



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

# SCSI

## "Z-100 LifeLine" SCSI Controller

by Steven Vagts  
Editor, "Z-100 LifeLine"

### "Z-100 LifeLine" SCSI Controller

#### SCSI Host Adaptor & Bootable EEPROM Board Copyright (C) 1992

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 vendors, research teams and the ultimate users.

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

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. Flanagan    SCSI programming
- Travis J. Barfield    Parts acquisition &  
Manufacturing
- Michael Zinkow        Z-DOS development
- John Beyers            BIOS programming &  
Z-DOS utilities

The following is a copy of portions of the original distribution documentation for the "Z-100 LifeLine" SCSI Host Adaptor/Bootable EEPROM Board. Publication here is for historical documentation only 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 256K 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 448K of system memory.
- MTR-ROM v3.1 or greater, which had to be installed in order to boot from the EEPROM device.
- MS-DOS v3.1 operating system, which was required to boot from the EEPROM device, 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:

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 v3.1 (or later) by following the instructions in your Z-100 User's Manual. Be sure to set the motherboard jumpers J-101 and J-102 both to position 1 (bridging the pins nearest the left edge of the motherboard).

2. Set the motherboard DIP switch S-101 to the following settings:

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

**Note:** 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.

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.

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.

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. Check the drive carefully to make sure 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.

**DANGER:** 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 on' 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. Since the DIP switch has been set so the default boot device is the EEPROM, you will need to use a manual boot sequence:

```
{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.
```

3. You should now be at the MS-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 words 'Go on' 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. DO NOT run this test 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 256K 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 READ-ONLY 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 programming 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".

### EEPROM Utilities Disk

This disk 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:

IO64W	.SYS	BIOS files which support the bootable EEPROM device.
IO64	.SYS	
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	.DVD	A RAM disk driver used during programming of the bootable EEPROM device.

**Note:** The EEMDISK was a 256K 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 256K from the RAM disk to the EEPROM device.
REEP	.EXE	A utility which read 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 256K 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 256K byte disk file (specified on the command line) to the EEPROM device.

EEPTTEST .EXE A utility to test the EEPROM device. This utility erased, then programmed the EEPROM device with test data (ALL current data was 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.

**Note:** Do NOT use the 128 Mb 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 512 Mb) are possible by changing and reassembling the BIOS.

**Note:** Using a BIOS without the Z-217 support will save about 3K of memory. It will also allow use of the drive letters E through H for the EEPROM device and SCSI partitions. 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 MS-DOS 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 MS-DOS distribution disk #1. This can be done by issuing the following 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 MS-DOS 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.

9. 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.

**Programming the EEPROM Device**

Creating the EEPROM Setup Disk - For use whenever you want to program your EEPROM device. A directory named 'EEP' on this disk contained all the files you may want on your bootable EEPROM device.

1. Format a new BOOTABLE MS-DOS v3.1 floppy disk using the command;

**FORMAT A:/S**

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

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

- Maximum 32 directory entries (including IO.SYS and MSDOS.SYS)
- Maximum 256K 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, or if using Z-DOS v4, the DRIVECFG.COM program. To do this, change to the 'EEP' directory, using the following command;

```
CD \EEP
```

Now invoke the CONFIGUR.COM or DRIVECFG.COM program. To do this, the file must be in your DOS path on another disk, or you may need to copy CONFIGUR.COM (or DRIVECFG.COM) into the 'EEP' directory temporarily.

When you are done configuring the IO.SYS file, be sure to write the changes to disk. If you copied CONFIGUR.COM (or DRIVECFG.COM) into the 'EEP' directory, you can now delete it.

#### Using the EEPROM Setup Disk to Program the EEPROM Device

12. Now re-boot your Z-100 from the EEPROM Setup Disk. The AUTOEXEC.BAT file will automatically perform the following steps to program the EEPROM device:

- The EEMDISK 256K 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, and then boot the EEPROM Setup Disk. The EEPROM device will be automatically programmed with the new files.

**Programming Hint:** A typical computer user may want to change his AUTOEXEC.BAT file from time to time. It would be nice to avoid having to reprogram the EEPROM device every time a small change is needed to AUTOEXEC.BAT. Use the AUTOEXEC.BAT file on your EEPROM device to simply call another AUTOEXEC.BAT file on your fixed hard disk. If your hard disk is assigned drive letter J, this could be done by using the following lines in your EEPROM's AUTOEXEC.BAT file;

**J:  
AUTOEXEC**

Now, whenever you want to make a change to the AUTOEXEC sequence, simply change the AUTOEXEC.BAT file on drive J. This avoids having to reprogram the EEPROM device.

#### Booting from the EEPROM Device

Now you should be able to boot from the EEPROM device. This can be done by pressing {B} at the hand prompt. If you followed our instructions for setting the motherboard DIP switch, S-101, under "Hardware Installation", the EEPROM device should be the default boot device. Otherwise, you may boot by pressing {B} at the hand prompt, then pressing {F4}, and then {RETURN}.

Here is a summary of the S-101 sections that affect the EEPROM device:

#### Switch Sections:

0	1	2	
Off	Off	Off	Boot from 5-1/4" floppy
On	Off	Off	Boot from 8" floppy
Off	On	Off	Boot from Hard Drive
On	On	Off	Boot from EEPROM device

(other settings undefined)

**Note:** If you would like your Z-100 to automatically boot up 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 512 mb in size, although partitions larger than 128 mb are considered non-standard, and require the BIOS to be changed and reassembled.

## 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. Additional instructions accompanied the MTR-100 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.

## 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 256K 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	= COX	from ADRPAL
Pin 5	= C8	from ADRPAL
Pin 6	= NCLR	from Pin 15 (/WENO)
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	= WENO	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 * /COX * sOUT + /pWR * /C8 * sOUT
/IOR = pDBIN * /COX * sINP + pDBIN * /C8 * sINP
/CLR = NCLR
/PREN = /CEF * /A0 * sINP + /CEF * A0 * sOUT
/WENO = /CEF * /A0 * /pWR * sOUT
/WEN1 = /CEF * A0 * /pWR * sOUT
/REN = /CEF * /A0 * pDBIN * sINP
/INCR = /CEF * A0 * sINP
```

**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 = COX Ports 0C0h through 0C7h registers  
 Pin 18 = CEF Address 0CEh or 0CFh (low)

### Logic Equations:

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

**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.

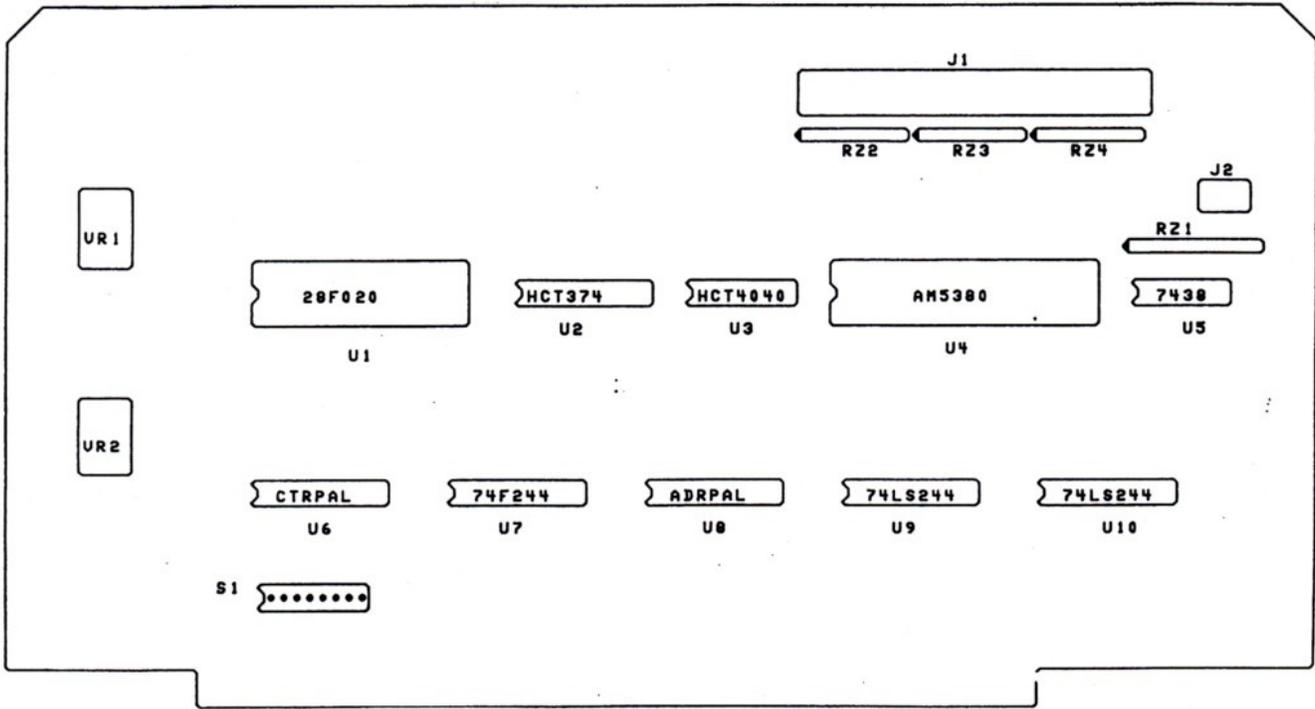
It is normal for VR1 to be hot to the touch. However, it has been found that some Z-100 power supplies provide as much as 11 volts input to this regulator, which may cause it to run hotter than normal. Even at 11 volts input, VR1 is still operating within its design criteria. However, the increased temperature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

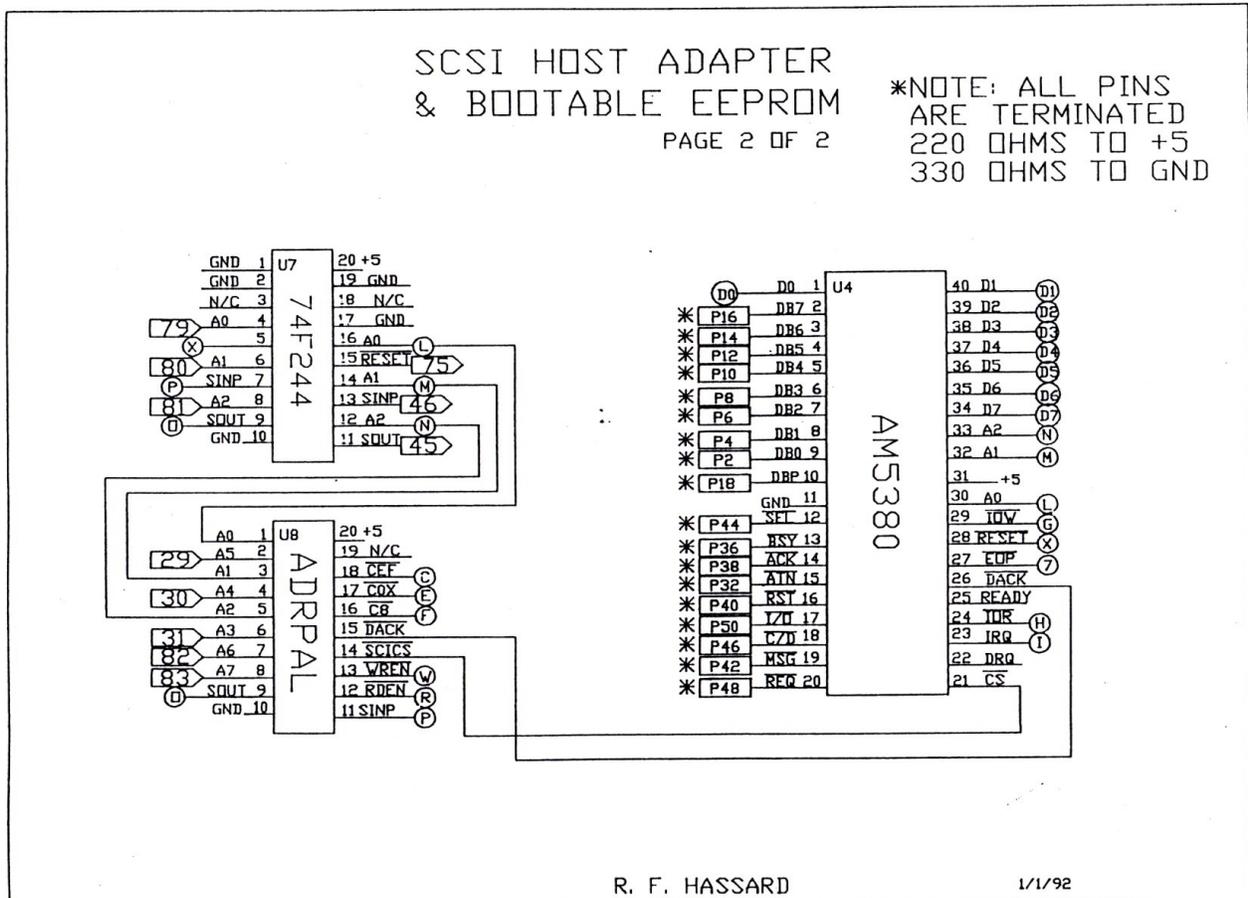
Measure the voltage at the input to VR1. If the voltage is above 9 volts, insert a 2.2 ohm, 2 watt resistor between the input and the regulator. This can be done by cutting the input line trace (along the left side of the board), and soldering the resistor in the holes which are provided. If the input 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	4.7K ohm, 10-pin SIP resistor network	770-101-R4.7K
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 T0-220	
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



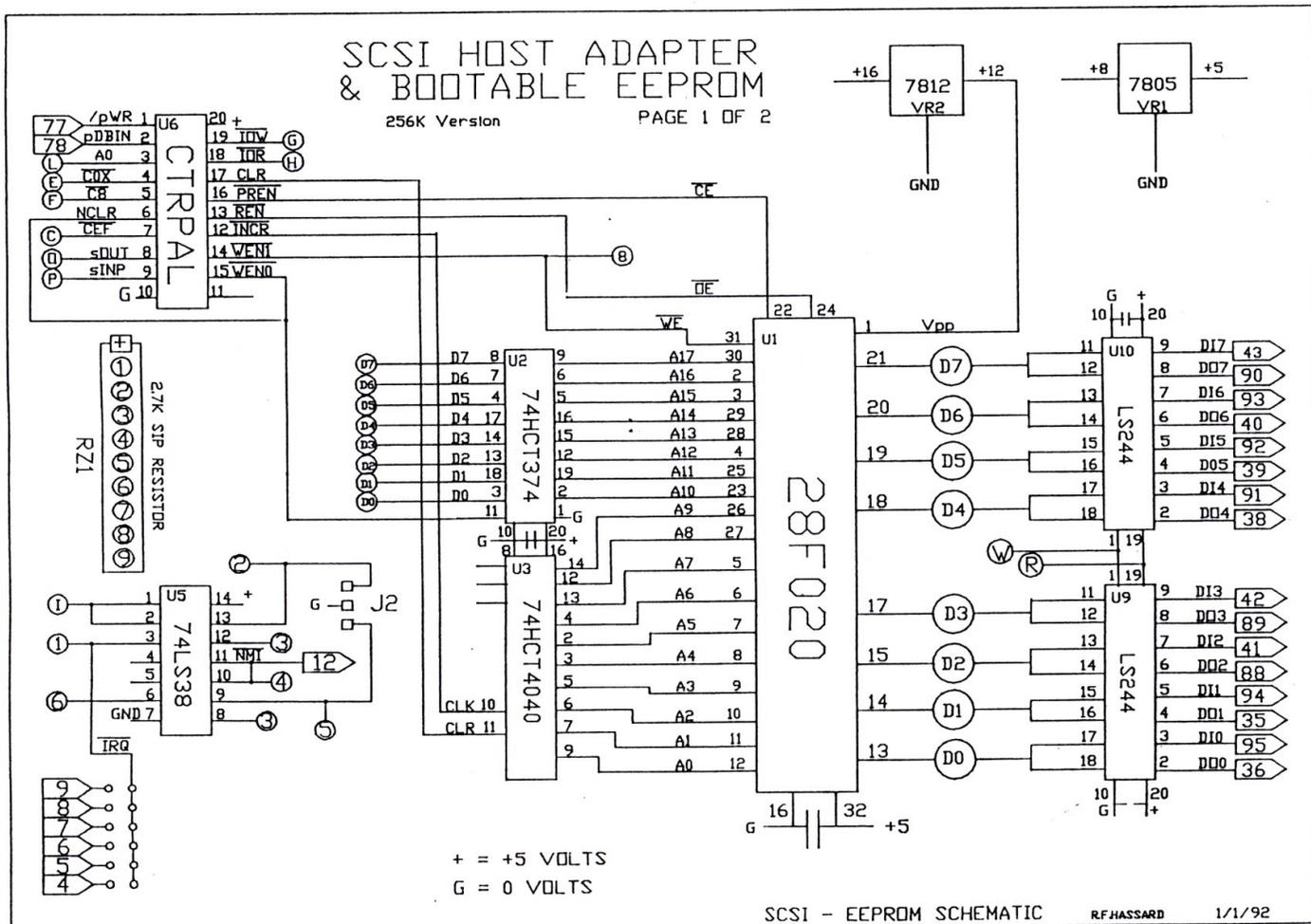
R. F. HASSARD

1/1/92

# SCSI HOST ADAPTER & BOOTABLE EEPROM

256K Version

PAGE 1 OF 2



SCSI Host Adaptor & Bootable EEPROM Schematic - Page 2 of 2

## SYMBOLS CROSS REFERENCE

There are 22 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 has been prepared.

Symbol	Signal Name	Active	Pages used on	Symbol	Signal Name	Active	Pages used on
A	ADSTB	HI	2	M	0AH	LO	2, 3
B	AEN	HI	1, 2, 3	N	DMAEN	LO	2
C	DMAS	LO	1, 3	P	EEPS	LO	1, 3
D	DATA		1, 2, 3	R	MRD	LO	2, 3
E	REN	LO	1, 3	S	MWR	LO	2, 3
G	HASCS	LO	1, 3	U	IRQ	HI	2, 3
H	IOR	LO	1, 2, 3	V	HLDAK	HI	2
I	IOW	LO	1, 2, 3	W	DREADY	HI	2
J	DREQ	HI	2, 3	X	HREADY	HI	2, 3
K	DACK	LO	2, 3	Y	RESET	LO	2, 3
L	DMACS	LO	1, 2	Z	HAS	LO	1, 3

# 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 512 mb capacity. Up to 16 partitions may be established on each SCSI drive. Partitions may be any size up to 512 Mb, although partitions larger than 64 Mb 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 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

## 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 has 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.

## Select Current Drive to Process:

The first step is selecting the drive to process. 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).

## 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.

Low level formatting will take a fair amount of time. The exact amount will depend on the size of your hard drive... the larger it is, the longer the prepping operation will take. When the low level format operation is complete, hit {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. When complete, hit {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. This is very wasteful of disk space.

It is recommended that you keep partitions under 32 Mb, if possible, for the most efficient disk space utilization. If necessary, partitions up to 64 Mb may be used with a stock Z-100 BIOS. Partitions larger than 64 Mb 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 128 Mb, without reassembling the BIOS.

For each partition, you will be asked to provide a name, and the partition size. The display to the right and left of the screen will show you 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 for use with MS-DOS. Each partition must be formatted. This last step is necessary in order for the LLSCSI driver to recognize the partition.

### 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 RESETTING 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.

Other changes to **LLSCSI.SYS** include:

#### 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 '-yyyyyy' 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 LSETUP 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 following 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.

## 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.

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](mailto:z100lifeline@swvagts.com)

Cheers,

Steven W. Vagts

