8M20 DISK EMULATOR MANUAL

VERSION 1.3

MESA ELECTRONICS
1329-D 61st Street
Emeryville, CA 94608
(415) 547-0837

TABLE OF CONTENTS

SAFETY

INSTALLATION AND OPERATION

REFERENCE INFORMATION

SAFETY PRECAUTIONS

LITHIUM CELL

The 8M20 disk emulator card contains Lithium Cells which can create a fire or explosion hazard if improperly handled. Do not expose the Lithium Cell to temperatures in excess of 100 Degrees Celsius or dispose of in fire. Do not attempt to charge the Lithium Cell or modify the associated circuity. Do not short circuit the Lithium Cell (take care not to set the 8M20 on conductive surfaces).

Note that some antistatic bags and foam are conductive and will result in rapid cell discharge if the 8M20 is allowed to contact them. Pink antistatic materials are generally non-conductive. If in doubt, check with an ohmmeter or remove lithium cells before packaging.

HANDLING PRECAUTIONS

STATIC ELECTRICITY

The CMOS integrated circuits on the 8M20 can be damaged by exposure to electrostatic discharges. The following precautions should be taken when handling the 8M20 to prevent possible damage.

A. Leave the 8M20 in its antistatic bag until needed.
B. All work should be performed at an antistatic workstation.
C. Ground equipment into which 8M20 will be installed.
D. Ground handling personnel with conductive bracelet through 1 megohm resistor to ground.
E. Avoid wearing synthetic fabrics, particularly Nylon.

INTRODUCTION

The 8M20 is a non-volatile memory module designed as a replacement for mechanical disk drives. The 8M20 will operate reliably in environments that are unsuitable for standard disk drives. Because the 8M20 requires no maintenance, it is ideal for applications with limited accessibility.

The 8M20 can use battery backed-up CMOS RAM for general read-write use, EEPROM for read-mostly use, or EPROM for low cost read-only file use. The 8M20 can use EPROM, CMOS RAM, or EEPROM on the same card for flexibility. On board firmware emulates a standard fixed disk controller, and allows booting from the disk emulator.

The 8M20 uses 32 pin JEDEC standard memory devices and has a capacity of 3M byte using 1M bit parts and will have an ultimate capacity of 12M bytes (RAM) and 24M bytes (EPROM) when the densest 32 pin parts become available. Total disk emulation storage capacity can be increased by the addition of more 8M20 cards (up to 4 total) which can be additional drives or be concatenated into one large drive.

Write inhibit logic protects data from inadvertent damage during power-up, power-down, or program malfunction. Data integrity is verified by CRC checking. Utility programs are supplied with the 8M20 for partitioning memory and converting the disk emulator memory image to a binary files for EPROM programming.

HARDWARE CONFIGURATION

GENERAL

The 8M20 has many jumper configurable options that must be properly set to match the application. Each jumper block will be discussed separately by function. In the following discussions, when the words "up", "down", "right", and "left" are used it is assumed that the 8M20 disk emulator card is oriented with its contact fingers pointing towards the person doing the configuration.

The default configuration of 8M20's, as shipped from the factory, is as follows:

MEMORY TYPE BANK 0: 128K BYTE RAM

MEMORY TYPE BANK 1: 128 BYTE RAM

FIRMWARE ADDRESS: "HIGH" (CC000H)

UNIT #: 0

SELECT LED ENABLED

ROM OPTION: OFF

SPEED/POWER OPTION: SET FOR LOW SPEED / LOW POWER

BACKUP POWER: DISABLED (MUST BE ENABLED FOR PROPER OPERATION)

HARDWARE CONFIGURATION

. .

DEFAULT JUMPER LOCATIONS

HARDWARE CONFIGURATION

MEMORY SOCKETS

The 8M20 has twenty four 32 pin sockets that accommodate JEDEC standard memory devices of various types. These sockets are arranged as two groups of twelve sockets. The first group will be referred to as Bank 0 and consists of the top row of sockets starting with U1 and ending with U12. The second group (Bank 1) consists of the bottom row of sockets starting with U26 and ending with U37. ***When installing memory chips, the pin one end (notched end) points to the top.*** Some of the utility programs refer to "socket numbers". The following table shows the correspondence between socket numbers and U numbers.

BANK 0:

U1	SOCKET 0	U5	SOCKET 4	U9	SOCKET 8
U2	SOCKET 1	U6	SOCKET 5	U10	SOCKET 9
U3	SOCKET 2	U7	SOCKET 6	U11	SOCKET 10
U4	SOCKET 3	U8	SOCKET 7	U12	SOCKET 11

BANK 1:

U26	SOCKET 12	U30	SOCKET 16	U34	SOCKET 20
U27	SOCKET 13	U31	SOCKET 17	U35	SOCKET 21
U28	SOCKET 14	U32	SOCKET 18	U36	SOCKET 22
U29	SOCKET 15	U33	SOCKET 19	U37	SOCKET 23

HARDWARE CONFIGURATION

. .

MEMORY SOCKET LOCATIONS

HARDWARE CONFIGURATION

MEMORY TYPE JUMPERS

These jumpers are used to configure the Bank 0 and Bank 1 memory arrays for the type of memory device used, and to determine whether Lithium Cell power is supplied to the arrays. Note that bank 0 and bank 1 can be configured independently to allow mixing of memory types on one 8M20 card.

The following table gives correct jumper settings for applicable memory types. The jumper blocks have 3 pins and 2 valid jumper locations, up and down. Jumper blocks W1 through W5 configure Bank 0, while Jumper Blocks W9 through W13 configure Bank 1.

JUMPER:					
Bank 0	W1	W2	W3	W4	W5
Bank 1	W9	W10	W11	W12	W13
EPROMs:					
128K X 8	down	down	down	down	down
256K X 8	down	down	down	down	down
512K X 8	down	down	down	down	down
1024K X 8	down	up	down	down	down
5V FLASH EEPROMs:					
128K X 8	down	down	down	down	up
256K X 8	down	down	down	down	up
512K X 8	down	down	down	down	up
EEPROMs:					
128K X 8	down	down	up	down	down
512K X 8	down	down	up	down	down
RAMs:					
128K X 8	up	up	up	up	up
512K X 8	up	up	up	up	up

HARDWARE CONFIGURATION

. .

LOCATION OF MEMORY TYPE JUMPERS

HARDWARE CONFIGURATION

FIRMWARE START ADDRESS

The firmware on the 8M20 occupies 16K bytes of address space and can be located at either of two jumper selectable locations in system memory. These locations are C4000H and CC000H. The suggested location is CC000H. ***You should make sure that the selected firmware start address does not overlap other memory in the system***. If these addresses cause a conflict in your system, the can be changed at the factory to any address in the 1M byte PC-BUS address range. Please contact MESA if you need to change the default firmware locations.

Jumper block W17 determines the firmware start address. when the jumper is in the up position, the CC000H start address is selected, when the jumper is in the down position, the C4000 start address is selected.

FIRMWARE OPTION

Jumper block W14 controls a currently unimplemented firmware option and must be left in the right hand position for proper operation.

. .

LOCATION OF FIRMWARE START ADDRESS
AND FIRMWARE OPTION JUMPERS

HARDWARE CONFIGURATION

UNIT SELECT

Up to four 8M20 cards can reside in a PC-BUS computer system. Each card in a system must be assigned a distinct unit number. One card must be programmed as unit 0 to enable the firmware ROM. Note that unit numbers are used only for hardware selection of individual cards and do not affect system configuration as regards number and size of drives. Unit numbers are set with jumpers W18 and W19. The following table shows the jumper positions for setting the unit numbers.

W18	W19	UNIT #
down	down	0
down	up	1
up	down	2
up	up	3

SELECT LED

On the upper right hand corner of the 8M20 circuit card, is an LED that indicates that the card is currently selected. This can be helpful when first installing 8M20(s), or as general indicator of disk activity. If low power consumption is important, the LED can be disabled. Jumper block W6 is used to enable or disable the LED. When W6 is in the down position, the LED is enabled, when W6 is in the up position, the LED is disabled. If an external LED indicator is needed, it can be connected to the three pin jumper block W6 with a three pin .1" female header. Pin one (top) is LED+ and pin three is LED-.

HARDWARE CONFIGURATION

. .

UNIT SELECT AND LED JUMPERS

HARDWARE CONFIGURATION

SPEED / POWER OPTIONS

The 8M20 is designed for low power operation. Typical power consumption is 100 mA when used with a 8 MHZ CPU clock rate. To further reduce power consumption, the 8M20 can be operated in a low power / low speed mode. This mode of operation tri-states the address buffers on the 8M20 when it is deselected. This lowers the power consumption to approximately 25 MA. The low power/ low speed mode increases the access time of the disk emulation memory by about 20 NS, which may require the use of faster memory (120 NS) devices when operating at an 8 MHZ bus speed. Jumpers W15 and W16 determine the speed / power mode. Both jumpers should be placed in the left hand position to enable the low speed / low power mode. Both jumpers should be placed in the right hand position to enable the high speed / high power mode.

BACKUP POWER

Backup power for CMOS RAM is provided by Lithium cells BA1 and BA2. There are two modes of backup power operation. Mode one uses identical cells for BA1 and BA2. In mode one, BA1 and BA2 are connected in parallel. In mode two, BA1 is a 3.6 volt cell and BA2 is a 2.5 volt cell. When mode two is used, circuitry on the 8M20 routes memory backup power from the cell with the highest voltage (normally BA1), when BA1 is exhausted, power will be taken from BA2. When this happens, a software readable status bit on the 8M20 indicates that BA1 should be replaced. Backup power is disabled for shipping by removing the shorting jumpers from jumper blocks W7 and W8. These jumpers should be reinstalled even if only EPROMs or EEPROMs are being used. The jumpers on jumper blocks W7 and W8 are installed differently for mode one and mode two. Be very careful that the jumper installation matches the backup power mode, as incorrect jumpering will result in rapid lithium cell discharge. (Check if BA1 is the same as or different than BA2 to determine mode).

Jumper setting for backup power mode 1 (identical Lithium cells)
W7 right hand position
W8 right hand position

Jumper setting for backup power mode 2 (different Lithium cells)
W7 left hand position
W8 right hand position

. .

LOCATION OF BACKUP POWER JUMPER
AND SPEED / POWER JUMPER

HARDWARE CONFIGURATION

EXTERNAL BACKUP OPTION

The 8M20 can be with supplied with external backup power when this option is present. The external power option is only used with backup power mode 1. Use of external power is appropriate in high ambient temperature environments where Lithium cell life is shortened by larger memory backup current and higher self discharge rates. External backup power must be a ripple free voltage in the range of 2.7 to 4.5 VDC. Backup power is supplied through 9 pin subminature D connector P2

Connector P2 pinout:

1	+ select LED
2	+ VCC through 300 ohm resistor
3	NC (do not connect)
4	+ backup power (2.7 to 4.5 VDC)
5	Lithium cell voltage (do not connect)
6	ground
7	ground
8	ground
9	ground

COMPATIBLE CMOS MEMORY DEVICES

RAM	**128K BYTE**	**512K BYTE**
HITACHI	HM66204LP-12	
MITSUBISHI		
MOSEL	MS88128-12	

EPROM	**128K BYTE**	**256K BYTE**	**512K BYTE**	**1024K BYTE**
HITACHI	HN27C101G-15			
MITSUBISHI	M5M27C101K-15			
TOSHIBA	XXXXXXXXXXX	XXXXXXXXXXX		

EEPROM	**128K BYTE**	**512K BYTE**
XICOR	X28C010	

INSTALLATION

Note: system power must be turned off when installing or removing the 8M20 or damage to the 8M20 or system may occur.

After the 8M20 has been properly configured for its application, it is simply plugged into any free slot of the PC/XT/AT motherboard or PC-BUS backplane. The bracket retaining screw should then be tightened to secure the 8M20 in its slot. If the 8M20 is configured with nonvolatile CMOS RAM and contains valuable data, be careful not to allow contact between the 8M20 circuit card and any conductive object to prevent destruction of data.

SOFTWARE CONFIGURATION

GENERAL

8M20 disk emulator cards can be configured as RAM-disks, EPROM-disks, and EEPROM-DISKS of widely different capacities. Drive capacity can be increased by chaining 8M20 cards together up to a limit of 4 cards total. The specific disk emulator configuration is controlled by information stored in the first 256 bytes each 8M20. This information is referred to as the "drive template".

Drive templates can be created and edited with the supplied program RDCONFIG. The drive templates are stored as files and are written to the disk emulator by the program INITRAMD. The software distribution diskette supplied with the 8M20 contains drive template files for a number of standard configurations. The README file on this diskette lists the disk emulator configuration for each file.

USING RDCONFIG TO EDIT A DRIVE TEMPLATE

If none of the supplied drive template files are appropriate for your application, the utility program RDCONFIG can create custom drive template files or edit existing files. RDCONFIG is invoked by typing RDCONFIG followed by the drive template file name. This can be any file name with any extension. If no extension is supplied, the default extension .RDT is used. After a drive template file is created, it can be used by the utilities INITRAMD or INITROMD to create a working drive.

CREATING DUAL DRIVES ON ONE CARD

The 8M20 has the capability of being divided into two independent drives. Each half (bank) of the 8M20 can be configured separately. For example: you can configure a 8M20 to be a two drive system with 512K bytes of EPROM as drive C: and 512K bytes of RAM as drive D:.

To combine two drives on a single card, you must create two drive template files with the appropriate memory types and sizes. These template files must be setup to access the same unit number but different bank numbers. After the drive template files are created, you must initialize each bank individually with INITRAMD or INITROMD. The 8M20 banks must of course have their jumpers configured to match the memory types chosen.

SOFTWARE CONFIGURATION

USING INITRAMD TO WRITE DRIVE TEMPLATE TO RAM DISK

The drive template is written to all subunits that comprise a single drive by INITRAMD. This utility is invoked by typing INITRAMD followed by the drive template file name. In addition to writing the template, INITRAMD initializes and checks all sectors. If there are not enough free slots to have all slave 8M20's installed in the system in which configuration is being done, you can install only the master 8M20 and as many slaves as will fit. In this case, INITRAMD initializes the master and the installed slave(s), and will report bad sector errors. The initialized slaves can be removed, the uninitialized slaves installed, and INITRAMD run again until all slaves are initialized. The master 8M20 must remain installed until all slaves are initialized.

Example of INITRAMD usage:

```
INITRAMD RAM1024          (Initialize a 1M byte RAMDISK with the
                            supplied drive template RAM1024.RDT)
```

DOS UTILITIES

If you are configuring a 8M20 on a system with a hard disk, you should make certain that you know which drive is which before you use FDISK or FORMAT.

Always verify that the drive you are configuring is the 8M20 and not your hard disk. A mistake at this point could be disastrous. It is a good idea to get a directory listing of the drive you intend to FORMAT before doing so.

After INITRAMD has been run, the system must be reset (with either CNTL ALT DEL or power down) so that it recognizes the presence of the disk emulator. The remaining steps for creating a working RAM disk are the same steps followed in installing a new hard disk. First, FDISK should be run. If you are configuring the 8M20 in a system with a hard disk, Option 5 (next drive) should be selected. Be careful at this point, if you do not select option 5, you may make all information on you hard disk inaccessible! Then Option 1 (create DOS partition) should be selected. When this is done, FDISK will display the usable disk emulator space and restart the system. After FDISK has been run, the disk emulator should be formatted with FORMAT. If only a single drive has been created, it will be drive C:. In this case, the FORMAT program would be invoked with the command FORMAT C:. After formatting, you will have a working disk emulator.

MAKING A BOOTABLE DISK EMULATOR

SOFTWARE CONFIGURATION

If you wish to have the system boot from the disk emulator, you must run the DOS utility SYS.COM to transfer the operating system to the disk emulator. This is done with the command SYS C:. After the SYS operation is complete, COMMAND.COM must be copied to the disk emulator. The system is now bootable. NOTE: Please check your DOS license agreement if you plan to use a bootable disk emulator in another system.

DISK EMULATOR UTILITIES

Four utilities have been provided for use with RAM and EEPROM disk emulators. CRCRAMD allows the CRC checking to be enabled or disabled. The advantage of disabling CRC's is that the disk emulator transfer rate is approximately double the checked rate. The disadvantage is that most types of data errors are not detected. PRTRAMD enables and disables a total drive write protect feature. NUMRAMD sets the hard disk drive number. This allows selecting the drive used as the boot drive (C:) in a system with a hard disk and a 8M20 or two 8M20 drives. The BOOTRAMD utility determines the boot sequence. Invoking any of these utilities with no parameters will cause usage information to be displayed.

Example of CRCRAMD usage:

```
CRCRAMD 0 0 off          (disable CRCs on RAM disk with master at
                           unit 0, bank 0)
```

Example of PRTRAMD usage:

```
PRTRAMD 0 0 on           (disable writes to RAM disk with master at
                           unit 0, bank 0)
```

Example of NUMRAMD usage:

```
NUMRAMD  1 0 0           (Make RAM disk with master at
                           unit 1, bank 0 be drive C:)
NUMRAMD  1 0 1           (Make RAM disk with master at
                           unit 1, bank 0 be drive D:)
```

Example of BOOTRAMD usage:

```
BOOTRAMD 0 0 hard      (attempt boot from hard (8M20) drive first)
BOOTRAMD 0 0 floppy    (attempt boot from floppy drive first)
```

SOFTWARE CONFIGURATION

MAKING EPROM-DISKS

There are two methods for creating 8M20 EPROM disks. The simplest method is to make an image of an existing RAM disk with the utility MKROMIMG. The MKROMIMG utility is used to convert the contents of a RAMDISK into a series of object files suitable for programming individual EPROMs. Each file corresponds to a single socket on a 8M20. MKROMIMG is invoked with the unit number and bank of the master 8M20. File names are assigned according to the unit and socket U number. For example, the file C00U05.RDI is an object file for the EPROM in socket U5 of an 8M20 with unit number 0. MKROMIMG creates socket by socket copies of the 8M20 data and cannot create files for programming EPROMS with larger capacities than the RAMs from which the object files were generated. The second method, using the utility INITROMD, avoids this limitation and allows EPROMs of up to 8 Mbit capacity to be used.

INITROMD creates an emulated drive on an existing hard disk or diskette. The hard disk or diskette must have enough free space to hold the emulation file (The size of the ROM disk drive you are creating) plus approximately 100 Kbytes. The drive on which the emulation file is created must also be bootable. All files and programs that you wish to be stored on EPROMS in ROM disk format are copied to this emulated drive in order to make a file system image. In addition, standard DOS utilities such as SYS and DIR can be used on the emulated drive. INITROMD requires that a number of ancillary files be present on the default drive and directory before it is invoked. The easiest way to accomplish this is to copy the entire 8M20 distribution diskette into a new directory (MESA8M20 for example) on a hard disk. In addition, the CONFIG.SYS file on the root directory of the boot drive must be modified to include the line DEVICE = 8M20ROMD.SYS.

When INITROMD is invoked, a drive template file name and a emulated drive name must be supplied as command line parameters. After INITROMD has finished creating the emulated drive, the emulated drive should be SYSed (If it must be bootable) and the desired files copied to it. Note that FDISK and FORMAT need not and should not be run on the emulated drive.

SOFTWARE CONFIGURATION

When all the desired information has been copied to the emulated drive, the utility DUMPROMD is used to create object files for EPROM programming. The object files created by DUMPROMD are assigned names according to bank and socket number exactly as described for MKROMIMG. DUMPROMD must be supplied with the emulated drive name on the command line. If the DUMPROMD operation is successful, the emulated drive is no longer needed, and should be removed by running the utility RMVROMD.

DO NOT use any disk packing or disk access optimizing software while the emulated drive exists as this will likely result in damage to random files and/or directories on the drive.

The driver file 8M20ROMD.SYS is created in the root directory by INITROMD. It is deliberately hidden and read/only. Do not copy or move the driver file to another system as it has appended to it a File Allocation Table (FAT) fragment that applies only to the current emulated drive.

NOTE: INITROMD cannot be used at present to create a bootable DOS 4.01 file system image due to a bug in DOS 4.01's SYS program.

You must use MKROMIMG to create bootable EPROM disks with DOS 4.01.

Example of MKROMIMG usage:

```
MKROMIMG  0  0          (Make ROM image from 8M20 RAM disk
                          emulator with master at unit 0, bank 0)
                        (The EPROM image files will be created
                          in the default directory)
```

Example of INITROMD usage:

First you must add the line DEVICE = 8M20ROMD.SYS to the CONFIG.SYS file on the drive where the emulation file is to reside.

```
INITROMD ROM512 /DC     (create emulation file for drive template
                          ROM512.RDT on drive C:)
```

At this point, the system must be reset (hard reset or control alt/del) to recognize the new drive. The new drive will be assigned the next available drive letter. That means that on a system with one hard disk (assumed in this example), the emulated drive would be drive D:.

SOFTWARE CONFIGURATION

SOFTWARE CONFIGURATION

If the EPROMDISK is to be bootable you must do the following two steps

```
SYS D:                  ( Put DOS on the emulated drive )

COPY COMMAND.COM D:     ( put command processor on emulated
drive )
```

Then all of the files that you wish to be on the EPROM disk are copied to the emulated drive.

```
COPY SMALTALK.EXE D:
COPY ROMFILE.ROM D:
```

When all of the desired files have been written to the emulated drive, the DUMPROMD is invoked to create individual EPROM object files

```
DUMPROMD D:             ( Create a set of EPROM object files from
                          the information in the emulated drive )
```

Note that EPROM size is determined by the drive template file created by RDCONFIG.

```
RMVROMD /DC             ( Remove emulated drive from system )
```

It is always a good idea to remove the emulated drive from the system after use.

OPERATION

LITHIUM CELL LIFE

Lithium Cell life is determined by four factors: memory standby current drain, Cell capacity, temperature, and system power off time. When factory configured as a 3M byte RAMdisk, the maximum standby current drain is 100 uA with 40 uA being typical at room temperature.

If customer supplied memory chips are used, the standby current should be measured to determine Lithium cell life. Standby current can be measured by removing the jumper at W1 (for bank 0) or W9 (for bank 1) and measuring the current across pins 1 and 2 (top and middle). If both banks are being used with CMOS RAM, these currents must be added to give the total current drain. Removing W1 or W9 disconnects backup power from bank 0 and bank 1 memory arrays and will destroy any information stored in them. If memory contents must be preserved, the voltage drop across 1000 ohm resistors R6 and R7 can be measured. The backup current in microamps is equal to (V(R7) + V(R6)) * 1000.

Lithium Cell capacity is 4.6 Ah with mode 1 backup circuitry. With mode 2 backup circuitry, primary cell capacity is 2 AH and secondary cell capacity is 2.3 AH. Elevated temperatures decrease Lithium Cell life by increasing memory standby current and accelerating self discharge in the Lithium cell. Lithium Cell life at a temperature of 60^{o}C is approximately one third of 25^{o}C Lithium Cell life. Memory backup current is drawn from the backup Lithium Cell only when system power is off.

Approximate Lithium Cell life in years of power off time at 25^{o}C:
(Mesa populated 3M byte 8M20 CMOS RAMdisk)

	min.	typ.
Mode 1	5.2	10
Mode 2	2.4	6.2 (until primary cell is exhausted)

SPECIFICATIONS

	Min	Max	Units
POWER REQUIREMENTS:			
Supply voltage	4.5	5.5	V
Supply current	---	100	mA Note 1
BUS LOADING:			
Input capacitance	---	20	pF
Input leakage current	---	15	uA
Output drive capability	300	---	pF
Output sink current	48	---	mA
LITHIUM CELL:			
Capacity (Mode 1)	4.6	---	Ah
Capacity (Mode 2)	2.0 + 2.3	---	Ah Note 2
ENVIRONMENTAL:			
Operating temperature range -I	-40	+85	^{o}C
-C	0	+70	^{o}C
Relative humidity	0	90	Percent

Note 1: Power consumption with CMOS RAMS or EPROMS, 8 MHz CPU

Note 2: 2.0 AH primary cell capacity, 2.3 AH secondary cell capacity.

W A R R A N T Y

Mesa Electronics warrants the products it manufactures to be free from defects in material and workmanship under normal use and service for the period of 2 years from date of purchase. This warranty shall not apply to products which have been subject to misuse, neglect, accident, or abnormal conditions of operation.

In the event of failure of a product covered by this warranty, Mesa Electronics, will repair any product returned to Mesa Electronics within 2 years of original purchase, provided the warrantor's examination discloses to its satisfaction that the product was defective. The warrantor may at its option, replace the product in lieu of repair. With regard to any product returned within 2 years of purchase, said repairs or replacement will be made without charge. If the failure has been caused by misuse, neglect, accident, or abnormal conditions of operation, repairs will be billed at a nominal cost.

> THE FOREGOING WARRANTY IS IN LIEU OF ALL OTHER WARRANTIES, EXPRESS OR IMPLIED. INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS, OR ADEQUACY FOR ANY PARTICULAR PURPOSE OR USE. MESA ELECTRONICS SHALL NOT BE LIABLE FOR ANY SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER IN CONTRACT, TORT, OR OTHERWISE.

if any failure occurs, the following steps should be taken:

1. Notify Mesa Electronics, giving full details of the difficulty. On receipt of this information, service data, or shipping instructions will be forwarded to you.

2. On receipt of the shipping instructions, forward the product, in its original protective packaging, transportation prepaid to Mesa Electronics. Repairs will be made at Mesa Electronics and the product returned transportation prepaid.

SCHEMATIC DIAGRAMS

SCHEMATIC DIAGRAMS

SCHEMATIC DIAGRAMS

SCHEMATIC DIAGRAMS

SCHEMATIC DIAGRAMS

SCHEMATIC DIAGRAMS

SCHEMATIC DIAGRAMS

SCHEMATIC DIAGRAMS

SCHEMATIC DIAGRAMS

SCHEMATIC DIAGRAMS

SCHEMATIC DIAGRAMS

SCHEMATIC DIAGRAMS

SCHEMATIC DIAGRAMS

SCHEMATIC DIAGRAMS

SCHEMATIC DIAGRAMS