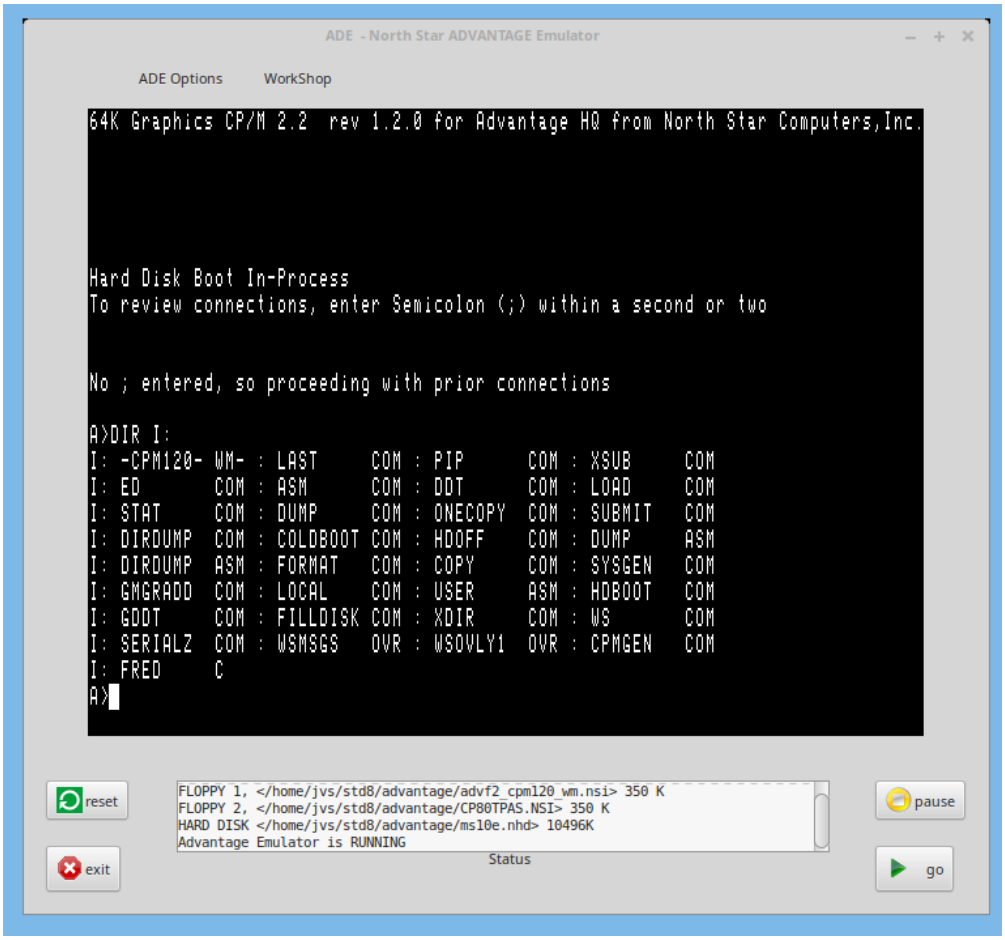
ADE

North Star ADVANTAGE Z80 Computer Emulator
User's Guide



Version 0.67 20th March 2021

This Page is Intentionally Blank

CONTENTS

ADE APPENDICES

1. INTRODUCTORY INFORMATION

1.1 Overview

ADE emulates the early 1980s North Star Advantage Z80 Computer.

ADE uses disk-image files which may contain any of the North Star Computers' Disk Operating Systems of the period which were designed for the graphical hardware of the North Star Advantage: North Star GDOS, CP/M, etc.

The Advantage used a double-density Floppy-Disk Controller which could access both sides of the disk and used 512-byte sectors giving 350K of storage. The Advantage originally had space for two floppy-disk drives. Later, hard-disk capability was added to the North Star Advantage by replacing one of the floppy-disk drives with a 5-inch hard drive. The hard drive was available in several capacities from 5 megabytes up to 30 megabytes.

ADE contains 128K of RAM, a Z80 microprocessor emulator, and a keyboard-handler. Other parts of the emulator are concerned with the North Star Advantage-specific components, such as the data and control ports of the serial and parallel I/O, the double-density floppy-disk controller with its boot-up PROM, the fixed-disk controllers and the bit-mapped display screen.

There are two drop-down menus, one of which emulates the operator's interaction with the hardware, such as inserting or removing floppy disks, and organizing the interaction between the host linux machine and the virtualised Advantage hardware. The other smaller menu looks behind the scenes at the internal virtual machine’s RAM contents.

For debugging or recording of the emulator’s operation, there is the ‘screenlog’ of the emulator’s screen output, and ‘xlog’ which contains the desired debugging information.

There are also a small status information display under the main screen-window, and two buttons on each side of the status display. The two buttons on the right make the virtual-machine stop and go. The two buttons, slightly out-of-the-way on the left are to reset the virtual-machine’s hardware, and to exit from the emulator.

1.2 ATTRIBUTIONS FOR OTHERS' CODE IN ADE

ADE's Z80 emulation code pretty much comes from yaze, a CP/M emulator written by Frank Cringle. North Star-specific amendments such as memory-mapped floppy-disk I/O and a few other additions such as Mode 2 interrupt code were made by Jack Strangio.

ADE's Z80 disassembly code comes from Marat Fayzullin's 1999 DAsm code with some local alterations.

The rest of ADE cannot be blamed on anyone else but myself.

Jack Strangio, February 2020

1.3 THANKS

I have the greatest appreciation for all those who have helped me in my rather idiosyncratic quest to write emulators of the North Star Computers' Horizon and Advantage hardware.

The North Star Horizon was my first computer way back in 1978. In those days, you could buy the Horizon ready-built for $2100 or you could save yourself $300 by taking a huge kit-box of parts and electronic components and assembling it yourself. That task took me more than 40 or 50 hours to complete over the course of several weeks in late 1978. The thousands of solder-joints literally burnt-out a new soldering iron. It says a lot for the quality of the instruction manual that most of the time I really had no idea what each step did but at the end (once my half-dozen wiring mistakes were fixed) I had assembled a computer which worked perfectly.

I'd like to mention a few of the people who have generously helped me:

Dave Dunfield, who gave me a lot of help in many different areas. He also had quite a few North Star floppy disk-image files that I have rummaged though over the years.

The Late Don Maslin, who got me started on the double-density floppy work by transferring a lot of data from my old 10-sectored disks to disk-image files.

Martin Brown, who helped me along the way with scanning old Disk-Controller manuals, without which I was more clueless than usual.

Howard Harte , whose regard for old computers means he has taken the trouble to maintain lots of North Star Manuals:

http://www.hartetechnologies.com/manuals/Northstar/

Bitsavers.org.

http://www.bitsavers.org/bits/NorthStar/

Thanks to them, there are still quite a few disk-image files around for the North Star Horizon.

Allison Parent, for indicating where I could get hold of information regarding the HD5X controller board.

Jon Hales, who has prodded me along when my enthusiasm for digging amongst the Advantage's documentation was flagging badly, and who was very generous with his time and extra Advantage software files.

Simon Coombs, who was able to supply me with a clean copy of the Advantage GDOS 2.0.0 DOSBASI disk along with the accompanying GHDOS Recovery disk for reinstalling the GHDOS 2.0.0. HD Utilities to a reformatted hard drive.

Holger Linning, who helped me along with many hardware items and also many ADVANTAGE disk images.

1.4 FLOPPY DISKS AND HARD DISKS SUPPLIED WITH ADE

Several floppy disks are supplied with ADE to get you up and going quickly. They are stored in the 'disks' subdirectory. (/home/USERNAME/advantage/disks/) Generally speaking, the floppy-images supplied with ADE are from true North Star Advantage floppies. We emulator nerds take pride in using either the original software disks themselves, or disk-images which are exact copies of those. A full list and description of the disks supplied is in the 'AAA_DISKS_README.TXT' file in that same directory.

Note that I have supplied two CP/M version 1.2.0 floppy images, CPMBASIC_120.NSI and CPMBASIC_120A1. These are practically identical, but the CPMBASIC_120A1 has been patched so that the A: drive will be searched whenever CP/M can't find a .COM file on the current CP/M drive. In other words, all of your .COM files can be placed on the A: drive and you won't need any copies of a .COM file on any other disk.

Unfortunately, I only have one sample of Advantage's GDOS/GHDOS and that seems to be a copy of a very early release, DOSGBASIC_200. At one point when it boots up GHDOS, it checks the version of the hard-disk and expects it to be 'version 1.0'. As far as I know there was only one hard drive supplied by North Star which was classed as version 1.0, and that was the original SG5A (5 megabytes). All other ensuing North Star hard drives are classed as 'version 2.0'.

If you manage to find a copy of the GHDOS disk Release 2.1.0, you can use all of the larger hard drives with that.

Consequently, I was forced to restrict the use of the DOSGBASIC_200.NSI (ghdos 2.0.0) boot floppy to working with a version 1.0 five megabyte hard drive, supplied as ADV_SG5A.NHD in the 'disks' subdirectory. It has been set up to have several 1-megabyte CP/M virtual disks, ranging from CP/M A: to CP/M E:, and provision for the two floppy CP/M drives, I: and J: which are used for transferring files to and from the larger virtual disks.

When ADE is booted with the CP/M Boot Disk, it will take you to the A: directory on the hard-disk on startup.

Larger hard drives are more useful in practice, of course, and you can make these easily with the 'mkhd' utility which is located in your 'advantage' top directory or in your ~/.local/bin $PATH. But the small hard drive image-file supplied should give you a good feel for the emulator to start with. There is another sample hard drive image on the website which is 30 megabytes. That one, MS30D-30MB.NHD, consists of 15 virtual CP/M drives, each one being about 2 megabytes.

****

Because of the 'HD version 1.0' restriction, restrict the accessing of the larger hard drives to CP/M ONLY, and not the GHDOS release 2.0.0 supplied, otherwise your hard-drive disk-image will be corrupted.

****

1.5 LIST OF SCREEN CAPTURES ON FOLLOWING PAGES.

ADE looks like a typical "green-screen" terminal of the 70's-80's period. Other possible colors for the screen output include white, amber and yellow.

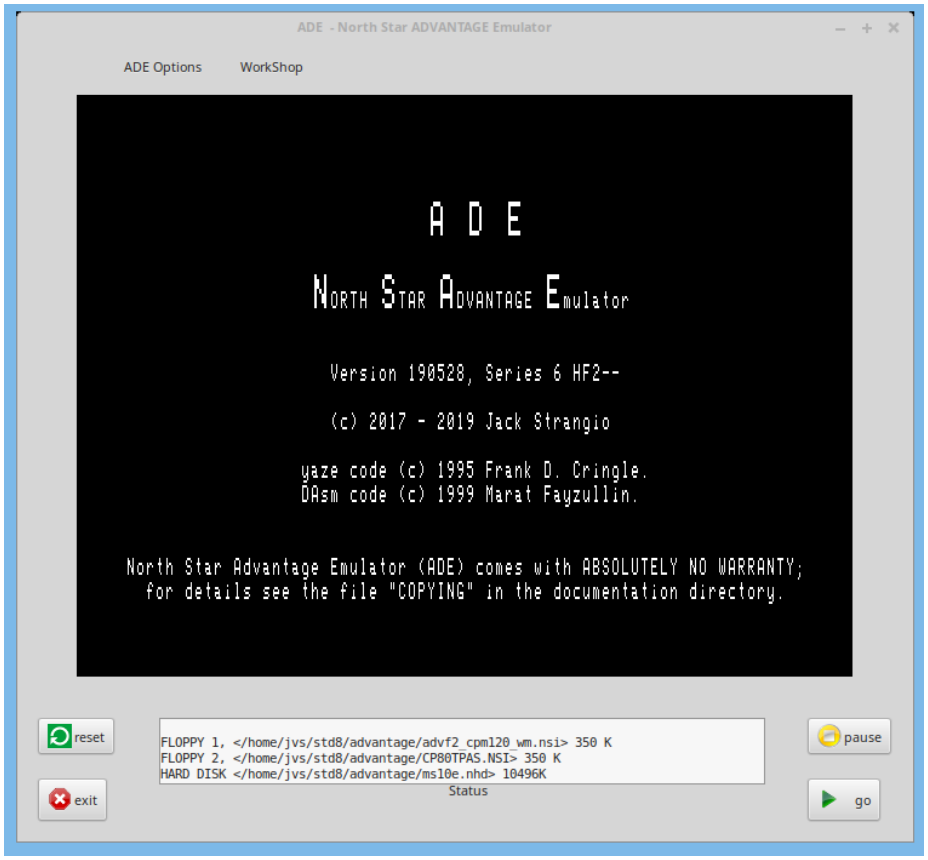


Fig. 1. Initial ADE Splash Screen. Z80 Emulator not yet running. White screen output.

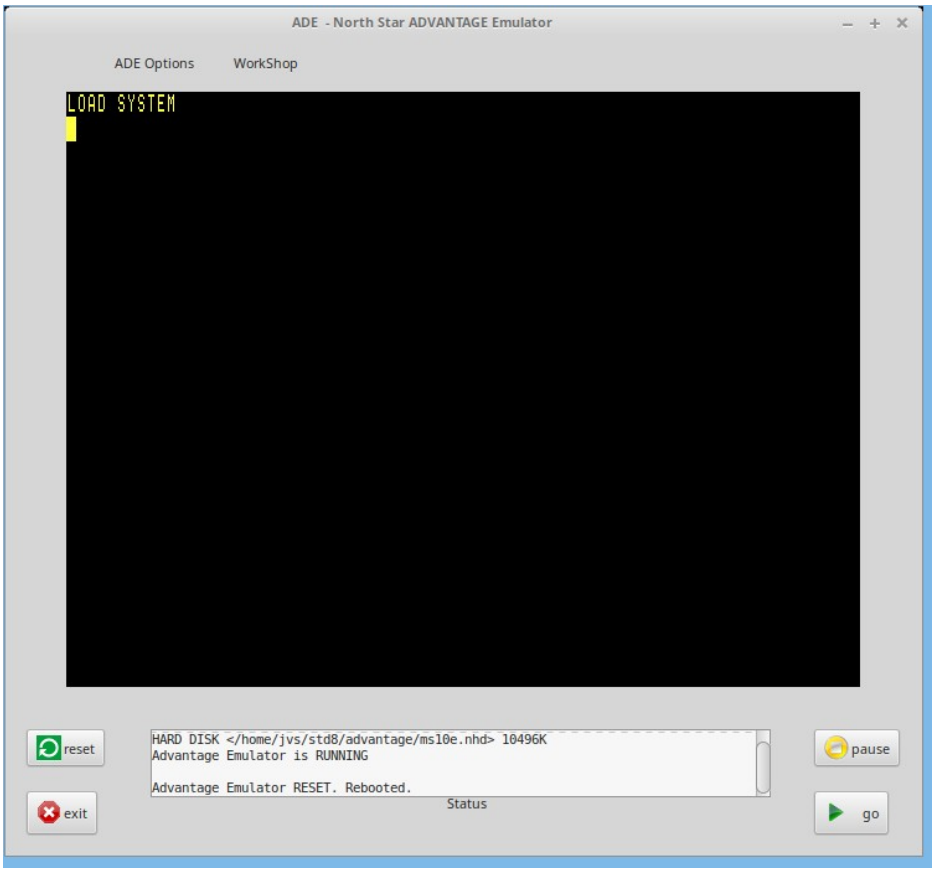


Fig. 2. After hitting ‘go’ button. “LOAD SYSTEM” screen. Z80 Emulator running. Hit ‘enter’ key to boot from floppy disk. Yellow screen output.

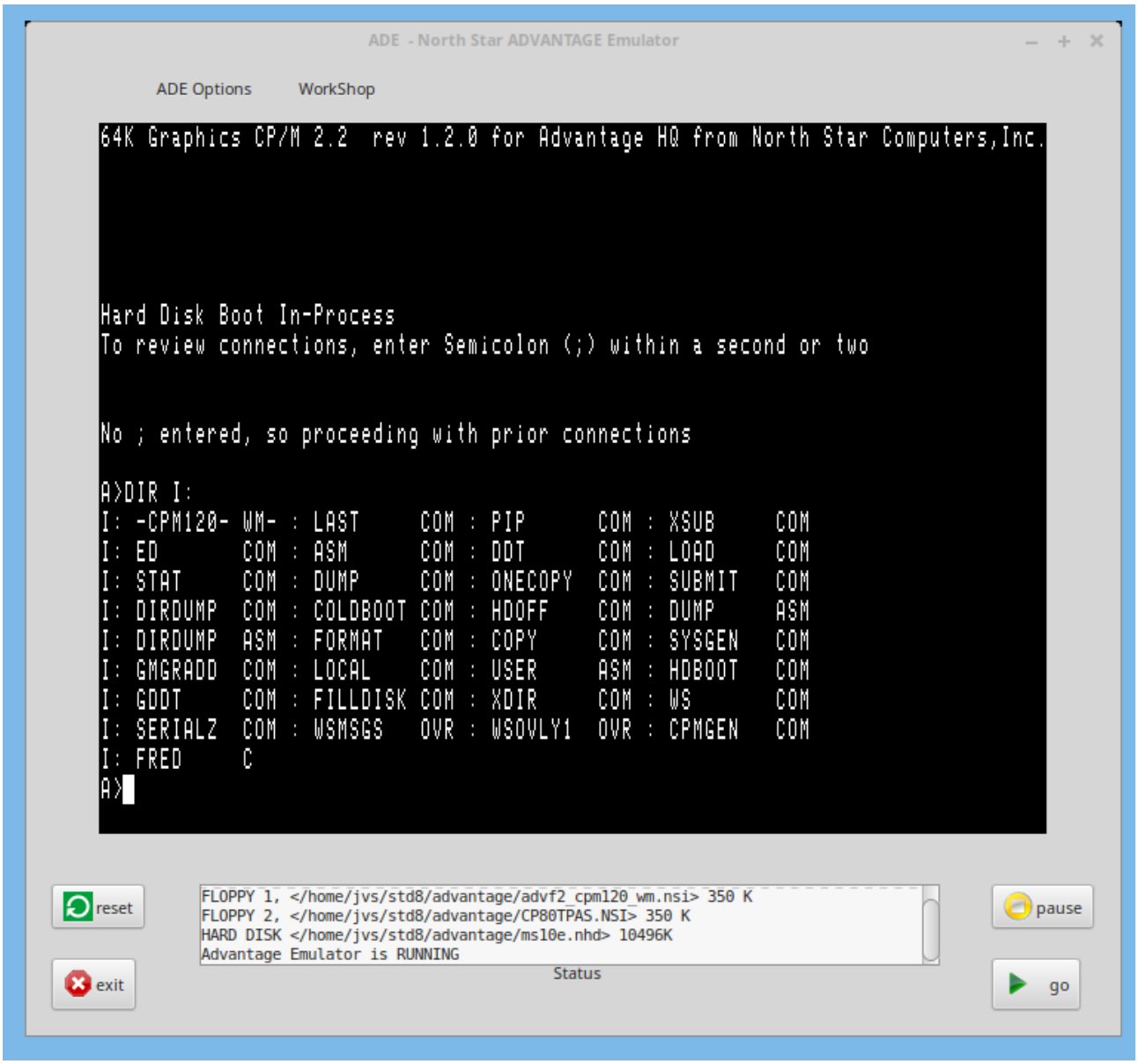


Fig. 3. CP/M Banner including Hard Disk Boot. Followed by a directory listing of the boot disk.

CP/M A: always belongs on the hard drive. Note the screen output is White in this image, the default is normally Green, with Amber and Yellow as alternative colors available with a recompile.

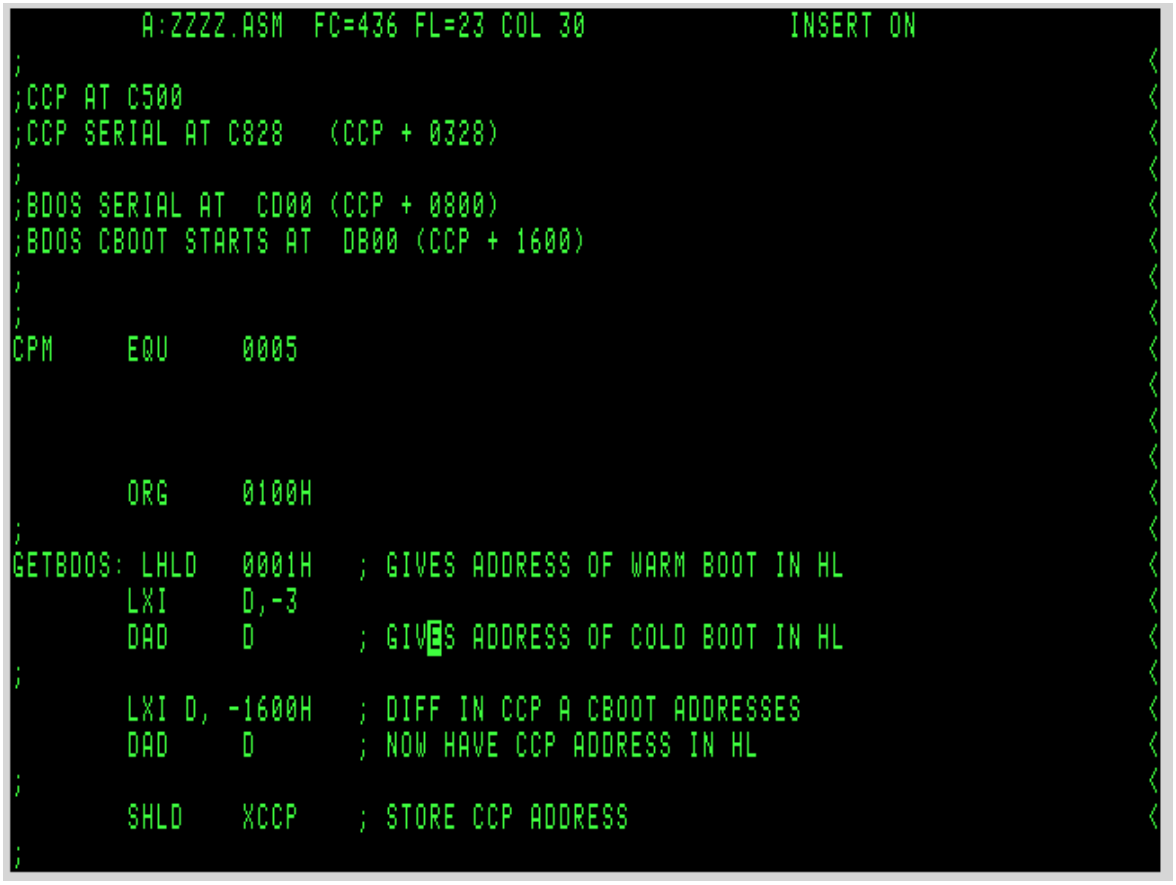


Fig 4. A version of Word Star running. Green screen output.

Fig. 5. DEMODIAG 1. Screen shots from the ‘DEMODIAG.NSI’ boot floppy.

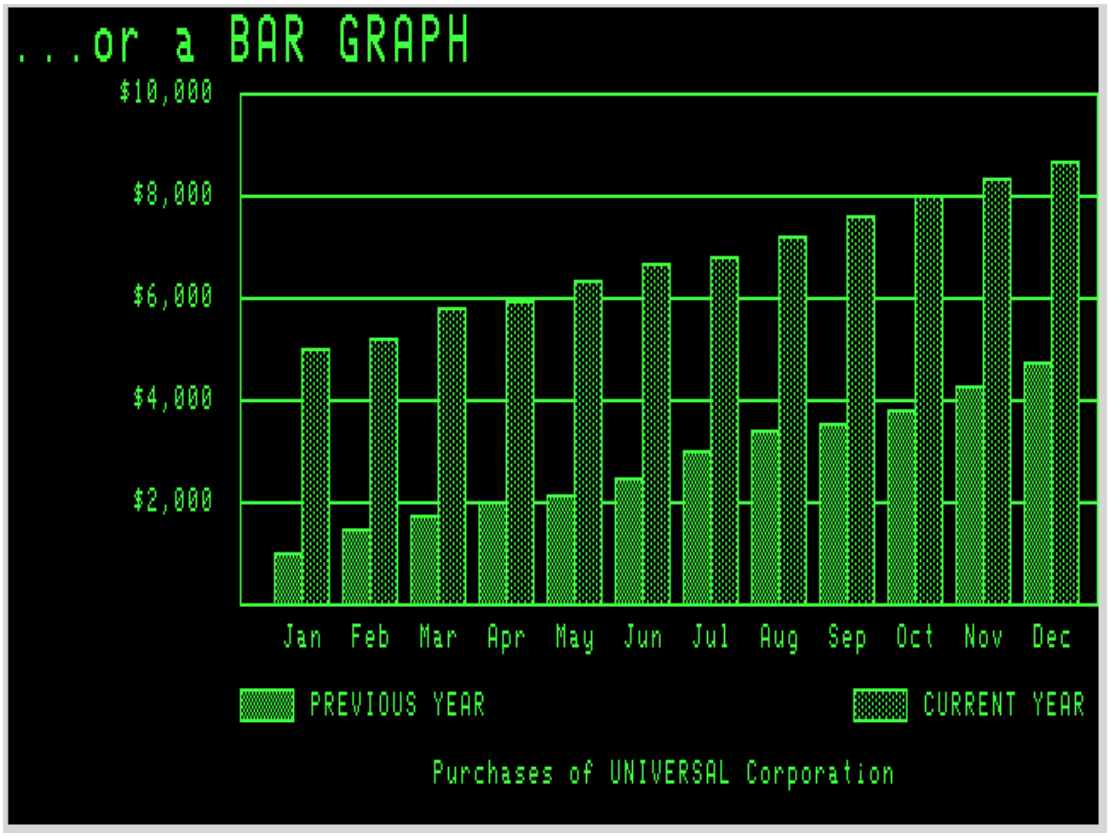


Fig. 6. DEMODIAG 2.

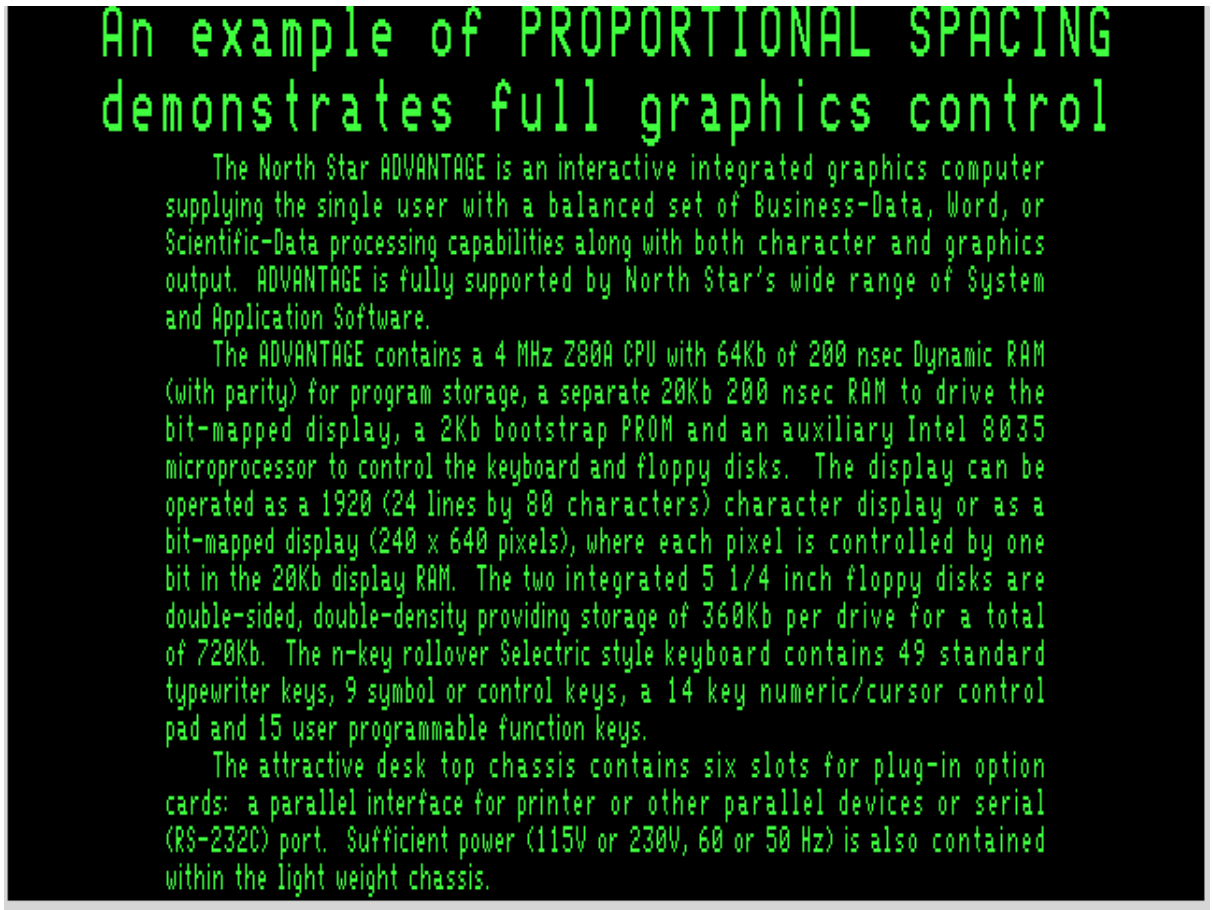


Fig. 7. DEMODIAG 3.

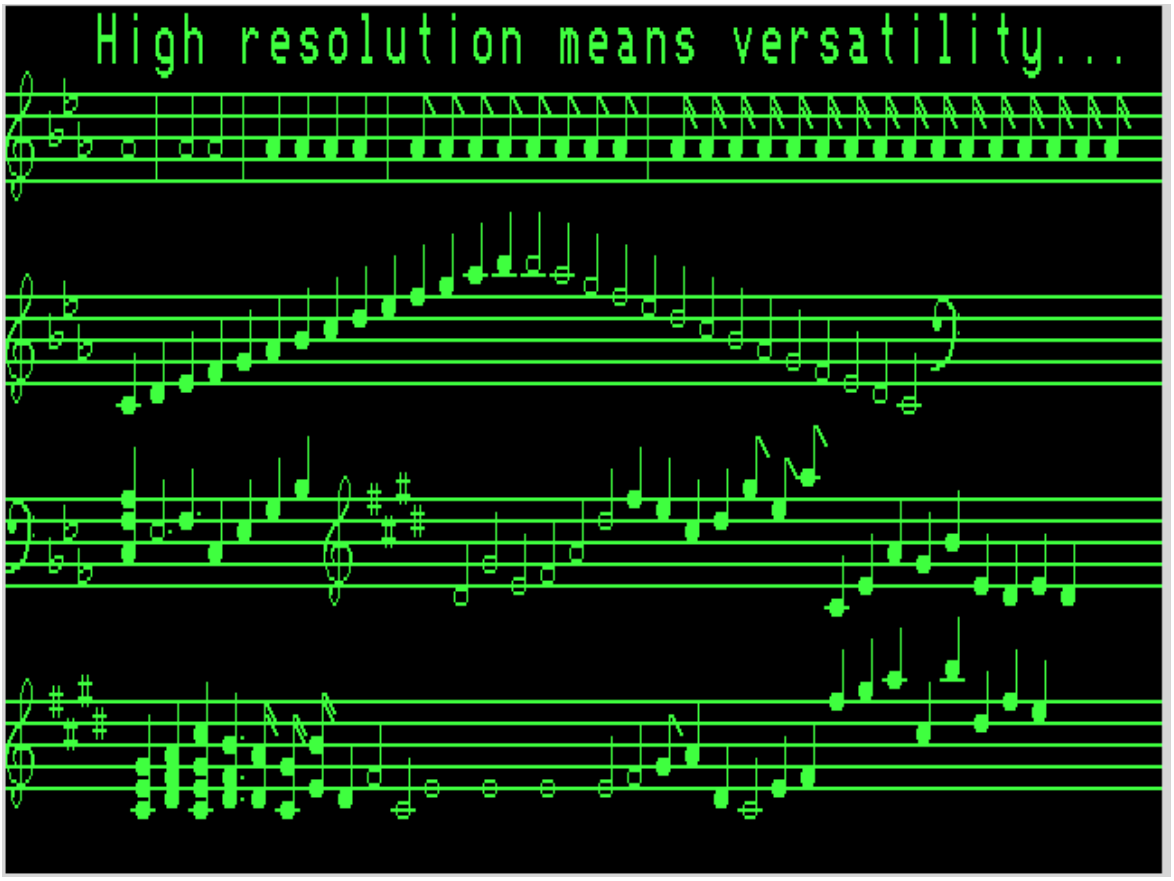


Fig. 8. DEMODIAG 4.

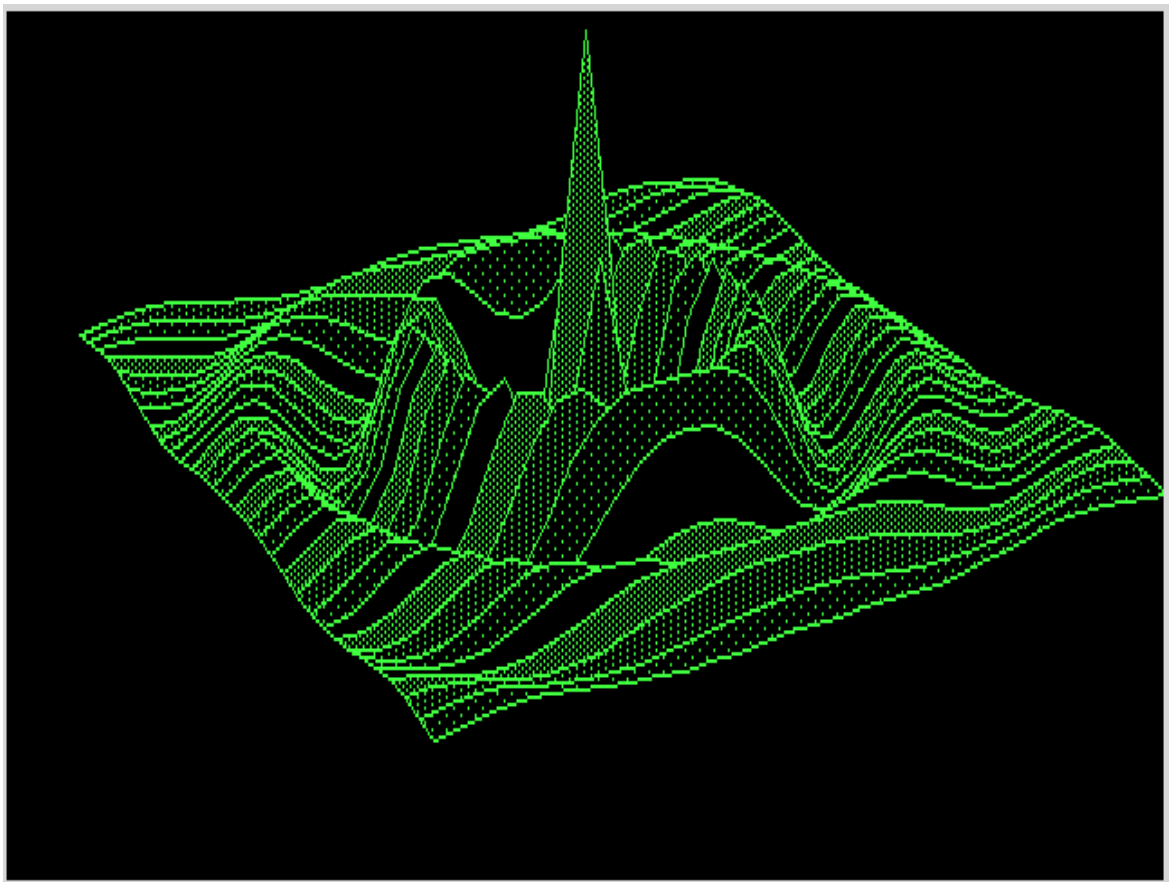


Fig. 9. DEMODIAG 5.

Fig. 10. DEMODIAG 6. Diagnostics - Floppy Test

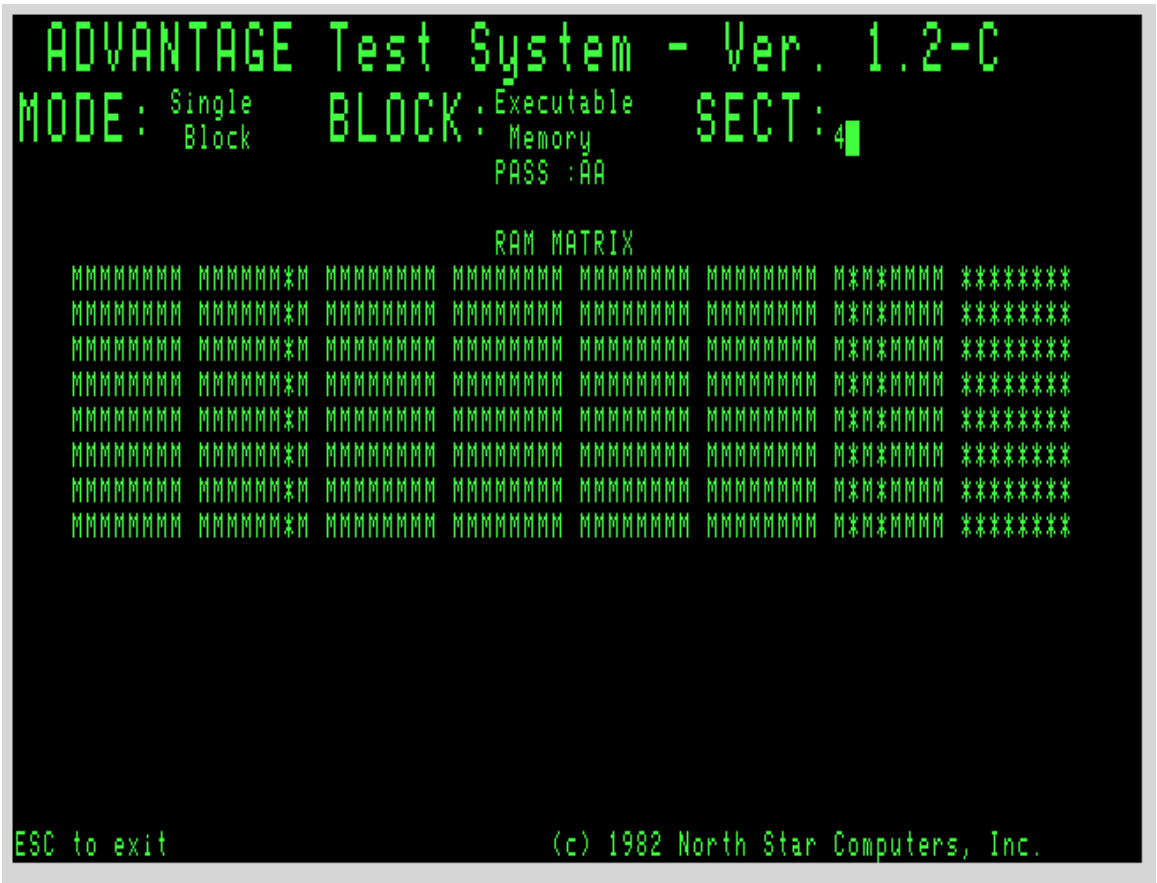


Fig. 11. DEMODIAG 7. Diagnostics - RAM Test

2. Obtaining and Building ‘ADE’

2.1 Linux Libraries required

Very few Linux libraries are required, apart from the standard packages installed on most Linux Distros.

The GUI Toolkit used is GTK+ Version 3, apart from a few deprecated functions from GTK+ Version 2.

This Toolkit can be installed using your Package Manager. If you’re using one of the Debian derivatives such as Debian itself, Mint, or Ubuntu, this can be done by installing **libgtk-3-dev** using Synaptic or even just

sudo apt install libgtk-3-dev

from the command line.

2.2 Get the source files

Download the ADE source code from http://itelsoft.com.au/code/ade_latest_6.tar.gz and move it to any convenient work directory. Untar and decompress the tarfile:

tar xvfz ade_latest_6.tar.gz

This will produce a subdirectory called **ade**. Move there.

cd ade

Compilation should be initiated with a simple **make** on the command-line**.**

If all goes well and the compile completes successfully, install the ade package with

make install

This will install the package in the **/home/***USERNAME***/advantage** work directory. So user ‘fred’ will find a directory called **/home/fred/advantage**.

A launcher icon will appear on the USERNAME’s Desktop. Clicking on that should launch the emulator. It can be ‘Drag n Dropped’ to the Desktop Panel. Alternatively, ADE can be invoked from the command-line if the **ade** executable file is to be found somewhere within your $PATH list.

ade

2.3 What’s in the /home/username/advantage work-directory?

The /home/username/advantage directory has several important files:

ade.conf the configuration file for ade which holds most of your personal preferences.

designates which disk image-files are mounted.
specifies which hardware cards will be inserted into which peripheral slots.
specifies what I/O files will be attached to the Advantage I/O ports.
preferred settings for capslock, hard-drive ‘speed’.
preferred ade-development settings.

Avoid editing the ade.conf file manually. It gets updated automatically every time you make different choices on the Options and WorkShop menus, and will hold those settings indefinitely over more than one session.

pio_out destination of text from the parallel-out port: the ‘LST:’ device in CP/M

sio_out destination of text from the serial-out port: the ‘PUN:’ device in CP/M

mkhd CLI utility for making hard Drive image files. See section 5.1, page 31.

2.4 Starting up ADE

Starting ADE can be done from the Desktop with the emulator's icon or from the command-line. On start-up, the program will show the title (splash screen) and will then wait for user input. Usually, the user will then just hit the 'go' button because the installation process also provides the default configuration file, **ade.conf**, which will be found in the ADE top directory, **/home/username/advantage**

ade.conf contains the default settings which are expected by the North Star ADVANTAGE computer:

Hardware 'slot' 1 contains a PIO card, which sends data to the parallel-out CP/M LST: device
Hardware 'slot' 2 contains an SIO card, which sends data to the serial-out CP/M PUN: device
Hardware 'slot' 6 contains a Hard Drive Controller card to enable use of a hard drive.

A boot floppy is in 'floppy drive' 1 at the minimum. Note that the system can also be booted from a floppy in Drive 2 and selected at the 'LOAD SYSTEM' prompt. See the 'North Star Advantage User Guide' for details.

Several ADE settings are also stored in the ade.conf file. Such as Capslock ON/OFF, and whether the hard drives runs FAST or SLOW. The **ade.conf** file should not be edited manually. While that can actually be done, any changes you make may not be permanent.

If for some reason, the default configuration is not present in the top directory, then a new configuration file needs to be made. This is simply done by providing the user's settings with the 'ADE Options' menu, and/or the WorkShop menu. See Section 3, page 18. Any time a setting is altered with these two menu-bars, the new setting is saved automatically into the **/home/USERNAME/advantage/ade.conf** file.

'ADE Options' menu: Things to be changed by the everyday user.

'WorkShop' menu: Settings for use during ADE development. Most users won't need to bother with these.

A sample ade.conf file:

```
##### Configuration File for North Star ADE Emulator (c) 190624
#####
##### Avoid Editing This File Manually. Any Changes You Make Can
##### Be Automatically Overwritten at Any Time.

hdd             /home/jvs/advantage/disks/ADV_SG5A.NHD
fd1             /home/jvs/advantage/disks/F2_CPM120_WM.NSI
fd2
disk_dir        /home/jvs/advantage/disks/
hd_delay        off
capslock        on
slot_hdc        6
slot_sio        2
slot_pio        1
sio_in
sio_out         /home/jvs/advantage/sio_out
pio_in
pio_out         /home/jvs/advantage/pio_out
=======
log             /home/jvs/advantage/xlog
screenlog       /home/jvs/advantage/screenlog
debug_level     0000
break_addr      0000
break_on        off
trap_addr       FFFE
trap_on         off
```

The configuration is divided into two parts. The upper section is concerned with user preferences. The lower section under the '======' separator is concerned with ADE development, and as such is unlikely to be of interest or changed by the majority of users.

2.5 Running North Star Advantage CP/M. The ‘go button.

Now hit the ‘go’ button. The screen will clear with a beep, followed almost immediately by another beep and a blank black screen with the words ‘LOAD SYSTEM’ at the top. These words come from the ADVANTAGE boot PROM. In almost every case, all that’s needed now is to hit ‘Enter’ which will boot CP/M from the floppy in drive 1 and then load CP/M Drive A: from the hard drive.

2.6. Pausing the Emulator. The ‘pause’ button.

In most cases you won’t need the pause button unless things happen to move too fast for you, for instance to change floppies before the software moves on. Otherwise, using the Emulator is just like using a normal computer.

2.7 Rebooting/Resetting the Computer. The ‘reset’ button.

Just like the real thing, a reset will wipe the screen and reboot the emulator from scratch. You will be immediately taken to the ‘LOAD SYSTEM’ blank boot-up page. Use this button sparingly, your work may be lost.

2.8. Finishing Up. The ‘exit’ button.

Pack it up and put it away. The ADE program closes down and the emulator window is closed. Settings in ade.conf will remain for next session.

2.9 The ‘Status’ window

In between the two pairs of buttons, left and right, is a small window which displays short one-line messages. This is used to show information or warnings regarding the progress of the emulator. A short beep may be heard when some messages are shown. Examples:

Advantage Emulator is RUNNING
Capslock is now ON
HARD DISK </home/fred/advantage/disks/ADV_SG5A.NHD> 4896K
Advantage Emulator RESET. Rebooted.
New Floppy “/home/fred/advantage/disks/newflop.nsi” Created

3.0 ADE Options Menu

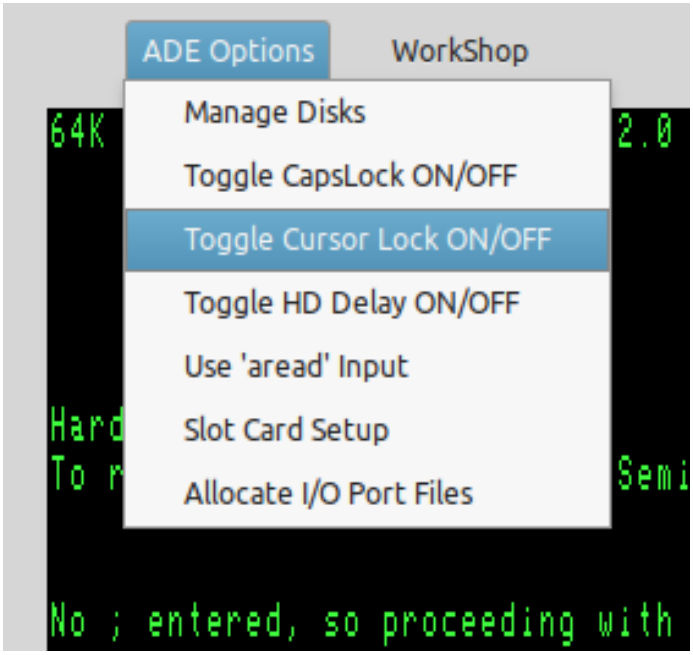


Fig. 12 ADE Options menu

3.1 Disk Management.

The Disk Management menu item allows the user to ‘eject’ floppies and hard drives from the Emulator. The first displayed window shows what floppy disks happen to be ‘inserted’ in Floppy 1 or in Floppy 2. It also shows which hard drive was installed when the Emulator was booted.

Each disk-drive has two buttons: a ‘Change’ button which will install a different floppy-image or hard-disk image. And an ‘Eject’ button which removes any image-file which was previously installed.

If the ‘Change’ button is hit, a file-chooser dialog window opens and allows the user to browse through the whole file-system looking for a floppy-disk image to install. Once the file is selected, hit the ‘Select’ button to confirm your choice. The file-chooser window will close, the floppy-image is ‘inserted’ into the selected floppy-drive and is then ready for use.

The directory which the floppy-image came from will be used as the default disk directory in future disk-image searches. For this reason it is handy to store all your North Star Advantage floppy and hard-drive image-files in one or two directories.

A fourth option in the Disk Management window will enable the creation of a new floppy-disk image. That new floppy-image can then be ‘inserted’ into Floppy1 or into Floppy 2 using the ‘change’ option as above. It is recommended that floppy images have the file extension of ‘.nsi’ or ‘NSI’

If the new floppy-disk’s name is not recognised to be an absolute filename (Absolute filenames start with a ‘/’, as with a filename like **/tmp/mynewfloppy.nsi**) it will be recognised as relative filename, and created relative to the default disk directory. For instance if the default disk directory is **/home/fred/advantage/disks** and the newly created floppy’s name is entered as **mynewfloppy.nsi**, the full absolute filename created would work out to be **/home/fred/advantage/disks/mynewfloppy.nsi**

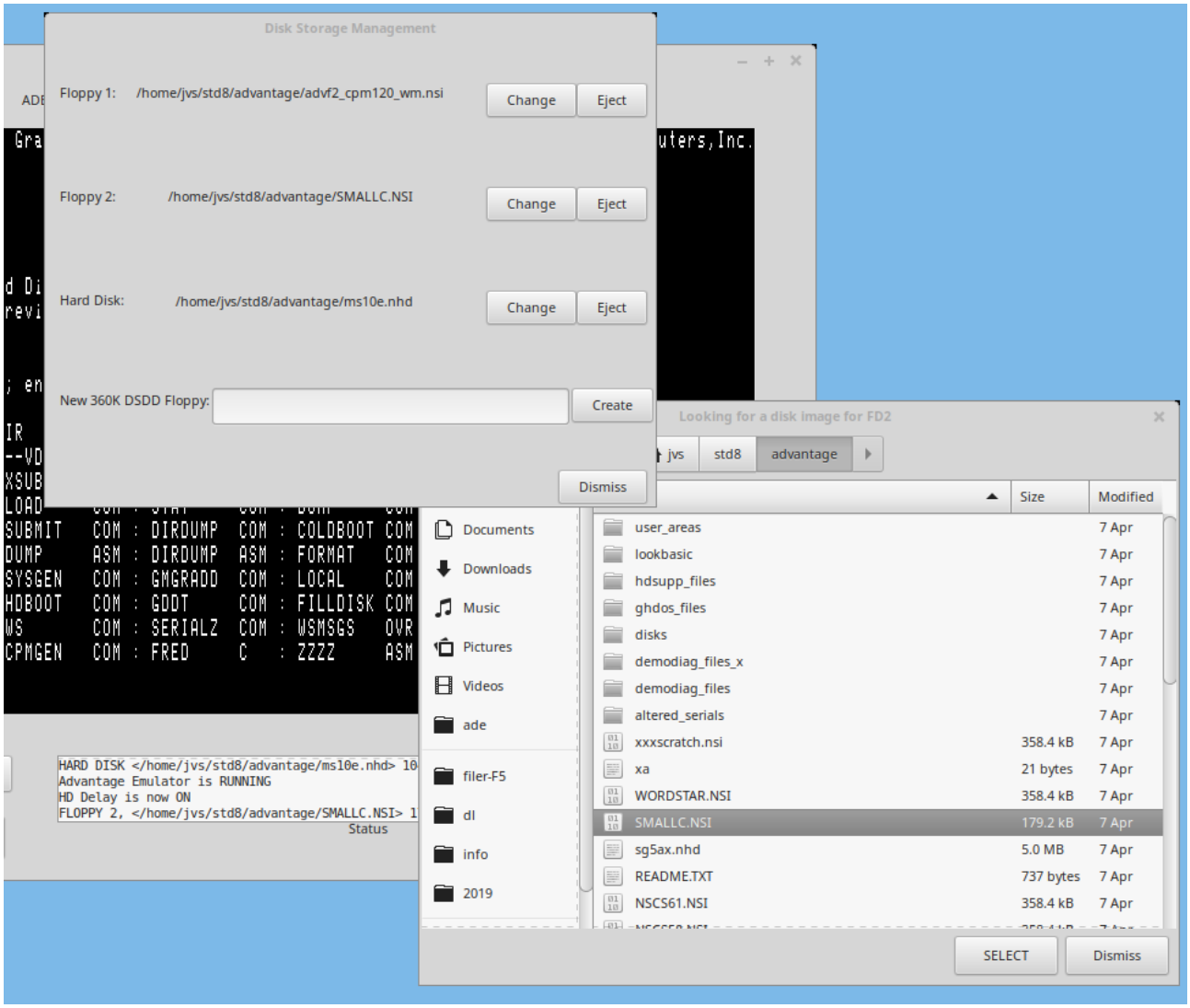


Fig. 13 Disk Selection Pop-Ups

Technical Notes on ADE Disk Configuration

As supplied, the ADE emulation will produce a ‘North Star Advantage’ which contains 2 floppy drives, 1 hard drive, 1 SIO card, and 1 PIO card. This was not really the standard configuration, as we have taken up the option of having two floppies as well as a hard drive. The original Advantage did not have the physical space inside the cabinet for doing that.

There is a side-effect to having both two floppies and a hard drive as well. The DEMODIAG test disk looks at whether it can detect a hard drive installed. If so, it insists that there can only be one floppy drive, and refuses to accept that there are two floppies. The program can be spoofed by temporarily uninstalling the hard drive by removing the hard drive ‘card’ from slot 6 (See section 3.5, page 23) in which case it it will happily test both floppies at the same time. Put it back in slot 6 after the floppy test.

Technical Note: The North Star floppy controller has the capacity to control four floppy drives. However, in a short-sighted decision when planning the Advantage and its hardware, the two drive-select bits in the Drive Control Register, which could have been used to designate 4 separate drives, can only designate two floppies, depending on whether bit 0 (floppy unit 1) or bit 1 (floppy unit 2) is high. Apart from that, Bits 2 and 3 are reserved bits, and also could have been used to designate an extra two floppies in a continuation of the single-bit drive-select rule. But that never happened.

3.2 Toggle HD Delay ON/OFF

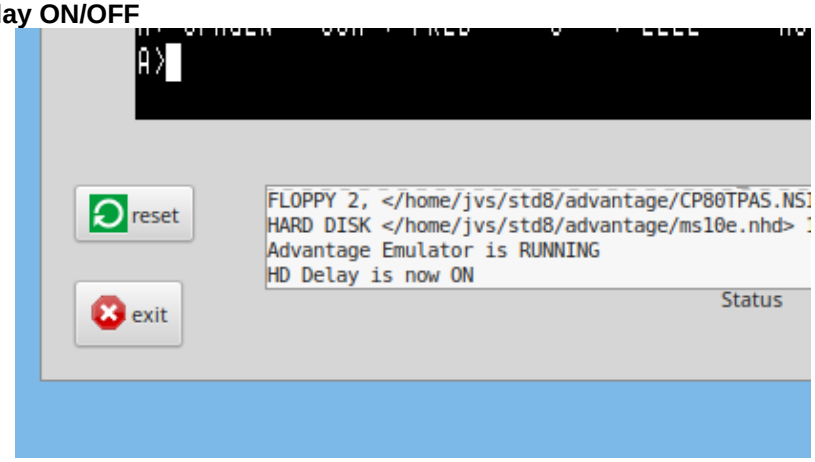


Fig 14. Status Window shows Hard Drive Delay Toggled ON (= SLOW).

ADE's 20 x speed emulation of the floppy disk drives and the hard disk drives is extremely fast! And the emulated 'correct speed' is still about 5 times faster than the REAL HD 'correct speed'. So the hard drive is deliberately slowed down even more so that the period allowed for entering a ';' to enable editing of the CP/M hard drive configuration is increased from about half a second to about 3 seconds. It is suggested that when you need to adjust the CP/M virtual-disk configuration, that the 'normal/slow' speed (HD Delay ON) is toggled on at the 'LOAD SYSTEM' screen, then the 'fast' speed (HD Delay OFF) is toggled back on, and left on, at the 'A> ' prompt. The default speed is 'Fast'.

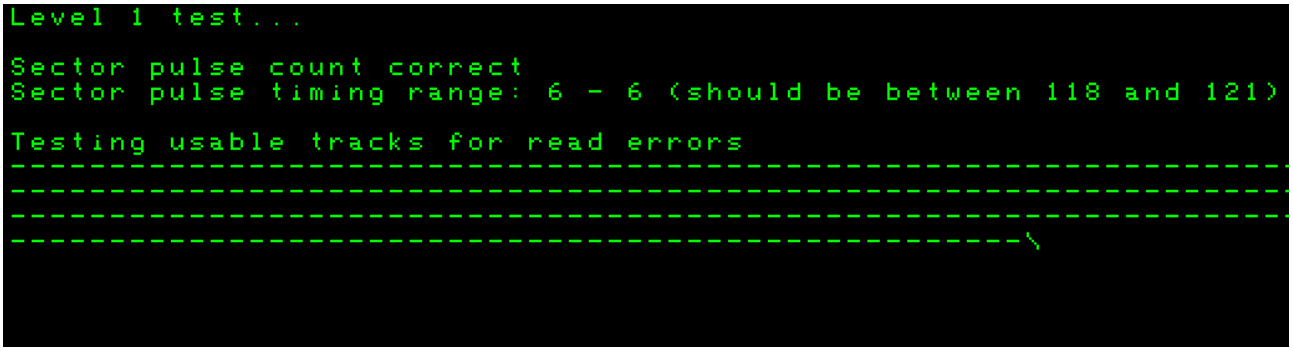


Fig 15. The hard drive (HD Delay is OFF) is 20 times faster than the 'correct speed'. (=FAST). The Level 1 Test on the HD Supplement Disk shows the pulse timing range is only 6 rather than 120.

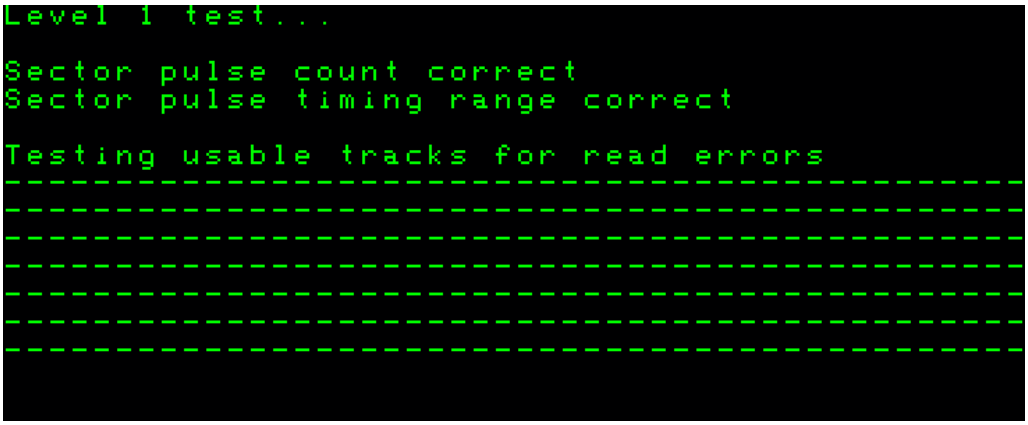


Fig 16. With HD Delay ON, (= SLOW) pulse timing range shows as "correct"!

It is very noticeable if the HD Delay is toggled SLOW/FAST while the HD Supplement Disk hard drive tests are in action.

3.3 Use ‘aread’ Input.

Read in an ASCII file from disk instead of having to type it all in manually. The ASCII file is read in line-by-line until it has all been entered. The keyboard then waits for user input, as it does normally

The Input File is selected with a file-chooser window. It is read in immediately after being selected.

Files read in with ‘aread’ will be processed in exactly the same way as they would if typed in at the keyboard. Excessively long lines will be rejected by the command-line processor of some operating systems, WordStar can ‘choke’ temporarily, but usually recovers.

3.4 Slots and Peripheral cards

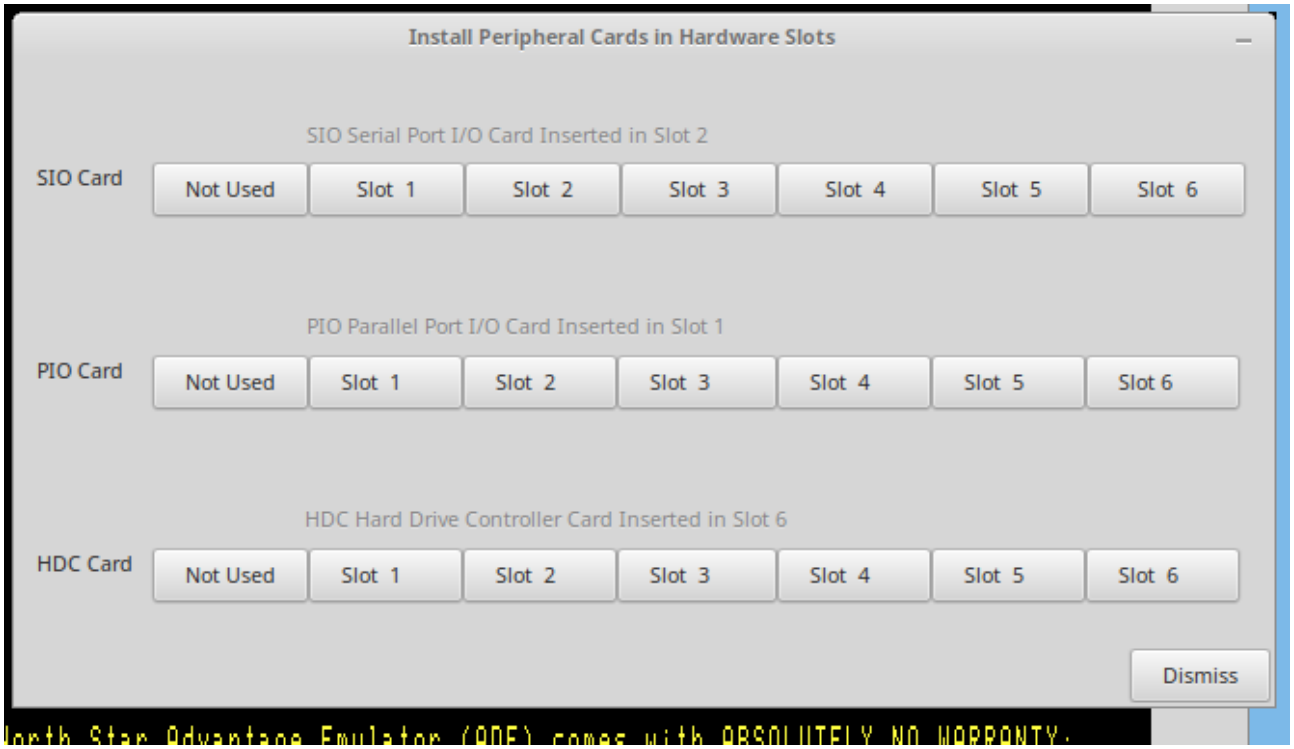


Fig 16. Peripheral Hardware Slot Management.

The North Star Advantage is normally set up so that the CP/M I/O Peripheral Card devices are allocated to slot numbers 1 and 2. Slot number one contains the device specified as the ‘LIST’ device. This could be either a parallel or a serial printer, according to whether a parallel card (PIO) or a serial card (SIO) is inserted into Slot Number 1. The default peripheral card in Slot Number 1 is a PIO card.

The ‘READER’ and ‘PUNCH’ devices are allocated to Slot Number 2. This pretty much requires these devices to be managed by a serial device, and the peripheral card in Slot Number 2 is software coded by North Star to be a serial card (SIO).

The Hard Disk Controller card is allocated by North Star’s software to Slot 6.

The North Star software assumes that the Peripheral Device cards will be inserted in the slots as specified above. Normally, no changes will be ever made to the Card-Slot Configuration, except by custom software.

In summary, ADE when started has the following configuration:

Slot 1	PIO Card
Slot 2	SIO Card
Slot 6	HDC Card

Each peripheral card has a numeric card-ID so that the Advantage can tell what sort of card is inserted into any particular slot.

PIO Parallel Card	DB hex
SIO Serial Card	F7 hex
HD Controller Card (Later Revision)	BF hex
HD Controller Card (Early Revision)	BE hex
Empty Slot	FF hex

The cards pre-inserted into the card-slots may be changed manually by using the monitor 'Slot Card Setup' menu item if deemed necessary. But the standard North Star Software would need to be altered to suit.

3.5 Allocate I/O Port Files

Attach or detach a unix file to or from an Advantage I/O Port. The PIO peripheral card acts as a parallel I/O port. The SIO peripheral card acts as a serial I/O port. In unix, everything is a file so one unix file or pipe is attached to to SIO-in port, and another to the SIO-out port.

Example: The 'List' device is allocated to the North Star Advantage Slot 1 which is where the PIO card is normally installed. Anything sent to the 'List' device will therefore show up as data in the file attached to the PIO output port. ('**/tmp/parallel_out**' as shown in the screenshot image.)

Note that some device names are used for both input and output, such as /dev/tty0. When you wish to use one of those, use the same device filename for both SIO IN and for SIO OUT, or for both PIO IN and PIO OUT.

When it comes to specific I/O ports, such as /dev/ttyS0 or perhaps /dev/ttyUSB0, the user will need to have sufficient permissions to access them. Generally these I/O ports have **dialout** group ownership in most distros, thus the user will need to be included in the **dialout** group to be able to access those I/O ports.

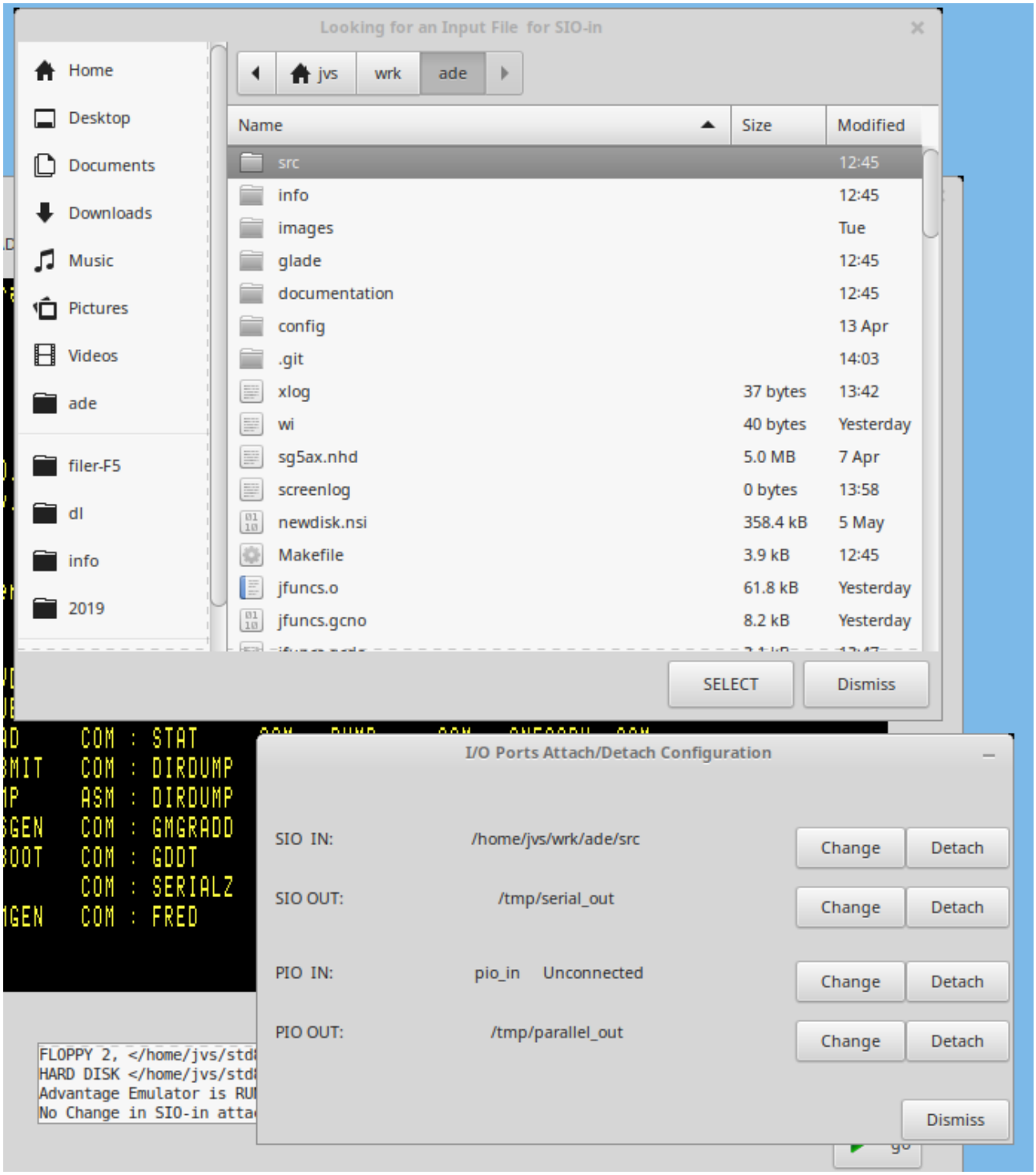


Fig 17. Attaching Linux Files/Pipes to Advantage I/O Ports

3.6 TEXT COLOR OF THE EMULATOR OUTPUT

A selection of colors is depicted in the various screen capture images. As this is pretty much a ‘set and forget forever’ option, it was decided against having a color-selection window as one of the ‘ADE Options’. To make a change it is simply a matter of selecting suitable values for the 24-bit RGB components, RED_LEVEL, BLUE_LEVEL, GREEN_LEVEL in the ‘ade.h’ file and recompiling. Some examples -

Green on Black: (as default)

RED_LEVEL	0x3F
GREEN_LEVEL	0xFF
BLUE_LEVEL	0x3F

Amber on Black:

RED_LEVEL	0xFF
GREEN_LEVEL	0xBF
BLUE_LEVEL	0x3F

Yellow on Black:

RED_LEVEL	0xFF
GREEN_LEVEL	0xFF
BLUE_LEVEL	0x3F

White on Black:

RED_LEVEL	0xFF
GREEN_LEVEL	0xFF
BLUE_LEVEL	0xFF

3.7 Toggle Caps Lock ON and OFF

Many of the older Operating Systems will not recognise the use of lower-case characters. While one can use the actual Caps Lock Key to turn on the CapsLock, it would also turn on upper-case for the host Operating System as well. This can be a nuisance.

North Star DOS only understands uppercase commands, so it's necessary to toggle Capslock ON when using DOS. CP/M automatically converts command-line lowercase to uppercase anyway, so the Capslock setting can be set to personal preference.

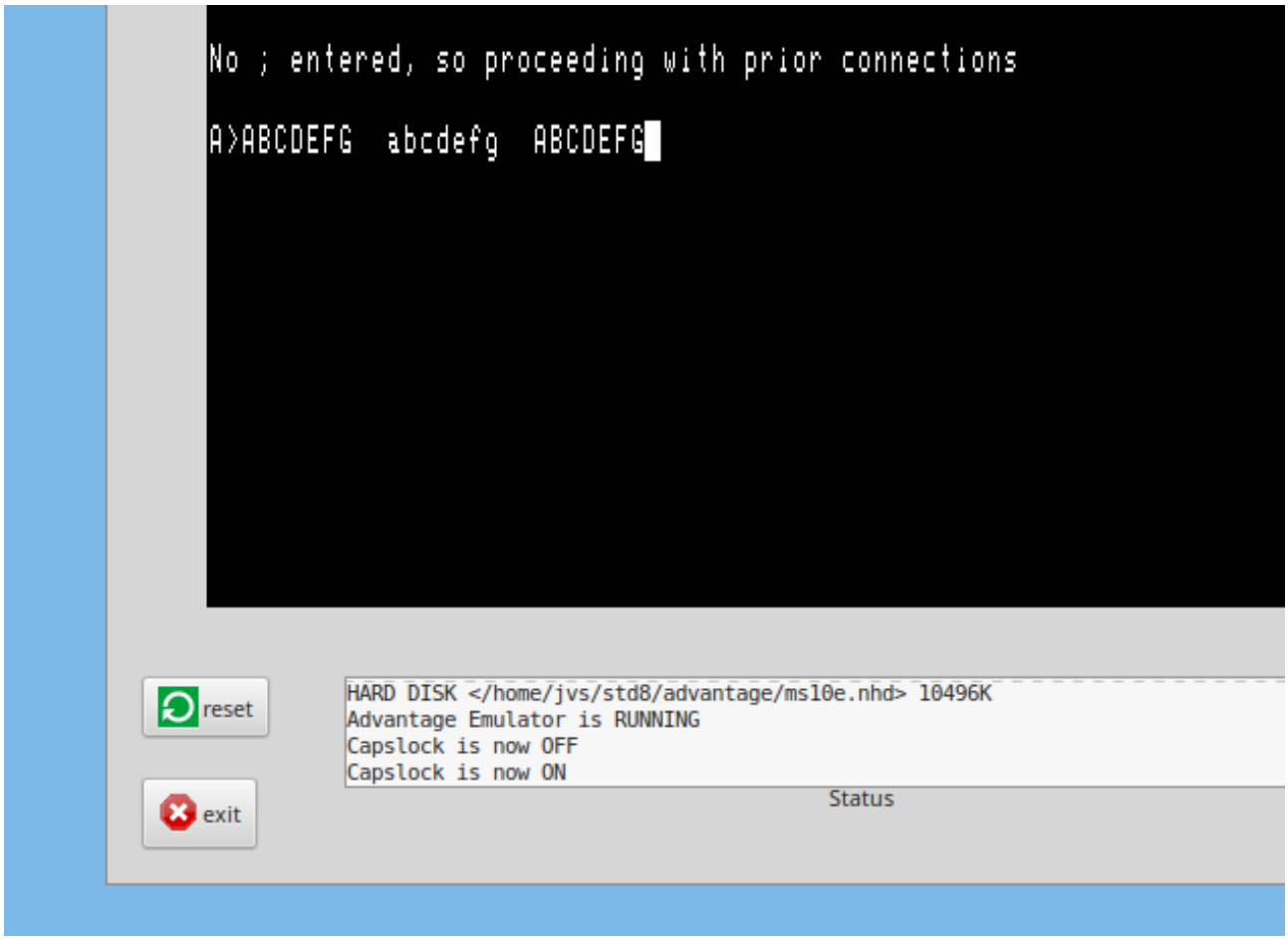


Fig 18 Caps Lock Toggled ON/OFF

In the screenshot above, Capslock starts out as being ON, showing the 'ABCDEFG' in uppercase. Then Capslock is toggled to OFF, as shown in the Status Window, with the next set of characters being lowercase 'abcdefg'. The Capslock is then toggled back ON, again showing in the Status Window, and the final 'ABCDEFG' is again uppercase

3.8 Toggle Cursor Lock ON and OFF

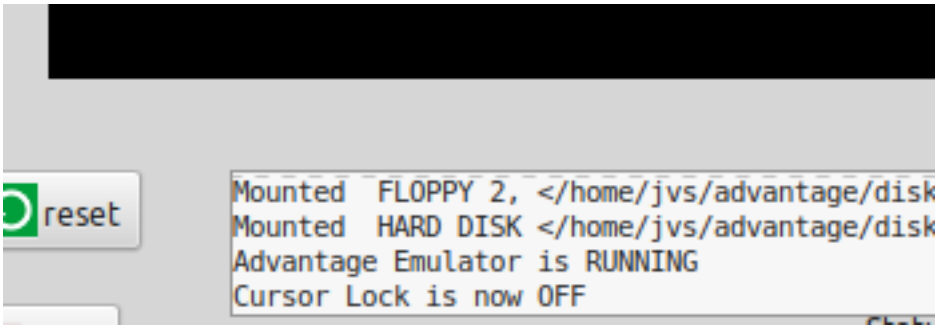


Fig 19. Cursor Lock Toggled OFF

When the CURSOR LOCK is toggled ON, the keypad 'Numbers Mode' is toggled OFF. The NUMLOCK LED on the PC keyboard should also show the opposite sense. In other words, when the Keypad 'Numbers Mode' is turned OFF, the NUMLOCK LED (Now **temporarily**, while ADE is running, the CURSOR LOCK LED) should turn ON.

3.9 Mimicking the ADVANTAGE Keyboard with the PC Keyboard

Generally speaking, the PC keyboard is pretty much the same as the ADVANTAGE keyboard, however there are some major differences. The PC keyboard has SHIFT, CONTROL, ALT and Windows 'Meta' keys, whilst the ADVANTAGE keyboard has only the SHIFT, CONTROL and COMMAND 'Meta' keys.

Both keyboards have a CAPS LOCK key, but the PC has a NUM LOCK key which has a indicator light showing when the NUMBERS mode is ON; whereas the ADVANTAGE has a CURSOR LOCK key which works in the opposite sense. That is, the indicator light is OFF when the NUMBERS mode is ON.

The PC keyboard has only 12 function keys. The ADVANTAGE keyboard has 15 function keys.

For a fuller discussion of how to work around the differences between the two types of keyboard, see 'Appendix C. ADVANTAGE KEYBOARD AND PC KEYBOARD: SAME AND DIFFERENT'

4.0 ADE DEVELOPMENT ASSISTANCE

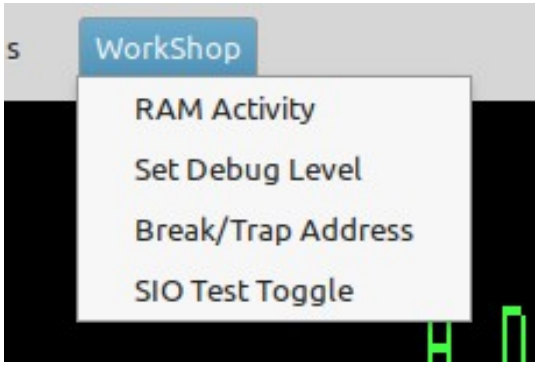


Fig 19. ADE Development menu: ‘WorkShop’

4.1 Display RAM in the North Star Advantage virtual machine.

Technical Explanation:

The 64K of Usable Virtual RAM (0000-FFFF) is made up of four 16K ‘pages’. These pages can be allocated non-exclusively in any order from a set of sixteen possible physical 16K pages between 00000 and FFFFF. Note that one physical page, say RAM 0, could be allocated twice in the 4 Virtual RAM pages. Those physical pages are:

```
0:    Physical RAM 0
1:    Physical RAM 1
2:    Physical RAM 2
3:    Physical RAM 3

4:    Physical RAM 4
5:    Physical RAM 5
6:    Physical RAM 6
7:    Physical RAM 7

8:    Video RAM 0
9:    Video RAM 1     (only the first 4K is available)
A:    Non-Accessible
B:    Non-Accessible

C:    PROM 0      ( Boot PROM is only 4K long but
D:    PROM 1              repeated 16
E:    PROM 2              times between C0000
F:    PROM 3              and  FFFFF )
```

Some possiblities for the Virtual RAM:

```
0000-3FFF:  Page 8 (Video RAM 0)
4000-7FFF:  Page 1 (Physical RAM 0)
8000-BFFF:  Page E (Boot PROM)
C000-FFFF:  Page 0 (Physical RAM 0)
```

This subsystem has usage similar to CP/M 'DDT' or MSDOS 'DEBUG'

Commands:

Upper or lower case commands are accepted
<xxx> is required parameter
[xxx] is optional parameter

Some of the commands below act on a single page (16 K block) of PHYSICAL Memory which has been previously specified with the ‘Page’ command , some others will act on the full 64K of VIRTUAL RAM.

compare
C <start address> <finish address> <start of compared block>
c 1a00 2000 2a00
Compare two equal-length blocks of memory. Only the bytes which are different will be displayed with location and values.

display
D [start address] [finish address]
d 0 12FF

Display the block of memory selected, showing bytes as hexadecimal and ASCII. If no start and end address specified, the command will continue for 100 H bytes from where it ended last.

examine/substitute
E <start address>
E 2CFF

Examine/change values at memory locations. The operation is stopped when no new value is entered, just a plain 'enter'.

fill
F <start address> <finish address> <fill byte>
f 1000 2000 55

Fill a block of memory with byte-value specified by <fill byte>.

hex
H <value> <value>
h 1267 abcd

Hex arithmetic results of the addition of two values and the subtraction of the second value from the first value.

load
L [load address]
l 2a00

Load the file (previously specified by the 'N' command) into memory. If a load-address is not specified the file will be loaded into location 0000 H.

move
M <source start address> <source end> <destination>
M 4d00 5000 6d00

Move the block of memory specified by the block's start and end into memory beginning at the destination address.

name
N <file name>
N xtest.bin.bas

Change active file-name which specifies which unix file will be used for 'load' and 'write' operations.

quit
Q
Quit from the RAM display subsystem back to the emulator's control console.

search
S <start address> <end address> "string"
S 0100 4fff "North Star"

Search for string delimited by quotes (") within memory block specified.
search [2]
S <start address> <end address> byte.byte...
S 0100 4FFF 38.4F.4D.60

Search for a list of bytes specified in hex. and joined by dots.

write
W [number of bytes in hex]
w 5c00

Write to the disk file previously specified by the 'N' command. Write the number of bytes specified.

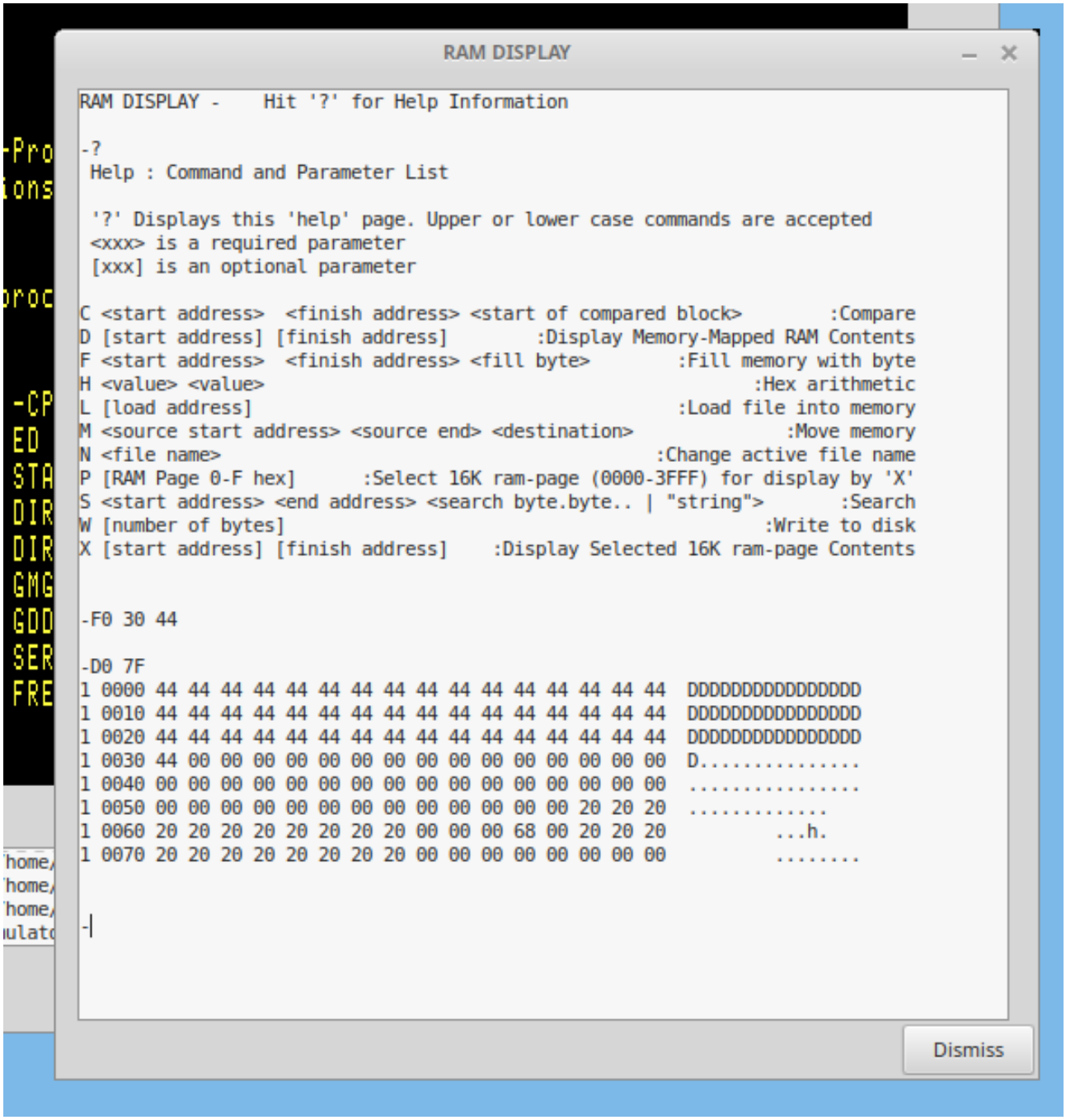


Fig 20. RAM Display Window

4.2 Set Debug Logging Level.

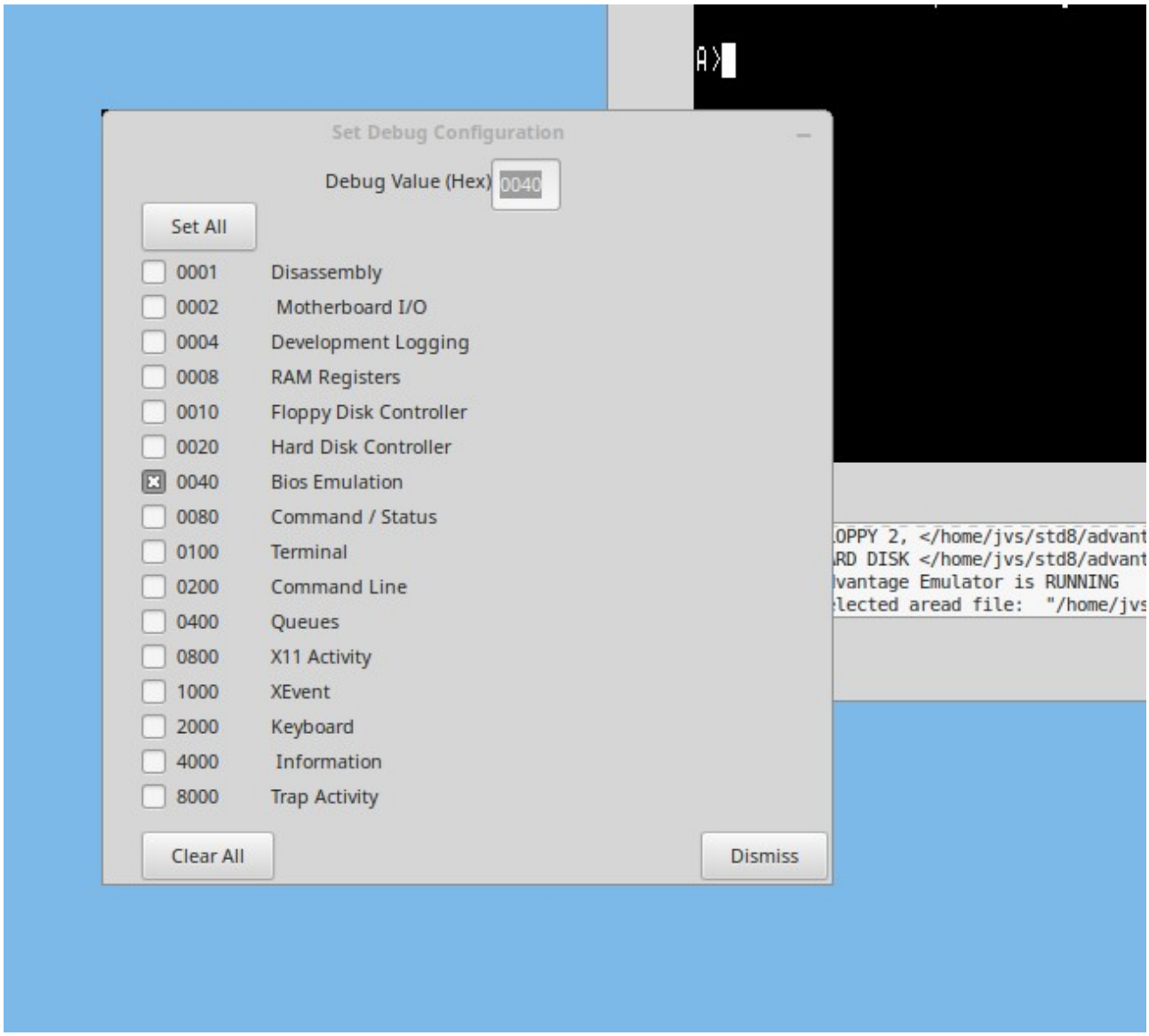


Fig 21. Setting the Debug Logging Level

The debug level can be set by typing a hex value into the small ‘Debug Value (Hex)’ entry field at the top of the pop-up window.

Alternatively the various check-boxes can toggled on or off by clicking in the selected box, or all check-boxes can be selected or all cleared by clicking on the appropriate button. Any debugging or information logged will be written to the ‘xlog’ file.

4.3 Setting Execution Breakpoint Address, and Trap Address

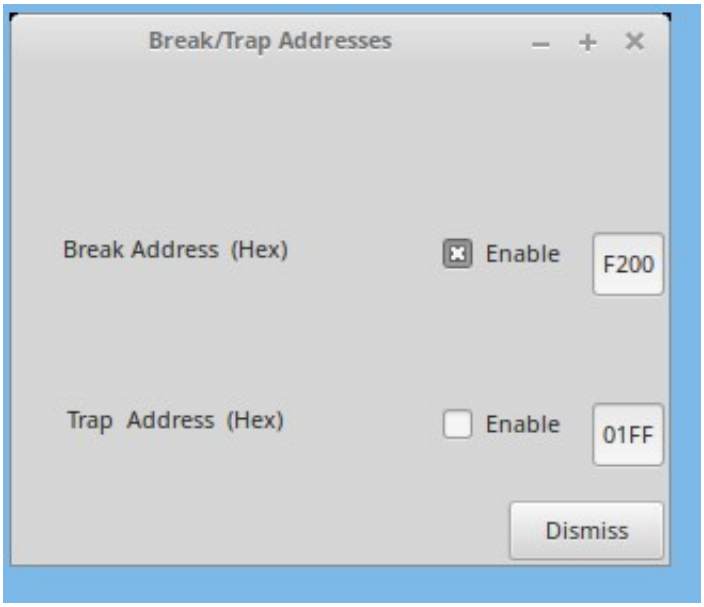


Fig 22. Enabling and Setting Break And Trap Addresses

break

Set a breakpoint address to stop the emulator at a pre-specified address. This is equivalent to the “PAUSE” button, but it occurs at a desired execution address. The contents of the RAM can then be examined by using the Ram Display functions. Hitting the ‘go’ button will resume execution from that breakpoint address and it will continue until that breakpoint address is again reached, unless the breakpoint is disabled while execution is stopped.

trap

Set a trap address to stop the emulator, perform a user-specified unix operation, return to the emulator and continue.

A dummy ‘trap’ function is included in the emulator source (trap.c) which merely prints the trap address and the register values. The trap function could be used to access parts of the host unix system or perform any other required operation.

Both the ‘break’ and ‘trap’ functions are enabled and disabled by the Check Buttons associated.

4.4 Log the debug information to Unix Disk File.

Automatically sends debugging/information output to the ‘xlog’ unix file.Take care, because the quantity of information sent to the log file can reach the maximum size (2 Gig in 32-bit systems, whole disk or whole filesystem in 64-bit systems) within a fairly short time.

Unless you're doing development on the North Star Advantage Emulator itself, it probably will not be useful to use any debug logging at all.

4.5 Log the Screen Output to Unix Disk File.

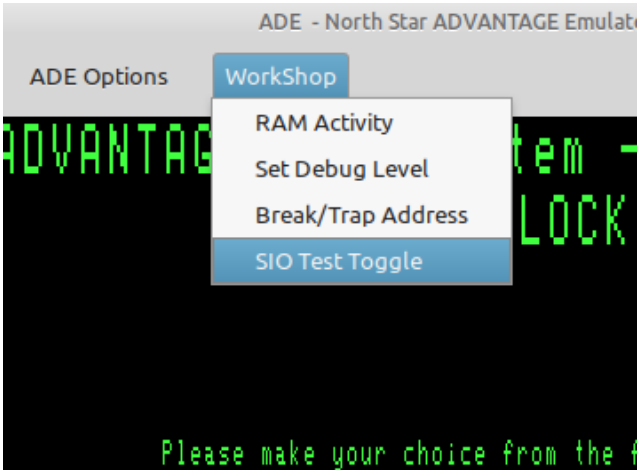
Automatically sends all ASCII screen text output to the ‘screenlog’ unix file. This can be handy to refer to if text output scrolls off the top of the screen before you can read it.

4.6 When running the SIO Card Test in the DEMODIAG Disks, Set the PN00000 Jumper as specified.

To test the SIO (Serial In-Out Port) boardlet, boot up the DEMODIAG disk.

Then select: 3 Diagnostics → 2 Single Block Mode → 5 Serial I/O Test (in the DEMODIAG-220.NSI disk)

The “Special SIO Test Jumper, PN 00000” must be inserted on all SIO boardlets to loop all SIO serial output back to the SIO serial input port. Since we don’t have a physical boardlet, this action is performed by toggling to ON the ‘SIO Test Toggle’ selection in the WorkShop Menu. At the end of the test, toggle the “SIO Test Toggle to OFF.



5. HELPER PROGRAMS

5.1 North Star Tools

Note: Only the **mkhd** program is installed by default. The other tools are compiled but will remain in the **..../ade/src/north_star_tools/** directory. If you want to use them often, you can copy them to your **/home/<username/.local/bin** directory.

5.1 mkhd (make hard-disk-image file)

mkhd is used to produce ADE hard-disk image files. The smallest of the images of the North Star 'standard' hard-disk types (as included in the HD5XTEST program) is 5 megabytes, the largest is 30 megabytes.

A typical example session with mkhd is shown (user input in **bold**):

```
centrepoint [jvs] /home/jvs/advantage/disks > mkhd


                        === mkhd ===
                        Version  3.2

Prepares a "Standard" 5-inch Hard-Disk Imagefile for use with North Star
Horizon Emulator (nse) and Advantage Emulator (ade) running HDOS.

Disk-image sizes available range from 5 MB to 30MB.

                                Usable    Usable  Shipping  Total   Total
No.   Type   Rev. Cylinders Heads Sectors  Capacity Cylinder Sectors Capacity

 1    SG5A   1.0    153      4     9792     4.90 M    153      9792   4.90 M
 2    TN5A   2.0    153      4     9792     4.90 M    153      9792   4.90 M
 3    MS5B   2.0    306      2     9792     4.90 M    336     10752   5.38 M
 4    RD5B   2.0    306      2     9792     4.90 M    319     10208   5.10 M
 5    SG5B   2.0    306      2     9792     4.90 M    306      9792   4.90 M
 6    TN5B   2.0    306      2     9792     4.90 M    306      9792   4.90 M
 7   CM10E   2.0    612      2    19584     9.79 M    650     20800  10.40 M
 8   MS10E   2.0    612      2    19584     9.79 M    656     20992  10.50 M
 9   CM15C   2.0    306      6    29376    14.69 M    306     29376  14.69 M
10   SG15C   2.0    306      6    29376    14.69 M    306     29376  14.69 M
11   RD15C   2.0    306      6    29376    14.69 M    319     30624  15.31 M
12   TN15C   2.0    306      6    29376    14.69 M    306     29376  14.69 M
13   MS15D   2.0    480      4    30720    15.36 M    522     33408  16.70 M
14   MS15E   2.0    459      4    29376    14.69 M    522     33408  16.70 M
15   CM20E   2.0    612      4    39168    19.58 M    650     41600  20.80 M
16   MS20E   2.0    612      4    39168    19.58 M    656     41984  20.99 M
17   RD20E   2.0    612      4    39168    19.58 M    639     40896  20.45 M
18   MS30D   2.0    459      8    58752    29.38 M    522     66816  33.41 M
19   CM30E   2.0    612      6    58752    29.38 M    650     62400  31.20 M
20   MS30E   2.0    612      6    58752    29.38 M    656     62976  31.49 M
21   RD30E   2.0    612      6    58752    29.38 M    639     61344  30.67 M

   Select ( '0' to exit) : 1

     Type: SG5A disk:   4.90 M usable capacity.   ---- Is that correct? y

    creating disk-image type SG5A,  4.90 M.

    Enter file name for this disk: /tmp/advantage5mb

Disk ImageFile:  /tmp/advantage5mb    requested.

Disk ImageFile: '/tmp/advantage5mb'   created OK.

Creating SYSTEM account. Do you want to include the TRANSIENT file? (Y/n) y

 TRANSIENT for the Advantage, or the Horizon?            (a/H) a
  Advantage TRANSIENT installed.

Done.
```

I suggest the use of the .NHD extension for these North Star Hard-Disk Image files. This extension, like most, is probably already in use elsewhere but is unlikely to be confused with our usage.

5.2 **<u>nshdls</u>** (nshd list directory)

nshdls <North Star hard-disk-image>

nshdls disks/SG5A-1.NHD

nshdls displays lots of information from the file-entries in the North Star Hard Disk index. Because the position of the file-entries within the directory depends on a hash function, the directories always seem jumbled – this was a trade-off way back in the days of slow hard-disks.

There are two options to **nshdls**:

- -d Show deleted files as well as existing files
- -l (long) Show more details about the directory and hard drive

```
centrepoint [jvs] /home/jvs/wrk/ade > nshdls -d disks/cm10ex.nhd


FileName  Size:Sectors---Blocks     Account    Filetype       Description


BUSINESS          0       0 blk     - - -          0     ACCOUNT (02) "BUSINESS"
CPMA           2031    4062 blk     SYSTEM         7     CP/M Virtual Disk File
TRANSIENT         0       0 blk     SYSTEM         1     NS **  DELETED  **
CPMB           2031    4062 blk     SYSTEM         7     CP/M Virtual Disk File
CPMC           2031    4062 blk     SYSTEM         7     CP/M Virtual Disk File
CPMD           2031    4062 blk     SYSTEM         7     CP/M Virtual Disk File
CPME           2031    4062 blk     SYSTEM         7     CP/M Virtual Disk File
CPMF           2031    4062 blk     SYSTEM         7     CP/M Virtual Disk File
CPMG           2031    4062 blk     SYSTEM         7     CP/M Virtual Disk File
CPMH           2031    4062 blk     SYSTEM         7     CP/M Virtual Disk File
SYSTEM            0       0 blk     - - -          0     ACCOUNT (01) "SYSTEM"
CPMWORK          47      94 blk     SYSTEM         6     CP/M Workfile
LEDGER          250     500 blk     BUSINESS       3     HBASIC data file
```

5.3 **<u>nshdbm</u>** (nshd bitmap)

nshdbm <North Star hard-disk-image>

nshdbm disks/SG5A-1.NHD

nshdbm gives a slightly more compact set of information than the long North Star Hard Directory Listing and also shows a graphical representation of the usage of the hard disk's file-space.

```
 /tmp/nse_120805/disks > nshdbm SG5A-1.NHD


          Bitmap and Directory for North Star Hard Disk Image "SG5A-1.NHD"  (4.896 Megabytes)

Total DIBs = 612 (maximum usable = 602, as also uses 1 DIB for System Track, 8 DIBs for File-Directory, 1 DIB for Test Track)
                            [ 1 DIB is 16 sectors unless a power-of-2 factor is applied ]


Dir    Dir              File             FileSize    Bytes  dib/dib    ndib0    ndib0     File  Type       File Type
Offset Type  Accnt  --  Name    -- Attr Sectors Blocks inLast Num/Sects Address Img_Offset  Type  Data       Description
_____________________________________________________________________________________________________________________________
011A00  1    0001  CPMA            80     1562  3124    200    6  64     0020   00040000    7   0001.00  CP/M Virtual Disk File
012000  1    0001  TRANSIENT       80       24    48    200    4  16     0003   00006000    1   1F00.00  Executable, TypeData=GO
012A00  1    0001  CPMB            80     1562  3124    200    6  64     0084   00108000    7   0001.00  CP/M Virtual Disk File
013400  1    0001  DT              80        2     4    200    4  16     0005   0000A000    1   5000.01  Executable, TypeData=GO
014400  1    0001  BACKUP          80       31    62    200    4  16     0006   0000C000    1   2600.00  Executable, TypeData=GO
016E00  1    0001  CLEAN           80        9    18    200    4  16     0013   00026000    2   0010.00  HBASIC program
017000  1    0001  CK              80        2     4    200    4  16     0010   00020000    1   5000.01  Executable, TypeData=GO
017020  1    0001  CO              80        4     8    200    4  16     0011   00022000    1   5000.01  Executable, TypeData=GO
017400  1    0001  RECMAIN         80       15    30    200    4  16     0012   00024000    2   001D.00  HBASIC program
01A600  1    0001  RECOVERS        80       24    48    200    4  16     0014   00028000    2   002E.00  HBASIC program
01C400  1    0001  RECEXP          80        3     6    200    4  16     0016   0002C000    2   0003.00  HBASIC program
01CC00  1    0001  BAKEXP          80        3     6    200    4  16     0017   0002E000    2   0005.00  HBASIC program
01DE00  1    0000  SYSTEM          01        0     0    000    0   1     0000   00000000    0   0000.00  ACCOUNT directory entry
01DE20  1    0001  CPMWORK         80       47    94    200    4  16     0001   00002000    6   0000.00  CP/M Workfile
01E400  1    0001  RECOVER         80       31    62    200    4  16     001A   00034000    1   2600.00  Executable, TypeData=GO
01EC00  1    0001  HBASIC          80       32    64    200    4  16     001C   00038000    1   2600.00  Executable, TypeData=GO
01F200  1    0001  BACKUPS         80       24    48    200    4  16     0018   00030000    2   0030.00  HBASIC program


128 DIBs per line. 'S' = System, 'D' = Directory, 'X' = TestTrack, '.' = Unallocated, 'o' = Allocated, 'U' = Multiple Allocation

SoooooooDDDDDDDDoooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo........................
................................................................................................................................
................................................................................................................................
...................................................................................................X
nullius [jvs] /tmp/nse/disks >
```

5.4 nshdcp (nshd copy file to unix)

nshdcp <North Star hard-disk-image> <Filename>

nshdcp SG5A-1.NHD HBASIC

nshdcp extracts a North Star HDOS file from the North Star Hard Disk image. The filename to be extracted is case-sensitive, although the huge majority of HDOS filenames are upper-case only.

Note that any CP/M files are contained within CP/M virtual disks which are large HDOS files. **nshdcp** will only extract the virtual disk file itself, rather than any indivual CP/M file contained within the virtual-disk file.

5.5 unskew-hd-image

unskew-hdimage <North Star Hard Disk Image> <unskewed image file>
OR
unskew-hd-image <unskewed image file> <North Star Hard Disk Image>

unskew-hd-image SG5A-1.NHD image-plain-a

unskew-hd-image can be dangerous to your hard-disk image-files. **Be careful!** It will be used mainly if you are trying to resurrect portions of files which have been lost by removing the interleaving of the sectors and giving a flat file with everything in correct order.

5.6 nsfilecalc (calculate filesizes in terms of NSDOS 256-byte 'blocks')

ns*filecalc*

```
nullius [jvs] /tmp/nse/disks > nsfilecalc

           North Star DOS/HDOS File-Size Calculator
                 copyright 2012 Jack Strangio

A North Star Floppy Disk file is restricted to a maximum length of 66
tracks on a DQ disk, or 660 sectors, 1320 blocks, 330 kilobytes.

A North Star Hard-Disk file is made from 'hunks' containing multiple
sectors.  These 'hunks' were originally so-named by North Star, but later
this name was changed to 'DIBs'.

Each DIB ('Data Incremental Block', similar to 'clusters', 'extents', etc.
in other operating systems) contains a multiple of 16 sectors.  There can be
a maximum of 128 DIBs per file.

Since this could really restrict the maximum size of a file, a power-of-2
factor can be applied to 16 giving 16, 32, 64, 128, or even up to 256 sectors
per DIB.  Consequently, it becomes possible to produce a file which can go up
to the maximum allowable file-size on a hard-disk: 65,535 blocks, 32,768
sectors or 16.384 megabytes.

Each file contains its own internal DIB-directory, which takes up the first
sector of the file itself.  Keep this 'loss' of the first file sector in
mind when creating your files on the hard-disk.  The Hard-Disk Directory (or
Index) merely tells HDOS where the file's first sector with its
DIB-directory is located upon the hard-drive.


 Bytes                                (1)
 North Star Blocks (256-byte)         (2)
 Hard-Disk Sectors (512-byte)         (3)
 North Star DIBs ('clusters','extents') (4)
 Kilobytes (1024 bytes)               (5)
 Megabytes (1000x1024 bytes)          (6)


         Select Units: ('0' to quit)  6
```

```
                Enter Value wanted : 3

File is:  3072000 bytes, 12000 blocks, 6000 sectors, 94 DIBs, allocation factor = 4, 3000.0 KB

    HDOS Command Line:     CR FILENAME[[,ACCOUNT],DISK_UNIT]  12000  4


****    That size of file has unused sectors in the last DIB.    ****

      If all sectors of the last DIB were to be included, the file's size would then become:

          3079680 bytes, 12030 blocks, 6015 sectors, 94 DIBs, allocation factor = 4, 3007.5 KB

    HDOS Command Line:     CR FILENAME[[,ACCOUNT],DISK_UNIT]  12030  4

nullius [jvs] /tmp/nse/disks >
```

nsfilecalc will notify you whether the file-size you have requested will not completely fill a DIB. For instance, if the disk space is being allocated in 64 block chunks, a file that's 65 blocks long will take up 128 blocks. So if you're making a CP/M virtual disk, it costs you no more to make your 'disk' have 128 blocks in size than a 'disk with only 65 blocks of disk space.

Therefore, if there is unused space left in the allocated disk area you may, if you want, increase the size requested up to the end of the last DIB. Hence the recommendation in the printout above of making a 12000-block disk into a 12030-block disk.

5.7 nsfd2u (copy NSDOS file from floppy-disk to unix)

nsfd2u <NSDOS disk-image>

nsfd2u D04B01.NSI

nsfd2u reads the files off a double-density North Star DOS disk image file and creates copies of those files in the unix file space.

The unix filenames will have the format of <Name of File>_<FileType>[_Go-Address]. The Go-address will only be used with a file of Type 1 (executable).

example 1.
The M5700 executable file is Type 1 and has a Go-Address of 5700 H; this has a unix file name of M5700_1_5700

example 2.
The BASIC program called OTHELLO is Type 2 (BASIC Program) and not being a executable Type 1 will have no Go-Address; this has a unix file name of OTHELLO_2

5.8 u2nsfd (copy file from unix to NSDOS floppy-disk)

u2nsfd <unix file> <NSDOS disk-image>

u2nsfd M5700_1_5700 MYDOSDISK.NSI

u2nsfd will copy a file from the unix file space onto a double-density North Star DOS disk image file.

If the above filename format (as in nsfd2u) is used for the North Star DOS filename in the unix file space, then the file will be added to the NSDOS disk directory complete with Type attributes and Go-Address if applicable. If the NSDOS directory already has a file of the same name, the new file will replace the earlier file.

If the above filename format is not used, the file-type defaults to Type 0 (undefined). This can then be altered using the TY command in NSDOS:

TY <filename> <File-Type> [Go-Address]

5.9 compact

compact <NSDOS disk-image>

compact MYDOSDISK.NSI

compact will 'compact' a North Star DOS disk image file. It will act similar to a defragmenting of the disk-image file by moving all files towards the beginning of the disk, eliminating any unused space between the files where previously deleted files once were. [Same as running the compact program in the emulator]

5.10 nsfdls (NS floppy-disk list directory)

nsfdls <*NSDOS disk-image*>

nsfdls MYDOSDISK.NSI

nsfdls lists the directory of the floppy-disk image file in the same format as the LI in NSDOS. [Same as running the LI program in the emulator]

5.11 mkfs.ns

mkfs.ns [-s] <disk-image filename>

mkfs.ns -s MYSSSDDISK.NSI

mkfs.ns creates an empty North Star DOS formatted floppy-disk image. It can produce either single-sided, single-density disk-images (88K) or double-sided, double-density disk-images (350K). The default size is 350K, if you use the '-s' option an 88K disk-image is produced. The first 8 characters of the filename are used as the disk-label. [Same as creating a new floppy in the Disk Manager screen]

5.12 jdz80 (Z80 disassembler)

jdz80 is a slightly improved version of Marat Fayzullin's 1999 DAsm, in which relative jump destination addresses are calculated and displayed rather than just displaying the relative jump offsets.

5.13 OTHER TOOLS

5.13 cpmtools

Life is simpler with cpmtools-2.7 (or later) which can be obtained from most linux repositories. This set of utilities can be used to copy files directly between North Star CP/M disk-images and the unix/linux file space. It will be necessary to add the following disk definitions to the cpmtools config-file **diskdefs** which is usually at /etc/cpmtools/diskdefs.

```
diskdef nsfd
  seclen 512
  tracks 70
  sectrk 10
  blocksize 2048
  maxdir 64
  skew 5
  boottrk 2
  os 2.2
end

diskdef nshd4
  seclen 512
  tracks 512
  sectrk 16
  blocksize 4096
  maxdir 256
  skew 0
  boottrk 0
  os 2.2
end
```

The added disk-definitions will enable cpmtools to understand the North Star CP/M disk formats, both the floppy-disk images and the larger CP/M Virtual Disk Images on the hard disk. (Note that you will need to copy the hard-disk CP/M Virtual Disk image-file off from the hard disk image-file by using the **nshdcp** program before you can start to use the cpmtools with it.)

The utilities in cpmtools include:

cpmls	list files in the North Star CP/M disk-image
cpmcp	copy files to and from the North Star CP/M disk-image
cpmrm	delete files from the North Star CP/M disk image
mkfs.cpm	prepare stub disk for CP/M. In my experience, this does not work properly. Instead, use mkfs.ns to produce an NSDOS disk then FORMAT it for CP/M.

5.14 screenlog

screenlog is not a tool as such but a record of ADE's screen output.

5.15 xlog

xlog is not a tool but is a record of all debugging information. Can make very large log files.

6 VARIOUS THINGS.

6.1 OTHER FILES REQUIRED

Various floppy-disk image files:

These are available from various sources. Most of them have a .nsi extension.

6.2 COMPILING LIBRARIES REQUIRED

The linux libraries required are GTK+ version 3, and libX11

6.3 VARIOUS USEFUL MANUALS

Most of the manuals are available from http://www.hartetechnologies.com/manuals/Northstar/ or from http://itelsoft.com.au.

Probably the most useful are:

North Star Advantage Technical Manual
North Star Advantage User Manual
North Star Advantage Graphics Manuals
North Star DOS Rev 5
North Star BASIC Version 6
North Star Advantage Emulator (ADE) User Guide (this manual)
North Star Hard Disk Operating System Manual
North Star CPM 2.2 Manual
North Star CPM 2.2 Preface to the Addendum
North Star CPM 2.2 Addendum

These are all included in the 'documentation' directory

6.4 BUGS

I feel I have got many bugs out, which makes ADE very usable. But there are still a few to go, apart from the things that could be done to make ADE not quite so rough-edged. It certainly is not yet anywhere near as elegant as I would like, and the fault-lines between the several programs that ADE is based upon are still very visible. Please inform me of any bugs that you discover. Email me at: jackstrangio@yahoo.com

6.5 TODOs

More realistic emulation of Parallel and Serial I/O, particularly the in-ports. There is a list in the TODO file.

6.6 AUTHOR and SUPPORT

Jack Strangio <jackstrangio@yahoo.com>

Website: http://itelsoft.com.au

APPENDIX A.

NORTH STAR HARD-DISK DATA FORMAT

DATA LAYOUT ON HARD DISK.

A hard-disk drive is actually a set of spinning disks (or platters). For each platter there are two heads, one above and one below the platter. Thus a hard-disk drive with two platters has four heads, and each head reads and writes on a separate 'surface'. Because all the heads are moved as a single unit from track to track on the platters, the set of tracks being read from is called a 'cylinder', so, in this case, there would be four tracks within each cylinder.

STRUCTURE OF SINGLE TRACK

Each North Star hard disk track consists of 16 sectors. Each sector has its own set of data fields. As the platter spins the disk-drive electronics supply pulses which specify when the first sector of the set of 16 sectors is reached by the read/write head (the index pulse), and when the start of each sector begins(the sector pulse). The index pulse is not retained by the North Star Hard-Disk Controller, but the sector pulse is latched on and is turned off by the Hard-Disk Controller itself.

STRUCTURE OF THE WRITTEN DISK SECTOR

When the sector-pulse is received from the hard-drive by the hard-disk controller, the controller waits a short period then begins sending a stream of zero bytes (00 H). This is to cushion variations in speed of the physical drive. After a enough time has passed, a Sync Byte (01 H) is sent to the hard drive to signify the actual start of the data to write on the disk sector.

The first set of real data written is the Sector-Label Header field, this is a set of nine bytes which identify which sector is being written. This information is later used when reading the disk, to ensure that the data being read is from the sector desired and not another sector.

The next data field contains the 512-bytes of data or program we want to store.

The last data field contains CRC information to ensure that the data has been written cleanly. If the data read back from the disk-sector does not match the store CRC value, there has been corruption of the data.

STRUCTURE OF THE SECTOR-LABEL HEADER FIELD

Example:

```
PHY  CYL  HED  LSl  LSh  STl  STh  CRC  CRC~
05   0C   83   BD   04   B0   04   09   F6
```

In typical North Star Computers fashion, the sector ID label is not that as suggested by Shugart in the ST506 protocol, but one which was designed by North Star themselves. However there are similarities.

Byte 1: PHYSICAL SECTOR

The lower 4 bits (Bits 0-3) are used to specify the physical sector on the track. The physical sector is the one calculated by skewing the reads to improve reading/writing speeds. The physical sector is calculated by adding 8 to the ODD logical sectors: logical sector 1 is at physical sector 9, logical sector 15 is at physical sector 7.

Bits 4 and 5 contain the 2-bit overflow of the CYLINDER byte (Byte 2) which then gives the CYLINDER byte a total of 10 bits which allows a maximum of 1024 cylinders

Byte 2: CYLINDER

This byte plus the extra 2 bits specified in Byte 1 allow 1024 cylinders.

Byte 3: Surface (Head Number)

The lower 3 bits are used to specify which head is selected.
The high bit (Bit 7) may used to specify whether the sector is write-protected or not.

Bytes 4 - 5: LOGICAL SECTOR NUMBER

These bytes contain the logical sector-number on the hard-drive. This number may differ from the physical sector number because of the skewing described above.

Bytes 6 - 7: SHIFTED TRACK NUMBER

These bytes contain the logical sector-number on the drive modulo 16. This can be thought of as either the disk-address of sector 0 on the track, or the 12 bits of the track number shifted up 4 bits. This supplies the physical sector address quite simply by adding the PHYSICAL sector in Byte 1 to this up-shifted track number.

example: (In hex numbers as it makes it easier to see.)

```
Logical sector : 04BD H
Track Number   : 004B H
Shifted Track  : 04B0 H
PHYSICAL       :   05 H
```

Physical Sector: 04B5 H

Byte 8: CRC SUM

This byte contains the lower 8 bits of the total obtained by adding all 7 previous bytes.

Byte 9: CRC BYTE COMPLEMENT

This byte contains the complemented CRC byte. (The sum of Byte 8 and Byte 9 is always FF H)

FURTHER EXAMPLE:

PHY	CYL	HED	LSl	LSh	STl	STh	CRC	CRC~
25	52	80	CD	DE	C0	DE	40	BF

Physical Sector: 5 (From Bits 0-3 of PHY)

Cylinder : 52 H (From CYL) + 0200 H (From Bits 4 & 5 of PHY) = 0252 H = 594 (Dec.)

Head : 0 (From Bits 0-2) of HED

Logical Sector : DECD H = 57037 (Dec.)

Physical Sector: 5 (From PHY) + DEC0 H (From Shifted Track) = DEC5 H = 57029 (Dec.)

CRC : 25 H + 52 H + 80 H + CD H + DE H + C0 H + DE H = 440 H = 40H

CRC~ : 40 H complemented = BF H (or BF H + 40 H = FF H)

APPENDIX B.

ADE's HARD-DISK IMAGE FILE STRUCTURE

The hard-disk image structure's size varies according to the number of sectors which were in the original physical hard disk.

The sectors are laid out as in physical sectors, rather than logical sectors. This means the sectors in the disk-image are interleaved, just as they are on the physical disk. There is an unskewing utility in the North Star Tools directory, but I don't think this would ever be used by most users of ADE.

NOTE: Validation that the file is truly a North Star Emulator hard-disk image depends solely on the presence of the North Star 'magic' bytes (00 H, FF H) at the start of the first sector of the hard disk image-file. This first sector is North Star's "Hard Disk Label" and contains much information about the size and layout of the hard disk.

If the two validation bytes are not found, ADE will not mount the file at all. While this means that a hard disk image file may become unusable very occasionally, it serves to guard against unwanted accidental damage to other types of files. If warranted, further tests for disk image validity may be included in later versions of ADE.

For producing ADE hard-disk image files of the 'standard' hard disks used by North Star Computers, see under North Star Tools, **mkhd,** Section 5.1, page 33.

APPENDIX C.

ADVANTAGE KEYBOARD AND PC KEYBOARD: SAME AND DIFFERENT

ADE EXPECTS A US 102-KEY PC KEYBOARD

ADE is configured to pretend that a standard US 102-key PC keyboard will mimic the usage of the North Star ADVANTAGE keyboard. Generally speaking, a PC hardware-scan-keycode for a particular key will be translated to the equivalent ADVANTAGE hardware-scan-keycode. However, some keys can not be directly translated, so I have made some workarounds in those cases.

META KEYS

The Meta Keys used on the ADVANTAGE are SHIFT, CONTROL and COMMAND. The ADE program itself uses an extra Meta Key.

The Meta Keys available on the PC are SHIFT Left, SHIFT Right, CONTROL Left, CONTROL Right, ALT Left, ALT Right, WINDOWS Left and WINDOWS RIGHT.

ADE can be configured to use one or more of the PC Meta Keys listed above to be allocated to the ADVANTAGE and ADE Meta Keys. Depending on your convenience, it's just a matter of amending the configurations defined in the 'ade.h' file and recompiling.

NOTE: Depending on your Linux distro, the ALT and WINDOWS keys may be configured in keyboard-shortcuts which capture the keystrokes before ADE can get hold of them. An example is ALT+F1 or WINDOWS+F1 which will show a Desktop menu. I have found it is convenient to use both the SHIFT Left and SHIFT Right as ADVANTAGE SHIFT, both CONTROL Left and CONTROL Right as ADVANTAGE CONTROL, using ALT Left as ADVANTAGE COMMAND, and then ALT Right as the ADE-Meta1 key. The actual defines are as shown here:

```
// ADVANTAGE->ADE Keyboard DEFINES
//
// Translates various PC keyboard keys to logical ADVANTAGE equivalents
//  PC Keyboard Physical Scan Codes

#define PC_WIN_L        0x85    //PC Windows Key Left
#define PC_WIN_R        0x86    //PC Windows Key Right
#define PC_ALT_L        0x40    //PC ALT Key left
#define PC_ALT_R        0x6C    //PC ALT Key Right
#define PC_CTRL_L       0x25    //PC CTRL Key Left
#define PC_CTRL_R       0x69    //PC CTRL Key Right
#define PC_SHIFT_L      0x32    //PC SHIFT Key Left
#define PC_SHIFT_R      0x3E    //PC SHIFT Key Right

#define PC_CAPS_LOCK    0x42    //PC CAPS LOCK Key
#define PC_NUM_LOCK     0x4D    //PC NUM LOCK Key

// CURRENT ASSIGNATION OF PC to ADVANTAGE KEYS

// Use BOTH PC SHIFT Keys

#define ADVANTAGE_SHIFT1        PC_SHIFT_L
#define ADVANTAGE_SHIFT2        PC_SHIFT_R


// Use BOTH PC CONTROL Keys

#define ADVANTAGE_CTRL1         PC_CTRL_L
#define ADVANTAGE_CTRL2         PC_CTRL_R
```

```
// Use only the ALT Left Key as ADVANTAGE COMMAND Key

#define ADVANTAGE_CMD1          PC_ALT_L
#define ADVANTAGE_CMD2          PC_NO_KEY

//Use only the ALT Right Key as the ADE META Key

#define ADE_META1               PC_ALT_R
#define ADE_META2               PC_NO_KEY

// Use the CAPSLOCK Key and the NUMLOCK/CURSOR LOCK Key as normal

#define ADVANTAGE_CAPS_LOCK     PC_CAPS_LOCK
#define ADVANTAGE_CURSOR_LOCK   PC_NUM_LOCK
```

NOTE: if, for example, you don't want to use BOTH of the PC SHIFT keys, you could change the relevant #define lines from something like this:

```
#define ADVANTAGE_SHIFT1        PC_SHIFT_L
#define ADVANTAGE_SHIFT2        PC_SHIFT_R
```

And if you wanted to use (say) the Right PC SHIFT key only as an ADVANTAGE SHIFT key, and leave the Left PC SHIFT KEY for another use, you could do something like this:

```
#define ADVANTAGE_SHIFT1        PC_SHIFT_R
#define ADVANTAGE_SHIFT2        PC_NO_KEY

#define ADVANTAGE_CMD1          PC_SHIFT_L
#define ADVANTAGE_CMD2          PC_NO_KEY
```

Important: Use PC_NO_KEY as a 'filler' for unused meta-key definitions.

In the above example we have the ***Left*** PC SHIFT key acting as the ADVANTAGE's COMMAND key, and the ***Right*** PC SHIFT key acting as a normal ADVANTAGE SHIFT key.

Feel free to adjust the ADE default meta-key translations to suit your own meta-key usage.

FUNCTION KEYS

The ADVANTAGE keyboard has fifteen function keys, F1 to F15. The PC keyboard has only twelve function keys, F1 to F12. To generate the equivalent of the three missing function keys, (F13, F14, F15) we use the ADE_META1 key plus F1, F2 and F3. Thus:

For F13, hold the ADE_META1 key down and hit F1
For F14, hold the ADE_META1 key down and hit F2
For F15, hold the ADE_META1 key down and hit F3

DIFFERENT UPPER AND LOWER CASE COMBINATIONS

There are two keys which differ in their combinations of upper and lower-case characters when hit.

Just use the SHIFT key or not as required to produce the character needed.

Note that this will only be a problem if you are trying to run the keyboard test program as given on the DEMODIAG-200.NSI disk image. In that case, hit SHIFT or not to produce the character needed instead of leaving the SHIFT key in the one position as directed by the keyboard test program.

CAPS LOCK KEY

The CAPS LOCK key works just the same in the ADVANTAGE as with the PC.

CURSOR LOCK KEY

The ADVANTAGE CURSOR LOCK does exactly the same function as the PC NUM LOCK. Except that they are named and are indicated in the opposite sense. The ADVANTAGE CURSOR LOCK is indicated as active when the Keypad keys ARE NOT in 'Numbers Mode', whereas the PC NUM LOCK is indicated as active when the Keypad keys ARE in 'Numbers Mode'.

ENABLE SCROLL LOCK KEY (OPTIONAL)

The Scroll Lock Key and its associated LED are often disabled in most Linux distros. In fact, many hardware manufacturers are omitting the Scroll Lock key and the associated LED from their keyboards altogether. My Lenovo laptop from 2021 has dispensed with them, whereas my Lenovo Desktop from 2016 has both.

If your keyboard does have the Scroll Lock LED, we can appropriate that LED to indicate when the ADVANTAGE Cursor Lock is active.

To do that, we will very likely need to enable the Scroll Lock LED. See this web page for a fuller discussion on enabling the Scroll Lock key and LED.

https://askubuntu.com/questions/127167/how-do-i-enable-scroll-lock

Basically, you adjust the file:

/usr/share/X11/xkb/symbols/zz

where the 'zz' indicates which keyboard configuration file is being used in *your* computer. In most cases with distros using the US 102-key keyboard, the 'zz' is just 'pc', so the file to adjust is most probably

/usr/share/X11/xkb/symbols/pc

but it also might well be

/usr/share/X11/xkb/symbols/us

The Scroll Lock key is designated as 'Modifier 3'. In many cases, that configuration line is simply omitted from the keyboard configuration file. To re-enable the Scroll Lock, we can re-insert the missing line in the

configuration file.

In the snippet below taken from /usr/share/X11/xkb/symbols/pc, we see that the modifier_map has Mod2 and Mod4, but no Mod3.

```
// Beginning of modifier mappings.
 modifier_map Shift  { Shift_L, Shift_R };
 modifier_map Lock   { Caps_Lock };
 modifier_map Control{ Control_L, Control_R };
 modifier_map Mod2   { Num_Lock };
 modifier_map Mod4   { Super_L, Super_R };
```

So we insert the missing Mod3 line, shown in **bold** below:

```
// Beginning of modifier mappings.
 modifier_map Shift  { Shift_L, Shift_R };
 modifier_map Lock   { Caps_Lock };
 modifier_map Control{ Control_L, Control_R };
 modifier_map Mod2   { Num_Lock };
 modifier_map Mod3   { Scroll_Lock };
 modifier_map Mod4   { Super_L, Super_R };
```

And then restart the X11 server to incorporate that extra information.

APPENDIX D.

CHANGES BETWEEN VERSIONS

Version 0.67 from Version 0.66

Changes to input modes of auxiliary ports, SIO and PIO, using the UNIX termios structure. Enables single character transfers, rather than whole-string transfers on inputs. Note that bidirectional I/O ports, such as /dev/ttyUSB0, will be need to be designated as the input port and also as the output port.

Version 0.66 from Version 0.65

Changes in organisation of development mode and distribution of public releases mode.

Version 0.65 from Version 0.64

Adds simulated ADVANTAGE keyboard, including arrow-keys and function-keys.
Now runs the most of the Keyboard Test as supplied with the DEMODIAG-200.NSI disk. Tests not passed are the multi-key rollover test as the ADE keyboard input buffer is very much larger than the 7-key queue in the ADVANTAGE. Also the repeat-key test as this is handled by the PC keyboard circuitry.

Version 0.64 from Version 0.63

Added SIO Input queue to allow better simulation of the SIO IO port.

Version 0.63 from Version 0.62

Debugged the display by allowing for a larger buffer for the Advantage Video RAM. Simulated use of SIO test jumper to allow the SIO Card test in the DEMODIAG-200.NSI disk to function properly.

Version 0.62 from Version 0.61

The changes are concerned mainly with more tidying of the source code. Along with automatic generalion of GTK+ resource files. Compiler warnings greatly increased, leading to cleaner header files.

Version 0.61 from Version 0.60

The Version 0.61 **ade** executable is not dependent on having several support files located in specified places. Thus the previous 0.60 version required two extra directories (‘glade’, ‘images’) within the user’s work-directory. Without having these directories of support-files restricting placement of the ‘ade’ executable to the work-directory, the ‘ade’ executable can now be placed in any directory within the user’s personal $PATH.

In keeping with current GNU/Linux usage, personal executables will be placed in the “/home/USERNAME/.local/bin” directory. If that directory is not already included in the user’s $PATH, it will be appended to the user’s $PATH in the ‘make install’ phase.