LISA BOOT ROM MANUAL

VERSION 1.3

Rich Castro
24 February 1984

OVERVIEW

This document replaces all previously written versions of the Lisa Boot ROM manual. It discusses the operation of the Lisa ROM contained on the system CPU board. This ROM contains a variety of diagnostic, setup and boot routines which are performed automatically upon power-up of a Lisa system. The various routines are outlined in the order they are performed, the ROM boot procedures are discussed, and the operation of the ROM monitor is explained. Also attached are separate appendices that cover error messages output by the ROM, miscellaneous information saved by the ROM on power-up, interfaces for ROM routines usable by external software, details on a special power-cycling mode provided for manufacturing use, and the procedure for creating a ROM master from its source code files.

The information contained in this manual corresponds to the latest release of the Lisa boot ROM, revision H.

CHANGES FROM LAST RELEASE

The latest changes made to the boot ROM have been done to correct errors dealing with the hard disk interface.

1) The boot ROM now always initializes the hard disk interface reset and parity reset lines as inactive outputs prior to attempting any interaction with a hard disk. Previously these lines were left initialized as inputs which resulted in a "floating" level on each line.

2) The hard disk interface reset routine now works properly for both Seagate and Widget type disk drives. Previously, reset of a Widget drive could result in a timeout error because of its longer delay in responding to a reset signal.

Minor changes have also been made to this document to correct erroneous statements and supply new information. These changes are noted by a vertical bar in the left hand margin of the changed page.

TABLE OF CONTENTS

1.0 INTRODUCTION

The boot ROM on the CPU board acts as the first test of the LISA system and enables booting from a variety of devices. It contains diagnostics that are executed each time the system is turned on, and also contains routines that enable startup from a floppy drive, an attached or builtin Profile type hard disk, or an "intelligent" I/O card.

As a general rule, any errors encountered terminate operation and the ROM attempts to output an error indication to the operator. The appendices contain a list of possible errors, as well as other miscellaneous information about the boot ROM. The following sections give details on the tests and procedures executed by the boot ROM in the order that they are actually run.

Note: Throughout this document hexadecimal numbers are denoted by a '$' prefix.

2.0 STARTUP TESTS/PROCEDURES

2.1) Reset Check

For normal power-up, or "cold-starts", the ROM executes its full set of internal diagnostics and boots from the selected device. However, since the system can be reset in other ways, such as via the reset button or a double bus fault, the ROM has two provisions for differentiating "warm-starts" from "cold-starts". Both use settings of the MMU registers as the key.

The first provision is for immediate resetting of the contrast level to avoid "screen flash". This is done by checking the setting of MMU register 127, context 0, normally set for access to special I/O space. If both limit and base registers retain their starting value, the ROM will immediately reset the contrast latch. The base register is initially set to 0, and the limit is set to $F00.

The second special provision in the ROM is for preserving memory on reset to facilitate debugging. This is done by checking MMU register 126, context 0, base and limit contents to see if they have changed. These registers are normally used for I/O space with the base set to 0 and the limit set to $900. They should remain constant unless the user wants to enable the reset feature. Enabling is accomplished by changing the limit register to $901 instead of its default value of $900, and leaving the base register set to 0. The ROM will then preserve as much of memory as possible on reset, and branch to its internal "monitor" that allows the user to examine the machine state or reboot as desired.

2.2) ROM Checksum

This test computes a 16 bit checksum of the entire ROM contents to check its validity. If the result is not zero as expected, testing is halted and the test loops indefinitely at a fixed address since further progress is probably not possible. The fixed loop address is $FE00C8.

If a hang occurs, the LISA screen may be blank or come on very bright with a random pattern, since the contrast control defaults to on and memory is uninitialized. The determining factor is the initial state of the video address latch, which is random at power-up time and thus may cause the system to access installed or uninstalled memory. The only definite means to determine that this has happened is to use a scope to see if the processor is continually reading the ROM at the loop address.

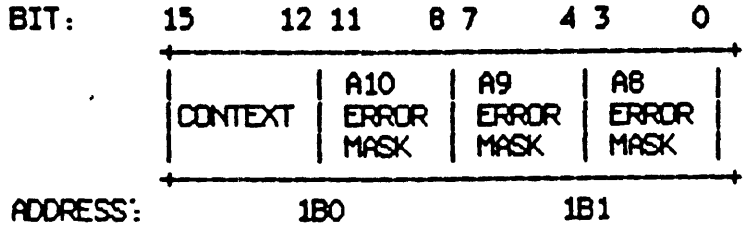
but has not yet started to access the MMU registers (which is the next test).

2.3) MMU Register Test

A read/write and address test is executed on the static RAM's that comprise the MMU registers. If an error is encountered in the critical supervisor registers, a single or double system reset signal is generated and the system hangs in an MMU exercise loop with the external indication being a blank screen. As with the ROM test, a scope must again be used to see if this is happening by checking the reset and MMU select signals. A single reset pulse indicates a read/write error was detected, while a double reset indicates an addressing error. Following the reset pulse(s), the ROM enters a loop that attempts to toggle every data and address line going to the MMU registers as an aid in determining the fault.

If no errors are encountered with the supervisor registers, the other context registers are tested and results are saved. If errors are detected here, a CPU error icon is displayed, along with error code '40' under the icon at the end of the power-up sequence. The results are saved at the following locations in memory for examination by the user (e.g., via the ROM monitor):

$1B0-$1B1 : The context in error and an "error mask" are saved at these locations. The context is saved as hex code 01, 02 or 03. The remaining three nibbles contain a bit pattern with each four bits corresponding to one of the MMU RAM chips. A '1' bit means an error at that bit location has been detected.



High 4 bits at location $1B0 contains the context setting
Low 4 bits at location $1B0 is for chip at A10 on CPU board
High 4 bits at location $1B1 is for chip at A9 on CPU board
Low 4 bits at location $1B1 is for chip at A8 on CPU board

If an error is encountered when trying to switch contexts, the ROM will

record the context that it tried to set and then abort the testing, returning the MMU to context 0 so further tests can proceed.

2.4) Memory Sizing and MMU Remapping

The memory is sized by a read/write test and the low and high physical memory addresses are saved. If no memory is detected, a second test is made to see if an I/O board is installed. If the I/O board is also missing, the ROM restarts the diagnostics, continually looping on the ROM checksum, MMU register and memory sizing tests as a means of burnin for the CPU board. At the end of each loop, the MSB of the video latch is also toggled such that an LED connected to this line will blink about ten times a second. The remaining bits of the video latch are set to $2F.

If memory cannot be accessed at all, but an I/O board is installed, an attempt is made to beep the speaker once, and the ROM then goes into a read/write loop at address one megabyte - 2 ($0FFFFE - long word address that spans both memory slots), using the pattern $AA55A55A. The video latch is also loaded with the default value $2F.

If a read/write error is detected, but some valid memory is found, an internal error indicator is set and the error bits are saved, but testing continues. The results are saved at location $184-185, as a word containing a '1' bit for each bit found in error (MSB at $184).

The MMU is then rewritten so that memory appears to start at address 0 and continue contiguously for the amount of memory contained in the system. All other MMU registers are set for invalid access except for registers 126 and 127 of context 0, which are set for I/O space and special I/O space (ROM and MMU access), respectively. The exact settings are as follows:

Context	Register	Base Contents	Limit Contents
0	0-7	Maps memory to start at 0 in 128K intervals	$700
	8-15	Set if system has >1024K of memory in 128K intervals	$700 for memory > 1024K $C00 for no memory
	16-125	0	$C00
	126	0	$900
	127	0	$F00
1	0-127	0	$C00
2	0-127	0	$C00
3	0-127	0	$C00

2.5) Preliminary Memory test

The first 2048 bytes of memory are tested next to ensure that some memory is functional and available for saving test results. If any errors occur, testing is aborted, the speaker is beeped twice with a low tone, and the ROM hangs in a read/write test loop at address 0 (on board in slot 2 if system has 2 memory boards), using the pattern $A55A. The video page is set to display this memory area by setting it to the first video page (0-32K).

If successful, low memory is initialized with interrupt and exception vectors and previous tests results (MMU and memory) are saved.

2.6) Parallel port VIA test

The ROM checks to see if the I/O board can be accessed, and if so, does a partial test of the VIA controlling the parallel port and contrast latch so that the contrast can be set. If all is OK, the contrast is turned off so that the following screen memory test is hidden from the user.

If the I/O board cannot be accessed, the ROM will abort further I/O board setup and testing, but continue with CPU and memory testing. The bad I/O board should result in either an I/O board error icon, with error code '58', or a CPU board error icon, with error code '41'. If the parallel port VIA is bad, the I/O board icon is displayed along with error code '51'.

2.7) Screen Memory Test

A test of screen memory is done next (last 32K of memory) so that it can be used to display information to the user. The contrast is turned off during this time so that the test is not visible on the screen. If any error occurs, the ROM starts searching for another area of memory for screen use, beginning from the top of memory down at 32K increments. If a good area is found, the video page is changed to that portion of memory, otherwise the standard default is used (last 32K of memory) and testing continues. Location $110 in low memory is written with the base address of the screen.

2.8) Continued I/O board testing and setup

At this point, the screen display is used to notify the user what test is in progress. The revision id of the boot ROM is also displayed in the upper right corner of the Lisa screen. The contrast is set to a mid-range default of $80, and the VIA controlling the keyboard and mouse interfaces is tested. If all is OK, a reset signal is sent to the keyboard and mouse interfaces, and a scan is then done to determine if either the keyboard or mouse are disconnected. (Note: Mouse disconnect check requires special setting of parameter memory to indicate that mouse should be attached). If either is disconnected, an error flag is set but testing continues.

Following the interface check the speaker is "clicked". This "click" serves two purposes: 1) it signals the user that all tests to this point have been executed, and 2) it indicates that the keyboard is now "operational" (if connected) and may be used to input "alternate booting" commands. At the same time, the speaker volume is set to a default mid-range value of 4.

2.9) CPU Board Test Completion

The remainder of the CPU board tests are executed to partially test the video and write wrong parity circuitry on the board. The video test checks to ensure that the vertical retrace signal is toggling, and then verifies that the system serial number can be read from the video prom. The parity test ensures that wrong parity can be forced and detected. If either of these tests fail, testing is aborted with a CPU board icon displayed along with one of the following error codes:

42 - Video logic error
43 - Parity logic error

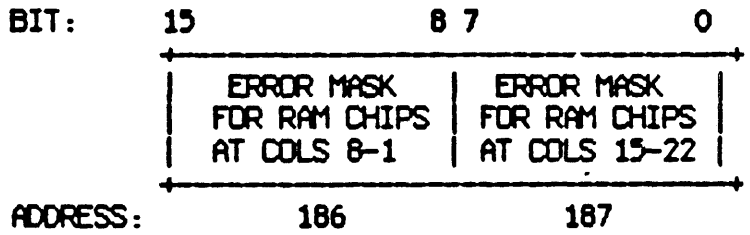
2.10) Memory Test

Next a full read/write and address test is performed on all of system RAM, minus the memory already tested (screen and first 2K). The default memory test from a "cold-start" consists of a pseudo random testing algorithm that is run with parity disabled and enabled. An optional mode, called extended memory testing, causes the same test to be run twice. This mode is invoked by setting of a special indicator bit in parameter memory, and is done automatically when in power-cycling mode. For "warm-starts" (e.g., after a reboot command from software or pressing of the reset button), only one pass of the test is run, with parity enabled, to minimize boot time.

The default cold-start test currently takes about 18 seconds for a full megabyte, with the extended mode approximately doubling this time to 36 seconds. The warm-start test reduces this time to about 8 seconds. While the tests are running, a display is shown on the screen indicating that memory is being tested.

If errors are detected, the results are saved in low memory and an icon for the memory board in error is displayed on the LISA screen at the end of the power-up sequence. A second line under the icon displays the error code as '70' for read/write errors, or '71' for parity errors. When a parity error is detected, the ROM automatically goes into a "search mode" to try and find the exact location in error down to the bit or parity chip level. The following memory locations save the results:

$186-$187 : bit pattern for memory at logical address 0 - 128K (row E). As with the MMU test, a '1' means an error in the corresponding memory chip. The results are saved as a word in memory with each bit corresponding to the memory chip for that bit as follows:



$188-$189 : bit pattern for logical addresses 128K - 256K (row D).
$18A-$18B : bit pattern for logical addresses 256K - 384K (row C).
$18C-$18D : bit pattern for logical addresses 384K - 512K (row B).
$18E-$195 : reserved for second 512K of memory (located on second board)
$196-$1A5 : reserved for future use
$1A6-$1A9 : parity error address (if error during memory test)
$1AA-$1AB : memory error address latch contents (physical address)*
$268-$26B : suspected (logical) error address for parity error
$26C-$26F : data written to suspected error address
$270-$273 : actual (logical) error address found during parity error search
$274-$277 : data read on parity error during search
$278-$27B : (physical) address read from parity error address latch
$27C : error row (0-7) if parity chip failure, where 0 = first 128K memory row (board 2, row E), 7 = last 128K memory row (board 1, row E)
$27D : error column (9 or 14) if parity chip failure
$294-$297 : Maximum memory address + 1

$2A4-$2A7 : Minimum memory address
$2A8-$2AB : Total amount of memory
$2AD : Board # in error if memory error detected

*Note: The memory error address saved must be shifted left by 5 to convert to the corresponding Lisa memory address. The low order 5 bits can be assumed to be zero.

2.11) I/O Board Test Completion

A test of the disk controller is performed to check the state of the DSKDIAG bit ensuring that the disk controller is ready. Then a read of shared memory is done to retrieve the results of the disk controller self-test, which is left in location $FCC003. If the resulting code is zero (no errors), a write test of shared memory is also done, followed by the issuing of a command to disable interrupts from both drives. If any errors are detected, an error '57' will be displayed. During this testing, the disk controller ROM id is also read and displayed next to the boot ROM id in the upper right corner of the Lisa screen. The id is also decoded and saved as a system type id at location $2AF.

A test of the Zilog SCC chip is tried next, using the internal loopback feature of this chip to check read/write of data for port B. Before doing this test, a check is also made to ensure read/write capability to/from the SCC interrupt vector register via channel A, and then the chip is set up to do asynchronous data transmission at its max baud rate. If any errors are found, an I/O board error icon is displayed along with error code '55' for port A errors or '56' for port B errors. In addition, the specific error is stored at memory location $2AC as follows:

$01 - Interrupt vector register read/write error (port A error)
$02 - Channel B transmit buffer timeout error
$04 - Channel B receive buffer full timeout error
$08 - Channel B data compare error

Finally a double "click" of the speaker is done to indicate that the keyboard is about to be scanned. The ROM does the scan to read any keyboard input that might have been input from the user (such as an alternate boot command). Next a read clock command is attempted, and the time and date information returned are saved. If any errors are encountered, an I/O board error icon is displayed along with one of the following error codes:

'52' - I/O board COPS error
'54' - Clock error

2.12) I/O slot Configuration Check

Each I/O slot is scanned for the presence of an installed card and, if found, the card id is read and saved in memory. The card id is a 16 bit number with the following format:

bsit nnnn nnnn nnnn

where

b = 1 if the card is bootable
s = 1 if the card has a status program
i = 1 if the card has an icon(s) to be displayed in the boot menu
t = 1 if the card is a test card
n = 12 bit card specific id number

If the card id indicates it has a status routine, a scan of its required onboard ROM is done to ensure that it can be read properly and the status routine is then executed. If any errors are encountered, they are displayed with an I/O slot card error icon, indicating which slot is in error, and an error code '92' for a ROM checksum error, or '93' for a board status routine error. In addition, the error code returned from the card's status routine is saved at memory location $1B5.

3.0 BOOT PROCEDURES

As mentioned previously, the ROM supports booting from a variety of devices, with the current version supporting either an 871 or SONY floppy drive, a Profile type hard disk builtin or attached to the parallel port (default), or any I/O slot card containing its own "boot ROM". The ROM checks to see if an alternate boot is desired by checking to see if valid keyboard input was detected. The following keyboard sequences are supported:

APPLE/1 - boot from upper builtin drive (871 floppy or hard disk)
APPLE/2 - boot from lower builtin drive (871 or SONY floppy)
APPLE/3 - boot from Profile attached to parallel port or builtin hard disk (default)
APPLE/4 - boot from I/O slot #1 (farthest from logic cards), lower port
APPLE/5 - boot from I/O slot #1, upper port
APPLE/6 - boot from I/O slot #2, lower port
APPLE/7 - boot from I/O slot #2, upper port
APPLE/8 - boot from I/O slot #3, lower port
APPLE/9 - boot from I/O slot #3, upper port
APPLE/ENTER (on numeric keypad) - abort boot, go to ROM "monitor"
APPLE/SHIFT/P - abort boot, go into power-cycling mode

This input is done by holding the 'APPLE' key down while depressing the other key(s) in the sequence. This must be done between the first and second clicks of the speaker during power-up, a window that lasts from 8-36 seconds (depends on amount of system memory and memory test mode).

If one of the boot sequences are detected, a "wait" icon is displayed and the ROM starts the booting process. For the floppy or hard disk this entails reading of block 0 from the boot device and then checking to see if the block has a valid "boot file id" of $AAAA. If invalid, a boot error icon is displayed along with the drive in error as appropriate (see Appendix A for details). Otherwise, the ROM transfers control to the loaded program (at address 128K) which then has responsibility to complete the booting process.

For I/O slot booting, the boot ROM checks for a valid boot ROM on the selected I/O card (see Hardware manual for details) and executes a status routine if one is present. If all is OK, the card's boot routine is executed by first loading it starting at address 128K and then jumping to it.

Direct access to the ROM monitor is provided with the APPLE/ENTER key sequence primarily for debugging purposes. Details of its operation are

explained in section 4.0.

Power-cycling mode is intended for manufacturing use to aid in burn-in testing of new systems. It is described in Appendix C.

If no alternate keyboard commands are detected, the ROM next scans to see if any key other than the alpha-lock or mouse button was depressed. If yes, a "boot menu" is displayed to the user containing choices for all of the valid boot devices. This includes a builtin floppy or hard disk drive, a Profile if attached and ready, and any I/O slot card that may be present. To check for a hard disk, the ROM tries a handshake over the parallel port. If a hard disk is attached, but in its "warm-up" mode, a wait icon is displayed for a maximum of 100 seconds before the full handshake is attempted.

The left side of the boot menu shows the device, and the right side displays a keyboard sequence that can be used to make the selection (the "apple" shown in the menu refers to the APPLE key on the keyboard). The boot device can then be selected in the standard LISA user interface fashion with the mouse, or via an alternate keyboard sequence which is displayed in the menu alongside the boot icon.

If no keyboard input was detected, the ROM does a check of parameter memory for a valid boot device id. If found, the boot takes place from the specified device. If not, a final alternate boot check is done by scanning the I/O slots. If both a test card (with id in the range hex 0-7FF) and an Applenet card are installed, the ROM boots from the Applenet card. If only the test card is installed, the ROM boots from it if it is bootable. Otherwise, the ROM assumes booting will take place from a Profile type hard disk attached to the parallel port (or builtin) and proceeds with an attempt to boot from there. For Lisa 2 systems (no internal hard disk), if no hard disk is connected, the ROM will next try the Sony floppy drive, and if no diskette is inserted will beep the speaker once and request a diskette. Otherwise, the "boot menu" is displayed accompanied by a speaker beep to alert the user.

If invalid keyboard input is detected, or the parameter memory setting is invalid, the ROM beeps the speaker and displays the "boot menu".

4.0 Monitor Operation

If any errors are encountered during power-up testing, or an APPLE/ENTER key sequence is detected, the ROM enters what is called its "monitor" mode. This actually consists of two parts as explained below.

4.1) Customer mode

In this preliminary mode, the error icon and code, if any, are displayed on the screen in an "alert box", along with "buttons" labeled as follows:

RESTART
CONTINUE
STARTUP FROM...

The "buttons" also contain an alternate keyboard sequence that can be used to activate them, which is displayed as an "apple" icon (standing for the APPLE key), followed by the number 1, 2 or 3. A selection can then be made by using the mouse and "clicking" once on the button, or by entering the alternate keyboard sequence. Invalid input causes a speaker "beep" and redisplay of the alert box.

Note: The ROM contains French and German translations for Customer mode and boot menu phrases. The language displayed depends on the id of the keyboard attached to the system. For other countries, such as Italy and Spain, all three languages are displayed.

4.1.1) Restart

This option is provided as a means of resetting the system from the keyboard to restart the power-up tests. It causes the ROM to do a "cold-start" of the system.

4.1.2) Continue

This option provides a means of continuing from a non-fatal error. It is displayed only when the error is of this type. The following errors are considered "non-fatal":

a) MMU error not in context 0
b) Serial number read error
c) Serial port error
d) Clock error
e) Read/write memory error not in the first 2K or the boot area (128K-256K)
f) I/O slot card errors
g) Most boot errors

Note: The CONTINUE button is also not displayed when the monitor is invoked directly from the keyboard by the APPLE/ENTER key sequence.

4.1.3) Startup From...

In the case of a boot failure, this option provides the means of easily retrying the boot from the same or another device. When it is selected, the "boot menu" is displayed as described in section 3.0.. A selection can then be made from this menu to try the desired boot device.

4.2) Service Mode

A fourth, unlisted option, is actually also available to the user from customer mode. It is invoked by entering a APPLE/S key sequence which causes the ROM to enter "Service mode".

This mode is provided primarily for engineering, manufacturing and field service use. It provides basic "peek and poke" capabilities, along with several additional diagnostic aids to help in debugging system failures. When invoked, the screen is cleared and a new "pull-down" menu is displayed with the following choices:

```
DISPLAY MEM
SET MEMORY
CALL PROGRAM
LOOP ON TEST
ADJUST VIDEO
POWER CYCLE
QUIT
```

Also displayed is a "window" labeled "Service Mode", which is used as an output area for those options that need it. As with the boot menu, these options have associated alternate keyboard sequences that can be used to activate them. In this case, however, the input does not need to be preceeded by the APPLE key, only the number displayed along side the desired option need be input. The options are described in the following paragraphs.

Invalid input causes a speaker "beep" and the message 'WHAT?' displayed in a dialog box on the screen. In most cases, hitting just the return key in response to a prompt will return control to the menu.

4.2.1) Display Mem

This option gives the user the capability of displaying the contents of system memory. Upon invoking, two additional prompts are sequentially displayed to request the starting address to display and a hex count of the number of bytes to display. The address must be input as 1-8 hex characters followed by either the "space" or "return" key. If the space key is hit, the count data can then be entered, otherwise the system will prompt for it if the return key is hit. The count must also be in hex and only the first four characters input are used. This gives a maximum display capability of 64K bytes, and the ROM rounds out the count to the next 16 byte boundary. The default is to display 16 bytes, which can be requested by simply hitting the return key in response to the "COUNT ?" prompt.

4.2.2) Set Memory

Setting of memory is enabled via this selection. Again, prompts are displayed to request a starting address and the data to write to that address. As with the display option, the address must be 1-8 hex characters and the address and data input can be separated by either a space or the return key. If the return key is hit, the system will display the prompt "DATA ?". The data must also be input as hex, but can be entered as more than just one "string". By separating input strings with spaces, the ROM will write to successive addresses with the data entered. In addition, the set operations performed can be byte, word or long operations, with the type determined by the length of the input string. For example, the input string 12 34 5678 9ABCDEF0 in response to the "DATA ?" prompt would cause a write of bytes at hex addresses 100 and 101, a word write at address 102, and a long write at address 104.

4.2.3) Call Program

This option provides the ability to invoke a routine at a specific address. An address value is requested as with the previous options, and the ROM then executes a JSR to the address entered. If the called routine properly executes an RTS when done, control is returned to ROM service mode which saves the contents of the registers and then returns to the service mode menu display.

Before doing the JSR to the input address, the ROM first initializes data registers D0-D7 and address regsiters A0-A5 by loading them from the "register save area" which is located at the following addresses:

$1C0-$1DF: Registers D0-D7 (one long word for each register)
$1E0-$1F7: Registers A0-A5 (" " " " " ")

Following a return from the called program, the registers are also saved in this area.

4.2.4) Loop on Test

This option allows any of the ROM diagnostics to be invoked. When invoked, it displays the lists of tests available and prompts for a selection. When the desired test is entered, the ROM displays a "testing window" showing which part of the system is being tested and then goes into an infinite loop on that test. The loop can only be

terminated by either hitting the reset button or powering the system on and off. An exception to this is the memory test which will terminate on a parity error.

4.2.5) Adjust Video

This option displays a "crosshatch" pattern on the screen for use in adjusting the video board circuitry for proper display. Hitting any key, or the mouse button, will terminate the display and return to the Service mode menu.

4.2.6) Power Cycle

This option provides a means of invoking power-cycling from service mode. It is described in detail in Appendix C.

4.2.7) Quit

This option will cause a return to the Customer mode.

5.0 ROM JUMP TABLE

A variety of ROM routines can be used by other software through a jump table located at address $0084 in the ROM space. The following is a list of the routines available:

Address	Routine
$XX0084	ROM Monitor
$XX0088	Display message
$XX008C	Write to MMU registers
$XX0090	Read hard disk block (via polling)
$XX0094	Read floppy sector (via polling)
$XX0098	Basic pseudo-random memory test
$XX009C	Reserved
$XX00A0	Reserved
$XX00A4	Read MMU registers
$XX00A8	Send COPS command
$XX00AC	Read clock/calendar setting (via polling)
$XX00B0	Display hex error code in decimal
$XX00B4	Set Contrast level
$XX00B8	Beep speaker
$XX00BC	Verify Checksum
$XX00C0	Write Checksum
$XX00C4	Read system serial number

Note that 'XX' in the ROM address depends on what MMU register is used to enable access to ROM space. The ROM sets up register 127 for this, which gives XX=FE. Refer to appendix C for details on the specific routine interfaces.

APPENDIX A: ERROR MESSAGES/TONES

The ROM outputs three types of error indicators: 1) a general icon indicating the 'global' nature of the error, 2) a more specific error code attempting to pinpoint the error, and 3) an error tone also identifying the error type. The general icons include the following:

CPU board
I/O board
Memory board
I/O slot card
Entire LISA system
Diskette
Keyboard
Mouse
Profile

The error codes implemented are as follows:

ERROR CODE	MEANING
23	Unable to read diskette
25	Unable to unclamp diskette
38	No boot file on diskette
39	Disk controller timeout error
40	MMU error
41	CPU selection logic error
42	Video circuitry error
43	Parity circuitry error
44	Unexpected NMI interrupt
45	Bus error
46	Address error
47	Other unexpected exception
48	Illegal instruction error
49	Line 1010 or 1111 trap
50	COPS VIA error
51	Parallel port VIA error
52	I/O board COPS error
53	Keyboard COPS error
54	Clock error
55	RS232 port A error
56	RS232 port B error
57	Floppy disk controller error

58	I/O board access error
59	I/O board COPS code error
60	I/O or keyboard error
61	Unable to initialize clock (power-cycling mode only)
62	Unable to set alarm (power-cycling mode only)
70	Memory read/write error
71	Memory parity error
75	Boot failure (boot file on device probably bad)
80	Hard disk not attached
81	Hard disk not ready
82	Bad response from hard disk
83	Non-zero status bytes returned from hard disk
84	Invalid boot file on hard disk
85	Hard disk timeout error
90	No I/O slot card installed
91	I/O slot card not bootable
92	Bad checksum on I/O slot card
93	Bad status returned from I/O slot card

The error tones possible are as follows:

ERROR TONE	MEANING
Lo	No memory detected
Hi	Default boot device not available
Lo, Lo	Error in first 2K memory test
Lo, Hi	General system error (e.g., bus error)
Lo, Lo, Hi	CPU board error
Lo, Hi, Lo	I/O board error
Hi, Lo, Hi	Keyboard error
Lo, Hi, Hi	General memory error
Hi, Lo, Lo	I/O slot error
Hi, Hi, Lo	Keyboard or mouse disconnected

Appendix B: ROM DATA SAVE AREAS

This section gives a complete listing of information saved by the boot ROM in LISA memory:

ADDRESS	CONTENTS
$180-183	Power-up status (0 = ok, any bit = '1' indicates error) Bit 0 - MMU error 1 - CPU selection logic error 2 - Video error 3 - Parity logic error 4 - Unexpected NMI interrupt 5 - Bus error 6 - Address error 7 - Miscellaneous exception (e.g., divide-by zero) 8 - Illegal instruction error 9 - Line 1010 or 1111 error 10 - Keyboard VIA error 11 - Parallel port VIA error 12 - I/O board COPS error 13 - Keyboard COPS error 14 - Clock error (can't read) 15 - RS232 port A error 16 - RS232 port B error 17 - Floppy disk interface error 18 - I/O card access error (bus error occurred) 19 - COPS reset code error 20 - I/O or keyboard error (can't determine which) 21 - Memory R/W error 22 - Memory parity error 23 - Keyboard disconnected 24 - Mouse disconnected 25 - I/O slot 1 error 26 - I/O slot 2 error 27 - I/O slot 3 error
$184-185	Memory sizing error results
$186-$196	Results of memory read/write tests (1 word/128K of memory)
$196-$1A5	Reserved for second megabyte of memory
$1A6-$1A9	Parity error memory address
$1AA-$1AB	Memory error address latch

$1AC-$1AF Save of reg D7 on exception errors
$1B0-$1B1 Results of MMU tests (context/data bits)
$1B2 Keyboard id
$1B3 Boot device ID
- $00 - Upper 871 floppy or builtin hard disk drive
- $01 - Lower 871 or SONY floppy drive
- $02 - Profile attached to parallel port or builtin hard disk (default)
- $03 - I/O slot 1, port 1
- $04 - I/O slot 1, port 2
- $06 - I/O slot 2, port 1
- $07 - I/O slot 2, port 2
- $09 - I/O slot 3, port 1
- $0A - I/O slot 3, port 2
- $0F - Power cycling
- $10 - Abort to ROM Monitor

$1B4 Device dependent error code

Floppy:
- 01 - Invalid command
- 02 - Invalid drive
- 03 - Invalid side
- 04 - Invalid sector
- 05 - Invalid track
- 06 - Invalid mask byte
- 07 - Disk not clamped
- 08 - Drive disabled
- 09 - Interrupts pending
- 10 - Invalid format parm
- 11 - ROM test error
- 12 - Random IRQ, NMI or BRK
- 20 - Disk write protected
- 21 - Unable to verify data
- 22 - Unable to clamp
- 23 - Disk read error
- 24 - Disk write error
- 25 - Unable to unclamp
- 26 - Unable to adjust disk speed (no speed marks on disk)
- 27 - Unable to adjust disk speed within timeout (found marks)
- 28 - Unable to write speed track
- 29-37 - Reserved
- 38 - Bad Header (for boot)
- 39 - Timeout error (for boot)

Hard Disk
- 80 - Not attached
- 81 - Not ready
- 82 - Unexpected response
- 83 - Nonzero status bytes
- 84 - Bad Header (for boot)
- 85 - Timeout error

I/O Slot
- 90 - Card not installed or can't access
- 91 - Invalid boot id
- 92 - Bad checksum
- 93 - Bad status

$1B5-$1B9	Device dependent error data		
	871 floppy	Hard Disk	I/O Slot
$1B5	Addr checksum errors	Error byte #1	Reserved
$1B6	Data checksum errors	Error byte #2	Reserved
$1B7	Retry count	Error byte #3	Reserved
$1B8	Reserved	Error byte #4	Reserved
$1B9	Reserved	Reserved	Reserved

$1BA-$1BF	Clock setting (Ey/ddd/hh/mm/ss/t format)
$1C0-$1C3	Register D0
$1C4-$1C7	Register D1
$1C8-$1CB	Register D2
$1CC-$1CF	Register D3
$1D0-$1D3	Register D4
$1D4-$1D7	Register D5
$1D8-$1DB	Register D6
$1DC-$1DF	Register D7 - Overall test result indicator ('1' bit is error). Bits same as power-up status value.
$1E0-$1E3	Register A0
$1E4-$1E7	Register A1
$1E8-$1EB	Register A2
$1EC-$1EF	Register A3
$1F0-$1F3	Register A4
$1F4-$1F7	Register A5
$1F8-$1FB	Register A6
$1FC-$1FF	Register A7 (User stack pointer)

$240-$25F	System serial number (format varies)
$260-$267	Scratch area
$268-$26B	Suspected (logical) error address for parity error
$26C-$26F	Data written to suspected error address
$270-$273	Actual (logical) error address found during parity error search
$274-$277	Data read on parity error during search
$278-$27B	(Physical) address read from parity error address latch
$27C	Error row (0-7) if parity chip failure, where 0 = first 128K memory row (board 2, row E), 7 = last 128K memory row (board 1, row E)
$27D	Error column (9 or 14) if parity chip failure

$280-$28F	Exception data save area
$280-$281	Function code
$282-$285	Exception address
$286-$287	Instruction register
$288-$289	Status register contents
$28A-$28D	PC at time of exception
$28E-$28F	Reserved

$290-$293	Supervisor stack pointer
$294-$297	High physical memory address + 1
$298-$299	I/O slot 1 card id
$29A-$29B	I/O slot 2 card id
$29C-$29D	I/O slot 3 card id
$29E-$2A0	Reserved
$2A1	Save of disk controller ROM id
$2A2-$2A3	Reserved
$2A4-$2A7	Low physical address
$2A8-$2AB	Total amount of memory
$2AC	SCC test results
$2AD	Memory board in error
$2AE	Results of disk controller self-test (0=no error)
$2AF	System type: 0 = Lisa 1 1 = Lisa 2 with Sony, old I/O board (slow timers) 2 = Lisa 2 with Sony, new I/O board (fast timers) 3 = Lisa 2/10 with Sony, new I/O, internal disk
$2B0-$2BF	COPS reset codes and keyboard input
$2C0-$400	ROM scratchpad/stack area

In addition, the ROM uses the following areas of parameter memory:

$FCC189	Boot device id (upper 4 bits)
$FCC18D	Mouse on/Extended Memory test indicators (bit 7 = 1 means mouse should be connected) (bit 6 = 1 means test all of memory)
$FCC1FD	Parameter memory validity checksum byte #1
$FCC1FF	Parameter memory validity checksum byte #2

Appendix C: External Interfaces

These pages give details on the various ROM routines available for use by external software. Unless specified otherwise, all registers are preserved. Addresses are given with 'XX' prefix since the exact address depends on the system MMU setting for ROM access. The boot ROM default is XX = $FE.

NAME: ROM "Monitor"

FUNCTION: This is an entry point into the "Customer mode" of the ROM monitor. It is intended to be used as an exit point for fatal errors during booting or other cases where LISABUG is not available. Registers are saved upon entry and display is possible for icons, error codes, and messages to the screen. Assumes stack pointer set to area that won't conflict with ROM low memory usage ($0 - $600), and MMU set to allow ROM read/write access to that area.

INPUTS REQUIRED:
D0 = error code, or 0 for no error code (lower word displayed, leading 0's suppressed)
A2 = ptr to icon*, or 0 for no icon (MSB = 1 if icon not compressed)
A3 = ptr to message for display, 0 for no message (must follow guidelines of display message routine)

Memory location $110 = base address of screen in logical range of 0 - 1MB

*Standard Lisa icon is a 6 byte (48 bit) wide by 4 byte (32 bit) high bitmap. ROM uses a proprietary algorithm to compress these icons in order to save ROM space. Software not using this algorithm should set the address MSB = 1 to indicate an uncompressed icon.

OUTPUTS: Icon, message and/or error code displayed. Video page set according to $110 setting, monitor "Customer mode" entered.

CALLING SEQUENCE: Jump to address $XX0084, no return to caller.

NAME: Display message

FUNCTION: Enables display of messages to Lisa screen using ROM character font. The video page must be preset by the calling program if not already set. The supported character set includes all upper case alpha characters, digits 0-9, space, carriage return, period, minus sign and /.

INPUTS REQUIRED: A3 = address of ASCII message terminated by a 0 byte.
D4 = column for left margin if message includes a CR
D5 = display row (0 - 31 decimal)
D6 = display column (0 - 88 decimal)

Memory location $110 = base address of screen in logical address range 0 - $F8000.

OUTPUTS: D5, D6 updated to new display position according to message.

CALLING SEQUENCE: JSR to address $XX0088.

NAME: Write MMU Registers

FUNCTION: This routine provides the ability to set two MMU registers at a time followed by updating of the register addresses if desired. The context must be preset by the calling program.

INPUTS REQUIRED: A2 = address of 1st MMU register
A3 = address of 2nd MMU register
A4 = return Address
A5 = address increment for MMU addresses
D0 = data value for 1st MMU register
D1 = data value for 2nd MMU register

OUTPUTS: A2, A3 updated by address increment.

CALLING SEQUENCE: Load A4 with return address, then jump to address $XX008C.

NAME: Read Hard disk Block

FUNCTION: This routine enables reading of 1 block from a Profile type hard disk attached via a parallel port. Uses polling to do read, disables interrupts during operation.

INPUTS REQUIRED: D1 = block to read
D2 = timeout count for hard disk ready (1 count = 10.6us)
D3 = retry count
D4 = threshold count
A1 = address to load header (20 bytes)
A2 = address to load data (512 bytes)

OUTPUTS: Carry bit set if error.
D0 = error code as follows:
00 = no error
80 = Hard disk not attached
81 = Hard disk not ready
82 = Unexpected response
83 = Nonzero status bytes
85 = Timeout error
D1 = error bytes (see Profile spec for format)

CALLING SEQUENCE: JSR to address $XX0090.

DESTROYS: D0, D1 and A0.

NAME: Read Floppy Sector

FUNCTION: Reads one sector from specified floppy drive via polling.
Disables interrupts during operation.

INPUTS REQUIRED: D0 = speed
D1 = drive/side/sector/track
D2 = read timeout (each count = 9.5us)
A0 = base address of disk shared memory
A1 = address to load header
A2 = address to load data

OUTPUTS: Carry bit set if error.
D0 = error code as follows:
00 = no error
XX = standard Floppy error codes (see Appendix A)
39 = timeout error

CALLING SEQUENCE: JSR to address $XX0094.

DESTROYS: A0 and D0.

NAME: Basic Memory Test

FUNCTION: Tests memory using a pseudo-random pattern. Memory range is completely written with pattern, then read back and verified starting from first address.

INPUTS REQUIRED: A0 = starting address to test
A1 = ending address (range must be multiple of 4 bytes)
A4 = return address

OUTPUTS: Zero condition code bit set if no error.
D3 = OR mask of errors, 0 if no error.

CALLING SEQUENCE: Load A4 with return address, jump to address $XX0098.

NAME: Read MMU Registers

FUNCTION: Enables reading of a pair of MMU registers in an "auto-increment" fashion.

INPUTS REQUIRED: D2 = context to read (0 - 3)
A2 = address of 1st MMU register
A3 = address of 2nd MMU register
A4 = return address
A5 = Increment for MMU addresses

OUTPUTS: D0 = contents of 1st MMU register
D1 = contents of 2nd MMU register
A2, A3 updated by A5 value

CALLING SEQUENCE: Load A4 with return address, then jump to address $XX00A4.

DESTROYS: D3

NAME: Send COPS command

FUNCTION: Sends command byte to COPS that controls keyboard and mouse interfaces. Disables interrupts during operation.

INPUTS REQUIRED: D0 = command to send.

OUTPUTS: Carry bit set if timeout error.

CALLING SEQUENCE: JSR to address $XX00A8.

NAME: Read Clock/Calendar

FUNCTION: Reads setting of COPS clock/calendar via polling. Disables interrupts during operation.

INPUTS REQUIRED: None.

OUTPUTS: Saves setting at following addresses:

$1BA = $Ey
$1BB = dd
$1BC = dh
$1BD = hm
$1BE = ms
$1BF = st

where y = year, ddd = days, hh = hours, mm = minutes, ss = seconds, and t = tenths.

CALLING SEQUENCE: JSR to address $XX00AC.

DESTROYS: A0-A2,D0,D1

NAME: Display hex error code in decimal.

FUNCTION: Displays error code as decimal value on Lisa screen. Assumes screen position set according to value in memory location $110.

INPUTS REQUIRED: D0 = code to display (lower word displayed)
D5 = pixel row for display (0 - 310 decimal)
D6 = byte col for display (0 - 88 decimal)

OUTPUTS: D5 updated to display row + 10.
D6 set for column 1.

CALLING SEQUENCE: JSR to address $XX00B0.

NAME: Set Contrast.

FUNCTION: Sets screen contrast with input value.

INPUTS REQUIRED: D0 = contrast value (00 - $FF, 00 = full on)
A4 = return address

OUTPUTS: None.

CALLING SEQUENCE: Set return address in register A4, then jump to address $XX00B4.

DESTROYS: A0

NAME: Beep Speaker

FUNCTION: Produces square wave from speaker according to input parameters.

INPUTS REQUIRED: D0 = desired frequency (00 - $AA, $AA = 333 hz)
D1 = duration (0 = .5 ms, each additional count = .5ms)
D2 = volume (0, 2, 4, ..., $E; 0=lowest)

OUTPUTS: Tone from speaker.

CALLING SEQUENCE: JSR to address $XX00B8.

NAME: Verify Checksum

FUNCTION: Computes checksum of memory or shared memory contents and compares to expected result which is assumed to be last word of the memory area read.

INPUTS REQUIRED: A0 = starting address to verify
D0 = number of words - 1
D1 = 0 for regular memory (uses MOVE.W for access)
= nonzero for shared memory (uses MOVEP for access)

OUTPUTS: Carry bit set if error.
D2 = expected checksum (last word read)
D3 = computed checksum

CALLING SEQUENCE: JSR to address $XX00BC.

NAME: Write Checksum

FUNCTION: Computes checksum of shared memory contents (with MOVEP instruction) using a 16 bit add and rotate algorithm, and then writes inverse of result to last word of specified shared memory area.

INPUTS REQUIRED: A0 = shared memory address (e.g., base address of parameter memory)
D0 = number of words - 2 of shared memory area (e.g., 30 for parameter memory)

OUTPUTS: None, result written in shared memory area

CALLING SEQUENCE: JSR to address $XX00C0.

NAME: Read system serial number

FUNCTION: Reads Lisa system serial number stored in video prom.

INPUTS REQUIRED: A0 = address for save of serial number

OUTPUTS: Carry bit set if read OK. Serial number saved in low nibbles of 32 consecutive byte field. Format varies according to date of system.

CALLING SEQUENCE: JSR to location $XX00C4.

DESTROYS: D0-D6, A1-A6

Note: ROM also automatically saves system serial number during power-up in memory locations $240-$25F.

Appendix D: POWER CYCLING

This mode is provided as an aid for manufacturing in doing system testing. Invoked by using the APPLE/SHIFT/P key sequence, it causes the ROM to continually cycle through its various power-up diagnostics, resetting the system after each pass to begin a new cycle. In addition, the ROM causes the system to power down after a specified time interval, with the COPS initialized so that the system will also "wake-up" after the same selected interval. The default for this is 1 hour (60 minutes), but this can also be "toggled" to a faster interval of 3 minutes on/3 minutes off by simply hitting the mouse button between the two speaker clicks. Hitting the mouse button during a later pass of the diagnostics will toggle the interval back to the 60 minute default.

A variety of messages are displayed in the power-cycling mode. At the end of each pass, the system will pause for about 5 seconds with the following messages on the screen in an "alert box":

POWER CYCLING AT XX
TIME IS A BB CC DD

where XX = the power on/off minute interval in decimal, and the time is shown as A=day, BB=hours, CC=minutes, and DD=seconds. On the first pass of power cycling the clock is set as day 1, time 00:00:00, so the run time message gives an indication of how long the test has been in progress.

A test of the floppy drive(s) is also performed during power-cycling, and requires a formatted diskette inserted in the drive. The ROM attempts to simulate expected normal system operation by exercising each drive for approximately 10 seconds every 2 minutes. This is done by checking the run time after each pass, and if 2 or more minutes have elapsed, the ROM executes a test that reads the first sector of every track on the top side of each disk. If any errors occur, the test is terminated and the message "FLOPPY TEST FAILED" is displayed. In addition, the message

FLOPPY ERROR COUNT IS XXXX

is always displayed at the end of this test, where XXXX = a running hex count of floppy test errors.

Power-cycling mode must be terminated by inputting the APPLE/ENTER key sequence, and will result in entry to ROM Customer monitor mode with the following messages displayed on the screen:

LOOP COUNT IS XXXX
TIME IS A BB CC DD
FLOPPY ERROR COUNT IS YYYY

Also displayed are the Customer mode option "buttons" since control is returned to the ROM at this point. As before, XXXX is a running hex count of the number of passes executed, the run time is expressed as days, hours, minutes and seconds elapsed since the testing was first started, and YYYY is the running hex count of floppy test errors.

Note: If the APPLE/ENTER method is not used to terminate power cycling, the parameter memory indicator for power cycling will not be erased and the system will start cycling again the next time it is powered up.

The following areas of parameter memory are used by the ROM when doing power cycling:

$FCC189	power cycling indicator = $CX, X = don't care
$FCC18D	memory test indicator (bit 6 = 1 for test all memory)
$FCC191	first pass flag (01 = no)
$FCC193	save of last hour value read
$FCC195	loop count high byte
$FCC197	loop count low byte
$FCC199	hour save needed flag (01 = no)
$FCC19B	save of last minute value read
$FCC19D	floppy error count high byte
$FCC19F	floppy error count low byte
$FCC1A1-1A7	save of last clock values read
$FCC1B1	save of last alarm value sent to COPS
$FCC1C1	minute counter for power cycling
$FCC1C3	cycling interval (in minutes)
$FCC1C5	minute counter for floppy testing

Appendix E: BOOT ROM CHECKSUM/ASSEMBLY PROCEDURE

The latest boot ROM's have the following Data I/O reported checksums:

Part #	Checksum
341-0175-H (high bytes)	7D42
341-0176-H (low bytes)	DA32

The boot ROM source code is contained in a series of 68000 assembly language text files. These files have the following names:

RMXXX.E.TEXT - comments and equates
RMXXX.K.TEXT - kernel diagnostic tests
RMXXX.S.TEXT - secondary diagnostic tests
RMXXX.B.TEXT - boot routines
RMXXX.M.TEXT - ROM monitor routines
RMXXX.G.TEXT - graphics routines

where XXX = internal ROM id (e.g., 248 for version 2, rev H (ASCII '48')).

To create a master ROM from these files, do the following:

1) A)ssemble the file RMXXX.E.TEXT to create a .CODE file called RMXXX.CODE, where XXX is defined as above. A list file can also be created if desired in the usual manner.

2) X)ecute the program CHECKSUM to compute the checksum for the ROM. Respond with the RMXXX name to both program prompts to create a .CODE file with the correct checksum.

If using an Apple][- Data I/O System 19 Prom Burner combination do the following:

3) Transfer the resulting RMXXX.CODE file to an Apple][diskette and place in the second drive of the Apple][connected to the Data I/O System 19 Prom Burner.

4) Turn on the Data I/O if not already on, and press "SELECT", "F1" (from the keyboard), and "START". The DATA I/O display should show 4 rows of horizontal lines. Note that the Data I/O UNIPAK module must also be installed if not already done.

5) On the Apple][, X)ecute the program BLOADER, and enter #5:RMXXX to the source file prompt.

6) Enter "N" to the AUTO PROGRAM? prompt.

7) Enter "N" to the SPLIT THE BYTES? prompt.

8) Enter "3FFF" to the ENDING ADDRESS[HEX]: prompt.

9) The loading of the ROM code file is shown by rapidly changing digits on the display of the Data I/O. Wait until fully loaded (Apple][screen will say "DONE").

10) On the Data I/O, press "SELECT", "A5", "ENTER", "2000", "START". This will split the loaded code file into high and low byte groups. High bytes will reside at locations 0-$1FFF of the Data I/O memory, with low bytes at $2000-$3FFF.

11) Press the "PROG" key, and enter "35" to the F prompt on the Data I/O display, and "33" to the P prompt.

12) Insert a blank 2764 EPROM into the socket indicated by the lit LED and then press "START".

13) When the Data I/O signals complete label the ROM with version number, letter "H" for high bytes, and the checksum as shown on the Data I/O display. Remove and set aside this EPROM.

14) Press "BLOCK LIMITS", "2000", "ENTER", "2000", "ENTER, "0", "ENTER", "SELECT", "A7", "START". This sequence moves the low byte group to 0-$1FFF memory area so it can be burned into an EPROM.

15) Press the "PROG" key, insert a blank 2764, and then press "START".

16) When this burn is done, label the ROM with version number, "L" for low bytes, and the checksum value from the Data I/O display.

Steps 3-16 can also be done using a Lisa - Data I/O System 29A Prom Burner combination as follows:

3). Transfer the resulting RMXXX.CODE file to a Lisa diskette and place in the Lisa connected to a Data I/O System 29A Prom Burner.

4) Turn on the Data I/O if not already on, and press "SELECT", "F1" (from the keyboard), "START", "START". The DATA I/O display should show "REMOTE MODE 0". Note that the Data I/O UNIPAK 2 module must also be installed if not already done.

5) On the Lisa, X)ecute the program BLOADER, which will give the initial

prompt "(D)ownload, (U)pload, (T)alk, Exit[CR]? ". Enter "D" for download.

6) Enter the appropriate "RMXXX" code file name to the "Enter filename[.code]" prompt.

7) Enter "16384" to the "Enter number of bytes (in decimal) to transfer" prompt. The Lisa will return the message "Romsize in hex = 4000".

8) The loading of the ROM code file is shown by a rapidly changing "clock" symbol on the display of the Data I/O. Wait until fully loaded (Lisa screen will redisplay initial prompt).

9) Hit "return" key on Lisa. Data I/O display should say "READY".

10) On the Data I/O, press "SELECT", "A5", "START", "2000", "START". This will split the loaded code file into high and low byte groups. High bytes will reside at locations 0-$1FFF of the Data I/O memory, with low bytes at $2000-$3FFF.

11) Enter the key sequence "COPY", "RAM", "0000", "START", "2000", "START", "DEVICE", "0000", "START". This instructs the Data I/O to copy the data loaded into RAM from address 0-1FFF hex into a prom device. The Data I/O display will then ask for a family/pinout code for the device to be used. For AMD parts, enter the sequence "79", "33". For Hitachi parts enter "AF", "33". For any other parts, refer to the Data I/O documentation for the correct codes to use.

12) Insert a blank 2764 EPROM into the socket indicated by the lit LED and then press "START".

13) When the Data I/O signals complete label the ROM with version number, letter "H" for high bytes, and the checksum as shown on the Data I/O display. Remove and set aside this EPROM.

14) Press "COPY", "RAM", "2000", "START", "2000", "START", "DEVICE", "0000", "START", followed by the family/pinout codes if using a different EPROM then the initial one used. This sequence copies the data from memory address 2000-3FFF hex into the prom.

15) Insert a blank 2764 EPROM into the socket indicated by the lit LED and then press "START".

16) When this burn is done, label the ROM with version number, "L" for low bytes, and the checksum value from the Data I/O display.