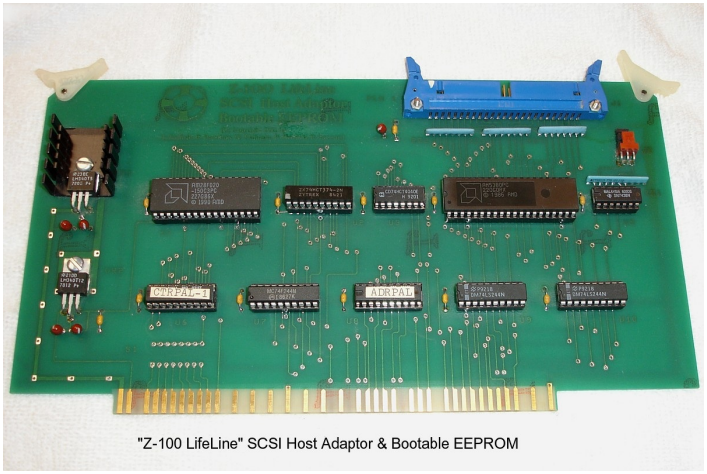
Z-100 LifeLine
Since 1989

March 2025

#WEB

This article was first published in issue #87, June 2003



"Z-100 LifeLine" SCSI Host Adaptor & Bootable EEPROM

"*Z-100 LifeLine*" SCSI Controller Manual

by Steven Vagts
Editor, "Z-100 LifeLine"

"*Z-100 LifeLine*" SCSI Controller Manual

SCSI Host Adaptor &
Bootable EEPROM Board

When Zenith and Microsoft support moved on from supporting the Z-100 series computer to developing PC hardware and software in the late 1980s, it quickly became apparent that any additional work on the Z-100 would have to be done within the Z-100 community. This was the driving factor for Paul Herman to begin publishing the "*Z-100 LifeLine*" in 1989. The purpose of the "*LifeLine*" was (and still is) to provide a central point for dissemination of information and development between venders, research teams and the ultimate users.

One of the first projects was the development of a new "*Z-100 LifeLine*" SCSI Host Adapter, also referred to as the LLSCSI Controller Board, to replace the aging "Winchester" MFM hard drives.

Originally conceived at the 1990 Z-100 Get-Together in Norfolk, Virginia, the SCSI/EEPROM board was a product of 1-1/2 years of research and development. Under the auspices of "*Z-100 LifeLine*", a development team was selected in November of 1990, and production units were first delivered in March of 1992.

A full team of volunteers began work on this project in 1990:

Paul F. Herman	Project Coordinator, EEPROM programming, Marketing
Robert F. Hassard	Engineering design, Prototype development
Robert W. Donohue	MTR-100 ROM and BIOS programming
William E. Flanagin	SCSI programming
Travis J. Barfield	Parts acquisition & Manufacturing
Michael Zinkow	Z-DOS development
John Beyers	BIOS programming & Z-DOS utilities

The following is a heavily modified portion of the original distribution documentation for the "*Z-100 LifeLine*" SCSI Host Adaptor/Bootable EEPROM Board. Publication here is for historical documentation and all rights are reserved.

Introduction:

The "*Z-100 LifeLine*" SCSI Host Adaptor/Bootable EEPROM Board, hereafter referred to as the LLSCSI/EEPROM Board, was a multifunction S-100 board designed for the Heath/Zenith Z-100 Series computer. It provided the following features:

- An industry standard **SCSI Host Adaptor.** This allowed you to connect fixed or removable media hard drives, tape backup units, CD-ROM drives, floptical drives, or any other device which included an imbedded SCSI controller, to your Z-100 computer.

Note: Software released with this board only supported fixed and removable media hard drives.

- A **Bootable EEPROM Device.** This non-volatile memory device, based on the AM28F020 flash programmable EEPROM, could be programmed at any time without removing it from this board. Programming software was provided with the board. The EEPROM device was fully bootable and could contain up to 256Kb of user selectable programs or files.

- A **Hardware Breakout Switch.** The breakout switch circuitry worked by generating a non-maskable interrupt (NMI) on the S-100 bus. Firmware to support the breakout switch for debugging was provided in the MTR-100 Monitor ROM, beginning with version 3.1.

System Requirements:

For proper use of all features, the LifeLine SCSI/EEPROM board required the following:

- An H/Z-100 computer. Use with other S-100 based computers was NOT supported and is unlikely to work.

- The EEPROM programming software, which was provided. This software required at least 448Kb of system memory.

- Monitor-ROM (aka MTR-ROM or ZROM) v3.1 up to ZROM v4.24 (more on this later), which had to be installed in order to boot from the EEPROM.

- Zenith's MS-DOS (aka ZDOS) v3.1 operating system, which was required to boot from the EEPROM and for the SCSI support software.

- Z-100 BIOS (IO.SYS) v3.10 or later. This BIOS included a device driver for the EEPROM device, plus other features not found in Zenith's BIOS. This BIOS was included with the SCSI/EEPROM board package.

Hardware Installation:

SyQuest SQ555 Cartridge Hard Drive and "Z-100 LifeLine" SCSI Controller

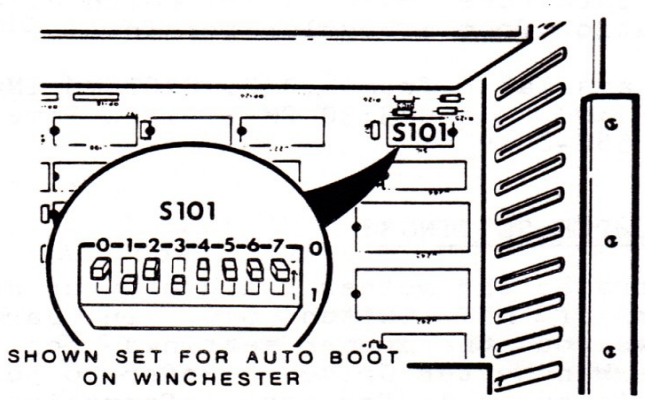
This section will describe the configuration used for testing the Z-100 LifeLine SCSI Controller board under ZDOS v3.10 and under ZDOS v4.06 (which updated the software for the LLSCSI board when the LLIDE Controller also came on the scene. We will use the SyQuest SQ555 SCSI drive with the LLSCSI controller.

The SCSI/EEPROM board required installation in accordance with the following steps:

CAUTION: Before doing anything to your Z-100 system, STOP and make duplicate copies of the software disks provided with this package. Put the originals in a safe place and use the copies.

1. Install the MTR-100 Monitor ROM, versions 3.1 to 4.24, by following the instructions in your Z-100 User's Manual. Be sure to set the mother-board jumpers J-101 and J-102 both to position 1 (bridging the pins nearest the left edge of the motherboard).

2. The motherboard DIP switch (located at the right rear corner of the motherboard, just forward of the S-100 Bus card cage) has several functions.



Z-100 Motherboard S101

The S101 switch sections are defined as:

Section:	Description:
0	Default boot device*
1	Default boot device*
2	Default boot device*
3	1= Auto boot, 0= Manual Boot
4	Not used
5	Not used
6	1= Dvorak keyboard, 0=Qwerty
7	0= 60 Hz, 1= 50 Hz operation

* Sections 0, 1, and 2 should be set to reflect the type of drive that the system is to boot from:

Section			Boot
0	1	2	Device Type:
0	0	0	5-1/4" Floppy Disk Drive
1	0	0	8" Floppy Disk Drive
0	1	0	Winchester Hard Disk Drive
1	1	0	EEPROM (SCSI) Device
1	1	1	NVsRAM (LLIDE) Device

For now, set S-101 to the following settings:

0	1	2	3	4	5	6	7
Off	Off	Off	Off	Off	Off	Off	Note

Note: The default boot device is now set for the 5" Floppy Drive. Set section 7 to OFF for 60 Hz operation, or ON for 50 Hz operation.

3. Set the jumpers on your SCSI hard drive for the correct logical unit number. Any number between 0 and 5 could be selected. Generally, your first drive will be unit 0, the second will be unit 1, etc. Unit number 6 is reserved for the SCSI Host Adaptor Board itself, and unit number 7 is reserved for a tape backup unit. The default device ID number on the SYQuest SQ555 is zero, so for us, no jumper is needed.

4. Install your SCSI hard drive(s). The installation details will vary and are left to you. Usually, today's small 3.5" drives should fit nicely somewhere on the inside of your Z-100's case. Power is generally provided via a standard drive power connector, similar to that used by a floppy drive. Be sure to follow the drive manufacturer's specifications when installing the drive.

Note: If you currently have a Z-217 controlled hard drive in your system, you may leave it installed. The SCSI/EEPROM board will coexist peacefully with the Z-217.

5. Install the SCSI/EEPROM Board in any available S-100 bus slot. If possible, leave at least one open slot in front of the board to aid cooling. I have NOT noticed any heating issues.

6. Mount the Breakout Switch in a convenient location so that it can be accessed with the case closed. Back-panel mounting should be fine for occasional use. Programmers who expect to make use of the breakout switch may want to connect it to a long cable, where it can be brought to the front of the machine when in use. Connect the breakout switch cable to connector J2 on the SCSI/EEPROM Board.

IMPORTANT NOTE: If you will NOT be using the breakout switch feature, you MUST place a shorting jumper over the LEFT two pins of connector J2 at the right side of the LLSCSI board.

7. Connect the 50-pin SCSI ribbon cable from the drive to the connector on the SCSI/EEPROM Board. Pin one is located toward the center of the board next to the logo. Double check to ensure you connect the cable correctly.

Testing the Hardware:

Use the following procedures to test your installation:

1. Turn the power on. You should hear the usual two beeps and get a hand prompt. If not, check the ROM installation, the settings of J-101 and J-102, and your cable connections.

Caution: DO NOT play with the monitor commands at this point. In particular, DO NOT issue the {**J**}ail command.

Note: The 'Jail' command (Monitor ROM v3.x) causes the CPU trap flag to be toggled. The trap flag is used to initiate single-step execution, and then using the '{**G**}o' command for a single step. To continue normal program execution, execute the 'Jail' command again to toggle the trap flag off. Beginning with MTR-ROM v4, the 'Jail' command was replaced by a much more comprehensive set of programming options.

2. Boot the system with MS-DOS v3.1 or later. You can boot from a floppy or a Z-217 controlled hard drive. The DIP switch has been set so the default boot device is the 5" Floppy Drive, so you can just press {**B**} for Boot, and press {**RETURN**}. Or you can manually select the boot device by using one of the following command sequences (If the device is installed):

{**B**}oot {**F1**} boots from the 5-1/4 inch floppy
{**B**}oot {**F2**} boots from the 8 inch drive
{**B**}oot {**F3**} boots from the Z-217 hard drive
{**B**}oot {**F4**} boots from the SCSI EEPROM, or
{**B**}oot {**F4**}{**p**} boots from the SCSI EEPROM
{**B**}oot {**F4**}{**s**} boots from the IDE NVsRAM

Where:
{**P**} or {**p**} is the Primary EEPROM device
{**S**} or {**s**} is the Secondary NVsRAM device

3. You should now be at the DOS prompt.

4. If you installed the breakout switch, try activating the switch. You should get a display of the CPU register contents, along with an unassembled assembly language instruction, followed by the MTR-100 hand prompt. If not, you may have installed the switch incorrectly.

5. Type the {**G**} monitor command. The word '*Go*' should be displayed on the screen. Now hit the {**RETURN**} key, and you should be back to the DOS prompt.

6. If you have made it this far, you can rest assured that your Z-100 is operating correctly, the new MTR-100 ROM is correctly installed, and the SCSI/EEPROM board is correctly installed.

7. Put the EEPROM Utilities Disk (your copy) in drive A. Run the **EEPTEST** program. If the test finishes successfully, your EEPROM device is working correctly.

Note: DO NOT run the EEPTEST after you have programmed the EEPROM device, since the test ERASES it!

8. Put the LLSCSI Utilities Disk (your copy) in drive A. Run the **LLINFO** program. Several screens full of data will scroll by very quickly. If the program finishes without crashing or issuing an error message, your SCSI host adaptor is working correctly and is able to communicate with the SCSI drive. If you do get an error message, or the system hangs, check your SCSI hard drive installation. Make sure the 50-pin cable is

connected properly, the drive has power, and the proper unit number is selected.

9. If you have passed all of these tests, your system is fully operational, and you are ready to continue programming the EEPROM device, and installing the SCSI software.

The Bootable EEPROM Device

Description:

The Bootable EEPROM portion of the SCSI/EEPROM Board was built around the AM28F020 flash programmable 256Kb EEPROM chip. It was organized as a non-volatile READ-only memory disk with 256 sectors of 1024 bytes each. A modified BIOS was provided (v3.10 or later) which included support for booting and reading the EEPROM device.

Once installed, the EEPROM device was assigned a drive letter and it could be read just like any disk drive (or a RAM disk). You could view the directory of files, run programs from the EEPROM device, or copy files and data FROM the EEPROM device.

However, you could not write to the EEPROM device in the normal fashion. It was considered by DOS to be a WRITE-PROTECTED device; any attempt to copy files to the EEPROM device would result in a disk error message.

Files and programs were installed on the EEPROM device by using special programing software provided with the SCSI/EEPROM Board. Once programmed, the EEPROM device could be bootable, and the contents of its memory would not be lost when power was turned off.

Programmers who would like to write their own software to access the EEPROM device should see the commented source files which were included on the EEPROM Utilities Disk. Additional information is also provided in the section titled "Theory of Operation", later.

Once all the tests have been completed and you are happy with the installation, if you wish to set the computer to autoboot to the EEPROM device, set the motherboard DIP switch S-101 to the following settings:

0	1	2	3	4	5	6	7
On	On	Off	On	Off	Off	Off	Note

EEPROM Utilities Disk

This disk was provided with the LLSCSI board and contained the programs needed to program the AM28F020 flash EEPROM device. All code was written by Paul F. Herman, and was released to the public domain for distribution with the SCSI/EEPROM Board. Full source code (in assembly language and Microsoft 'C' v5.1) was included.

Disk contents (For ZDOS v3.10):

AUTOEXEC	.BAT	For use on the EEPROM Setup Disk
CONFIGUR	.COM	Modified version of CONFIGUR for use with the v3.10 BIOS.
READ	.ME	Last minute info; manual changes
IO64W	.SYS	4 BIOS files which support
IO64	.SYS	the bootable EEPROM device.
IO128W	.SYS	
IO128	.SYS	
FLAGS	.COM	A public domain program by William C. Parke which allowed changing the file attributes of a file.
CONFIG	.SYS	For use on the EEPROM Setup Disk.
EEMDISK	.SYS	A RAM disk driver used during programming of the bootable EEPROM device.

Note: The EEMDISK was a 256Kb RAM disk which must be installed and prepared before running the PEEP.EXE program. The EEMDISK included a boot record required for proper booting of the EEPROM device after programming.

PEEP	.EXE	A utility to program the bootable EEPROM device. This utility looked for the EEMDISK RAM disk in memory, and then programmed the EEPROM device by copying all 256Kb from the RAM disk to the EEPROM device.
REEP	.EXE	A utility which reads from the EEPROM device. This utility was not required for programming the EEPROM device, but was provided as a way to read the EEPROM device memory. Output of REEP.EXE was a disk file (specified on the command line) which was 256Kb bytes long.
WEEP	.EXE	A utility which wrote to the EEPROM device. This utility was not required for programming the EEPROM device, but was provided as an alternate way of writing to the EEPROM device. WEEP.EXE will write a 256Kb byte disk image file (specified on the command line) to the EEPROM device.
EEPTEST	.EXE	A utility to test the EEPROM device. This utility erased, then programmed the EEPROM device with test data (ALL current data is DESTROYED). It then test read the EEPROM device using several methods.
SOURCE	<DIR>	This directory contained all the source code files for the EEPROM software utilities.

Programming the EEPROM Device - ZDOS v3.x

Note: There are several unique differences between creating a ZDOS v3.x LLSCSI Setup Disk and creating a ZDOS v4.x LLSCSI Setup Disk. Also, I have NOT been able to get the EEPROM Boot capability to work with a ZROM v4.3. Please see the section "Programming the EEPROM Device - ZDOS v4", for a complete description of my efforts, later.

First, you must create an EEPROM Setup Disk - for use whenever you want to program the EEPROM device. A directory named 'EEP' on this disk contains all the files you may want on your bootable EEPROM device.

1. Format a new BOOTABLE ZDOS v3.1 floppy disk using the command;

    FORMAT A:/s/v

2. Create a directory named 'EEP' on the EEPROM Setup Disk. This can be done with the command;

    MD A:\EEP

3. Copy the following files from the EEPROM Utilities Disk to the **ROOT** directory of the EEPROM Setup Disk;

    EEMDISK.SYS
    CONFIG.SYS
    AUTOEXEC.BAT
    PEEP.EXE

4. Copy the appropriate BIOS (IO.SYS) file from the EEPROM Utilities Disk into the 'EEP' directory of the EEPROM Setup Disk. There are four different BIOS files from which you can choose, depending on your desired system configuration:

IO64W.SYS	BIOS supported SCSI partitions up to 64Mb. Included support for Z-217 hard drives E thru H.
IO64.SYS	BIOS supported SCSI partitions up to 64Mb. Did NOT include Z-217 support.
IO128W.SYS	Supported SCSI partitions up to 128Mb. Included support for Z-217 hard drives E thru H.
IO128.SYS	Supported SCSI partitions up to 128Mb. Did NOT include Z-217 support.

Note: Do NOT use the 128Mb BIOS unless you must have partitions this large. The cluster sizes used for these large partitions is very wasteful of disk space. Even larger partitions (up to 512Mb) are possible by changing and reassembling the BIOS.

Note: Later versions of IO.SYS were distributed with later versions of ZDOS, up to ZDOS v4.06. Please contact me for a copy of the latest LLSCSI distribution software.

Note: Using a BIOS without the Z-217 support will save about 1.5Kb of memory. It will also allow use of the drive letters E through H for the EEPROM device and SCSI partitions.

Note: When you copy the appropriate BIOS file to the EEPROM Setup Disk, rename it to **'IO.SYS'**. It is important that IO.SYS be the **FIRST** file copied to the 'EEP' directory.

5. Make a copy of your ZDOS v3.1 distribution disk #1. This copy should NOT BE write-protected.

6. Use the FLAGS.COM program provided on the EEPROM Utilities Disk to remove all flags from the MSDOS.SYS file on the copy of your ZDOS distribution disk #1. This can be done by issuing the command;

    FLAGS A:MSDOS.SYS /

This step is necessary because the MSDOS.SYS file is normally a hidden system file and cannot be viewed or copied. Removing the 'hidden' flag from this file allows you to copy it from your EEPROM Setup Disk.

7. Copy the **MSDOS.SYS** file from the ZDOS disk to the 'EEP' directory of your EEPROM Setup Disk. This should be the **SECOND** file in the directory.

8. Copy the **COMMAND.COM** file from the MS-DOS disk to the 'EEP' directory of your EEPROM Setup Disk. It should be the **THIRD** file in the directory.

9. Run the command **DIR** on the \EEP directory. You should now have three files in the 'EEP' directory on your EEPROM Setup Disk... IO.SYS, MSDOS.SYS, and COMMAND.COM. They MUST be in that order! If not, go back and start over.

10. Technically speaking, these three files are all that you will need to make the EEPROM device bootable. If you like, you may proceed with the programming procedure to see how it works, and then add more files to your bootable EEPROM device at a later time. Sooner or later, you will almost certainly want to include the following files on your bootable EEPROM device;

AUTOEXEC.BAT	- Specifies automatic startup procedure
CONFIG.SYS	- Specifies device drivers to load, etc.

You may also want to include the device drivers you plan to install at boot up time. These may include a driver for the SCSI device, a floppy device driver, an ANSI device driver, etc. You may also include utilities which are needed by the AUTOEXEC or CONFIG files. For instance, memory resident utilities that you load at boot time.

You may add as many files as you like to your bootable EEPROM device, with the following limits:

- Maximum 64 directory entries (including IO.SYS and MSDOS.SYS)
- Maximum 256Kb of files (including boot record and system files)

11. Before using the EEPROM Setup Disk to program the EEPROM device, you may want to configure the IO.SYS file using the MS-DOS CONFIGUR.COM program. To do this, change to the 'EEP' directory, using the following command;

CD \EEP

Now invoke the CONFIGUR.COM program.

Note: If you attempt to use Zenith's original CONFIGUR.COM Program, you may encounter two problems:

- It may complain that this is the wrong BIOS version. A new version of CONFIGUR.COM is included with each new ZDOS release. Do NOT mix versions of these programs.

- The IO.SYS file must ALWAYS be located as one of the three systems files; IO.SYS, MSDOS.SYS, and COMMAND.COM, located first and in that order in the ROOT directory.

Therefore, if you need to configure your new BIOS, copy the appropriate IO.SYS file into the ROOT directory, run CONFIGUR.COM, then place it in the EEP directory of the EEPROM Setup Disk.

Finally, the AUTOEXEC.BAT file on the distribution disk ran and programmed the EEPROM each time the disk was booted. The AUTOEXEC.BAT file contained the lines:

COPY EEP I:
PEEP

Rather, you may wish to rename this file to EPROM_PG.BAT, so you can have more control, allow changes to the \EEP directory, and make changes to the EEPROM less automatic. Remember, some file changes will require a fresh reboot before reprogramming the EEPROM.

Using the EEPROM Setup Disk to Program the EEPROM Device

12. Now re-boot your Z-100 from the EEPROM Setup Floppy Disk. Run the EPROM_PG.BAT file to program the EEPROM device:

- The EEMDISK 256Kb RAM disk is installed when the EEPROM Setup Disk is booted.
- All files are copied from the 'EEP' directory to the RAM Disk.
- The PEEP.EXE program is used to program the EEPROM device by writing the memory image of the RAM disk into the EEPROM device.

13. This completes the EEPROM programming procedure. Put your EEPROM Setup Disk away as a backup, in case you need to reprogram the EEPROM device in the future.

Whenever you need to reprogram your EEPROM device, simply modify the files in the 'EEP' directory of the EEPROM Setup Disk, if needed, then reboot the EEPROM Setup Disk and run EPROM_PG.BAT.

Booting from the EEPROM Device

Now you should be able to boot from the EEPROM device. This can be done by pressing {**B**}oot {**F4**} and {**RETURN**} at the hand prompt.

If you wish to make the EEPROM device the default boot device, set the motherboard S101 switch to:

0	1	2	3	4	5	6	7
On	On	Off	Off	Off	Off	Off	Note

Note: If you would like your Z-100 to boot automatically when you turn ON the power or do a {**Ctrl**}-{**RESET**}, put S-101 section 3 to the ON position.

Special Note: The Am28F flash EEPROM devices used in the bootable EEPROM device may be reprogrammed up to 10,000 times. This should be ample to allow you to change your boot files any time you like, as system requirements change.

The SCSI Host Adaptor

The SCSI Host Adaptor portion of the LifeLine SCSI/EEPROM Board was designed around the NCR-5380 SCSI chip. It provided an industry-standard interface to devices which include an imbedded SCSI controller.

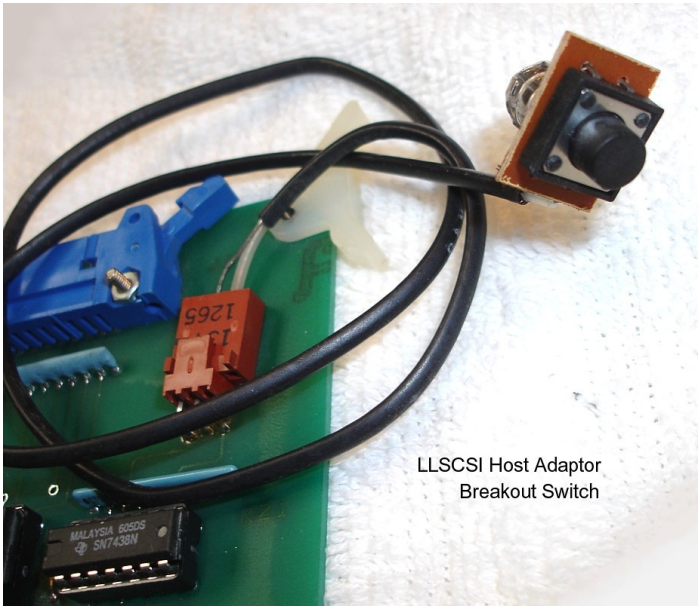
Software (LLSCSI Device Driver and Utilities) was provided to support fixed and removable media SCSI hard disks, such as Seagate, Rodime, and Quantum SCSI hard drives and SyQuest removable cartridge SCSI hard drive. Hard drives from other manufacturers may also work.

The LLSCSI software supports drives with capacities up to 512 megabytes. Your SCSI drive may have from one to 16 partitions. Each partition may be up to 512Mb in size, although partitions larger than 128Mb are considered non-standard, and require the BIOS to be changed and reassembled.

Please see the section, "**LLSCSI DEVICE DRIVER & UTILITIES**", later, for the procedures to prepare the SCSI drive of your choice for use.

The NMI Breakout Switch

The breakout switch portion of the LifeLine SCSI/EEPROM Board is a tool for programmers. It allows you to break out of any executing program, perform various debugging chores, and then continue execution. The switch works by generating a non-maskable interrupt (NMI) on the S-100 bus.



In order to use the breakout switch for its intended purpose, you need special software to support it (a non-maskable interrupt routine, to be specific). The MTR-100 Monitor ROM v3.1 and later includes debugging capability which utilizes the NMI breakout switch. I have added an article regarding the use of a Breakout Switch on the "LifeLine Website" with details about how the switch was used with the ROM.

Use of the breakout switch was not recommended unless you were a programmer who understood assembly language programming. If you do not fall into this category, you may not want to install the breakout switch.

IMPORTANT: If you do not install the breakout switch, a shorting jumper must be placed over the two pins of J2 nearest the center of the board. Otherwise, your Z-100 may fail to operate correctly.

Programming the EEPROM Device - ZDOS v4.x

Note: As I mentioned earlier, there are several unique differences between creating a ZDOS v3.x LLSCSI Setup Disk and a ZDOS v4.x version.

Note: I have NOT been able to get the EEPROM Boot capability to work with a ZROM v4.3. Further, if you were hoping to use ZDOS v4.06, it requires ZROM v4.3 to work. So that pretty much puts the kibosh on using the newest ZDOS.

Ever hopeful, I will briefly list the procedures for creating the ZDOS v4.06 LLSCSI Setup Disk, then explain the testing that I went through in the hope that one of you may have the solution.

If I hear of a solution from someone, I will certainly pass it on with an update here.

As before, to create an EEPROM Setup Disk to reprogram the EEPROM device, you must create a directory named 'EEP' on this Setup disk that contains all the files you may want on your bootable EEPROM device. Briefly, you must:

1. Boot to ZDOS v4.06, from a floppy or the hard drive, and FORMAT a new BOOTABLE ZDOS v4.06 floppy disk containing the ZDOS v4.06 versions of IO.SYS, MSDOS.SYS, and COMMAND.COM. Use the command;

    FORMAT A:/s/v

2. Run the DRIVECFG.COM utility on drive A: to create a new EEPROM drive. I set my drive configuration to include 4 floppy drives, A:-D:, four MFM hard drives, E:-H:, an imaginary drive I: for drive A:, and select J: to be the EEPROM device.

When you press drive letter, **J:**, the utility will present a list of drive choices. If you press {**F11**} for an EPROM device, the computer displays:

    "Select (S)CSI or (I)DE Controller Card [CR = SCSI]... _"

Press {**RETURN**} or {**S**}{**RETURN**} for the SCSI controller.

Save and Exit the program. Your IO.SYS file is now ready for use.

3. Create a directory named 'EEP' on the EEPROM Setup Disk. This can be done with the command;

    MD A:\EEP

4. Copy the IO.SYS file you just modified from the ROOT directory into the 'EEP' directory of the EEPROM Setup Disk. This must be **FIRST** file in the directory.

5. Copy the **MSDOS.SYS** file from the Root directory to the 'EEP' directory of your EEPROM Setup Disk. This must be the **SECOND** file.

6. Copy the **COMMAND.COM** file from the Root directory to the 'EEP' directory of your EEPROM Setup Disk. It must be the **THIRD** file.

7. Run the command **DIR** on the \EEP directory. You should now have three files in the directory - IO.SYS, MSDOS.SYS, and COMMAND.COM. They MUST be in that order! If not, go back and start over.

8. Copy the following files from the EEPROM Utilities Disk to the **ROOT** directory of the EEPROM Setup Disk:

EEMDISK.SYS
CONFIG.SYS
AUTOEXEC.BAT
PEEP.EXE

9. Rename AUTOEXEC.BAT to EPROM_PG.BAT, as I mentioned earlier. You must also modify the contents to change the drive letter to reflect whatever drive letter you chose for programming the EEPROM. On mine, I had to change the drive letter to 'K:' for programming my EEPROM, so my EPROM_PG.BAT file contains:

COPY EEP K:
PEEP

10. In addition to your usual files and buffers lines, CONFIG.SYS in the Root directory should contain the lines;
DEVICE=EEMDISK.SYS
DEVICE=LLSCSI.SYS

11. Copy LLSCSI.SYS and the CONFIG.SYS file to the \EEP directory as LLSCSI.SYS must be run when the EEPROM boots, then delete the "DEVICE= EEMDISK.SYS" line from this copy of CONFIG.SYS in the \EEP directory. It will not be needed.

12. You may also want to include other programs, ZDIR.COM, a clock routine like ZCLK.COM, or maybe greeting routines or screens, such as my HELLO.SCN, that would run at boot time.

You may add as many files as you like to your bootable EEPROM device, with the following limits:

- Maximum 64 directory entries (including IO.SYS and MSDOS.SYS)
- Maximum 256Kb of files (including boot record and system files)

After creating both DOS versions, 3 & 4, of my LLSCSI Setup Disks, I was ready for testing...

ZROM v4.3 FAILS!!

OK, I have done everything that I can think of, but no matter what I try, I can NOT get the EEPROM to boot while the ZROM v4.3 is installed!

Everything works great with the older versions, ZROM v3.2 and v4.24. The EEPROM boots great, it comes up with my opening graphics screens correctly, the SCSI drive partitions are all found and all disk drive operations are flawless.

The stage is set...
As I needed to check out this LifeLine SCSI Host Adaptor for this article, I chose the existing hardware, running with the ZROM v4.3 and ZDOS v4.06 on a new motherboard running first at 9.3MHz, but then slowed to 5.3MHz after the unexpected failures.

I was using a SyQuest SQ555 44Mb SCSI cartridge hard drive which had been fully tested on this test bed computer while using the CDR-317 SCSI Controller. These tests are described in a separate article available on the "*Z-100 LifeLine*" website.

Before I even began, I wanted to ensure that the setup worked. So, I ran the EEPROM test utility, EEPTEST.COM, before doing anything else;

A:\>EEPTEST {CR}

The computer responded with:

"EEPTEST v1.11 Test EEProm Memory
(C)Copyright 1991 by Paul F. Herman Inc."
IF YOU CONTINUE, YOUR EEPROM DISK WILL BE ERASED! Are you sure you want to continue (Y or N)?" **Y**

"Erasing EEPROM disk ... This will take a few seconds ... Okay."
"Programming EEPROM ... Okay."
"Reading EEPROM (Hi-Lo test) ... Okay."
"Reading EEPROM (Sequential test) ... Okay."
"A:\>"

So, this showed the EEPROM was working great.

The new LLSCSI SETUP Disk with DOS v4.06 must be rebooted to make sure the new IO.SYS was working and configured the computer for programming the EEPROM. It showed drive J: was the EEPROM on the SCSI controller and drive K: was set for programming the EEPROM, specifically:

"J: EEPROM on SCSI LifeLine Board p"
"K: 256K EEMDISK (EEPROM programming)"

"MTR ROM v4.3 768K RAM 64K COLOR Video
8088 9.3446 MHz CPU Speed"

"Using default CONFIG.SYS optn."
"DSKPAK SCSI Device Driver Version 1.00"
"Copyright (C)1991, William E. Flanagin,"
"Licensed for Sale by Paul F. Herman Inc."
"All Rights Reserved"

If a memory cartridge was installed in the SyQuest SQ555 drive, the LLSCSI.SYS driver would be installed. Otherwise, you get the message;

"No Drives Found Configured ERROR"
"Driver NOT Installed Due To Prior ERROR"

NOTE: I experienced NO problems with the SyQuest SQ555 cartridge drive at any time. It worked flawlessly with any ZROM & ZDOS combination. So I will not be referencing the driver any further in this discussion. All difficulties were with recognizing and reading the EEPROM.

I programmed the EEPROM using the command:

EEPROM_PG.BAT

which also worked flawlessly, and confirmed the files were indeed on the EEPROM by listing the drive directory using the command;

ZDIR J:/a

I would next complete a Warm Boot, using the key combination {**CTRL**}-{**RESET**}, and would try to boot to the EEPROM with the command;

{**B**}oot {**F4**}

The computer would display:

*"Input BOOT string <CR>**f4**"*
"Booting Primary EEPROM Unit 0"
"(C)1992 Z-100 LifeLine v1.15,
Booting EEPROM Device ..."

And then stall. If I press the Breakout Switch, the computer continues with:

"No System"
" AX BX CX DX SP BP SI DI
"07FF 002E 0000 F808 0200 0000 0517 BC50
DS SS ES ODITSZ A P C"
1400 1FA0 E000 1111000001000110"

"1400:04F1 EBFE JMP 04F1"

I will not bother listing the code here, but I can make it available, if anyone is interested. However, the jump is intentional, it is not an error. The code is looking for the File FAT (File Allocation Table) on the EEPROM and can not find it, so assumes there is no system, reports that fact, then does an endless loop until someone resets the computer.

According to the source code of EEMDISK.ASM, several memory SEGments are used with the program:

ROMDATA begins at ORG 0000:03FE
BIOS begins at ORG 0040:0000
BOOTRUN or LOADER is at ORG 1400:0400

If these SEGments are working properly, each show the location of the File FAT. Let me summarize what is found where:

SEGment:	**ZROM v4.24**	**ZROM v4.3**
0000:042B	EEMDISK	Yes (ok)
0000:05B6	(C)1992 Boot Msg	Yes (ok)
0000:0C00	File FAT	NO!
0040:002B	EEMDISK	Yes (ok)
0040:01B6	(C)1992 Boot Msg	Yes (ok)
0040:0800	File FAT	NO!
1400:042B	EEMDISK	Yes (ok)
1400:05B6	(C)1992 Boot Msg	Yes (ok)
1400:0C00	File FAT	NO!

As I understand the code, the BOOTRUN or BOOT LOADER from the last part of EEMDISK.SYS is loaded into memory SEGment 1400 from the contents of the EEPROM in the following order:

- The portion of the EEMDISK.SYS device driver that creates and runs the Boot Loader is loaded first into the memory at SEGment 1400:0400. This contains the applicable messages to be displayed on the screen while booting.
- The "PC" boot code is entered next.
- The final part is the File FAT.

If the File Fat can not be found, then there is no system, the BOOTRUN or LOADER (SEG 1400) is not created, and the boot process halts.

As I ran into difficulty, I tried slowing to 5MHz, then eventually tried the earlier ZROM v4.24 and ZDOS v3.10, looking for a working system. After all, something must have worked for the developers, or the system would have never made it to distribution! Remember, this LLSCSI Controller was available long before the IDE controller, ZROM v4.3 and ZDOS v4.06 were developed.

Anyway, if the EEPROM was programmed using ZDOS v4.06 with ZROM v4.3, but I replaced the ROM with ZROM v4.24, the EEPROM would attempt to boot, would display the v4.06 opening screen showing the drive configuration table, but would fail, after displaying the drive table with;

(Drive Table Listing)
"I: Imaginary Drive mapped to A:"
"J: EEPROM on SCSI LifeLine Board p"

And below the table;
*"** Initializing Motherboard Parity **"*
"Version mismatch between BIOS and system ROM"
(Hand prompt)

So, with the only difference being the ROM was now ZROM v4.24, the File FAT must have been found on the EEPROM and the EEPROM attempted to boot, but was stopped because of the BIOS & ZROM version mismatch!

I had to floppy boot to the ZDOS v3.10 EEPROM Setup Disk and reprogram the EEPROM. Then I found that the system worked great using ZROM v4.24 and ZDOS v3.10. The correct, working bootup using EEMDISK.SYS v1.11 goes like this:

*"{**B**}OOT {**f4**}"*
"BIOS version 3.10"
"Drives E: - D: floppies on the Z207"
"Drives E: - H: hard disk partitions on the Z217"
"Drive I: is the ROM drive"
"DskPak SCSI Device Driver version 1.00"
"Copyright (c)1991, William E. Flanagin,"
"Licensed for Sale by Paul F. Herman Inc."
"All Rights Reserved"

"Number of logical drives configured 2"
"Starting with logical drive J: through"
Logical drive K: respectively"

"Ms-DOS Version 3.10"
"Copyright (c)1985, Zenith Data Systems Corporation"
"I>ECHO OFF
"Strike a key when ready . . ."

My graphics screen, HELLO.SCN is displayed fine. The ZBE ZCLK Date & Time is displayed.

"Strike a key when ready . . ."

ZDIR of the EEPROM root directory is displayed; with the label 'SCSI EEP.ROM'.

"I:\>"

Note: The primary differences between v1.11 and the later versions of EEMDISK.SYS, is that the "PC" boot code was introduced with v1.13 and the IDE capability was introduced with v1.15.

For EEMDISK v1.13, the screen display is:
"{B}oot {F4}"
"(C)1992 Z-100 LifeLine v1.13,
Booting EEPROM Device . . ."

Then the rest is the same as for the version 1.11, beginning with *"BIOS Version 3.10"*.

And for EEMDISK v1.15, the screen display is:
"{B}oot {F4}"
"(C)1992 Z-100 LifeLine v1.15,
Booting EEPROM Device . . ."

Then, again, the rest is the same as for the previous versions.

In spite of the differences in EEMDISK.SYS versions, the above test results were the same on both ZROM v3.2 and ZROM v4.24.

Final Food for Thought...

As you can see from my selection of partition names during the preparation of the SCSI cartridge above, I am in the habit of making a separate partition for each of the operating systems that I may use with a hard drive, so I created ZDOS3 (40%), ZDOS4 (50%) and CPM (10%). However, these SCSI drives are different in that they will NOT be bootable - the EEPROM does that.

That creates several new considerations that I did not worry about with my MFM hard drive systems. IDE devices with the IDE Controller had the same restrictions.

* The EEPROM can only be programmed to boot one operating system! It has to be reprogrammed to boot to another OS.

* And speaking about my CP/M partition, is it even possible to program the EEPROM device to boot CP/M? While CP/M does not use a directory structure that we use on our Setup Disk, would it be possible to load the CP/M files into the DOS \EEP directory, perhaps using a READCPM utility, then boot to the Setup disk in DOS to load the \EEP data into the EEPROM? The EEPROM certainly would not care what the operating system was. Data is just 00's and 11's right?

* Then, there is the SCSI drive itself. Like the EEPROM and any other storage devices, such as an MFM drive, it does not care what operating system was in use? It just stores data. Has one of you geniuses already thought of how this could be accomplished?

* While programming the EEPROM is easy and a separate setup disk could easily reprogram the EEPROM for different operating systems, it would still be confusing and a process to be avoided. You may as well just keep a separate computer for each operating system with which to work!

* The LLSCSI device driver detects all three partitions and creates 3 drives available for use by the operating system that was booted. It does not care if one was for CPM. Intermixing files would be a real problem, especially between ZDOS3 and ZDOS4!

* At 42Mb, you will need to figure out how you want to divide the partitions into more manageable sizes; perhaps by use - Spreadsheets, Word Processing, Database, Taxes, etc., or directory listing programs will show PAGES of files with every use. And I have a pair of 88Mb cartridges!

Theory of Operation

The LifeLine SCSI/EEPROM Board serves three functions; a SCSI host adaptor, a bootable in-circuit programmable EEPROM, and a breakout switch.

First, a description of the overhead. U9 and U10 are tri-state drivers which provide an on-board bi-directional data bus. The on-board bus interfaces to the S-100 data-in and data-out lines. The bi-directional data bus serves both the EEPROM (U1) and the SCSI host adaptor (U4), plus the sector latch (U2), which is discussed below.

U2 acts as a sector latch, allowing any of the EEPROM's 256K sectors to be accessed. U6 (CTRPAL) is a programmable logic array which synthesizes the proper timing and enable signals. U8 (ADRPAL) is another PAL which, along with U7, provides port address decoding. U3 is a sector offset address counter which allows successive bytes to be read from the sectors of the EEPROM. And finally, U5 provides debouncing for the non-maskable interrupt switch.

Communications with the SCSI/EEPROM board is done with I/O ports. The SCSI host adaptor uses ports 0C0h through 0C7h to read or write its registers, and port 0C8h for data in/out. The EEPROM uses port 0CEh for reading data or writing to the sector latch, and port 0CFh for writing data or incrementing its address counters.

These port assignments were carefully chosen to avoid conflicts. Should there be a conflict, the port addresses could be changed by reprogramming the ADRPAL and rewriting the software. It might be easier to resolve the other party to a conflict.

The EEPROM is organized like a pseudo disk, into 256 sectors of 1024 bytes each, for an overall total of 256Kb bytes. A sector is read by first loading the sector latch (port 0CEh) with the sector number, and then reading port 0CEh 1024 consecutive times. After each read from port 0CEh, the sector offset address counter (U3) must be incremented by reading from port 0CFh. The data read from port 0CFh is meaningless... the read is only done to increment the counter. The sector offset address counter is reset by a write to the sector latch.

Writing to the EEPROM (programming) can also be done without removing the EEPROM chip from the board. This is made possible by the flash programmable technology of the Am28F020 chip. However, programming of the EEPROM involves critical timing operations which are beyond the scope of this discussion. See the source code for the EEPROM Utilities for an example of how the EEPROM is programmed.

CTRPAL Equations

This PAL creates the control signals to read and write to the EEPROM and SCSI host adaptor chip. It also creates the signal to increment the sector offset address counter. This device may be a PAL-16L8 or GAL-G16V8.

INPUTS:

```
Pin 1   = pWR    CPU write strobe (active low)
Pin 2   = pDBIN  CPU read strobe (high)
Pin 3   = A0     CPU address
Pin 4   = C0X    from ADRPAL
Pin 5   = C8     from ADRPAL
Pin 6   = NCLR   from Pin 15 (/WEN0)
Pin 7   = CEF    from ADRPAL
Pin 8   = sOUT   CPU OUT to PORT signal (high)
Pin 9   = sINP   CPU IN from PORT signal (high)
```

OUTPUTS:

```
Pin 12  = INCR   signal to increment sector
                 offset address counter
Pin 13  = REN    EEPROM read strobe (low)
Pin 14  = WEN1   EEPROM write strobe (low)
Pin 15  = WEN0   Address latch write strobe (low)
Pin 16  = PREN   EEPROM enable (low)
Pin 17  = CLR    Sector counter reset (high)
Pin 18  = IOR    SCSI read strobe
Pin 19  = IOW    SCSI write strobe
```

Logic Equations:

```
/IOW    = /pWR * /C0X * sOUT + /pWR * /C8 * sOUT
/IOR    = pDBIN * /C0X * sINP + pDBIN * /C8 * sINP
/CLR    = NCLR
/PREN   = /CEF * /A0 * sINP + /CEF * A0 * sOUT
/WEN0   = /CEF * /A0 * /pWR * sOUT
/WEN1   = /CEF * A0 * /pWR * sOUT
/REN    = /CEF * /A0 * pDBIN * sINP
/INCR   = /CEF * A0 * sINP * pDBIN
```

Note: Signals preceded by a forward slash represent low signals.

ADRPAL Equations

This PAL deciphers board addresses to create enable signals for CTRPAL, the SCSI host adaptor chip, and the data bus multiplex control. This device may be a PAL-16L8 or GAL-G16V8.

INPUTS:

```
Pin 1   = A0     System address bus
Pin 2   = A5     System address bus
Pin 3   = A1     System address bus
Pin 4   = A4     System address bus
Pin 5   = A2     System address bus
Pin 6   = A3     System address bus
Pin 7   = A6     System address bus
Pin 8   = A7     System address bus
Pin 9   = sOUT   I/O out indicator
Pin 11  = sINP   I/O in indicator
```

OUTPUTS:

```
Pin 12  = RDEN   Data bus read strobe (active low)
Pin 13  = WREN   Data bus write strobe (low)
Pin 14  = SCICS  Select SCSI control registers
Pin 15  = DACK   Select SCSI data registers
Pin 16  = C8     Port 0C8h register
Pin 17  = C0X    Ports 0C0h through 0C7h registers
Pin 18  = CEF    Address 0CEh or 0CFh (low)
```

Logic Equations:

```
/CEF    = A7 * A6 * /A5 * /A4 * A3 * A2 * A1
/C0X    = A7 * A6 * /A5 * /A4 * /A3
/C8     = A7 * A6 * /A5 * /A4 * A3 * /A2 * /A1 * /A0
/DACK   = /C8 * sOUT + /C8 * sINP
/SCICS  = /C0X * sOUT + /C0X * sINP
/WREN   = /CEF * sOUT + /C0X * sOUT + /C8 * sOUT
/RDEN   = /CEF * /A0 * sINP+/C0X * sINP+/C8 * sINP
```

Note: The signal /DACK had a pencil change in my copy of the written manual that appeared to change the equation to +/C8 * sOUT + A0. We need to check this out.

Note: Signals preceded by a forward slash represent low signals.

Engineering Notes:

Heat Dissipation of VR1 (5 volt regulator)

Regulated 5 volt power to the LifeLine SCSI /EEPROM Board is provided by voltage regulator VR1. Typically, the input voltage to this regulator is about 8 to 8.5 volts, and must be a minimum of 7 volts. The current requirements of the board are about 0.5 amp.

It is normal for VR1 to be hot to the touch. However, it has been found that some Z-100 power supplies provide as much as 11 volts input to this regulator, which may cause it to run hotter than normal. Even at 11 volts input, VR1 is still operating within its design criteria. However, the increased temperature may result in decreased life.

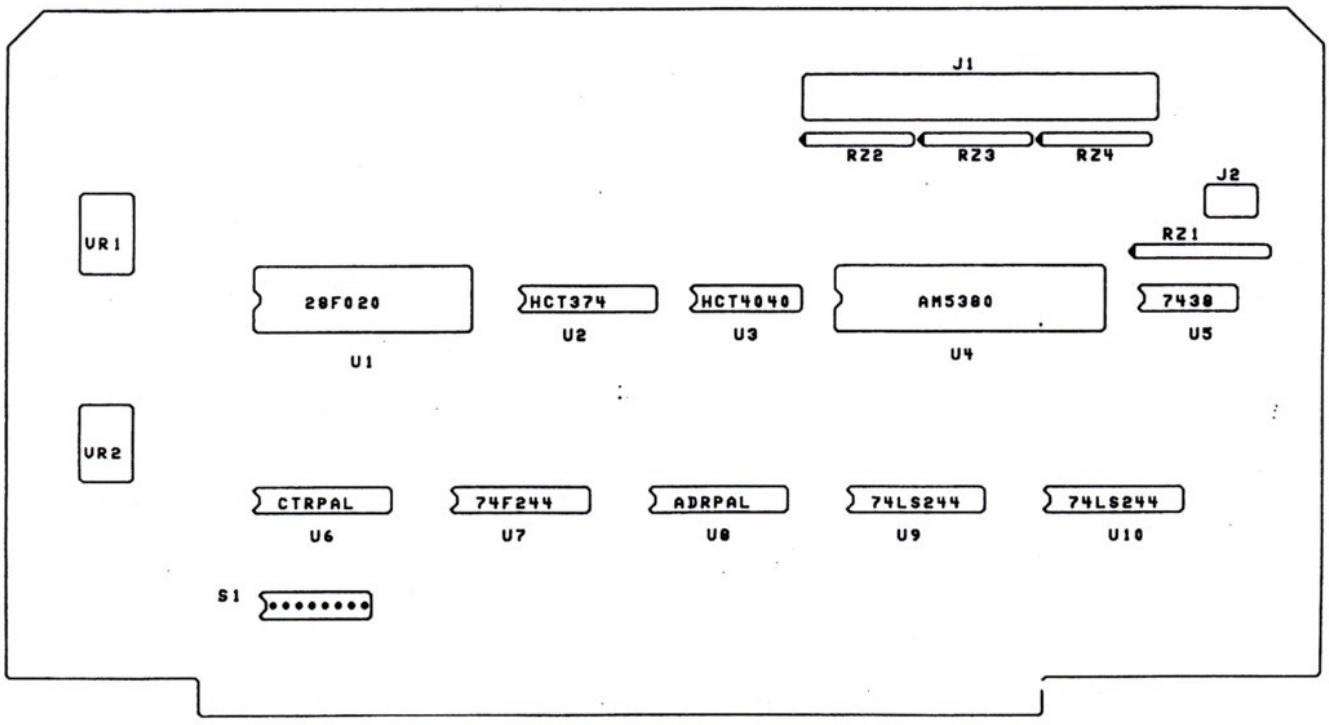
If VR1 gets hot enough to burn your finger, you may want to consider making a modification to the board. Only attempt this modification if you have a voltmeter, and are experienced in electronic repairs.

Measure the voltage at the input to VR1. If the voltage is above 9 volts, insert a 2.2 ohm, 2 watt resistor between the input and the regulator. This can be done by cutting the input line trace (along the left side of the board), and soldering the resistor in the holes which are provided. if the input voltage measures greater than 10 volts, you may want to consider adding two 2.2 ohm resistors in series with each other.

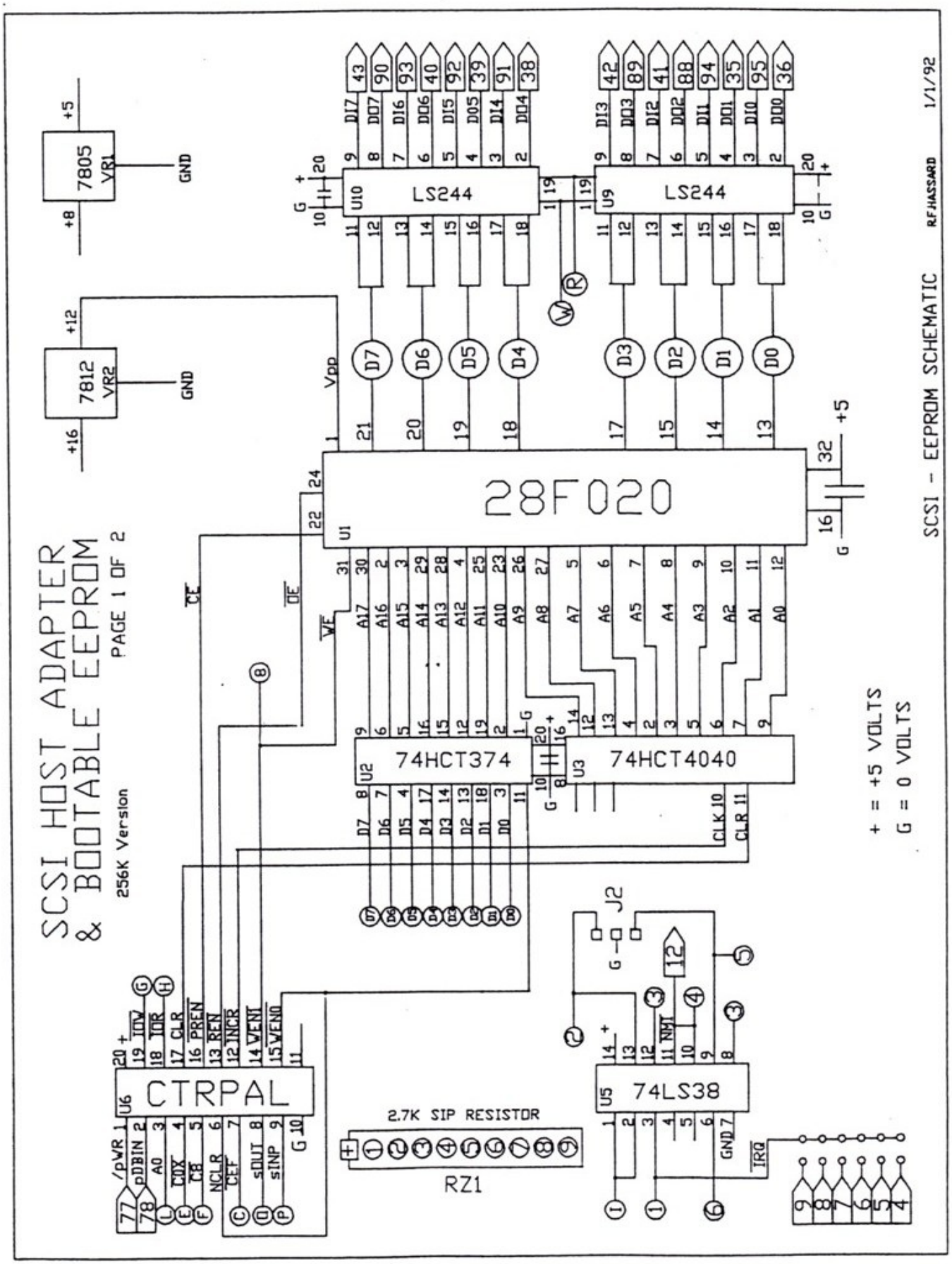
~~~~~~

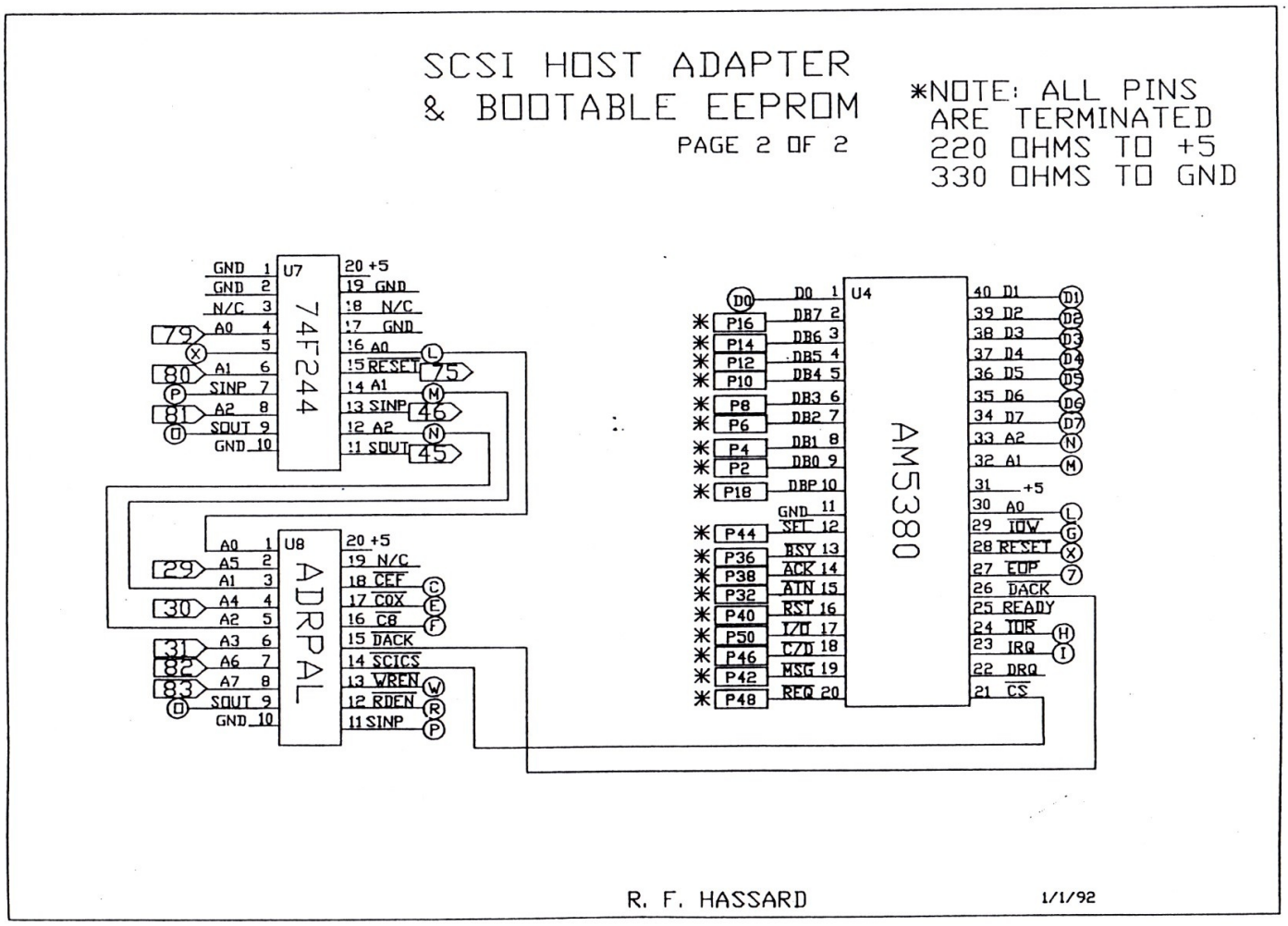
Parts List:

Qty:	Part #:	Description:	Digikey Part #:
1		Custom S-100 printed circuit board	
1	U1	Am28F020-150 flash programmable EEPROM	
1	U4	Am5380-PC SCSI host adaptor chip	
1	U3	74HCT4040 12 bit binary counter	CD74HCT4040E
1	U2	74HCT374 octal D tri-state flip-flop	CD74HCT374E
1	U5	74LS38 quad 2 in NAND buffer	DM74LS38N
1	U6	Custom CTRPAL PAL16L8-15CN	
1	U7	74F244 quad buffer/line driver	74F244PC
1	U8	Custom ADRPAL PAL16L8-15CN	
1	U9	74LS244 tri-state octal line driver	DM74LS244N
1	U10	74LS244 tri-state octal line driver	DM74LS244N
1	VR1	7805 5V, 1.5A voltage regulator	LM340T-5
1	VR2	7812 12V, 1.5A voltage regulator	LM340T-12
1	RZ1	2.7K ohm, 10-pin SIP resistor network	770-101-R2.7K
3	RZ2-RZ4	220/330 ohm, 8-pin SIP resistor network	770-85-R220/330
5		6.8uF, 25v tantalum capacitor	P2048
11		0.1uF, 25v capacitor	P4887
1	J1	50-pin SCSI connector	CHR50G-ND
1		Lock ejector set	CLKS01-ND
1	J2	3 contact header	WM4301
1		3 contact terminal housing	WM2001
1		Shorting jumper	929950-00-ND
1		SPDT momentary contact switch	
1		Heat sink for TO-220	
1		14-pin DIP socket	A9314
1		16-pin DIP socket	A9316
1		20-pin DIP socket	A9320
1		32-pin DIP socket	ED3632
1		40-pin DIP socket	A9340
2		6-32 x 1/4 machine screw and hex nut	H154, H220
2		#6 lock washers	H240
2		2-56 x 5/16 machine screw and hex nut	H155, H212



IC Placement on the LLSCSI/EEPROM Board
~~~~~~

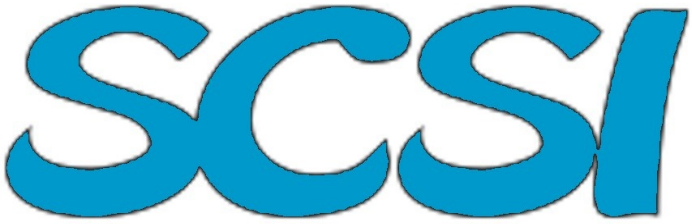

SCSI HOST ADAPTER
& BOOTABLE EEPROM
PAGE 1 OF 2
256K Version
SCSI - EEPROM SCHEMATIC
RF HASSARD
1/1/92
28F020
74HCT374
74HCT4040
LS244
LS244
CTRPAL
74LS38
7805
VR1
7812
VR2
2.7K SIP RESISTOR
RZ1
+ = +5 VOLTS
G = 0 VOLTS



SYMBOLS CROSS REFERENCE

There are 26 symbols used in the schematics to show signal connections between pages, or, in some cases, from point to point on the same page. Because of the complexity of so many symbols, the following table may help.

Symbol	Signal Name	Active	Pages used on
A	WEN0	LO	1
B	WEN1,WE	LO	1, 2
C	CEF	LO	1, 2
D	D0-D7 Data		1, 2
E	C0X	LO	1
F	C8	LO	1, 2
G	IOW	LO	1, 2
H	IOR	LO	1, 2
I	IRQ	HI	1, 2
J	IRQ	LO	1
K	DACK	LO	2
L	A0	HI	1, 2
M	A1	HI	1, 2
N	A2	HI	1, 2
O	SOUT	HI	1, 2
P	SINP	HI	1, 2
Q	INCR,CLK	LO	1
R	RDEN	LO	1, 2
S	SCICS,CS	LO	2
T	REQ	LO	2
U	REN,OE	LO	1
V	PREN,CE	LO	1
W	WREN	LO	2
X	RESET	LO	2
Y	NMI	LO	1
Z	CLR	HI	1



LLSCSI DEVICE DRIVER & UTILITIES

Introduction

The LLSCSI Device Driver allows you to use SCSI hard drives on a Z-100 computer which is equipped with a SCSI host adaptor. The driver and setup program allows you to use multiple drives, each up to 512Mb capacity. Up to 16 partitions may be established on each SCSI drive.

Partitions may be any size up to 512Mb, although partitions larger than 64Mb will require a modified Z-100 BIOS.

System Requirements:

In order to use the LLSCSI driver software and utilities in this package, you will need the following:

- A Heath/Zenith Z-100 series computer
- One of the following SCSI host adaptors;
  - = LifeLine SCSI Host Adaptor/ Bootable EEPROM Board
  - = Lomas Data Products SCSI host adaptor
- MS-DOS v3.1 (or later) operating system
- Fixed or removable media hard drive with embedded SCSI controller. At the time this software was released (March 1992), drives from the following manufacturers were supported:
  - = Conner SCSI drives
  - = Seagate 'N' series SCSI drives
  - = Quantum SCSI drives
  - = Rodime SCSI drives
  - = Syquest removable cartridge SCSI drive
  - = Other untested SCSI drives may also work

LLSCSI Distribution Disk Contents

The LLSCSI Distribution Disk contained the following files:

LLSETUP	.COM	Program which preps, certifies, partitions, and formats the SCSI hard drive
LLSCSI	.SYS	The SCSI device driver
LLINFO	.COM	Program which displays information about the SCSI drive
LLDUMP	.EXE	Program which displays absolute sector information from the SCSI drive
LLDFLS	.COM	Program which displays SCSI drive sector defect list

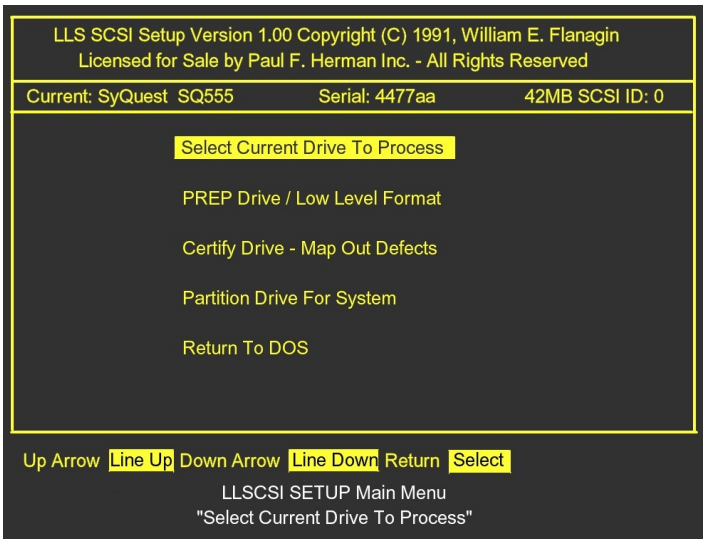
Note: The LLSCSI software version for the Lomas Data Products SCSI host adaptor had each file name prefixed with **'LDP'** instead of **'LL'**.

Setting Up the SCSI Hard Drive

The **LLSETUP** program is used to setup the SCSI hard drive. This will be your first step when installing a new SCSI drive. LLSETUP performs a low-level format (PREP), certifies the disk surface and updates the bad sector defect list, allows you to partition the drive, and formats the partitions for use with MS-DOS.

Note: If you are using a removable media SCSI drive, the LLSETUP program activities must be performed for EACH removable cartridge.

LLSETUP Menu Screen



When the **LLSETUP** program first begins, it takes inventory of the SCSI bus to see what devices it finds. Each SCSI device will have a unique logical unit number assigned to it.

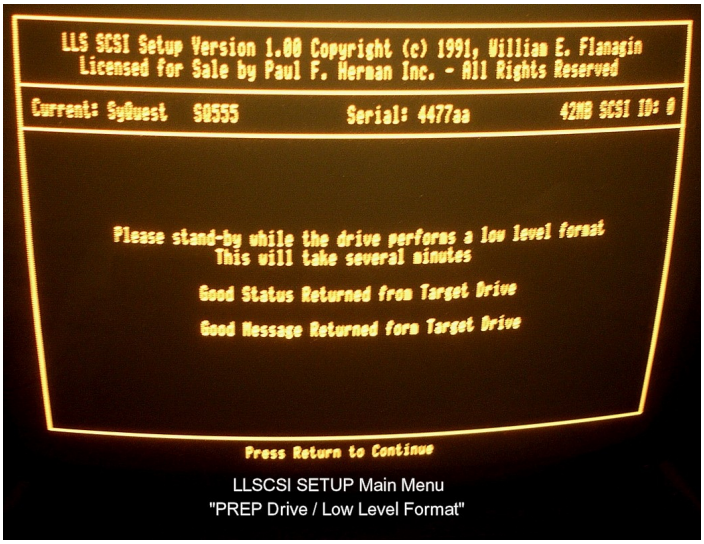
When you install your SCSI drive, you should set the jumpers so that it is configured as logical unit 0, 1, 2, 3, 4, or 5. Typically, your first drive would be unit 0, the next drive would be unit 1, and so forth. Logical unit 6 is reserved for the SCSI host adaptor itself, and unit 7 is reserved for a tape backup unit (not supported by this LLSCSI driver).

As you can see in the above picture, the program found my SyQuest SQ555 cartridge hard drive. The program then lists the Main Menu, with five choices. We will discuss each...

Select Current Drive to Process:

The first step is selecting the drive to process. Just press {RETURN} to select.

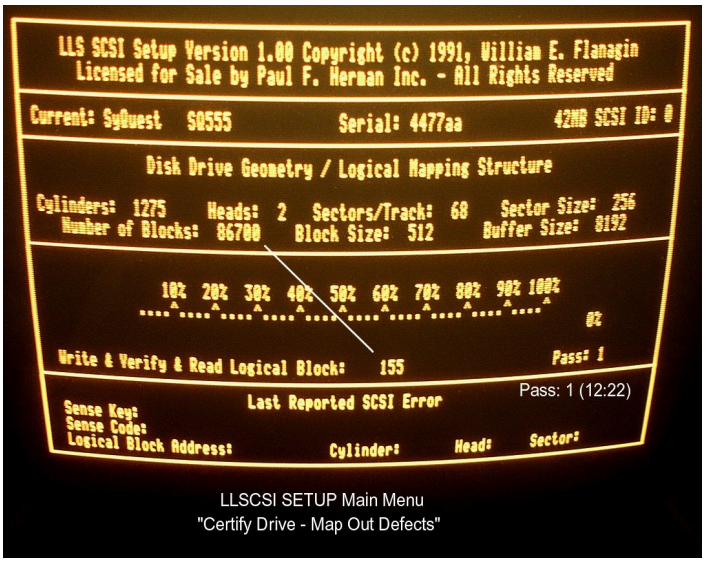
Prep Drive / Low Level Format



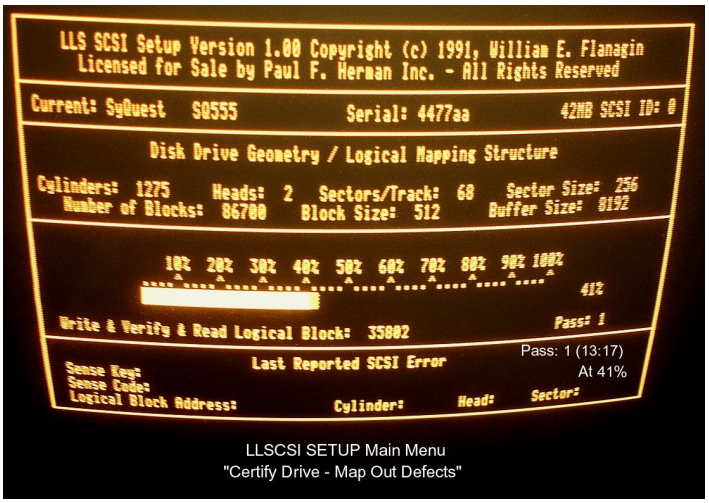
All new drives must be prepped. If a drive has previously been used in a PC compatible or other computer system, it must still be prepped, since the low level formatting performed by the LLSETUP program is NOT the same as used by other SCSI software or manufacturers.

Low level formatting will take a fair amount of time - 4 or 5 minutes on my 44Mb cartridges. The exact amount will depend on the size of your hard drive. When the low level format operation is complete, press {RETURN} to return to the main menu.

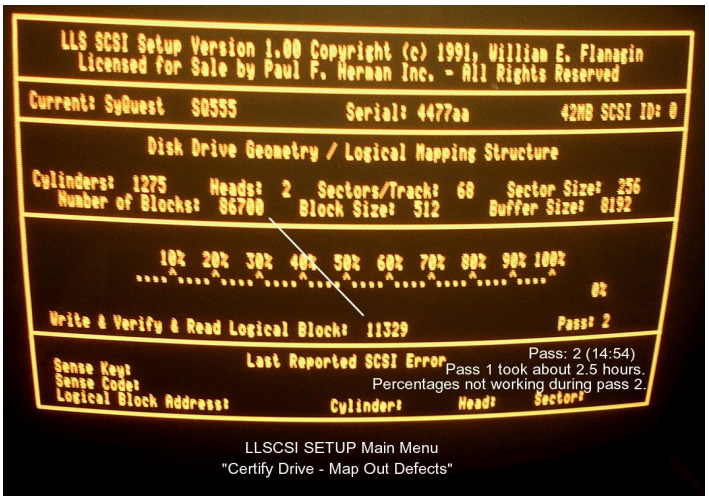
Certify Drive / Map Out Defects:



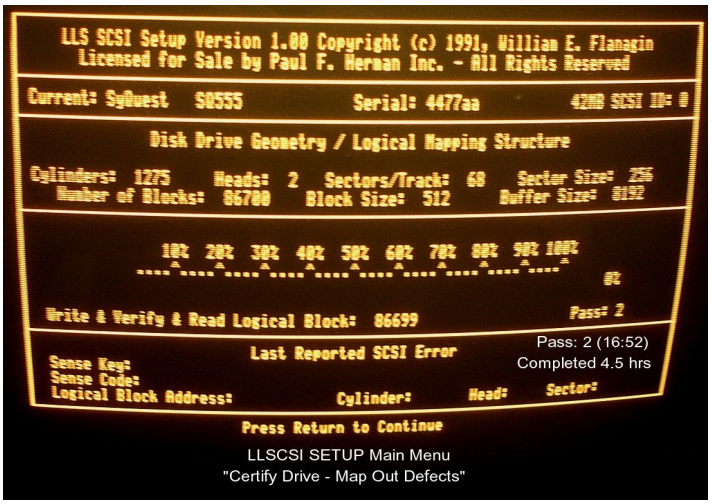
This is an optional, but recommended step. The certification process causes the entire disk to be checked for defects, and if any are found, they are added to the drive defect list. This prevents them from being used.



Drive certification will be a time consuming task. Here we are about an hour in and we are checking logical block 35802 out of 86700 blocks or about 41% in Pass 1.

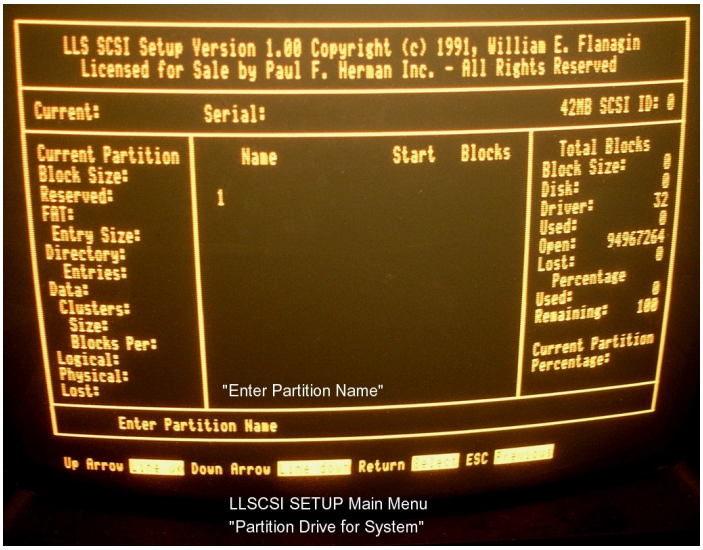


Here Pass 1 is complete (~2.5 hrs) and Pass 2 has begun. For some reason, the percentage bar does not work during this second pass.



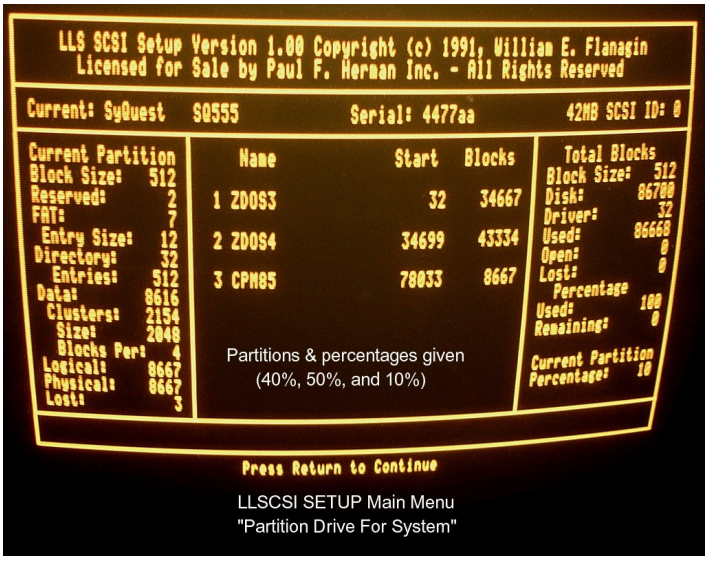
When complete, press {RETURN} to return to the main menu.

Partition Drive for System



You must establish at least one partition on each SCSI hard disk. LLSETUP allows you to have as many as 16 partitions, if you desire. Each partition may be any size up to the entire drive, but you should avoid unnecessarily large partitions, since they may result in large cluster sizes - very wasteful of disk space.

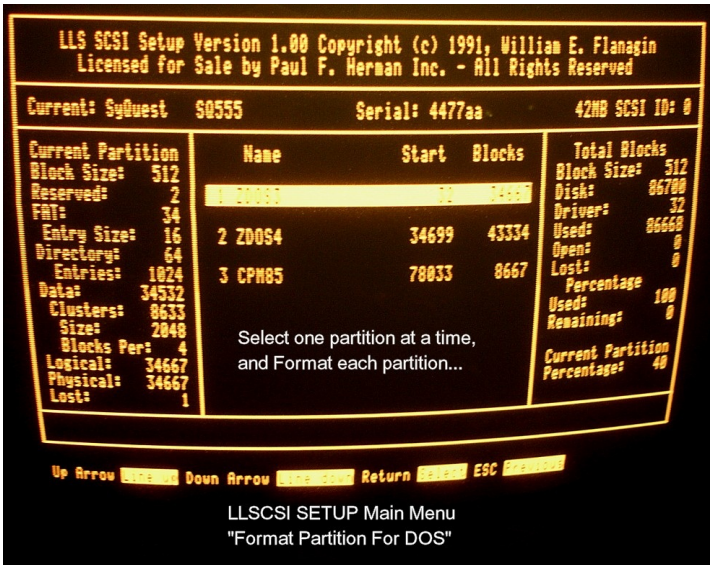
It is recommended that you keep partitions under 32Mb, if possible, for the most efficient disk space utilization. If necessary, partitions up to 64Mb may be used with a stock Z-100 BIOS. Partitions larger than 64Mb will require a modified BIOS.



Note: If you are using the LLSCSI/EEPROM Board, BIOS versions are included which allow you to use partitions up to 128Mb, without reassembling the BIOS.

For each partition, you must provide a name and the partition size. The display to the right and left of the screen will show statistics for the current partition, and for all partitions.

Format Partition for DOS:



The last step in setting up the SCSI hard drive is formatting each partition for use with DOS. Each partition must be formatted. This last step is necessary in order for the LLSCSI driver to recognize the partition.

The above picture shows the typical screen as it is being formatted. It is repeated for the other two partitions. When all the partitions are completed, pressing {RETURN} takes you back to the Main Menu, and selecting “Return to DOS” takes you back to the DOS prompt.

Installing the LLSCSI Driver

The LLSCSI device driver is installed by adding a line to the CONFIG.SYS file on your boot device... like this;

DEVICE=LLSCSI.SYS

If you do not want the device driver to be in the root directory on the boot drive, the complete path of the device driver may be given. See your DOS manual for more details on the syntax of the 'DEVICE=' command.

After adding the 'DEVICE=' command to your CONFIG.SYS file, you must reboot in order for the device driver to load.

During the bootup process, the **LLSCSI.SYS** driver will take an inventory of all SCSI devices by attempting to communicate with each logical unit number. If valid data is not returned within a predefined timeout period, it will give up, and go on to try the next logical unit number. It does this for each logical unit number on the SCSI bus.

Since the LLSCSI.SYS driver polls the SCSI bus whenever it is loaded, this means you can add or remove new SCSI devices any time, without any changes to the driver or software. Just make sure you run LLSETUP on each new device when it is first installed.

The /W Switch

It is possible that when you first turn the computer on, the LLSCSI.SYS driver will exceed its timeout limit before your SCSI drive spins up to speed. If this happens, the driver will not recognize the drive. This might particularly be a problem if you are booting from the EEPROM device of the SCSI/EEPROM Board, since this device gets through the boot sequence very quickly.

This problem may be corrected by simply RESETing the computer after the drive is up to speed, and letting the driver load again. Or, you may want to set your Z-100 for a "manual" boot, and wait a few seconds after power up to issue the boot command.

If your drive consistently fails to come up to speed within the timeout period, you may use the **'/W'** (Wait) switch when loading the LLSCSI driver. To use this 'Wait' switch, use a command of the following form in CONFIG.SYS;

DEVICE=LLSCSI.SYS /W:n

where 'n' is a delay constant which causes the driver to wait before beginning to poll the SCSI bus. Start with a low number and keep increasing it until your SCSI drive is consistently recognized during the power-up polling process.

The disadvantage of using the '/W' switch is that the driver will wait every time you reboot - even if the drive is already up to speed. For most new drives, a wait value of just a few seconds should be satisfactory, and in many cases, no wait period will be required at all.

One other suggestion... if your drive is just a little too slow to come up to speed at power on, you might try assigning it logical number 5, instead of unit number 0. This will not increase the amount of time it takes the driver to poll the SCSI bus, but it will postpone polling of your drive, since it is near the end of the logical unit number list.

The /V Switch

The **'/V'** (Verbose) switch causes the driver to display a complete description of each SCSI partition during the bootup process.

Improved LLSCSI.SYS

Subsequent work on the LLSCSI/EEPROM Board resulted in three additional LLSCSI.xxx files that were alternate forms of LLSCSI.SYS. They were distributed with the DOS v4 distribution packages for the Z-100. The new files were:

```
LLSCSI.FRW = fast read/write
LLSCSI.FR  = fast read
LLSCSI.SLO = non-accelerated read/writes
```

It was found that some drives worked better than others. Experimentation is the only way to determine if your drive will work with faster commands. The problem is that if your drive cannot accept the faster commands during a write, data WILL BE LOST.

These files were provided with NO guarantees. Most drives will work with LLSCSI.SLO or LLSCSI.FR.

Additional Note: LLSCSI.FRW may not work reliably at 10 MHz or faster.

Still later changes to **LLSCSI.SYS** included:

The New /U Switch

The new **/U[:][xxx]** switch is an **Unconditional Wait**. Each 'xxx' is a 1/2 second wait. After the wait period, processing continues checking for a Wait Until Ready request. If no 'xxx' is given, a value of 20(10 seconds) is used.

Additionally, if **CLEAR_PU_WAIT** is set to TRUE, the unconditional wait is cancelled if the MTR-ROM did not find a Power On Reset condition at boot time. This is done since the unconditional wait should only be necessary if we are allowing the drives to spin up to speed.

The New /W Switch

The new **/W[:][xxx]** Switch is now **Wait Until Ready**. Each of the 7 possible units are checked with 'Test Unit Ready' command starting with unit 7. The Wait is exited as soon as any drive shows ready. A 'Reset' command is done between each group of tests.

After 'xxx' groups of tests, if no unit becomes ready, processing continues as if no Wait Until Ready was requested.

If the compile time option **WAIT4ALL_READY** is set to TRUE, the /W switch is modified to have the format **/W[:][xxx][-yyyyyyy].**

The 'xxx' is still the wait count but '-yyyyyyy' is a list of the physical drive numbers of the drives to check for ready. If no drives are given, the /W command reverts back to the functions as described above. We do not exit the check until the wait count runs out or all the drives show ready.

This is useful if you have more than one drive and they don't all show ready at the same time. For example, a command line switch **/W:45-015** will try a maximum of 45 times, exiting when drives 0, 1, and 5 all have shown ready.

The New /T Switch

A compile time option now eliminates the /T(est) switch to save both memory and disk space. If enabled, the /T command line switch will generate **screen messages and diagnostics** on internal errors.

New Multiple Drive Speed Selection

DMA_CNFG_DRIVES must be set to TRUE at compile time. You may then pass to LLSCSI (on the command line) the speed selection for READS and WRITES for each drive. The syntax is as follows:

/#:[R][W]

where:
is the physical drive
R indicates the desire for FAST READs
W indicates the desire for FAST WRITEs.

The actual speed of the FAST option is controlled by the compile time options in SCIPARMS.ASM. If you do not select the FAST option on the command line, then the speed is the same as setting FULL_SEC_READS or FULL_ SEC_WRITES to FALSE at compile time.

For example, if drives 1 and 5 can not handle the fastest V20 WRITES, set the following:

/0:RW /1:R /5:R

In this case, choosing to compile with **V20_WRITES_1** set to TRUE will permit drive 0 to read/write at the fastest possible speed, but drives 1 and 5 will only do FAST READs.

The New "PC" Master Boot Record

LLSCSI now checks for a "PC" Master Boot Record in Physical Sector Zero, if it does not find the old LLSETUP format. "Extended" partitions are supported. The number of Sectors Per Track and number of Heads can only be determined if at least one of the partitions in the Master Boot Record has both non-zero Cylinder and Head for either the Starting or Ending location.

CONVERT.EXE is Now Required for Recompile

In order to save EEPROM disk space, the use of CONVERT.EXE (same as BIOS) is now required after a successful recompile.

The New Generic Input/Output Control

GIOCTL is now fully supported, enabling the use of "PC" FORMAT.COM to format a partition.

Maximum Drives Supported

LLSCSI now supports a maximum of 25 logical drives, but will not pass back to DOS an amount greater than 'Drive Z:'.

Smaller Size When DOS 4

A compile time option now disables version independence to achieve a smaller disk and memory size.

Use of BIOS Stack and Disk Buffer

To further save resident memory requirements, LLSCSI has another compile time option that uses the BIOS stack and/or BIOS disk buffer.

Removable Media Hot Swapping

An attempt has been made to support "Hot Swapping" for removable media.

The compile time option **REMOVEABLE_MEDIA** must be set to TRUE and the Boot Loader on the disks must be "PC" compliant. Due to the lack of 'removable media devices', however, this has not been checked out.

LLS_xxxx.Vxx

During continued experimentation, it was found that for optimum performance, LLSCSI needed a small change when compiling for DOS versions 2 & 3 and for different computer processing units (CPUs).

This resulted in the following files:

LLS_8088.V3
LLS_8088.V3P
LLS_8088.V4
LLS_8088.V4P
LLS_V20.V3
LLS_V20.V3P
LLS_V20.V4
LLS_V20.V4P

The .V3? files are good for version 2 or 3 of MSDOS. The .V4? files are for Z-DOS v4.

The .V?P files are compressed using PKLITE.

To try the newer files, rename your original LLSCSI.SYS file to LLSCSI.ORG (for original) and then copy the appropriate file you wish to try to LLSCSI.SYS.

Other Utility Programs

LLINFO

This program displays information about the SCSI drive. This information will only be of interest to those familiar with the intricacies of the SCSI world. It is included in the package mainly for two reasons;

- Running LLINFO is a quick way to see if your SCSI host adaptor and drive are operating correctly. LLINFO does not require the LLSCSI.SYS driver to be installed.

- The information provided by LLINFO may be helpful in assisting you in supporting new SCSI drives or devices, and may aid in diagnosing problems. Output from LLINFO may be directed to the printer using the command;

LLINFO>TEST.DAT

By default, LLINFO will provide information about SCSI logical unit 0. If you wish to display information for another logical unit number, use the following syntax;

LLINFO n

where 'n' is the unit number, from 0 to 5. The number used for 'n' must be separated from the LLINFO command by exactly one space.

LLDUMP

This program allows you to display absolute sector information from the SCSI drive. It will prompt you for the logical unit number you wish to investigate.

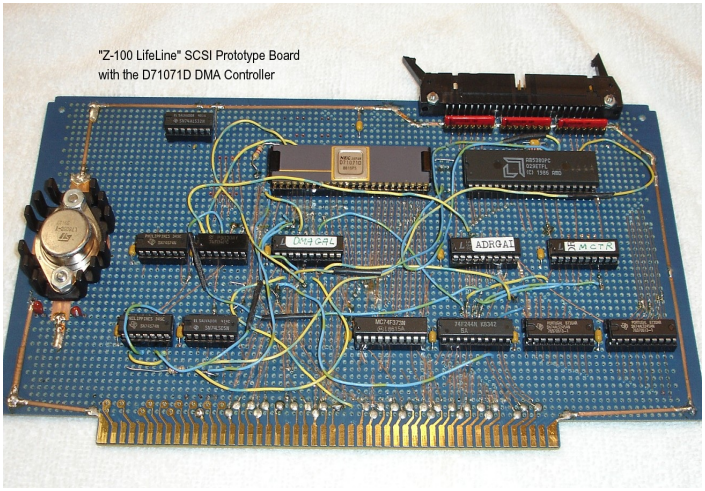
LLDFLS

This program displays the SCSI drive sector defect list. By default, it reports the defect list for logical unit 0. To access other SCSI units, use the following syntax;

LLDFLS n

where 'n' is the unit number, from 0 to 5. The number used for 'n' must be separated from the LLDFLS command by exactly one space.

NOTE: All these changes were before my time and as I could not afford nor needed a SCSI drive at the time, I could not even be considered a casual observer. So, while I tried to make sense of these last few pages of information, I tried to just gather and document the information as best I could.



SCSI DMA Host Adaptor and Bootable EEPROM Board

Source Information:

The following is based upon a paper by Robert F. Hassard of Walnut Creek, CA, dated May 1, 1992, and the development team's original SCSI Host Adaptor/Bootable EEPROM Board distribution documents.

Note: As I understand it, Robert was working on another SCSI Host Adaptor that was to use a DMA Controller. In fact, I have what appears to be his prototype board for this upgrade. However, I have NO other documents or correspondence that shows what became of the project. If you know anything that could shed some light on this project, please contact me.

The source information for Robert Hassard's DMA concept is from the following:

- The H/Z-100 Tech Manuals and Schematics
- The iAPX-88 Book supplied with the H/Z-100 Technical Manuals
- NEC Electronics Data Book, pages 7-91 thru 7-125, on the NEC uPD71071 DMA Controller
- Advanced Micro Devices Flash Memory Products 1990 Data Book/Handbook
- Advanced Micro Devices Personal Computer Products, pages 4-3 thru 4-30, on the Am5380 SCSI Controller
- Various other documents on PALs & GALs, and other Integrated Circuits

Theory of Operation:

This is a rather complicated board, so it will be discussed in four parts:

1) Data Bus Management
2) The EEPROM
3) The SCSI Interface
4) DMA Operations

DATA BUS MANAGEMENT:

Normally, the H/Z-100 has two data buses; the Data In Bus to transfer data from memory or a peripheral IN to the CPU, and the Data Out Bus to transfer data from the CPU OUT to memory or a peripheral.

But, on this board, both the Data In Bus and the Data Out Bus are bi-directional. The reason for that is that the DMA Controller takes the place of the CPU and the only way that it can send data to memory is on the Data Out Bus and the only way it can receive Data from memory is on the Data In Bus.

In addition, there is a third bi-directional Data Bus on the board linking the various components of the board together. This Bus interfaces to the Data In and Data Out Buses through a pair of 74HCT245 Tri-state Transceivers. (**Note:** the 74LS245 is NOT Tri-state, but has a tri-state counterpart, the 74LS645.)

The DMA Controller has two states (or cycles); Slave Mode and Master Mode.

When in the **Slave Mode**, the Data In and Data Out Buses are normal.

When DMA is in the **Master Mode**, the direction of those two buses is reversed. When one Transceiver is Enabled, the other is Disabled.

The Data Bus is enabled for writing by the signal sWO*. It is enabled for reading by the signals RDEN*, which is generated in the ADRPAL, and MRD*, which is generated by the DMA Controller. Both of these signals are Tri-state.

FLASH EEPROM:

The EEPROM is addressed as a Port because it is used as a pseudo disk organized into 256 sectors of 1024 bytes each. The EEPROM is addressed by a latch which holds the sector number and by a counter which counts up from 0 to 1023. The Sector Latch is loaded by an OUT to Port 0CEh. The latched sector is read by 1024 looped reads from Port 0CEh.

The EEPROM is written to by sectors as above. The sector number is latched in the same way, using Port 0CEh. However, the writing is done in loops of 1024 bytes each to Port 0CFh.

Each time the EEPROM is written to or read from, the Address Counter must be incremented. This is done by reading Port 0CFh. The data read in this case is meaningless.

Automatic incrementing is NOT possible because of the Write Algorithm used in programming the EEPROM. The EEPROM Address Counter is reset by the same signal that enables loading the Sector Latch (OUT to 0CEh).

SCSI INTERFACE:

Because the SCSI data transfer to and from memory is done by DMA, the only communication with the Am5380 Interface is for programming purposes. It is programmed using Ports 0C0h thru 0C7h. Otherwise, the DMA Controller controls the Interface circuitry.

However, this function is important. Not only are the Interface's Registers read to determine status conditions, but writing to them is essential for controlling the SCSI operations and the Start of DMA operations.

DMA CONTROLLER:

The DMA controller is programmed using Ports 080h thru 08Fh. These Ports correspond to the programming registers of the uPD71071 and all may be byte written or byte read.

The interface with this DMA controller is fairly simple. The 8-bit Data Bus uses the Transceivers described above under Data Bus Management.

Address lines A7-A0 are connected directly to the S-100 Bus.

Address lines A15-A8 are latched out of the data bus.

Address lines A19-A16 are connected direct to the S-100 Bus.

Address lines A23-A20 are NOT connected because those lines are uni-directional out of the H/Z-100 Main Board.

There is an open collector lines driver for disabling the H/Z-100 buses during DMA operation. A tri-state driver to assert the new status and program signals to the H/Z-100 memory. And a combination of logic in flip-flops and a PAL to create the memory control signals during a DMA operation.

There are two transfer modes available for use; **DEMAND**, which matches the Am5380 SCSI Controller's Normal Mode, and **BLOCK**, which matches the 5380's Block Mode.

In both of these Modes, data is transferred by bytes. The DMA controller provides the control signals to actuate the SCSI Host Adaptor and Memory so that as data enters the Data Bus from one, it is picked off by the other depending on the programming instructions. This process continues until the Address Counter reaches Terminal Count, or the SCSI Host Adaptor issues an Interrupt (IRQ) signal.

The speed of Data Transfer is controlled by a READY signal generated by the SCSI Host Adaptor. This signal is combined with the RDY signal generated by the Refresh Wait circuitry so that either may delay a transfer.

I hope that you enjoyed the history and documentation of the first "*Z-100 LifeLine*" project, the LLSCSI Controller Board.

If you have any questions or comments, please email me at:
z100lifeline@swvagts.com

Cheers,

Steven W. Vagts