CSR test is as follows:

```
set clock_on in CSR0
read CSR0
If clock_on = 0 then call error_message

clear clock_on in CSR0
read CSR0
If clock_on = 1 then call error_message

set output_enable in CSR0
read CSR0
If output_enable = 0 then call error_message

clear output_enable in CSR0
read CSR0
If output_enable = 1 then call error_message

set step_mode in CSR0
read CSR0
If step_mode = 0 then call error_message

clear step_mode in CSR0
read CSR0
If step_mode= 1 then call error_message

set free_run in CSR0
read CSR0
If free_run = 0 then call error_message

clear free_run in CSR0
read CSR0
If free_run = 1 then call error_message

set group_a_on in CSR0
read CSR0
If group_a_on = 0 then call error_message

clear group_a_on in CSR0
read CSR0
If group_a_on = 1 then call error_message

set group_b_on in CSR0
read CSR0
If group_b_on = 0 then call error_message

clear group_b_on in CSR0
read CSR0
If group_b_on = 1 then call error_message

set sel_gr_a in CSR0
read CSR0
If sel_gr_a = 0 then call error_message
```

```
clear sel_gr_a in CSR0
read CSR0
If sel_gr_a = 1 then call error_message

set set_errors in CSR0
read CSR0
If set_errors = 0 then call error_message
read CSR1
If CSR1 < > $1ff then call error_message
```

the error messages look like:

csr test: expected xxxx in csrx but read yyyy after setting (or clearing) the zzzz bit in csrx.

If the Sequencer passes the CSR test, the next screen will be:

```
test passed
Type any key to exit
```

"D"

4.4.2.6. Address Modifier Test

(Note: this test is only applicable for the Bit 3 Versions.)

Upon selecting D from the clock Tests menu, the user begins the Address Modifier test. This test checks to see that the Sequencer only responds to valid address modifier codes. The screen will clear, and the following message will appear:

```
Now commencing address modifier test
```

The Address Modifier test performs its functions by attempting reads of the Sequencer CSR0 with various invalid address modifiers. The returned VMEbus status is expected to be bus error - if it is not, an error message is printed. Pseudocode for the address modifier test is as follows:

```
for address_modifier = 0 to 40
   read Sequencer CSR0
   if VME status = ok and address modifier is not valid
     then errmsg1
   if VME status not ok and address modifier is valid
     then errmsg2
for ct = 1 to 100
   generate random address modifier
   read Sequencer CSR0
   if VME status = ok and address modifier is not valid
     then errmsg3
   if VME status not ok and address modifier is valid
     then errmsg4
```

the various error messages are:

1) Read CSR with an am of xx and received no berr

2) Error: Read CSR with a valid am of xx and received no dtack
3) Read CSR with a random am of xx and received no berr (Bit 3 error)
4) Error: Read CSR with a valid random am of xx and received no dtack

If the Sequencer passes the address modifier test, the next screen will be:

test passed
Type any key to exit

Valid address modifiers are $39 and $3D.

"E"
4.4.2.7. Static RAM test
Upon selecting E from the Clock Tests menu, the user begins the Static RAM Test. This test verifies the integrity of the static data memory. The screen will clear, and the following message will appear:

testing static ram

The first test involves writing, reading and comparing of known data to all of the RAM. The known data in this case is the address to which the data is going. The following message is output:

using address as data

If this test passes, the next test begins - this time the data $AAAA is written to every location. The following message is output:

using aaaa as data

If this test passes, the next test begins - this time the data $5555 is written to every location. The following message is output:

using 5555 as data

If this test passes, the next test begins - this time random data is written to every location. The following message is output:

using random data

If compare errors do occur, the following error message is output:

data error at addr xxxxxx, wrote xxxx, read xxxx

Only the first (Note: this is now controlled by the user with menu pick "M") errors are output, but the rest are counted, and at the end of the compare the total number of errors is output:

total errors in test =xxxx

If there was a compare error in a test, the following tests don't execute - so if the test using AAAA as data has a compare error, the next two tests (5555 and random) don't execute. If all of the tests execute and the Sequencer passes, the message "tested ok" is output, otherwise, "test failure" is output. The final message, as always, is "Type any key to exit".

"F"

4.4.2.8. Dynamic bank A RAM test

Upon selecting F from the clock Tests menu, the user tests the Group A Dynamic Data Memory to verify its integrity. The known data in this case is the address to which the data is going. The screen will clear, and the following message will appear:

testing dynamic ram - bank A

The first test involves writing, reading and comparing of known data to all of the Dynamic Group A RAM section. The following message is output:

using address as data

If this test passes, the next test begins - this time the data $AAAA is written to every location. The following message is output:

using aaaa as data

If this test passes, the next test begins - this time the data $5555 is written to every location. The following message is output:

using 5555 as data

If this test passes, the next test begins - this time the random data is written to every location. The following message is output:

using random data

If compare errors do occur, the following error message is output:

data error at addr xxxxxxxx, wrote xxxx, read xxxx

Only the first ((Note: this is now controlled by the user with menu pick "M")) errors are output, but the rest are counted and at the end of the compare the total number of errors is output:

total errors in test=xxxx

If there was a compare error in a test, the following tests don't execute - so if the test using AAAA as data has a compare error, the next two tests (5555 and random) don't execute. If all of the tests execute and the Sequencer passes, the message "tested ok" is output, otherwise, "test failure" is output. The final message, as always, is "Type any key to exit".

"G"

4.4.2.9. Dynamic bank B RAM test

Upon selecting G from the clock Tests menu, the user tests the Group B Dynamic Data Memory to verify its integrity. This test verifies the integrity of the Dynamic Group B section of Sequencer memory. The screen will clear, and the following message will appear:

testing dynamic ram - bank B

The first test involves writing, reading and comparing of known data to all of the Dynamic Group B RAM section. The known data in this case is the address to which the data is going. The following message is output:

using address as data

If this test passes, the next test begins - this time the data $AAAA is written to every location. The following message is output:

using aaaa as data

If this test passes, the next test begins - this time the data $5555 is written to every location. The following message is output:

using 5555 as data

If this test passes, the next test begins - this time the random data is written to every location. The following message is output:

using random data

If compare errors do occur, the following error message is output:

data error at addr xxxxxxxx, wrote xxxx, read xxxx

Only the first ((Note: this is now controlled by the user with menu pick "M")) errors are output, but the rest are counted and at the end of the compare the total number of errors is output:

total errors in test=xxxx

If there was a compare error in a test, the following tests don't execute - so if the test using AAAA as data has a compare error, the next two tests (5555 and random) don't execute. If all of the tests execute and the Sequencer passes, the message "tested ok" is output, otherwise, "test failure" is output. The final message, as always, is "Type any key to exit".

"H"

4.4.2.10. Next Sequencer Address Ram test

Upon selecting H from the clock Tests menu, the user begins the Next Sequencer Address RAM test. This test verifies the integrity of the next address section of Sequencer memory. The screen will clear, and the following message will appear:

testing next address ram

The first test involves writing, reading and comparing of known data to all of the next address RAM section. The known data in this case is the address to which the data is going. The following message is output:

using address as data

If this test passes, the next test begins - this time the data $AAAA is written to every location. The following message is output:

using aaaa as data

If this test passes, the next test begins - this time the data $5555 is written to every location. The following message is output:

using 5555 as data

If this test passes, the next test begins - this time the random data is written to every location. The following message is output:

using random data

If compare errors do occur, the following error message is output:

data error at addr xxxxxxxx, wrote xxxx, read xxxx

Only the first ((Note: this is now controlled by the user with menu pick "M")) errors are output, but the rest are counted and at the end of the compare the total number of errors is output:

total errors in test=xxxx

If there was a compare error in a test, the following tests don't

execute - so if the test using AAAA as data has a compare error, the next two tests (5555 and random) don't execute. If all of the tests execute and the Sequencer passes, the message "tested ok" is output, otherwise, "test failure" is output. The final message, as always, is "Type any key to exit".

"I loop any option 'B-H' until key pressed"
When "I" is selected the user is prompted to enter B - H or ESC to eXit. Any other input is ignored. "The individual tests run almost the same when looping, except that there is an added " field that is updated on how many loops have been executed.
This is good for troubleshooting the electronics with a scope as the tests just repeat.

"J loop all options 'B-H' until key pressed"
Upon selecting option J from the main menu the user enters the loop on all options B - h world. This allows the user to loop on all of the looped tests until a key is pressed. The screen will clear, and the tests B - H will begin executing in sequence. When H is done, the loop starts again with B. This continues until a key is pressed.
The individual tests run almost the same when looping, except that there is an added message on most errors:

lpct = xxx

this lpct is just a count of how many times the test B - H have been run. Also, errors won't result in some tests not being executed - the tests will loop until a key is pressed whether it is passing or failing.

"K error message destination"
Allows the user to re-route most error message. If it says "SCREEN" then all output will be put to the CRT. If any other filespec is put there than most output will go to disk with some summary messages still always coming to the screen.

"L additional information"
If this is on then more information will be output on some tests. This may be useful in seeing what is going on.

"M how many of the first ram failures to display"
This allows the user to set how many RAM errors messages to display before suppressing the rest. This provides a way of turning off and controlling the message suppression feature.

"N help"
Displays this file one screen at a time. There is no going backwards except to hit the ESC key and start over. At the end of the file the filename will be displayed so the user may print this out or look at

it with another utility such as "browse" or even edit this file with a text editor.

step through menu

Help
Change coarse key value. current= 100 buckets
Step through buckets by using above value.
or step 1,2,3,4,5,6,7,8,9 for their respecitve no. of buckets.
Plot menu of sequencer memory using step through mechanism.
Goto a bucket
eXit back to the main menu

bucket #: 0

Enter the letter of your choice:

Line	Statu
TL0)	down
TL1)	down
TL2)	down
TL3)	UP
TL4)	down
TL5)	UP
TL6)	UP
TL7)	UP
TL8)	down
TL9)	down
TL10)	down
TL11)	down
TL12)	down
TL13)	down
TL14)	down
TL15)	down
TL16)	down
TL17)	down
TL18)	down
TL19)	down
TL20)	down
TL21)	down
TL22)	down

D0 Sequencer Step Sub-Sub-Menu Help Notes
4/11/91

This page is similar in function as the original Version 1.03 main menu pick one except that the user interface has been slightly changed and the ability to plot the memory using the step through algorithm has been added.

A lot of the following is mostly quoted from section 4.4.3 through 4.4.3.4 of the original Version 1.03 Users Guide D0 Control and Test Program by Bill Graves. My changes and comments are in parenthesis preceded by the word "Note:".

4.4.3 Option N: Step Through (of F1)

Upon selecting option N, the users enters the step through menu. The current state of the Sequencer (CSR0) is saved and the Sequencer is: put in step mode, turned on, outputs are enabled, and put in free run mode. When one exits Step Through, the previous state of the Sequencer is restored. Step Through is a way of verifying the contents of Sequencer RAM. It allows one to examine the potential state of the timing lines at any desired bucket. Step through will only work if a file has been loaded into the Sequencer. The Line Status column of the Step Through menu shows the RAM contents of the current bucket (displayed by the bucket #xxx in the center of the screen)...

"Help"
Displays this file one screen at a time. There is no going backwards except to hit the ESC key and start over. At the end of the file the filename will be displayed so the user may print this out or look at it with another utility such as "browse" or even edit this file with a text editor.

"Change coarse key value. current= xxxx buckets"
(4.4.3.1: old Option V: Assign Coarse Key)

The purpose of (...this option..) is to allow one to change the value of the coarse key (key s) from its default of 100. Valid values for the coarse key are 10-1113. ...the user will be prompted "type in the value you wish the coarse key to have:" valid responses result in the value of the coarse key being changed. Invalid responses result in error messages, and the user is prompted again for a valid response. A response of less than 10 results in the message "A fine key already has this value" being output. A response of greater than 1113 results in the message "Values cannot exceed 1113" being output.

(Note: User input is in decimal unless preceded by a '$' which means that user input is in hex.)

"Step through buckets by using above value."
(4.4.3.2. Option S: Step Coarse)

When the user selects option S, the bucket number is increased by the value of S, and the state of the timing lines at that bucket is displayed.

4.4.3.3. Options 1-9: Step Fine

When the user selects options 1 - 9 the bucket number is increased by the selected number, and the state of the timing lines at that bucket is displayed.

"Plot menu of sequencer memory using step through mechanism." This is a new menu with it's own help notes. It allows several means of displaying all of memory buckets. It's draw back is that it is a little slow in gathering all of the information in this manner. It is especially slow in the Token Ring Version.

4.4.3.4. Option G: Goto a bucket

Upon selecting option G., the user is prompted: "Type in the value of the bucket you wish go to:". Valid responses are from 0-1112. If the response contains an non number key, the message "Stick to number keys, please" is output, and the user is prompted again for a valid response. If the response is a negative number, the message "No negative values allowed" is output, and the user is prompted again for a valid response. If the response is greater than 1112, the message "Values cannot exceed 1112" is output, and the user is prompted again for a valid response, the bucket number is changed and the state of the timing lines is displayed.

```
 step test plot menu or eXit, Quit or ESC

Help
Minimum (first) bin to start from.      current:  0
Ending bin.                             current:  1112
0 (down) is represented by              current:  .
1 (up) is represented by                current:  ^
Text display to screen of timelines

Display static address lines (toggle)   current:  TOP ....
Auto increment on x-axis by             current:  100
Ibm pro/graph printers=on.epson=off.    current:  ..   OFF
Graphic display to screen of timelines

List of output files.          current mask: *.tl
File to act on.                     current: read.tl
Save  (write)  current  timelines  to file: read.tl

Browse file = read.tl
Print  file = read.tl  (use "print" in DOS first!)
Copy   file = read.tl to "lpt1".
Open a door to DOS
```

Step Timeline Display/Plot Sub-Menu Detailed Help Notes

These help notes are the same in function as the Memory Timeline ..." ones except that the data plotted is obtained using the step through mechanism and is much slower especially over the network. A lot of the internal variables on this page are the same as the ones on "F1" plot.

This is really more of an exercise of the Sequencer hardware than it is to plot it; the plot is just to display the results of exercising the hardware. The Sequencer is put in step mode, turned on, outputs are enabled, and put in free run mode. Step Through starts at bucket 0 reads that location, increments the bucket, reads the next location and so on until all 1113 buckets have been stored in the P.C.'s RAM. So it somewhat automates the previous page. The two pages should give the same results. It is a way of verifying the contents of Sequencer RAM. It allows one to examine the potential state of the timing lines at any desired bucket. Step through will only work if a file has been previously loaded into the Sequencer.

"Help."
Displays this file one screen at a time.
This page reads the Sequencer hardware selected on "F1" in the same way that the "F1" "N step through menu" uses the hardware. It interferes with the normal operation of the clock. Please refer to the "Fermilab Users Guide D0 Clock Control and Test Program V1.03" by Bill Graves 1/31/90 section 4.4.3 through 4.4.3.4.

"Minimum (first) bin to start from."
Determine the first bin to start displaying to the screen. Valid bin numbers are 0 to 1112. Input is clamped inside of the valid range.

"Ending bin."
Determine the last bin to start displaying to the screen. Valid bin numbers are 0 to 1112. Input is clamped inside of the valid range.

"0 (down) is represented by"
This allows the user to choose the ASCII character which will represent the binary value of 0.

"1 (up) is represented by"
This allows the user to choose the ASCII character which will represent the binary value of 1.

"Text display of screen of timelines"
Reads the sequencer memory by "stepping" through it and displays the timelines in a character based mode modulo 24 of the bucket number.

"Display static address lines (toggle)" (TOP or BOTTOM)

Controls where on the plot and for the timeline files the Static Address lines will be positioned. When set to "TOP" the normal timelines will be displayed first, 1 to 24, and then the sixteen address lines, 25 to 40. When this is set to set to "BOTTOM" the sixteen address lines will be on the bottom followed by the 24 timelines which will be numbered 25-40. This flag also controls the plot on the "download timeline menu". This switch controls both the "Save (write) current timelines" and the "Graphic display to screen of timelines" menu picks on this sub-men as well as the "Make a timeline text file..." and "Display a timeline graph of current files" on the "Download" sub-men pick of "F1".

file "DZERO SEQUENCER HARDWARE DESCRIPTION"
mnemonic description by M. Fachin, C. Rotolo 1/18/90

"upa" = upper parity ="HPARITY" bit D11 of dynamic memory.
"syn" = synch ="SYNC_REF" bit D12 of static address memory.
"bc" = beam crossing="BC_REF" bit D12 of static address memory.
"lpa" = lower parity ="LPARITY" bit D14 of static address memory.
"bct" = bc trigger ="BC_TRIG" bit D15 of static address memory.

"Auto increment on x-axis by"
Pertains to the next menu pick. This allows the user to change the size of how many bins to jump when incrementing the plot from the Graphic sub-menu. This was put here instead of the graphic due to space limitations.

"Ibm pro/graph printers=on.epson=off."
If "current= ... OFF" then this program is set to work with an Epson FX (9 pin) dot matrix printer when the Epson DIP switch set to Epson mode. (See your printer manual as they vary.)
If current= ...ON" then this program is set to work with the IBM Professional and Graphics printers and compatible (including Epson set to IBM compatible mode). (If your plot comes out with funny vertical separations in it then change this and try it again!)

"Graphic display to screen of timelines"
Reads the Sequencer memory into the P.C. by "stepping" through Sequencer memory. It then plots the timelines. This has it's own menus associated with it allowing the user to expand the plot or look at a particular section of the timelines in detail in a graphical manner. It only reads the Sequencer once and keeps reploting what it read until the user presses the "Escape" key or another function key. To make use of the "Increment.." feature set the " End x" to the same as what "Auto increment" is set to first and then press "I" for "Increment..". The x-axis maximum is clamped and there is auto wrapping logic in the code on the x-axis minimum. These clamps and wrapping are only valid on the "Increment.." feature so they may be manually over-ridden.

All 1113 buckets of all 40 timelines are always being plotted. If the scaling is changed during the plot it is possible to not see anything happening and not to see a plot. It may be necessary to strike a key (the spacebar is usually safe) to get the screen to refresh after changing one or more of the following:

"Quit" goes all the way out to the main menu.

"Print" tries to open "LPT1" as a file and then write an image of the screen to it.

"Top y" meaningful values here are 2-40. Can be used with below to select a range of timelines. Changing this will automatically adjust the number of y divisions to be the number of timelines you are plotting but you may over-ride this with the "Y divis." pick yourself.

"Bot y" meaningful values here are 1-39. Can be used with above to select a range of timelines. Changing this will automatically adjust the number of y divisions to be the number of timelines you are plotting but you may over-ride this with the "Y divis." pick yourself.

"Strt x"Determines which R.F. bucket (0-1112) will be lined up at the left plot axis.

"End x"Determines which R.F. bucket (0-1112) will be lined up at the right plot axis.

"X divis." How many x-axis divisions will be on the plot.

"Y divis." How many y-axis divisions will be on the plot.

"Grid" Turn on and off the dashed lines on the inside of the plot.

"Auto erase" Is probably not very useful here but it can be used to stop erasing the screen after the user changes one of these menu picks.

"Line data" Turns on and off connecting the buckets with lines. It can be hard to see a single pixel but one might only be interested in them. Note that x-axis plot for an EGA and VGA adapter is 512 x-axis pixels between the margins. It is possible to have this off and still look (mostly) like lines when packing 512 or more buckets into this space.

"Increment x-axis..." Adds the value displayed and set on the calling menu to both the "Strt x" and "End x" values to save time. This has some auto-wrap logic.

"List of output files."
Provides an easy way to see what files already exist. The user is prompted to edit the mask or if "Enter" is pressed the current mask will be used.

"File to act on."
This is the file name that this page will use to write to, or print from.

"Save (write) current timelines to file: ..."
Reads the Sequencer memory by "stepping" it as above into P.C. memory and then writes this information to the "File to act on."

WARNING:
THERE MUST BE ENOUGH AVAILABLE MEMORY IN YOUR MACHINE FOR THE FOLLOWING TO WORK. (If you are calling this program from another there may not be enough left for these to dynamically use the extra memory that they need.)

"Browse file"
Allows the user to look at the data without going to DOS. The "Browse" utility must be available in the current path for this to work.

"Print file"
Passes the name of the file to the DOS "print" queue buffer.
WARNING: this can have unwanted side effects!
If the DOS "print" queue buffer was not previously run or set up before this program was loaded then DOS sets up the "print" queue buffers in the memory after the memory this program is using. Then when this program is exited DOS will not recover the program memory because the "print" buffers were loaded after it. The only (normal) way to recover this memory is to reboot as DOS cannot recover fragmented memory.
(This problem may be avoided by allocating memory at boot time by placing the following two lines in your autoexec.bat file:
"mode LPT1:
print/d:LPT1"
The exact form of this may vary slightly from clone to clone so see the DOS "mode" and "print" commands for your machine.)
WARNING:
It is possible to exit or "Open a door to dos" and delete the file you sent to the printer before the it has finished printing. If this happens then only part of the file will be printed.
If the file is a longer file then the DOS "print" buffers and the printer's buffers then the file will be deleted before it gets to the printer.
(This is true unless you are using a more sophisticated "spooling" print queue utility than the standard simple DOS one.)

"Copy file..."
Uses the DOS "copy" command to copy the file to the DOS device "lpt1" which is normally the printer.
Use this instead of "Print" to avoid the memory buffer problems mentioned above. (A possible penalty here is that it may tie up the machine longer while the

file is being printed. This depends on the file length and the length of the printer's hardware buffer.)

"Open a door to DOS"
Passes whatever the user type's to DOS without leaving the program until the user hits the "Enter" key. Then it comes back into the program. "CT" is still using up considerable memory of DOS's precious 640k bytes so there may not be enough to do what you want.

```
    token ring selector fanout control or eXit, Quit or ESC
yellow=hot pick unless in a column menu then "enter" exits column menu.
arrow keys moves blue cursor & change vals.  reverse green=current pick.
Device name (file)=    test73a.sf      List devices (files)
Node # of vme=  $073A                  Open a door to dos
Module address= $10300000              Vme diagnostics
Address modifier= $39                  Time delay: 15ns
Help   Save   Restore   Erase   Browse (device) file    Global save/restore
1pick=rev    all stat        1pick=rev      1pick=rev        1pick&stat   1pick
P2:TL Router P3:TL Selector  P3:Busy Router P2:Busy Selector Clock Out    Outpu
------------ -------------- -------------- ---------------- ---------    -----
In0-3 to:    Out0-3 from:    BusyIn to:     BusyOut from:
0: TL00-03   Ch0 = 18         0:Busy0        0: Busy0          MCLK       direc
1: TL04-07   Ch1 = 20         1:Busy1        1: Busy1          PCLK       retim
2: TL08-11   Ch2 =  2         2:Busy2        2: Busy2        disabled
3: TL12-15   Ch3 =  3         3:Busy3        3: Busy3
4: TL16-19   Ch0 enabled      4:Busy4        4: Busy4
5: TL20-23   Ch1 disabled     5:disabled     5: disabled
6: TL Sel    Ch2 disabled     6:selector     6: router
7: disabled  Ch3 disabled     7:disabled     7: disabled
 register readings follow:  flashing display may indicate vme read error(s)
router: $FF  ch0 $12              busy sel/router: $F7    con. stat. reg.= $F7
             ch1 $34                                      local/remote= remote
             ch2 $22
             ch3 $23
```

F2 Sub-Menu: D0 Clock Selector Fanout Help Notes
11/5/90

This page no longer reads the last selected address and Downloads it automatically although the user is allowed to do this by pressing the "R" key for "Restore". The page reads the last selected "Device" (address file) to get the address and then reads the hardware and displays it's current settings. Nor are the last downloaded settings automatically saved to the file unless "auto save on exit is enabled" on the main menu and the user type 'y' or 'Y' to the question asking if he wants to over-write the existing file. Also the user may at any time manually over-write the existing file with the "Save" option below. The last device selected is stored in the file "SF_DEF.TXT" which is still automatically updated by the program.

This page allows the user to change which Selector Fanout Module (SFM) is being read by the program. While not in any of the column menus and a hot key is not pressed the program is constantly doing byte reads of the (byte) registers and comparing them to values it has last read. If they are different then the display flashes annoyingly.

Light gray (white) is information of how to interpret the display and helps divide the page into sections. The Capital letters in the top part are hot and cause a menu pick to be activated. If the user does not want to change the value of the menu pick, the "ESC" or "Enter" key should abort changing the value.

The information "1pick=rev" is a reminder that the item in reverse video (green) is the current reading from that column. "all stat" is short for "all status" meaning that everything in the column may be changed and is being read back. "1 pick&stat" means that there is both. Either MCLK or PCLK is selected and that "enabled" or "disabled" will be displayed below it. This section has not changed from the previous program.

The section under "register readings follow" is the raw hex reading from the byte registers that were previously on the top of the old program. Hopefully they have been moved to more roughly correspond to their decoded columns and be a little more meaningful.

"Device name (file)"
If the "Backspace" key is hit first then the string may be deleted and changed one character at a time; else if a normal ASCII character is struck the whole string will be deleted and the user may input a new string. The arrow keys are not implemented in editing this field.

A device name is really any legal DOS filespec which may include a disk and sub-directory as part of the filename, up to 79 characters is valid. See a DOS manual for further information on DOS filenames. When the user enters a new device name the program looks for that file.

If the file is found the program reads in the Node,

VME module base address (menu pick "M"), and the address modifier in the file. Using this information the program then continuously does byte reads of the (byte) registers and compares them to the values it has last read. If they are different then the display flashes annoyingly while not in a column menu or a hot key is not being processed.

If the file is not found the assumption is that the user wants to create, add, or make a new device! It goes through a series of questions to determine how you want to do this. The possibility exists to use the present settings of "N","M", and "A" or to change them interactively. If "Enter" is pressed the old value will prevail. ESC will get you out of the question tree. The files that are created are human readable and have comment strings to the right of them so that they may be easily manipulated through DOS: "copy", "rename", "delete", and/or edited with an editor program.

"Node # of vme"
This is a hex (byte) address of which D0 Control Crate the SFM is tied to through the Vertical Interconnect (VI).

If this is changed the "Device name" above is changed by the program to "UNKNOWN.SF". This mechanism is enforced by the program as a way of trying to prevent accidental corruption of a Device and it's address. There is no backward lookup mechanism for looking up a (file) name from an address.

"Module address"
This is the full 32 bit VME base address the SFM is set to. Please note that if one is using an address that is greater than 24 bits one must be using a Bit3 406 adapter and the menu pick "3irty-two bit add" must be set to "ON". When using the 24 bit addressing #403 Bit3 adapter the "3irty-two bit add" must be set to "OFF". If the #403 is being used the high byte of the 32 bit address will be ignored.

If this is changed the "Device name" above is changed by the program to "UNKNOWN.SF". This mechanism is enforced by the program as a way of trying to prevent accidental corruption of a Device and it's address. There is no backward lookup mechanism for looking up a (file) name from an address.

"Address modifier"
This is the address modifier that the Bit3 will use to talk to the VME crate. It is always in the file of data but is not used if this program is doing I/O over the Token Ring. That way only one set of Device files need be maintained no matter which interface is being used.

If this is changed the "Device name" above is changed by the program to "UNKNOWN.SF". This mechanism is enforced by the program as a way of trying to prevent accidental corruption of a Device and it's address. There is no backward lookup mechanism for looking up a (file) name from an address.

A word about address modifiers is in order here. This version has been modified so that it is always doing BYTE reads of the SFM byte registers so one needs to put in a valid BYTE VME address

modifier. Further if the program is being used over a Vertical Interconnect then one needs to change the high nibble to $0. For example if the hardware normally responds to $3D then when using the VI one would use $0D.

"List devices (files)"
Allows the user to do a directory of files on the disk. The current mask is displayed which the user may change by typing in a new one: if the "Backspace" key is hit first then the string may be deleted and changed one character at a time; else if a normal ASCII character is struck the whole string will be deleted and the user may input a new string. The arrow keys are not implemented in editing this field. When the user hits "Enter" then the program will shell out to DOS and do a directory if there is memory available in the PC.

"Open a door to dos"
Allows the user to shell out to DOS and do what ever they can with the remaining memory available including editing any files. To return to the program hit "Enter" with out typing in a string.

"Vme diagnostics"
Goes into another part of the original program that is a command interpreter. For help on this please refer to the end of the F1 help notes.

"Time delay"
Is the same thing it was in the original program.

"Help"
Displays this file one screen at a time.

"Save"
Reads the current values and writes them to the "Device name" file.

"Restore"
Reads the current "Device name" file and writes the values to hardware.

"Erase"
Deletes the current "Device name" file from the current sub-directory (or filespec). If this key is hit accidently the file may easily be recovered by immediately hitting the "Save" menu pick which will re-write the file with the current address program settings and current hardware readings.

"Browse"
Allows the user to look at a "Device" file. The format is a human readable ASCII hexadecimal number followed by a descriptive text string describing which byte register it is.

"Global save/restore"

Brings up a sub-menu which allows saving and restoring from a file containing a list of "Device" (file) names. This menu has it's own help file.

5.2 Control Columns

The control columns are the vehicle by which the user controls the Selector fanout. One can move from column to column only when the "cursor" is at the top of the columns. One moves from column to column via the right and left arrow keys. When you get over the column you would like to change hit either the down arrow key, or the return key. The "cursor" will move down to the currently selected function (in the TL Selector column, the cursor just moves to the first option). In general one moves the cursor up and down in the column (using the up and down arrow keys) until the cursor is over the desired selection, then hits enter to select the option. The exceptions are in the TL Selector column, and in the Clock Out column (see below).

5.2.1. TL Router

There are eight options in this column. The following table displays what gets written to the TL Router register when the user selects a particular option:

TL Router Selection	Value Written to TL Router Register
TL0-3	0
TL4-7	1
TL8-11	2
TL12-15	3
TL16-19	4
TL20-23	5
P3 I/O	6
nowhere	7

5.2.1. TL Selector

The TL Selector is the most unusual column in the Selector Fanout Control menu. The user goes down to the desired rows with the up and down arrow keys as usual, but uses the left and right arrow keys to select the desired state of each row. In the first four rows, the values range from 0 to 23. In the bottom four rows, the values are enabled and disabled. When the entire column is in the desired state, the user hits enter to write the data to the various TL Output Selector registers. The following table shows the actual data written to those registers for each possible selection:

TL Output Selection	Value Written to TL Selector Register
Ch = 0, enabled	0
Ch = 1, enabled	1
Ch = 2, enabled	2
Ch = 3, enabled	3
"	
"	

Ch = 23, enabled	23
Ch = 0, disabled	32
Ch = 1, disabled	33
Ch = 2, disabled	34
Ch = 3, disabled	35
"	
"	
Ch = 23, disabled	55

5.2.3. Busy Router

There are seven options in this column. The following table displays what gets written to bits 0-2 of the Busy Selector/Router register when the user selects a particular option:

Busy Router Selector	Value Written to bits 0-2 of Busy Selector/Router Register
Busy0	0
Busy1	1
Busy2	2
Busy3	3
Busy4	4
selector	6
nowhere	5

5.2.4. Busy Selector

There are seven options in this column. The following table displays what gets written to bits 4-6 of the Busy Selector/Router register when the user selects a particular option:

Busy Selector Selector	Value Written to bits 4-6 of Busy Selector/Router Register
Busy0	0
Busy1	1
Busy2	2
Busy3	3
Busy4	4
router	6
nowhere	5

5.2.5. Clock Out

Although there are only three rows in this column, there are four possible selections. The MCLK and PCLK options work like the others do, but when one moves down to the enabled/ disabled row, the left and right arrows keys are used to toggle between enabled and disabled. When the entire column is in the desired state, the user hits enter to write the data to bits 1 and 2 of the Control and Status Register. The following table shows the exact data written to bits 1 and 2 for each combination of options:

Value Written to bits 1 and 2 of

Clock Out Selection	Control and Status Register
MCLK and enabled	0
PCLK and enabled	1
MCLK and disabled	2
PCLK and disabled	3

5.2.6. Output

There are two options in this column. The following table displays what gets written to bit 0 of the Control and Status register when the user selects a particular option:

Output Selection	Value Written to bit 0 of Control and Status register
direct	0
retime	1

global save/restore for selector fanout modules

Help for this page
List files
Process file of sf device filenames. current: DEFAULT.LIS
Browse above file.
Save (read hardware and write to disk).
Restore (read disk and write to hardware).

Open a door to dos

eXit, Quit or ESC

D0 Clock Selector Fanout Global Save/Restore Help Notes
11/5/90

"Help for this page"
Displays this file one screen at a time.

"List files"
Allows the user to do a directory of files on the disk. The current mask is displayed which the user may change by typing in a new one: if the "Backspace" key is hit first then the string may be deleted and changed one character at a time; else if a normal ASCII character is struck the whole string will be deleted and the user may input a new string. The arrow keys are not implemented in editing this field. When the user hits "Enter" then the program will shell out to DOS and do a directory if there is memory available in the PC.
NOTE:
When making new "*.lis" files with your favorite editor the string "*sflist" must be the first line of the file. Also there should not be apparent blank lines including the last line. The end-of-file marker should be at the end of the list to avoid the disconcerting message: "device file: not found=not done". A blank line or null string probably contains a hidden carriage return (or other hidden ASCII characters). If the character is a carriage return (Enter key) then that gets filtered out by the PASCAL "readln" procedure and does not get echoed to the screen in the error message. Carriage returns and other hidden characters are not legal characters as part of a filename.

"Process file of sf device filenames."
Allows the user to change the file name to either Save to disk ("Upload") or Write from disk to hardware ("Download"). This file is a list of "Device name" (files) to be processed. The name may be changed similarly to "List devices" above. There is no provision built into the program to maintain these lists. It is expected that any normal (program) text editor be used to create these files such as DOS's "EDLIN".

"Browse above file."
Allows the user to look at the contents of the above filename.

"Save (read hardware and write to disk)."
Reads the "Process file.." device names and tries to open them. If it finds the file it reads the address, node, and address modifier, then the program reads the hardware and re-writes the device file with the latest readings.

"Restore (read disk and write to hardware)."
Reads the "Process file.." device names and tries to open each device file, read the contents and then writes to hardware the values in the device file.

"Open a door to dos"

If there is enough memory available this will allow the user to use any DOS commands including an editor to create, and modify new "*.lis" files.

token ring pcc control or eXit, Quit or ESC
yellow=hot pick

error bits: RF_MISSING SYNC_TIMING SYNC_MISSING

1 sync delay = 1 ns
2 mclk delay = 2 ns
3 mode = normal
4 set errors
5 clear errors

Address (pcc base add.) $10070000
Token ring node $073A
Bit three add. mod. $39

List files
File for current device TEST73A.PCC

Save to above file.
Restore from above file to hardware.
Open a door to dos.

Vme diagnostics

F3 Sub-Menu: D0 PCC Help Notes
12/16/90

This page is similar in function as the original Version 1.03 main menu pick three except that the user interface has been slightly changed and that some menu picks have different letters and have been re-located. Also a few more DOS functions have been added including saving and restoring to a user specified file.

This page no longer reads the last selected address and Downloads the Control Status Register automatically although the user is allowed to do this by pressing the "R" key for "Restore". The page still reads "DEFAULTS.PCC" to get the address and then reads the hardware and displays it's current settings. Nor are the last downloaded settings automatically saved to the file unless "auto save on exit is enabled" on the main menu and the user type 'y' or 'Y' to the question asking if he wants to over-write the existing file. Also the user may at any time manually over-write the existing file with the "Save" option below. The last device selected is stored in the file "SF_DEF.TXT" which is still automatically updated by the program.

"Address"
This is the full 32 bit VME base address the PCC card is set to. Please note that if one is using an address that is greater than 24 bits one must be using a Bit3 406 adapter and the menu pick "3irty-two bit add" must be set to "ON". When using the 24 bit addressing #403 Bit3 adapter the "3irty-two bit add" must be set to "OFF". If the #403 is being used the high byte of the 32 bit address will be ignored.

"Token ring node"
This is a hex (byte) address of which D0 Control Crate the PCC is tied to through the Vertical Interconnect (VI).

"Bit three add. mod."
This is the address modifier that the Bit3 will use to talk to the VME crate. It is always in the file of data but is not used if this program is doing I/O over the Token Ring. That way only one set of Device files need be maintained no matter which interface is being used.

A word about address modifiers is in order here. This version has been modified so that it is always doing BYTE reads of the SFM byte registers so one needs to put in a valid BYTE VME address modifier. Further if the program is being used over a Vertical Interconnect then one needs to change the high nibble to $0. For example if the hardware normally responds to $3D then when using the VI one would use $0D.

"List of files"
Allows the user to do a directory of files on the disk. The

current mask is displayed which the user may change by typing in a new one: if the "Backspace" key is hit first then the string may be deleted and changed one character at a time; else if a normal ASCII character is struck the whole string will be deleted and the user may input a new string. The arrow keys are not implemented in editing this field. When the user hits "Enter" then the program will shell out to DOS and do a directory if there is memory available in the PC.

"File for current device"
Chooses the filename to "Save", "Restore", or "Use" as below. If the "Backspace" key is hit first then the string may be deleted and changed one character at a time; else if a normal ASCII character is struck the whole string will be deleted and the user may input a new string. The arrow keys are not implemented in editing this field.
If the file is found then the program reads in the above file's addresses and filenames to read the PCC without changing the values in the PCC control registers from the file as does "Restore" in the menu pick below. If the file is not found a message informing the user that the no address values have been changed. The user is free to change any address or value and to save into the same file.

"Save"
Reads the current values and writes them to the "File for current device:" above.

"Restore"
Reads the current "File for current device:" above and writes the values to hardware.

"Open a door to dos"
Allows the user to shell out to DOS and do what ever they can with the remaining memory available including editing any files. To return to the program hit "Enter" with out typing in a string.

"VME diagnostics"
Goes into another part of the original program that is a command interpreter. For help on this please refer to the end of the F1 help notes.

```
      F8 menu:  peek/poke vme
Hex output toggle (on=hex. off=dec.)  current=  ..   OFF
Address  to poke.                     current:   8388607
Location to peek.                     current:   8388607
Copy poke address to peek address
Value to write.                       current:         0
Modifier (address) for vme i/o. current:       57

Peek word 1 time.
Write word 1 time.

Read continuously.
Forever write (until a key is pressed).

Both poke & peek 1 time.
Do poke & peek continuously

1 process file vme.dat
2 change above file to process.        current=  vme.dat
3 directory of files.               current mask=  *.dat
```

```
      F8 menu:  peek/poke vme
Hex output toggle (on=hex. off=dec.)  current=   ..   OFF
Address  to poke.                     current:   8388607
Location to peek.                     current:   8388607
Copy poke address to peek address
Value to write.                       current:         0

Node no. fadc vme crate in hex        current: $073A

Peek word 1 time.
Write word 1 time.

Read continuously.
Forever write (until a key is pressed).

Both poke & peek 1 time.
Do poke & peek continuously

1 process file vme.dat
2 change above file to process.         current=  vme.dat
3 directory of files.              current mask=  *.dat
```

"F8" Detailed Notes

"Hex output" controls whether this pages numeric values are displayed in Hex or Decimal. Especially useful with F2, F6, and F8 to have the data in the base of your preference.

"Address to poke "
Which VME address other menu picks this page that write to VME memory will act on.

"Location to peek"
Which VME address other menu picks this page that read from VME memory will act on.

NOTE:
When reading standard VME memory 16 bit words that represent 16 bit quantities as words, the bytes will be correctly displayed by this page else if the data at that address was byte data then the bytes will be swapped.
(The rest of the "ct" program is always swapping bytes to the hardware but displaying them normally to the user. Save/ restore files are this way except for the Sequencer timeline files. Generally the addressing of the clock hardware is backwards from standard VME hardware.)

"Copy poke address to peek address"
An easy way to make both addresses above the same.

"Value to write"
This changes the value that will be written out when "W", "C", "B", or "A" is pressed.

"Node no. fadc vme crate in hexadecimal..."
{ Token Ring Version only.}
Which VME control node is being addressed.

"Modifier (address) for vme i/o..."
{ Bit 3 versions only. }
This is the address modifier that the Bit3 will put on the VME bus when it peeks and pokes. Changing this changes it for the whole program! If the VME Bit3 board is physically in the FADC crate then the recommended value here is 61 decimal ($3D). If the VME Bit3 board is in a D0 control crate and addressing an FADC crate over the Vertical Interconnect then the recommended value is 13 decimal ($D).

"Peek word 1 time"
Reads the address entered in "Location address..." once and displays the result.

"Write word 1 time"

Outputs the word (unsigned 16 bit) displayed in "Value to write" to the VME "Address to poke...".

"Read continuously"
Same as "Peek word 1 time" but does it until the user stops the process.

"Continuous write"
Same as "Write word 1 time" but does it until the user stops the process.

"Both poke & peek 1 time"
Does a write followed by a read cycle to the VME via the Bit3 using the addresses and value specified above.

"Alternately poke & peek continuously"
Does a write followed by a read cycle to the VME via the Bit3 using the addresses and value specified above until the user stops the process.

'1 process file 'xxx.yyy
Is a way for the user to generate and send an arbitrary list of word reads and writes into VME address space from the file "xxx.yyy" in the default directory of the default disk that "pcplot" is currently using. The first line of the file should begin with the string "*list". The second line is ignored by the program but useful as a column header reminder for humans. The columns in order are the VME address, the value to write, the VME address modifier, write or read, a line comment field. A space must be between the fields. In the fourth column a "1" writes to VME and anything else, "0" will read and display the value read to the screen and the address it read it from. If there is a lot of reading going on the user may toggle on and off the printer with control-Print Screen.
An actual extract of the file is from one used to test a VBD:

```
*list
 add val addmod '1'=write  comment_string
 0 $9C $29 1 ! reset and write protect
 0 $C $29 1  ! set write protect of the VBD off
 8 $2 $29 1  ! crate type = Central det.
 $A $8008 $29 1 ! low 16 bit address of event counter
 .
 .
 .
 $486820 $1 $3D 0 !read of Z.S. chan length register slot 13 chan 0
 $486800 $1 $3D 1 !write pattern to slot 13 chan 0
 .
 .
 .
```

'2 change above file to process.'
Allows the user to specify the name of the file to process. The first line of the file should = "*list". See above.

'3 directory of list files.'
Allows the user to see a list of files on the disk using the mask. The user is allowed to change the mask interactively after selecting this pick or use the default mask by hitting the "Enter" key.