MICROMATION

DOUBLER

OPERATOR'S
MANUAL

D01 000 REV B
This manual cannot be reproduced, in whole or in part, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without prior written approval from Micromation, Inc.

If you would like to duplicate, regardless of the means, any portion of the text, write a letter to us requesting permission first. Specify the pages needed and the application of the duplicated text when filing the request. We reserve the right to refuse duplication rights.

Micromation Incorporated makes no representations or warranties with respect to the contents of this document and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose. Further, Micromation reserves the right to revise this publication and to make changes as necessary in the content hereof without obligation of Micromation to notify any person or organization of such revision or changes.
# MICROMATION

**DOUBLER**

**FLOPPY DISK CONTROLLER**

## TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section 1</th>
<th>Section 2: Theory of operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Introduction</td>
<td>6 Introduction</td>
</tr>
<tr>
<td>1-1 Hardware Interface</td>
<td>6-1 Disk system operation</td>
</tr>
<tr>
<td>1-2 Bus Interface</td>
<td>6-2 Disk read/write</td>
</tr>
<tr>
<td>1-3 Disk Interface</td>
<td>7 The S-100 bus interface</td>
</tr>
<tr>
<td>1-4 Special Interface Requirements</td>
<td>7-1 The control bus</td>
</tr>
<tr>
<td>2 Controller Options</td>
<td>7-2 The data bus</td>
</tr>
<tr>
<td>2-1 Necessary Hardware</td>
<td>7-3 The address bus</td>
</tr>
<tr>
<td>2-2 Console Device Connection</td>
<td>8 Address bus decoding</td>
</tr>
<tr>
<td>3 DOUBLER Installation</td>
<td>8-1 The address decoder PROM</td>
</tr>
<tr>
<td>3-1 Preparation</td>
<td>8-2 The C2-PROM</td>
</tr>
<tr>
<td>3-2 Baud Rate Selection</td>
<td>8-3 The scratch pad RAM</td>
</tr>
<tr>
<td>3-3 Jumper Options</td>
<td>8-4 The read and write control</td>
</tr>
<tr>
<td>3-4 Board Insertion</td>
<td>8-5 The UART</td>
</tr>
<tr>
<td>3-5 Cable Installation</td>
<td>9 Disk drive interface</td>
</tr>
<tr>
<td>3-6 Booting the Systems</td>
<td>9-1 The phase lock oscillator</td>
</tr>
<tr>
<td></td>
<td>9-2 The byte sync counter</td>
</tr>
</tbody>
</table>

## Section 1

### 1 Introduction

1

### 1-1 Hardware Interface

1

### 1-2 Bus Interface

1

### 1-3 Disk Interface

2

### 1-4 Special Interface Requirements

2

### 2 Controller Options

3

### 2-1 Necessary Hardware

4

### 2-2 Console Device Connection

5

### 3 DOUBLER Installation

5

#### 3-1 Preparation

5

#### 3-2 Baud Rate Selection

6

#### 3-3 Jumper Options

6

#### 3-4 Board Insertion

6

#### 3-5 Cable Installation

6

#### 3-6 Booting the Systems

7

### 4 Use Of CP/M with the DOUBLER

10

#### 4-1 Diskette Handling and File Maintenance

10

#### 4-2 Standard CP/M Utilities

11

#### 4-3 Micromatin Utilities

12

#### 4-4 Other Files

15

#### 4-5 The Micromation BIOS

16

### 5 CP/M Interversion Compatibility

17

## Section 2: Theory of operation

### 6 Introduction

18

#### 6-1 Disk system operation

18

#### 6-2 Disk read/write

19

#### 7 The S-100 bus interface

20

#### 7-1 The control bus

20

#### 7-2 The data bus

21

#### 7-3 The address bus

21

#### 8 Address bus decoding

21

#### 8-1 The address decoder PROM

21

#### 8-2 The C2-PROM

21

#### 8-3 The scratch pad RAM

22

#### 8-4 The read and write control

22

#### 8-5 The UART

23

#### 9 Disk drive interface

23

#### 9-1 The phase lock oscillator

24

#### 9-2 The byte sync counter

25
9-3 Wait states (XRDY or PRDY) 26
9-4 The cyclic redundancy check 26
9-5 Encoding and write precompensation 27
10 The RS232 serial interface 27
11 Power on Jump 28
12 Power supply 28

Appendix A: The Disk Format 29

Appendix B: C2-PROM listing 31

Appendix C:
Changing Operating System Characteristics 46
C-1 Changing the System Size 46
C-2 Changing the IOBYTE default and Step Time 48
C-3 BIOS and BOOT Alteration 52

Appendix D: Schematic diagrams
The DOUBLER is a high performance floppy disk controller designed for the S-100 bus. Its proper installation in an S-100 system will provide reliable operation at the highest speed and capacity currently possible with floppy disk technology. The controller is designed to interface directly with CP/M*, an operating system which has file management facilities and software utilities comparable to the most advanced computer systems. To ensure full utilization, the user of a DOUBLER system should be acquainted with the options available and its general method of operation.

1-1 HARDWARE INTERFACE

The DOUBLER will provide all the functions required for single or double density floppy diskette operation for S-100 bus computer. It uses the IBM 3740 format for single density recording (26 sectors of 128 bytes each). In double density, the DOUBLER uses a version of the IBM 2D format modified for 52 sectors of 128 bytes to maintain compatibility with CP/M. The board uses the system processor to control drive functions. The transfer of disk data is done under program control in order to assure reliable operation. The board requires two thousand (2K) bytes of system address space: 1K is used by the 2708 EPROM and the remainder is used for an on-board RAM scratchpad and memory-mapped I/O locations. All I/O operations are handled through memory locations in the board's address space, so no I/O mapped ports are required.

1-2 BUS INTERFACE

Detailed interface information is available by referring to the schematics included in appendix D. Address decoding for the board is accomplished by a 256 x 4 bipolar PROM. When the proper address appears on the bus, the appropriate strobes are generated. Further decoding of the strobes for input and output functions provides strobes for individual operations. A power-on-jump is accomplished by disabling the memory at location 0 after a reset with the phantom line. The on-board EPROM is then enabled and a jump instruction to the bootstrap routine is sent to the processor. The board synchronizes the disk drive with the processor by holding the system ready line low until the disk interface is ready for another byte of data. Circuitry on the board prevents the ready line from remaining low for a period of time longer than that required to transfer eight bytes of data from the disk.

* CP/M is a registered trademark of Digital Research, Inc.
1-3 DISK INTERFACE

Software routines in the EPROM direct the generation of signals necessary to control the disk drives. The output signals are latched in a latch/driver. The input signals are read by the system through an input port. Circuitry is included to cause the drive head to unload if there has been no disk access during eight revolutions of the diskette.

When reading data from the disk drive, the controller derives its data clock from the data on the diskette by generating a signal called PLO. This signal represents a phase lock oscillator and maintains synchronization with the disk data without being sensitive to the shift of individual bits. During write operations, the PLO is derived from the crystal clock and controls the data pulses written to the disk. During write operations in double density, the PLO signal is shifted under special conditions to cause the data written to the diskette to be "pre-compensated." Special circuitry prevents inadvertent writing to the diskette by unintentional memory accesses.

Error checking of all disk operations is done by computation of the sector's CRC. The CRC checking is done in hardware, so the system is capable of reading or writing consecutive sectors.

1-4 SPECIAL INTERFACE REQUIREMENTS

The DOUBLER and Micromation CP/M are currently configured to interface Shugart single-sided or Remex or YE Data double-sided drives. In addition, the controller can be factory configured for PerSci model 277 drives. Call Micromation for other drive types supported. This is especially important if you are thinking of upgrading your system with the DOUBLER.

FOUR-DRIVE SYSTEMS: The DOUBLER and associated CP/M are designed to support from 1 - 4 drives without modification. Each drive must be from the same manufacturer and be of the same type (single- or double-sided), however.

DRIVES REQUIRING ABOVE TRACK 43 CONTROL: Some floppy disk drives require a signal to indicate that the head is positioned above track 43. This signal is available from the controller and is generally connected to the disk interface connector on pin 8. Some drives require this signal on a different line, however, and the user should verify the operation with the particular drive being used.
2 CONTROLLER OPTIONS

The DOUBLER has several options to make it the most powerful controller available. Their functions and factory settings are reviewed below. Note that all the jumpers described below except the WAIT jumper are in the form of traces. To disable the function, the trace must be cut. (A header can be installed to facilitate re-enabling the jumper.)

POJ: Connection of this jumper (located near device 7C) enables the power-on-jump function of the DOUBLER. This feature operates by driving the "phantom" line on pin 67 of the S-100 bus low after a reset. This should disable the output buffers of the low memory. Check with your system technical manual of the memory used in this area to ensure that it supports the "phantom" line. If it does not, the controller's power-on-jump cannot be used, and another method of transferring control of the processor to the bootstrap routine must be used. When enabled, the DOUBLER's power-on-jump circuitry causes the system to jump to the cold bootstrap routine located in PROM on the controller board. The DOUBLER is shipped with the power-on-jump enabled. To disable it, remove the jumper between the pads marked "POJ".

PHANTOM: This jumper (located near device D4) connects the line used to disable RAM while the DOUBLER executes a power-on-jump. It ordinarily is connected to pin 67 of the S-100 bus, but can be jumpered to any other pin which the user's system supports. The reference manuals of the system should be checked to determine whether any other boards use pin 67. A few processor boards use this line to output the refresh signal from Z-80 processors. This should be disabled or disconnected if the DOUBLER's power-on-jump circuitry is utilized.

XRDY & PRDY: The DOUBLER uses the ready line to synchronize the processor with disk data. Different systems, depending on front panel or dynamic memory design, require that peripherals use XRDY and PRDY on the S-100 bus. The DOUBLER can use either line. It is shipped with jumpers (located near device D4) enabled to use both XRDY and PRDY. If either adversely affects the system operation, it may be disconnected by cutting the trace where marked.

WRITE: In order to prevent unintentional writing on a diskette, a write enable jumper (located near device 7A) is installed. This jumper must be in place in order to write on a diskette. If the floppy disk drives that are being used do not support write protect (all Micromation systems do), it is recommended that this jumper be removed until the system has been operated successfully and whenever the user wants to ensure that a diskette is not written upon.

HEAD: Standard Shugart-type drives support a signal named HEAD LOAD on the disk interface cable. This signal is used to load and unload the head of a selected drive, so the drive select buffers may remain enabled. Other drives, such as PerSci, use the drive
select lines to unload the head. With these drives, the drive select lines must be disabled in order to unload the head. The DOUBLER unloads the head of a selected drive if a read or write operation has not occurred during the past eight revolutions of the diskette. Ordinarily this is done by disabling the HEAD LOAD line. If the controller is used with PerSci-type drives, the HEAD jumper (located near device 10A) must be switched to its alternate position. This will disable the drive select buffers to unload the head.

WAIT: To facilitate operations with Z-80 processors, a WAIT jumper (in the form of a header located near device D5) is available. This causes a wait state to be added only when the board is addressed. This is necessary to enable the on-board 2708 EPROM to be accessed and to allow time for the disk control circuitry to be properly set-up. This wait state will not affect overall system speed since during disk operations the speed of the system is controlled by the transfer speed of the disk data. The WAIT jumper must be connected when the DOUBLER is used with any 8080 system or when the DOUBLER is installed in a Z-80 based system operating at 4 MHz. If a 2 MHz Z-80 processor is used, WAIT should not be connected. The DOUBLER is shipped with the WAIT jumper installed. It should be removed only when operating with a 2 MHz Z-80 processor.

2-1 NECESSARY HARDWARE

Micromation DOUBLER floppy disk systems are designed to operate with all standard S-100 systems with 2 to 4 MHz 8080 or Z-80 processors. Since the controller performs a power-on-jump, it can be used in systems with or without a front panel. The operating system requires at least 16K bytes of RAM contiguously addressed in the lowest addresses of memory. The controller occupies 2K bytes of memory generally located at the top of the addressable memory range, from F800 to FFFF. The user should ensure that there are no memory address conflicts with other boards in the system. The DOUBLER may be addressed at locations C000, D000, E000 or F000 by obtaining special PROMs from Micromation.

2-2 CONSOLE DEVICE CONNECTION

A console device is necessary to communicate with the system. The DOUBLER includes a full function UART to communicate with RS-232 type terminals. The software provided with the system is programmed to use the UART on the controller to communicate with the console device. Optional software drivers for the Processor Technology SOL and NorthStar Horizon are also available from Micromation.

NOTE: The DOUBLER derives the clock input for the UART from the CLOCK signal on pin 49 of the S-100 bus. This must have a 2 MHz frequency for the UART to function. Check your system manual to
ensure that pin 49 has a 2 MHz clock on it. If your system does not, install the necessary jumpers to provide the requisite signal.

CONSOLE DEVICE INSTALLATION: Installation of the hardware is straightforward. To initially bring up the system, the minimum amount of hardware should be used. This is just the disk controller, processor board, and, at least, 16K to 32K of memory in the lowest address of RAM. An RS-232 terminal should then be connected to the controller board. The controller has a 10 pin socket header with cable on the right side of the board. The pins are labelled to indicate their RS-232 function. The following table may be used to connect an RS-232 terminal. A ten foot cable with RS-232 connector is available from Micromation. Most RS-232 terminals require only three signals (ground, transmitter data, and receiver data) to be connected. The other signals are for terminals which require handshake operation.

<table>
<thead>
<tr>
<th>DOUBLER HEADER PIN</th>
<th>SIGNAL NAME</th>
<th>TERMINAL RS-232 PIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>GND</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>TxD</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>RTS</td>
<td>20</td>
</tr>
<tr>
<td>5</td>
<td>DSR</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>RxD</td>
<td>2</td>
</tr>
</tbody>
</table>

GND The ground signal sets a common reference between the output and input devices.

TxD Transmitter Data is that data output from the CPU to the terminal.

RTS Request to Send informs the terminal that the CPU has some data to output. This signal is used only when handshaking is necessary.

DSR Data Set Ready active indicates that the terminal is ready to receive data. The terminal sends a signal called Data Terminal Ready (DTR) to this pin. The processor checks this bit before it sends the character. This, also, is only necessary with terminals that require handshaking.

RxD Receiver Data is that data output from the terminal to the processor.
3 DOUBLER INSTALLATION

3-1 PREPARATION

Before installing the DOUBLER in your system, clean off the S-100 edge connector fingers with alcohol on a cotton swab. This will remove any oxidation or finger prints. Do not use any other cleaning agents (e.g., a solvent, emory cloth, ink eraser, etc.), as these may damage the connector fingers.

3-2 BAUD RATE SELECTION

Any baud rate from 110 to 9600 can be selected with the proper jumper. The baud rate selection jumpers are in the upper right side of the DOUBLER, just to the right of the RS-232 connector. Do not confuse the two. The baud rate jumpers are in the form of a 16-pin connector; the RS-232 connector has 10 pins.

The selectable baud rates are labelled on the board. The DOUBLER is shipped with a jumper setting the rate at 2400. Before installing the board, remove the jumper and place it in the position that corresponds with the setting on your terminal. Most terminals also feature a selectable baud rate. Ensure that the setting on the DOUBLER matches the setting on your terminal. If you are in a quandary as to which setting to choose, 9600 BPS (bits per second) is a popular rate.

3-3 JUMPER OPTIONS

Read the CONTROLLER OPTIONS section above and install or remove the appropriate jumpers for your system.

3-4 BOARD INSERTION

- Turn off power to the computer and floppy drives.
- Install the DOUBLER in a slot in the S-100 mother board. Place the controller as close to the processor card as possible. Ensure that the card is pressed down into the edge connector all the way. Note that S-100 fingers are offset from the side preventing mis-orientation of the DOUBLER in the card cage.

3-5 CABLE INSTALLATION

- Connect the 50 conductor ribbon cable from the floppy disk drives to the 50-pin connector on the DOUBLER. Pin 1, indicated by a red stripe on the edge of the cable, must be connected to header pin 1 on the left side (as viewed from the component side) of the board.
- Connect the RS-232 terminal to the controller. Again, pin 1 of
the 10-pin cable attaches to the left side of the DOUBLER connector.

NOTE: Pin 1 of this cable is typically indicated by a red stripe. If no such stripe is present on the cable, an arrow or indentation molded into the plastic cable connector also designates pin 1.

The figure below illustrates the top of the DOUBLER for use with cable installation and baud select.

![Diagram of DOUBLER connector and cables](image)

**DOUBLER CABLES AND BAUD SELECT**

### 3-6 BOOTING THE SYSTEM

Before loading the operating system, read the CP/M Licensing Agreement and mail the registration card to Digital Research. Only registered owners are entitled to updates.

Of course, you should become familiar with the CP/M operating system as well. Begin with the booklet An Introduction to CP/M Features and Facilities. This should be followed with CP/M 2.2 User's Guide for CP/M 1.4 Owners for a discussion of the updates from the previous version included in the new release. The remainder of the documents shipped with the O/S (operating system) can be read when the need arises.

NOTE: CP/M version 2.2 is being shipped at the time of this writing. As new revisions are distributed by Digital Research, Micromation makes the necessary modifications and ships them with the units. To accommodate this flux, 2.x will be used as a descriptor in the examples that follow.
To boot the system,

- Turn on the power to the floppy drives and the terminal.

- Turn on the power to the computer and press the reset button.

- Ensure that the distribution diskette from Micromation is write protected. For systems with drives that check the write protect notch of the diskette, leave the notch exposed. (The figure below shows the location of the notch.) If your drives do not support this feature, remove the WRITE GATE jumper described above from the DOUBLER.

- Insert the distribution disk in drive A with the label facing up (see the illustration below) and close the door. In Micromation systems, drive A is the bottom drive where the drives are mounted vertically; the left drive in our systems with horizontal mounting.

```
DRIVE A  DRIVE B

WRITE PROTECT NOTCH

LABEL FACING UP

DISKETTE ORIENTATION
(Horizontal Mounting)
```

- If the computer has a RUN/STOP switch, hit RUN. If the DOUBLER's power-on-jump is not being utilized, use a front panel or monitor to cause the system to execute the program at location F800 (or at the base address of the controller if it is located at a different location).

- Drive A should home (go to track 0), step twice, and within two or three seconds display

```
62k CP/M - Micromation ver 2.x
A>
```

where the first line is the sign-on message and "A>" is the CP/M prompt. "A" indicates the current drive.

- To view the contents of the disk, type "DIR". This is the CP/M command to display the directory of the files on the diskette. The system responds with the names of the files.
This is the sequence for booting the system. It is referred to as a cold boot and need only be performed when the system is first turned on. Subsequently, a control-C can be used to load the system when necessary. This is called a warm boot and is only mandatory when diskettes are changed. In the event of a program crash, a cold boot may be necessary to get out of the program and back to the operating system.

The following chapter discusses the utilities (called transient commands in the CP/M documents) provided on the distribution diskette and provides some exercises to demonstrate their use. We strongly recommend that the operator(s) perform(s) these exercises to get hands-on experience in the use of the computer system.
4 USE OF CP/M WITH THE DOUBLER

4-1 DISKETTE HANDLING AND FILE MAINTENANCE

There are some precautions that should be observed to ensure that your files aren't inadvertently lost due to operator error or the slings and arrows of outrageous fortune.

1) **Handle the diskettes with care.** Keep your fingers away from the exposed areas on both sides of the disk. Store diskettes in the paper sleeve whenever they are not in the computer (dust accumulates, otherwise, and shortens the life of the read/write head on the floppy drive). Since diskettes are a magnetic media, be sure to keep them away from magnets. You should also keep the diskette out of direct sunlight and away from extreme heat. Never remove the mylar disk from its protective paper enclosure nor abuse the disk by folding or bending it. When possible, write the label before applying it to the diskette. Subsequently, use only felt-tip pens for additional notes.

2) **ALWAYS make a back-up of your data files.** In many cases great time and effort is spent creating these files. It is much easier to make back-up copies on an on-going basis than to re-enter the whole thing over again. Make a back-up of any files changed *every time* they're changed. (After the program has terminated and the O/S prompt has re-appeared, of course.)

3) **NEVER power down the disk drives during program execution nor with a disk in a drive and the door closed.** Clear the drives of all diskettes before turning off the power to the drives (opening the drive door will do). Powering down during program execution will, at least, render the data files suspect or, worse, trash a couple of sectors rendering the whole file useless. Turning off power with a diskette in the drive (but not during program execution) may also trash a sector or two as the head responds to powering down.

4) **Write protect your important diskettes.** The little notch on the left side of the leading edge (see the figure above) causes write protect when exposed. Cover it with tape (or the silver squares frequently provided with this type of diskette) and the diskette can be written on. The CP/M system disk provided with the DOUBLER has the notch exposed so the disk cannot be written on or formatted (erased).

If these precautions are observed, very few problems will arise and those that do can be easily repaired with the back-up disks.
4-2 STANDARD CP/M UTILITIES

These two utilities will be used frequently. Diskettes used for program execution (and development) should contain these programs for diskette monitoring (with STAT) and file back-up (with PIP).

STAT.COM: This utility is used primarily to indicate the status of the disk space. It can indicate, for instance, the amount of space remaining for file storage, how much space a certain file takes up, how much space a group of files takes up, etc. Slightly different invocations display or change the current logical assignments of the peripheral devices.

PIP.COM: PIP is short for Peripheral Interchange Program. It is most frequently used to transfer files from one disk to another in multi-drive systems. It is also used to transfer files between devices (from disk to list device, from paper tape reader to disk, etc.).

The next three programs may be useful, depending upon the application of the your system.

ED.COM: This is the CP/M text editor for creating and altering files. Although ED can be used for word processing, it is not recommended. There are several word processing programs available that are much better suited to this application. Primarily, it is useful for creating and editing program files.

SUBMIT.COM: A limited form of batch processing is available with SUBMIT. This is most useful when a sequence of transient commands is frequently performed. A source file containing these commands is created with the editor for subsequent execution(s). There is no limit to the number of times this file is SUBMITTED.

XSUB.COM: XSUB enhances SUBMIT by allowing line input to programs in addition to CCP input. Refer to the CP/M 2.2 User's Guide for CP/M 1.4 Owners for a description of XSUB.

The following utilities are used to generate a new system and to transfer the system from one disk to another.

MOVCPM.COM: This utility is used to move the CP/M system from one location in memory to another. CP/M should be loaded into the uppermost part of memory allowed by the computer system unless a special application requires differently.

SYSGEN.COM: Primarily, this utility is useful for transferring the system image between different density disks (single to double, double to single). Additionally,
SYSGEN is used when a new system image is created and recorded. An illustration of its primary use is given in DISKETTE PREPARATION below.

The next four utilities are pertinent to assembly language programming. Many end users will never need these programs.

ASM.COM: This is the CP/M 8080 assembler. A program of commands from the 8080 instruction set can be assembled rendering two files, x.HEX and x.PRN, where 'x' is name of the program file. The .PRN file contains the original program listing plus the machine code. The .HEX file contains only the 8080 machine code in the Intel "hex" format.

LOAD.COM: The LOAD command converts a .HEX file created by ASM to a COM file (which indicates that it contains machine executable code). Once an assembly language program has been LOADed, it can be invoked by merely typing the file name when the CP/M prompt is displayed on the console. That is, the program has the status of a CP/M utility.

DUMP.COM: The DUMP program displays the contents of the designated file on the console device in hexadecimal form.

DDT.COM: DDT, which stands for Dynamic Debugging Tool, allows interactive testing and debugging of programs. An assembly language program can be tested, altered, patched, executed and/or repaired "on the fly" under DDT.

4-3 MICROMATION UTILITIES

The utilities described below were developed by Micromation to reconcile CP/M with Micromation equipment and to make computer operation easier. Note that the following describes the utilities provided with the standard Micromation CP/M system.

FORMAT.COM: Before using them to store data, diskettes must be formatted. This procedure writes a code on the diskette identifying each track (77 concentric circles around the diskette) and sector (26 or 52, depending upon the density, sections within each track). Thus, each location on the diskette is uniquely identified and can be individually accessed. There are two formats for standard 8" diskettes: single and double density. The diskette shipped with the DOUBLER is recorded in single density and contains about 250 kilobytes (250K) of read/write space. Diskettes with a double density format have about 500K bytes of file space.
To format a diskette in drive B in double density enter:

**FORMAT BD@**

Where FORMAT indicates the program, B the drive containing the diskette and D the density.

To format a diskette in drive B in single density enter:

**FORMAT BS@**

Where S indicates single density.

The diskette in drive A can be formatted. However this will erase the diskette. Do not attempt to boot from the diskette in drive A after it has been formatted.

**IMPORTANT:** The format program should be used with care. When old diskettes that have information stored on them are formatted, all data is erased as part of the process. If a used diskette requires formatting (e.g., to make it double density from single density) be aware of this fact. There is no way to recover files erased by FORMAT.

**SYSTFORM.COM:** This program is like the format utility but formats the system tracks on the diskette only.

**VERIFY.COM:** This utility calculates cyclic redundancy check characters from all sectors and compares them to the CRCCs written on the diskette in the drive queried. If an error exists its location will be displayed, if not the prompt is returned.

**DENSITY.COM:** A program to determine and display the density of the diskette in the drive queried.

**SDIR.COM:** This utility displays an alphabetized disk directory when called.

**COPY.COM:** Often, it is easier or necessary to copy an entire diskette rather than transfer files one at a time. The COPY utility is provided for this purpose. Note that COPY will only transfer data between like-density formatted diskettes (single to single or double to double).

There are three forms of COPY, each for a different task.

- **COPY A** (copy All system and data tracks)
- **COPY S** (copy just the System area, tracks 0,1)
- **COPY D** (copy just the Data area, tracks 2 - 76)
Prompts are displayed by the program requesting the source disk (the master from which the copy is made) and the destination disk. Use of COPY is illustrated in DISTRIBUTION DISKETTE DUPLICATION below.

FILES.COM: The directory entries for a disk and the blocks used are indicated by this transient. The first three 8-digit groups of numbers are the file name and type; the fourth indicates the extent in the first two digits and the number of records used in the extent (in hex) in the last two digits (the middle 4 digits have no significance); and the remainder indicate the specific blocks used. Note that this utility can be invoked on the current disk only.

CPM62.COM: This is not a utility. It is a duplicate of the Micromation CP/M operating system set up for operation in 62K of RAM. It is provided as a convenience in new system generation. Refer to Appendix C for an example of its use.

RAMTEST.COM: This utility was written to check the RAM in Micromation systems. It may or may not run in non-Micromation systems. Refer to the listing (RAMTEST.ASM) on the distribution diskette to see if it has utility.
4-4 OTHER FILES

DISKDEF.LIB: DISKDEF is used with the Digital Research Macro assembler. It has no utility beyond its use with this program. Refer to the CP/M 2.2 Alteration Guide.

DEBLOCK.LIB: This file is supplied by Digital Research and contains sector blocking and de-blocking algorithms. Refer to the CP/M Alteration Guide for a discussion of this feature.

CBIOS.LIB, BIOS.LIB, BOOT.LIB: These files are distributed by Digital Research as examples of the BIOS and BOOT programs. They are for reference only. To alter the system, use MM2BIOS.ASM and M2BOOT.ASM described below.

LIST.SUB, STEP.SUB: These files are used with the CP/M SUBMIT utility to change the IOBYTE and step time respectively. See Appendix C-2 for a description of their use.

The remainder of the files on the distribution diskette, with file type ASM, are the source files of the Micromation generated utilities. Many users will find these files immaterial for day to day operation. However, system builders may find these very useful, especially when developing dedicated application packages. Most of the ASM files have corresponding COM files, some don't. Those that don't are discussed below.

C2PROM.ASM: This file contains the code of the DOUBLER PROM at board location D4. It is provided for reference. As such it can be used to develop special application packages that require knowledge of and/or access to DOUBLER routines and their locations.

MM2BIOS.ASM: MM2BIOS is the source file of the BIOS portion of CP/M. Micromation has written this section to allow system alteration with a minimum of fuss. Refer to THE MICROMATION BIOS below for a description of the default characteristics.

M2BOOT.ASM: M2BOOT, like the custom BIOS, is another part of the CP/M O/S. Its role is to load the system. If you change the size parameter in MM2BIOS, a corresponding change must be made in M2BOOT. The two files must then be re-assembled and inserted in the system.
4-5 THE MICROMATION BIOS

The BIOS (Basic Input/Output System) portion of CP/M is custom tailored to accommodate the Micromation hardware. In addition, several parts are conditionally assembled to suit the user's application. "Conditionally assembled" means that portions of the program are not included during assembly unless a flag is set true. To reset them to alter the system configuration, see New System Generation below.

The system characteristics are established in BIOS and are as follows. (Your system may have slightly different characteristics if it was configured for a non-Micromation hardware environment.)

- a system size of 62K
- the serial port on the DOUBLER for console (CON:) device with an appropriate driver
- 3 parallel ports on the Multi I/O board assigned to the line printer (LPT:) option for LST: with a driver routine for a Centronics type dot matrix printer
- the serial port on the Multi I/O Board assigned to the TTY: option for LST: with a driver routine for a serial interface printer

The following table summarizes the relationship between logical and physical device assignments as established in BIOS at cold boot.

CON: = CRT: (through the DOUBLER serial port)
RDR: = TTY:
PUN: = TTY:
LST: = LPT: (Centronics 703/779 printer through Multi I/O board parallel ports)

Although RDR: and PUN: are assigned to TTY:, they are not supported in the BIOS. Attempts to output to PUN: or input from RDR: will not work.

The list device (LST:) is assigned to the line printer (LPT:) option. The BIOS currently contains a driver for a Centronics 703/779 printer to correspond with this assignment. This is a parallel input dot matrix printer connected through the parallel ports on the I/O board.

The next section has twofold significance. First, it presents the procedures for backing up the distribution diskette, for making a double master from the distribution diskette, and, finally, for making work disks for use in the day to day operation of your computer system. Second and equally as important, execution of
these procedures provides hands-on experience in use of the utilities for the operator.

5 CP/M INTER-VERSION COMPATIBILITY

The operating system shipped with the DOUBLER is the latest version of the popular CP/M O/S from Digital Research. (As of this writing, the version number is 2.2. This is subject to change as new revisions are distributed.) In the single density recording format, there's complete compatibility between this and previous versions. In double density recording, there is an important difference. This difference will destroy the data stored in a file when transferring it from a diskette recorded under an earlier version to one recorded under version 2.2 (or later) or vice versa. Use the following procedure to move files from diskettes recorded in double density by previous versions to double density disks created under the new 2.2 system.

1) Put the old system master in drive A. Use your old FORMAT program to initialize enough disks in drive B to accommodate the files presently on your double density disks.

2) Using the old version of PIP, transfer the program and data files from the double density disks to the newly created single density disks. Do not transfer any utilities (also referred to as transient commands); the ones provided with your new system diskette will replace those from the previous version.

3) Place a single or double density CP/M version 2.2 (or later) disk in drive A and your single density disk created above in drive B. Use PIP from the new version to transfer the program and data files from B to A. Do not use the version of PIP from the earlier revision of CP/M.

Transfer all the files from your double density disks created under a previous CP/M version to the new one in this manner.

Do not use the Micromation COPY utility to make 2.2 duplicates of your 1.4 or earlier double density disks. This will render the files on the 2.2 disk unreadable.

Do not use any utilities from previous versions. Use only the ones provided on the distribution diskette.
SECTION 2
THEORY OF OPERATION

6 INTRODUCTION

The DOUBLER is a byte oriented floppy disk controller. It has an on board PROM that allows for bootstrap start, and holds primitives that control hardware systems on the board. Features include single and double density disk formats with automatic selection of the format on the disk, an RS-232 serial port with baud rate select, and a variety of control configurations for the S-100 bus.

The disk controller transfers data under program control on a sector by sector basis. In CP/M compatible single density format there are 26 sectors per track with 128 bytes per sector transferred at a data bit rate of 250KHz (IBM 3740 standard). In the double density format there are 52 sectors per track with 128 bytes per sector transferred at a data bit rate of 500KHz.

The intent of this theory of operation is to describe the hardware systems on the DOUBLER. Since there are many references to the schematic diagrams, component parts are referenced by the page number of the schematic followed by the part number. The part numbers on the schematic also refer to the column and row that the package occupies on the board. After the part number, the part type is listed in parentheses. For instance 2IC10C (74LS374) refers to page 2 of the schematic, IC 10C (which is the IC in column 10 at row C), of the type 74LS374.

6-1 DISK SYSTEM OPERATION

When the disk system is to be accessed the intent of the operator is translated by CP/M into a sequence of events. The drive to be used, selection of read or write, and the file to be found, are provided, indirectly, by the operators actions. These parameters are then processed by the operating system to access the appropriate portion of the disk.

Disk I/O is performed by a sequence of calls to the disk access subroutines, and by hardware that performs the ongoing processes of encoding, decoding, and phase lock to the serial data stream. When a request for disk access is made, the operating system reads the file to be written to the disk or allocates memory space to accept the file read from the disk. The operating system must then select the drive to be accessed, load the read/write head and move it to the proper track on the disk, phase lock the controller to the serial data stream on the disk, locate the sector to be operated on, perform a read or write record operation, and determine if the transfer is completed. If the transfer is not complete the next record is selected,
located, and written to or read from, until the transfer is completed.

The routines used to control the disk drive, and interface to the operating system are in BIOS and the C2-PROM (Appendix B is a listing of the C2-PROM). The routines in the C2-PROM are an extension of BIOS. They are closest to the DOUBLER hardware, while BIOS holds the more executive functions. A listing of the MICROMATION custom BIOS and C2-PROM can be found in the distribution diskette, and information on the standard model BIOS is included in the CP/M reference manuals.

The parameters used to access the sector on the disk are held in the scratch pad RAM. For example, TRACK, SECTOR, DMA (the address of a memory buffer used for the source or destination of data during transfers), and DENBYTE are registers in the RAM that pertain directly to sector read/write operations. A complete list of these registers is included in the C2-PROM listing.

When a disk is accessed for the first time after being inserted in the drive, it is logged and tested for density. Testing for density is executed by trying to read the SYNC FIELD HEADER on track 2 in single density. After 30 unsuccessful tries at single density, double density is tested. DENBYTE, in the scratch pad RAM, is set according to test results.

6-2 DISK READ/WRITE

During disk read and write operations the operating system loads the head, steps to the selected track, and tests DENBYTE for the density of the disk. A call to the SYNC subroutine then finds the sync field header (see Appendix A if unfamiliar with the sector format) in the ID FIELD and establishes the synchronization of PLO and the byte sync counter with the moving disk data. Once in sync, the operating system looks for the sector ID MARK. When it is found, the track intended is checked against the track read. If it is the correct track, a sector by sector search is executed until the selected sector is found. CRC is checked during these operations to ensure that the disk has been read correctly.

After the proper sector has been found, read or write of the DATA FIELD begins. When a read operation has been requested, the operating system looks for the DATA MARK, then reads the 128 bytes of data in the sector, and finally checks the CRC to confirm the accuracy of the data. When a write operation has been requested, the operating system writes a new SYNC FIELD HEADER and a DATA MARK in the data field, then writes 128 bytes of data followed by the CRC.

Read or write operations are a byte by byte transfer of a sector of data to or from the system memory area marked by the disk memory address, DMA, (not to be confused with direct memory access). After a sector has been read, the operating system
takes the information in the memory area marked by DMA and uses it to build the file being read. The operating system then provides parameters on a new sector to be transferred, and transfers it, or ends the read operation. After a sector has been written, another sector of data is placed in the memory area marked by DMA, and transferred, or the write operation is ended.

7 THE S-100 BUS INTERFACE

The DOUBLER's S-100 bus interface has three sections; the control bus, the bidirectional data bus (D0-D7), and the address bus (A0-A15).

7-1 THE CONTROL BUS

The control bus is used to control data transfers between the processor and memory or peripherals. The DOUBLER uses the following signals:

PDBIN is used by the processor to indicate that a valid address is on the address bus and that it is reading data on data bus lines D0-D7 from memory or an I/O port.

/PWR is used by the processor to indicate that a valid address is on the address bus and that it is outputting data on data bus lines D0-D7 to memory or an I/O port.

SINP and SOUT are used by the processor to indicate input or output, respectively, to an I/O port. They are similar to the PDBIN and /PWR signals and are active when these signals are in coincidence with /IOREQ. These lines disable the DOUBLER when active.

SINTA indicates that the processor is in an interrupt mode. The DOUBLER is disabled when this signal is active.

/PHANTOM disables the RAM that occupies the same memory position as the DOUBLER. It is active whenever the DOUBLER is enabled.

PSYNC is a synchronizing signal used with a 4MHz CPU clock to synchronize wait state requests to the processor machine cycles.

/PRST (reset) is used on the DOUBLER board to generate a power on jump which accesses the jump to BOOT instruction in the resident firmware.

XRDY and PRDY are wait state inputs to the processor. One of these control lines (user option) is used by the DOUBLER floppy disk interface to make the CPU wait, on a byte by byte basis, during data transfers to and from the disk.
7-2 THE DATA BUS

The bidirectional data bus handles data transfers between the DOUBLER and the processor. It is isolated from the on board data bus by a bidirectional tri-state buffer, consisting of 1IC10D (74LS244) and 1IC11D (8304).

This buffer writes data to the DOUBLER board whenever /PWR is active, and reads data to the bus when PDBIN is active and the board is enabled by the address decoder. When the DOUBLER is not enabled the data bus buffers are in a high impedance state.

7-3 THE ADDRESS BUS

The address bus is used to enable and select registers that communicate with the disk system on the DOUBLER board, and to access the UART, scratch pad RAM, and the C2-PROM.

8 ADDRESS BUS DECODING

Decoding of the high order address bits takes place in the address decoder PROM 1IC9C (74S287). The low order bits are connected directly to the devices addressed or to the read/write control.

8-1 THE ADDRESS DECODER PROM

The address decoder generates the /RAM, I/O, PROM, and BD signals from address lines A9-A15. Decoding of these lines takes place in the bipolar PROM, 1IC9C (74S287). Note that address inputs to 1IC9C do not correspond to address bus lines. The A3 input to 1IC9C is set by the NOR of control bus signals SINP, SOUT, and SINTA. If any of these lines are active the decoder PROM output lines, and the board, are not enabled. BD is active whenever /RAM, I/O, or PROM are active. BD is used as the board enable, and inverted, it drives the /PHANTOM line.

The DOUBLER occupies the memory space from F800H to FFFFH. This area is divided into three sections as follows.

<table>
<thead>
<tr>
<th>Section</th>
<th>Address Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>C2-PROM</td>
<td>F800H - FBFFH</td>
</tr>
<tr>
<td>Scratch pad RAM</td>
<td>FC00H - FDFFH</td>
</tr>
<tr>
<td>I/O</td>
<td>FE00H - FFFFH</td>
</tr>
</tbody>
</table>

8-2 THE C2-PROM

The C2-PROM, 6IC9D (2708), is enabled by the PDBIN and PROM enable signals, accessed by address lines A0-A9, and based at address F800H. It holds the bootstrap loader and routines that control the DOUBLER. Refer to the C2PROM.ASM listing in Appendix B.
8-3 THE SCRATCH PAD RAM

The 64 byte scratch pad RAM, 6IC9A (4036), is accessed by address lines A0-A5. It is selected by the /RAM signal from the address decoder. RAM output enable and R/W are controlled by the /WR signal. This RAM is assigned the dedicated registers and the stack used by the routines in C2-PROM and BIOS.

8-4 READ AND WRITE CONTROL

The read/write control generates strobes that operate the read, write, control, and status latches on the DOUBLER's internal data bus. Its inputs are address lines A0-A2, /WR, /PDBIN, and I/O. This circuit consists of two, eight wide data distributors, 1IC7D and 1IC6D (74LS138). Both of these are enabled by the I/O signal from the address decoder PROM. 1IC6D and 1IC7D are also enabled by /PDBIN and /WR, respectively, /PDBIN enables the read control; /WR enables the write control. Low active strobes generated by the read/write control memory map are listed in firmware as follows.

<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>/WR</th>
<th>/PDBIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE00H</td>
<td>WRCONT</td>
<td>RDSTAT</td>
</tr>
<tr>
<td>FE01H</td>
<td>WRCLK</td>
<td></td>
</tr>
<tr>
<td>FE02H</td>
<td>WRUART</td>
<td>RDUART</td>
</tr>
<tr>
<td>FE04H</td>
<td>WMRKCRC</td>
<td>RDMRKC</td>
</tr>
<tr>
<td>FE05H</td>
<td>WMRK</td>
<td>RDMARK</td>
</tr>
<tr>
<td>FE06H</td>
<td>WRDATA</td>
<td>RDDATA</td>
</tr>
<tr>
<td>FE07H</td>
<td>WRCRC</td>
<td>SYNCPORT</td>
</tr>
<tr>
<td>FE0AH</td>
<td></td>
<td>UARTSTAT</td>
</tr>
</tbody>
</table>

These strobes and their corresponding latches perform the following functions.

WRCONT loads the drive control latch, 5IC10A (74S374), which operates the drive control lines.

RDSTAT reads the drive status latch, 5IC11A (74LS224), which contains the drive status lines.

WRCLK writes the clock pattern to the sync mark latch, 2IC12D (74LS273). This value is then compared with the clock pattern from the SYNC FIELD HEADER to synchronize with the moving disk data.

WRUART and RDUART write and read to the UART, 6IC13D (8251), used for the RS-232 serial interface.

WRMRK is a strobe that writes an ID or a DATA MARK to the disk (depending on the status of the MRKA signal).
RDMRK is a strobe that holds the processor until a MARK is read (or the timeout triggers) in order to synchronize the byte sync counter. It is also used to clear the head load counter.

WRMRKCRC and RDMRKCRC are strobes that perform the same functions as WRMRK and RDMRK, respectively, and also preset the cyclic redundancy check circuit.

WRDATA and RDDATA these strobes activate the DISKWR and /DISKREAD signals respectively. Addressing these ports transfers data to or from the disk on a byte by byte basis.

WRCRC is a strobe used to gate the cyclic redundancy check character into the serial data stream.

SYNCPORT is a strobe used in the synchronization of the byte sync counter.

UARTSTAT accesses the control and status latches in the UART.

Address line A2 is used to enable the DISKWR and /DISKREAD signals. All strobes listed above in the address range of FE04H - FE07H also transfer data to or from the disk when active.

8-5 THE UART

The UART, 6IC13D (8251), is based at FE02H, and selected by the /RUART and /WRUART signals from the READ/WRITE CONTROL. Address line A3 is connected to the UART C/D input, which accesses its control and status latch, based at FE0AH.

9 DISK DRIVE INTERFACE

The operating system controls and monitors the drives via the drive control latch (WRCONT), which operates the drive control input lines, and the drive status latch (RDSTAT), which contains the drive status outputs. These latches and their pin connection to the disk drive list as follows:

<table>
<thead>
<tr>
<th>BIT</th>
<th>PIN</th>
<th>WRCONT</th>
<th>PIN</th>
<th>RDSTAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>D0</td>
<td>36</td>
<td>/STEP</td>
<td>22</td>
<td>/RDY</td>
</tr>
<tr>
<td>D1</td>
<td>34</td>
<td>/DIR</td>
<td>10</td>
<td>/SEEK DONE</td>
</tr>
<tr>
<td>D2</td>
<td>SD/DD</td>
<td></td>
<td>18</td>
<td>/HEAD</td>
</tr>
<tr>
<td>D3</td>
<td>32</td>
<td>/DRIVE D</td>
<td>20</td>
<td>/INDEX</td>
</tr>
<tr>
<td>D4</td>
<td>30</td>
<td>/DRIVE C</td>
<td>24</td>
<td>/SECTOR</td>
</tr>
<tr>
<td>D5</td>
<td>28</td>
<td>/DRIVE B</td>
<td>44</td>
<td>/WRITE PRT</td>
</tr>
<tr>
<td>D6</td>
<td>26</td>
<td>/DRIVE A</td>
<td>42</td>
<td>/TRACK 00</td>
</tr>
<tr>
<td>D7</td>
<td>12</td>
<td>/RESTORE</td>
<td></td>
<td>CRCSTAT</td>
</tr>
</tbody>
</table>
CRCSTAT and SD/DD are listed in the latches but do not connect to the drives. The other signal pin connections to the drives are:

<table>
<thead>
<tr>
<th>PIN</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>/ABOVE 43</td>
</tr>
<tr>
<td>38</td>
<td>/DISK WRITE DATA</td>
</tr>
<tr>
<td>40</td>
<td>/WRITE GATE</td>
</tr>
<tr>
<td>46</td>
<td>/RAW DATA</td>
</tr>
</tbody>
</table>

9-1 THE PHASE LOCK OSCILLATOR

The heart of the floppy disk system is the phase lock oscillator, PLO, which generates signals used to synchronize to the moving disk data. Phase lock is the process of synchronizing an oscillator to an external signal. In order to work, control of the output frequency of the oscillator and a system to detect the frequency differences between the external signal and the oscillator must exist. The final part of the phase lock system is a feedback loop that allows the detector to control the oscillator.

In the DOUBLER, control of the PLO output frequency is achieved by digitally dividing the signal from an oscillator with a frequency twenty times higher than the frequency to be generated. The divider (a preset counter string) can be set to divide by any integer in the domain of 1 to 45.

Detection of frequency difference is executed by latching the divider status when a transition of the external signal (raw disk data) occurs.

The feedback loop is completed by a PROM that is coded to use the divider status as an input to output a preset value that will correct the difference in frequency.

The DOUBLER's PLO consists of a 20 MHz crystal oscillator, 4 Y1 and 4IC4A (74S04), which drives a presetable counter string, 4IC2A (74LS163) and 4IC3A (74LS161-A). The nominal count for double density is 20D, which yields a 1 MHz clock rate which is approximately equal to the frequency of the serial stream from the disk. The nominal count for single density is 40D which yields 500KHz. Notice that these values are twice the data transfer rate. This is necessary to accommodate the interleaved data and clock bits.

Preset values are provided for the counter string by one of two PROMs, the R1, 4IC1B (74S471), for read and the WA, 4IC2B (74S471), for write. The counters are continuously being incremented by the 20MHz clock. The logic conditioned raw data stream is used as a clock input to the latch, 4IC1A (74LS174). Data inputs to this latch are connected to the stage outputs of the counter string. When a pulse comes down the raw data stream
the current count of the counter string is stored in the latch. Outputs of this latch connect to the address inputs of the R1 PROM. If there is an error in timing between the disk data stream and the PLO, the preset value of the counter string is changed by the contents of the PROM to correct the timing error.

Since data on the disk is frequency modulated, changes in timing caused by modulation must be ignored, so the R1 PROM is coded to compensate only the larger errors in timing. If there is no incoming pulse during the current PLO count cycle, (when reading a zero data bit for example), the latch is cleared. If the incoming pulse is on time, the latch stores zero. Either of these conditions set the counter string to the nominal value, and PLO remains synchronous to the disk data stream.

PLO is the clock input of the byte sync counter, and is also used to shift disk data into the shift register, 21C11B and 21C11C (74LS164), used for conversion of raw disk data to eight bit parallel bytes. PLO is locked to the signal from the disk any time information is read from the disk. (Write operations involve a read of the ID field to verify the track and locate the sector to be written to.) Since the frequency to be locked is known, phase lock can be achieved in a few cycles.

During write operations, after the sector has been located, PLO becomes a frequency synthesizer and is set to the nominal frequency. Data is brought from the data bus to the disk write data latch, 21C10B (74LS165), and is shifted serially out by the PLO signal. This serial stream goes through the CRC, multiplexer, encoder, and precompensator, and then to the disk drive.

9-2 THE BYTE SYNC COUNTER

The byte sync counter, 31C4B (74LS161), provides signals used to separate clock bits from data bits and define the beginning and end of bytes in moving disk data. Outputs from this counter are also sent to the multiplexer, 3IC3B (74LS157), to insert clock bits into the disk write data and provide information for write precompensation (see section 9-5).

The byte sync counter is clocked by the /DPLO signal. This signal is PLO delayed by 50N$s$ which is one cycle of the 20MHz oscillator. /DPLO is used so that circuits using PLO can settle before action is taken on their outputs.

The byte sync counter is set, if not in sync, when SYNCPLO is addressed by the operating system. The counter is set so that C/D is high with respect to data bits in the moving disk data, EOC is high during data bit 7 and EOW is high during clock bit 0.

The SYNCMARK signal is connected to the "B" load input of the counter. If SYNCPLO is addressed and the counter is not in sync the high on the "B" load input is loaded. This steps the counter toward byte synchronization.
Detection of SYNCMARK starts in the shift register, 2IC11C and 2IC11B (74LS164), which is used to separate clock bits from data bits, and convert moving disk data into parallel bytes. Alternating stages of this shift register are connected to the disk read data latch, 2IC10C (74LS374), and the SYNCMARK comparator, 2IC12B and 2IC12C (74LS266). The SYNCMARK comparator sees the clock bits from the shift register and codes loaded by the operating system into the sync mark latch, 2IC12D (74LS273). When the codes in the sync mark latch and the shift register match, the SYNCMARK signal goes low. This sets EOC from the byte sync counter, 3IC4B (74LS161), and loads the multiplexed data pattern, from the SYNC FIELD HEADER, into the disk read data latch, 2IC10C (74LS374). The operating system reads this latch and, if the proper code is found, verifies sync and continues the read or write sector operation.

9-3 WAIT STATES (XRDY or PRDY)

Data recovery from the disk is slower than the cycle time of the CPU. Consequently the XRDY or PRDY lines are held low to hold the processor in a wait state and allow the current byte of the disk read or write operation to be transferred. The byte sync counter output, EOC, marks the end of a byte. EOC is used to lift the wait state and transfer a complete data byte from the DOUBLER to the data bus, or from the data bus to the DOUBLER.

If for some reason the DOUBLER cannot complete the byte transfer, a timer, 1IC13A (4040), is used to prevent loss of dynamic RAM data. It lifts the wait state before the refresh timing limits of dynamic RAM are exceeded.

Wait states must also be generated when a 4MHz processor clock is used. PSYNC and BD (board enable) coincidences are detected in an AND gate 1IC5D (74LS00). When coincidence is detected XRDY and PRDY become active, generating one wait state per board access. This is needed to allow adequate access time for the C2-PROM. A jumper labeled "WAIT" can be found at location D-4 on the DOUBLER. It must be installed to operate at 4MHz.

9-4 THE CYCLIC REDUNDANCY CHECK

The cyclic redundancy check, CRC, is an ongoing process carried on by the CRC IC, 3IC8B (8506). During write operations a complex logic creates a unique code from the data sent to the sector, called the cyclic redundancy check character (CRCC), which is appended to the sector. During disk read operations the CRCC is read from the disk and compared with the character calculated from the data just read. If an error condition is found, the operating system attempts to read again. If subsequent retries fail, the operating system displays an error message.
9-5 ENCODING AND WRITE PRECOMPENSATION

The double density recording format pushes the recording medium to the limit. When two transitions of magnetic polarity are written adjacent to each other, they interact, causing a shift in timing. This shift makes the data stream unreadable. The solution is to write adjacent pulses shifted, so that their interaction yields the correct position in time along the disk serial data stream. This corrective shift of timing, prior to write, is called write precompensation. In the DOUBLER, compensation in write timing is achieved by moving the position of clock bits with respect to data bits. The mechanism for doing this is in the PLO. The PLO is a digitally controlled frequency synthesizer. During read, frequency control is used to achieve synchronization with the moving data stream on the disk. During write, clock pulses are offset in time by changing the count of the counter string, 4IC2A and 4IC3A (74LS161A), on a bit by bit basis.

During write, control of the PLO counter string is executed by the WA PROM, 4IC2B (74S471). The address inputs to this PROM represent a portion of the serial data stream mixed with clock. The data outputs of the PROM are connected to the preset inputs of the counter string. Coding in the PROM presets the counter string to provide write precompensation.

WA PROM output D7 is the serial data stream sent to the disk during write. Coding in the PROM also holds the algorithms for encoding disk data. Both single and double density codes are in the WA PROM. Write precompensation is not used in single density. So for single density write the counter string is always set to the nominal count, which yields a 500KHz output frequency.

10 THE RS-232 SERIAL INTERFACE

The terminal interface consists of an RS-232 serial interface designed around the UART, 6IC13D (8251), on the DOUBLER. The UART clock is derived from system clock by counters, 6IC14D (74LS161), and 6IC14A (4024). Baud rate is selectable via a jumper from the outputs of the counters. There is a provision for use of the on board 20MHz clock to generate the 2MHz UART clock via 6IC3C (74LS90 not provided), if the host system has a different clock frequency.

The following is a list of the RS-232 connections:

| J2-1 | TxD | transmitter data |
| J2-2 | GND | ground |
| J2-3 | /RTS | request to send |
| J2-5 | /DSR | data set ready |
| J2-6 | /DTR | data terminal ready |
| J2-7 | /CTS | clear to send data |
| J2-9 | /RxD | receiver data |
To connect a video display terminal, signal ground, TxD, and /RxD are all that need be connected. The other signals are for handshake arrangements not usually necessary for terminals.

11 POWER ON JUMP

When power is first applied to the computer, or the reset button is pushed, a reset (/PRST) pulse is generated activating the power on jump circuit on the DOUBLER which generates a /POJ signal. With this signal active the C2-PR0M is enabled, and the JMP COLDBOOT instruction at address F800H in the PROM is executed. Since the high order address bits are decoded by the address decoder, F800H appears to be 0000H when /POJ is active.

COLDBOOT reads tracks "0" and "1" (the system tracks) of the diskette in drive "A" into memory locations 0000H-007FH, and then executes the transferred code by performing a jump to 0000H, to load the system into memory.

12 POWER SUPPLY

The DOUBLER runs on the S-100 bus supply line voltage. On board regulation produces +5v, -5v, +12v, and -12v. The +5 volt line can draw more than an ampere, while the other lines draw less than 50 milli amperes.

Ceramic disk capacitors are distributed along the power rails, in accordance with good digital logic design, to reduce noise.
Appendix A:

THE DISK FORMAT

The MICROMATION DOUBLE DENSITY FORMAT divides the disk into tracks (numbered 0 - 76). Each track has 52 sectors. The position of the tracks is set by the position of the read/write head in the disk drive. Sectors are sequentially positioned within the track starting at the index.

Each sector is divided into two parts; the ID FIELD and the DATA FIELD. The ID FIELD is written when the disk is formatted and is used to find the sector to be read from or written to. This is called soft sectoring.

Gaps are inserted between the sectors, and between the ID and DATA fields. These allow the write current to be turned on without destruction to the information recorded on the disk. Gaps are recorded with the hex number 4E, which identifies them as gaps when the operating system is locating sectors.

The ID field marks the start of a sector, identifies it and verifies the track. The first six bytes of the ID FIELD are the SYNC FIELD HEADER. These have a unique pattern written into the clock pulses and are therefore readily distinguished from the rest of the serial data stream. The SYNC FIELD HEADER is used to synchronize the hardware to the serial data stream. The next byte is the ID MARK. This verifies that the SYNC FIELD HEADER found is in an ID FIELD, and prepares the operating system to read the track and sector bytes. The ID MARK is followed by a byte written with the hex number FE, which ensures that an ID FIELD, not a gap, has been found. This is necessary since gaps are written with random information, that may mimic an ID FIELD, when the write current is turned on.

The DATA FIELD is used to store the 128 bytes of data recorded in a sector. There is a gap that separates it from the ID field. So the SYNC FIELD HEADER, and the DATA MARK, in the DATA FIELD are needed to synchronize the hardware again. When the disk is formatted, the SYNC FIELD HEADER and the DATA MARK are written, and the data area of the DATA FIELD is filled with the hex number E5.

When data is written to the disk, only the DATA FIELD is changed. A new SYNC FIELD HEADER and DATA MARK are written, followed by the data to be recorded in the sector.

CRC, cyclic redundancy check, bytes are appended to the fields in sectors. They are used by the hardware system to verify that the serial data was read without errors.
**MICROMATION DOUBLE DENSITY FORMAT**

<table>
<thead>
<tr>
<th>FORMAT</th>
<th>4E 4E'S</th>
<th>SECTOR 1</th>
<th>SECTOR 2</th>
<th>SECTORS 3-51</th>
<th>SECTOR 52</th>
<th>4E'S</th>
</tr>
</thead>
</table>

**FORMAT**

<table>
<thead>
<tr>
<th>4E 00</th>
<th>ID MARK</th>
<th>FE TRACK</th>
<th>SECTOR</th>
<th>CRC</th>
<th>CRC</th>
<th>4E 00</th>
<th>DATA MARK</th>
<th>DATA</th>
<th>CRC</th>
<th>CRC</th>
<th>4E</th>
</tr>
</thead>
</table>

**ID FIELD**

<table>
<thead>
<tr>
<th>ID &amp; DATA MARKS:</th>
<th>A1 DATA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0A CLOCK</td>
</tr>
</tbody>
</table>

**DATA FIELD**
APENDIX B:
C2-PROM LISTING

WARNING: This listing is provided for information only. It may not be the exact information for your DOUBLER. Refer to the listing provided in the distribution diskette for exact information.

; PROM ROUTINES FOR MICROMATION DOUBLER, VERSION C.2
; THE C.1 VERSION HAS NOPs IN SYNC ROUTINE TO ALLOW MORE FREQUENT REFRESH OF DYNAMIC RAMS
; IT ALSO SETS UP THE SIDE BIT EARLIER TO MEET SETUP TIME FOR Y-E DATA DRIVES
; THIS VERSION HAS THE FOLLOWING CHANGES FROM C.1:
; HAS FIX FOR C.1 BUG IN SETTING UP DENSITY
; ENABLES INTERRUPTS AFTER FINDING CORRECT SECTOR
; HAS SLOWER STEP AND SETTLE TIMES
;
FEB 11, 1980

BASE ORG 0F800H
BUFF EQU BASE+400H ; SCRATCHPAD RAM

; START OF HARDWARE PORT DEFINITIONS

WRCONT EQU BASE+600H
WRCCLK EQU WRCONT+1
WRUART EQU WRCONT+2
WMRMKRCRC EQU WRCONT+4
WMRK EQU WRCONT+5
WRDATA EQU WRCONT+6
WRCRC EQU WRCONT+7
RDSTAT EQU WRCONT
RDUART EQU WRCONT+2
RDMRKRCRC EQU WRCONT+4
RDMARK EQU WRCONT+5
RDDATA EQU WRCONT+6
SYNCPEORT EQU WRCONT+7

; START OF RAM VARIABLE DEFINITIONS

ERRORBYTE EQU BUFF ; NO. OF ERRORS DURING RETRIES
DENBYTE EQU BUFF+1 ; 0 FOR SINGLE DENSITY
; 4 FOR DOUBLE DENSITY
READWRITE EQU BUFF+2 ; 0 FOR READ
; 10H FOR WRITE
CONTROLBYTE EQU BUFF+3 ; RAM IMAGE OF RDSTAT OR WRCONT
TRACK EQU BUFF+4
PRESDISK EQU BUFF+5
LOGINTAB EQU BUFF+6 ; FOR EACH DRIVE
; 0 IF DRIVE HAS NOT BEEN LOGGED IN
; 55H IF DRIVE HAS BEEN LOGGED IN
SECTOR EQU BUFF+0AH
DMA EQU BUFF+0BH ; DMA ADDRESS
DISK EQU BUFF+0DH
TESTNEXT EQU BUFF+0EH ; 55H IF WANT TO TEST DENSITY
; OF NEXT TRACK
TWOSIDE EQU BUFF+0FH
STEPTIME EQU BUFF+10H
ABOVE43 EQU BUFF+11H ; 10H IF (TRACK)<44D
; 50H OTHERWISE
TRACKTAB EQU BUFF+12H
DENMAP EQU BUFF+16H ; SAME CONVENTION AS DENBYTE
TRY1 EQU BUFF+20H
RETRYCOUNT EQU BUFF+21H
CURRDRIVE EQU BUFF+22H
TESTMAX EQU BUFF+23H ; NO. RETRIES FOR DENSITY TEST

STEPSETTLE EQU 15
HEADSETTLE EQU 40
STACK EQU BUFF+64D

JMP COLDBOOT
JMP HOME
JMP SELDSK
JMP SETTRK
JMP SETSEC
JMP SETDMA
JMP READ
JMP WRITE
JMP SKEW
JMP SETDEN

WRITEPROTECT:
CALL DISKREADY1 ; LOADS HEAD
; WAITS TILL DISK READY
; RETURNS (RDSTAT) IN B
MOV A,B
ANI 04
; WRITEPRT BIT FROM DRIVE
RZ
LDA RDMARK
; RESETS HEAD LOAD COUNTER
RET

READ:
; ENTRY POINT FOR READ ROUTINE
XRA A
; (READWRITE)= 00 FOR READ
JMP GO

WRITE:
; ENTRY POINT FOR WRITE ROUTINE
MVI A,10H
; (READWRITE)=10H FOR WRITE
GO  STA READWRITE 
LHLD  DENBYTE ;(L)=(DENBYTE) 
LDA  CONTROLBYTE 
CMA 
ANI  0FBH ;MASK OUT BIT 2 (SD/-DD = 0) 
ORA  L 
CMA  
STA  WRCONT  
CALL  DISKREADY1 
LDA  SECTOR 
MOV  C,A ;(C)=(SECTOR) 
LDA  TRACK 
MOV  B,A ;(B)=(TRACK) 
XRA  A 
STA  ERRORBYTE ;(ERRORBYTE)= 0 
MOV  A,L 
ORA  A ;TEST FOR SINGLE DENSITY  
JZ  SD  
READDD: CALL ;DOUBLE DENSITY READ OR WRITE  
BLOOP  
MVI  SYNC ;SYNC ON HEADER 
;FOUND HEADER  
LDAX  M,0AH ;FIND OA CLOCK FOR ID MARK  
LDA  D ;SYNC WITH -EOW  
LDA  RDMRKRC 
CPI  0A1H ;DATA FOR ID MARK  
JNZ  BLOOP ;FOUND ID ADDRESS MARK  
LDAX  D ;BYTE AFTER ID MARK SHOULD BE FE  
CPI  0FEH 
JNZ  BLOOP ;FOUND FE BYTE  
LDAX  D ;TRACK BYTE FROM DISK  
CMP  B ;(B)=(TRACK) 
JNZ  TERROR1 ;TRACK ERROR  
LDAX  D ;SECTOR BYTE FROM DISK  
CMP  C ;(C)=(SECTOR) 
JNZ  BLOOP ;WRONG SECTOR. TRY AGAIN  
LDAX  DI ;DISABLE INTERRUPTS BEFORE  
;CHECKING ID CRC  
LDAX  D ;READ 1 BYTE PAST ID CRC  
LDA  RDSTAT ;CHECK ID CRC  
RAR  
LDAX  D ;ID CRC ERROR  
LDAX  D 
LDA  ABOVE43 
MOV  B,A 
LDAX  D 
33
MOV M,B ; (WRCLK) = (ABOVE43)
LDAX D ; NOW 5 BYTES INTO GAP
MVI B,9
GLOOP
LDAX D
DCR B
JNZ GLOOP

LDAX D ; NOW 15 BYTES INTO GAP
LDA READWRITE
ORA A ; CHECK FOR WRITE
LDAX D ; 16 BYTES INTO GAP
JNZ WRITEDD

LDAX D
LDAX D
MVI M,0FFH
LDAX D
LDAX D
LDAX D ; 21 BYTES INTO GAP
INX D ; (D) = SYNCPORT
LDAX D ; SYNC ON FF CLOCK PATTERN
DCX D ; (D) = RDDATA
MVI M,0AH ; (WRCLK) = 0A
LHLD DMA ; CLOCK PATTERN FOR DATA MARK
LDAX D ; SYNC WITH -EOW
LDA RDMRKCRC ; GET DATA PATTERN FOR DATA MARK
CPI 0A1H
JNZ ERROR ; MISSING DATA MARK

RXFER
LDAX D
MOV M,A
INX H
MOV B,D
LDAX D
MOV M,A
INX H
MOV C,E
LDAX B
MOV M,A
INX H
MVI E,0E1H
LDAX B
MOV M,A
INX H
LDAX B ; 4 BYTES OF DATA
ERROR: ;ARRIVE HERE ON ANY OF FOLLOWING CONDITIONS
;
; 30H TRACK ERRORS
;
; 3DH CRC ERROR
;
; MISSING DATA MARK
;
; DATA CRC ERROR

MVI A, 0EFH ;RETURN EFH IN ACC
ORA A ; (UNSUCCESSFUL READ)
STA WRCLK
RET

TERROR: ;ARRIVE HERE ON TRACK ERROR IN SINGLE DENSITY

CALL ERRORCOUNT ;INCREMENT ERRORBYTE
JNZ ALOOP ;TRY AGAIN IF LESS THAN 30H

NO MVI A, 0EFH ;30H TRACK ERRORS
ORA A ;RETURN EFH IN ACC
STC ; (UNSUCCESSFUL DISK OPERATION)
STA WRCLK
RET

TERROR1: ;ARRIVE HERE ON TRACK ERROR IN DOUBLE DENSITY

CALL ERRORCOUNT ;INCREMENT ERRORBYTE
JNZ BLOOP ;TRY AGAIN IF LESS THAN 30H
JMP NO
ERRORCOUNT:
  LXI  H,ERRORBYTE
  INR  M ;INCREMENT ERRORBYTE
  MOV  A,M
  CPI  70H
  RET

WRITEDD: ;DOUBLE DENSITY WRITE
;ARRIVE HERE 16 BYTES AFTER ID FIELD
  MVI  A,4EH
  STAX  D ;WRITE 4 BYTES OF 4E
  STAX  D
  STAX  D
  STAX  D
  XRA  A
  STAX  D ;WRITE 6 BYTES OF 00
  STAX  D
  LHLI  DMA
STAX  D
STAX  D
STAX  D
LXI  B,WRMRKRCR
STAX  D
STAX  D
MVI  A,0A1H
STAX  B ;WRITE DATA MARK (A1)
MVI  C,0E1H ;START WRITING DATA TO DISK FROM MEMORY
WXFER
  MOV  A,M
WLOOP
  STAX  D
  INX  H
  INR  C
  MOV  A,M
  STAX  D
  INX  H
  MOV  A,M
  STAX  D
  INX  H
  MOV  A,M
  INX  H
  STAX  D
  MOV  A,M
  JNZ  WLOOP
;WHEN WE ARRIVE HERE WE'VE WRITTEN
; 31*4=124 BYTES TO DISK
  STAX  D
  INX  H
  MOV  A,M
  STAX  D
  INX  H
  MOV  A,M
  INX  H
  STAX  D
  MOV  A,M
  STAX  D ;128TH BYTE TO DISK
MVI A,0FFH
STA WRCRC ;WRITE 2 BYTES OF DATA CRC
STA WRCRC
STAX D ;WRITE 3 BYTES OF FF
STAX D
STAX D
XRA A ;RETURN 00 IN ACC
STA WRCLK ;(SUCCESSFUL WRITE)
RET

;SINGLE DENSITY ROUTINES
;ENTRY POINT IS SD (BELOW)

WRITESD: ;ARRIVE HERE 6 BYTES PAST ID FIELD
MVI A,0FFH
STAX D ;WRITE 3 BYTES FF (BYTES 7,8,9)
STAX D
STAX D
XRA A
STAX D ;WRITE 6 BYTES 00 (BYTES 10-15)
STAX D
LHLD DMA
STAX D
STAX D
STAX D
STAX D
STAX D ;BYTE 15 OF GAP
MVI A,0FBH ;WRITE DATA MARK FOR SINGLE DEN
STA WMRKCRC
MVI C,0E1H
JMP WXFER ;JUMP TO COMMON WRITE ROUTINE

SYNC: ;ROUTINE TO SYNC ON HEADER
LXI H,WRCLK
LXI M,0FFH
LXI D,SYNCPORT
CLOOP LDAX D ;SYNC ON FF CLOCK IN HEADER
ORA A ;SHOULD HAVE 00 DATA
;FOUND SYNC PATTERN
NOP
NOP
DCX D ;(D)=WRDATA=READDATA
RZ
JMP SYNC

SD: ;SINGLE DENSITY ENTRY POINT
ALOOP CALL SYNC ;FOUND HEADER
MLOOP MVI M,0C7H ;CLOCK PATTERN FOR ID MARK
LLOOP LDA RMRKCRC
ORA A
JZ LLOOP
CPI 0FEH
JZ NLOOP
MVI M,0FFH
LDA SYNCPORT

37
NLOOP:
ORA A
JZ MLOOP
JMP ALOOP

LDAX D
;FOUND DATA MARK
CMP B
;TRACK BYTE FROM DISK
JNZ TERROR
;TRACK ERROR
LDAX D
;SIDE BYTE FROM DISK (IGNORE)
LDAX D
;SECTOR BYTE FROM DISK
CMP C
JNZ ALOOP
;WRONG SECTOR. TRY AGAIN

DI

LDAX D
;FOUND CORRECT TRACK AND SECTOR
LDAX D
;DISABLE INTERRUPTS BEFORE
LDAX D
;CHECKING ID CRC
LDAX D
;CRC BYTE
LDAX D
;CRC BYTE
LDAX D
;GAP BYTE 1
LDA RDSTAT
;CHECK ID CRC
RAR
LDAX D
;GAP BYTE 2
LDAX D
;GAP BYTE 3
JC ERROR
;ID CRC ERROR
LDAX D
;GAP BYTE 4
LDA ABOVE43
MOV M,A
LDAX D
;GAP BYTE 5
LDA READWRITE
ORA A
;CHECK FOR WRITE
LDAX D
;GAP BYTE 6
JNZ WRITESD
;SINGLE DENSITY READ
LDAX D
;READ 6 BYTES OF GAP
LDAX D
LDAX D
LDAX D
LDAX D
LDAX D
MVI M,0FFH
(LWCLK)=FF
LXI B,RDDATA
LDAX D
;GAP BYTE 14
INX D
;(D)=SYNC_PORT
LDAX D
MVI M,0C7H
;CLOCK PATTERN FOR DATA MARK
MVI E,04
;(D)=RDMRKCRC
LDAX B
;GAP BYTE 16
LDAX D
;READ DATA MARK
ANI 0FCH
CPI 0F8H
;DATA PATTERN FOR DATA MARK
JNZ ERROR
;MISSING DATA MARK
MVI E,0E0H
;FOUND SINGLE DENSITY DATA MARK
32*4=128 BYTE TRANSFER
LDAX B
LHLD DMA
JMP RLOOP ;JUMP TO MAIN READ ROUTINE

TEST: ;TESTS DENSITY OF DISKETTE IN LOGGED-IN DRIVE
;RETURNS 00 IN ACC IF DOUBLE DENSITY
;RETURNS 0F IN ACC IF SINGLE DENSITY
;RETURNS 0A IN ACC IF TEST FAILS
XRA A
STA TESTMAX ;(TESTMAX)=0
TEST1 XRA A
STA ERRORBYTE ;(ERRORBYTE)=0
CALL DISKREADY ;LOAD HEAD
LXI B,WRCONT
LDA CONTROLBYTE
ORI 80H ;SET CONTROLLER FOR SIDE 0
ANI 0FBH ;TRY DOUBLE DENSITY
STAX B

LOOP6: ;DOUBLE DENSITY TEST
LXI H,WRCLK
MVI M,0FFH
LXI D,SYNCPORT ;SYNC ON FF CLOCK IN HEADER
LOOP7 LDAX D ;READ DATA PATTERN
INR L ;ABORT AFTER 256 TRIES
JZ RETRY
ORA A ;DATA SHOULD BE 00
JNZ LOOP7 ;FOUND HEADER
DCX D ;(D)=READDATA
MVI L,01 ;(H)=WRCLK
MVI M,0AH
LDAX D ;SYNC WITH -EOW
LDA RDMRKCRC ;LOOK FOR ID MARK
CPI 0A1H
JNZ RETRY ;FOUND ID MARK
LDAX D ;FE BYTE
LDAX D ;SECTOR BYTE
LDAX D ;CRC BYTE
LDAX D ;CRC BYTE
LDAX D ;GAP BYTE 1
LDAX B
RAR
JC RETRY ;CHECK ID CRC
XRA A ;ID CRC OK
RET

RETRY CALL ERRORCOUNT
JNZ LOOP6 ;SINGLE DENSITY TEST
;ARRIVE HERE AFTER 30H TRIES AT DOUBLE DENSITY
SDTEST XRA A
STA ERRORBYTE ;(ERRORBYTE)=0
LDA CONTROLBYTE
ORI 84H ;SET UP SIDE 0, SINGLE DENSITY
STAX B ;TO WRCONT
SDLOOP1 MVI E,07 ;(D)=SYNCPORT
LXI H,WRCLK
MVI M,0FFH ;SYNC ON FF CLOCK PATTERN
SDLOOP2 LDAX D ;GET CORRESPONDING DATA
INR L ;ABORT AFTER 256 TRIES
JZ RETRY1
ORA A ;DATA SHOULD BE 00
JNZ SDLOOP2 ;FOUND HEADER
DCX D ;(D)=READDATA
MVI L,01 ;(H)=WRCLK
MVI M,0C7H ;LOOK FOR C7 CLOCK
LDAX D ;SYNC WITH -EOW
LDA RDMRKCRC
CPI 0FEH ;DATA FOR ID MARK
JNZ RETRY1 ;FOUND ID MARK
LDAX D ;TRACK BYTE
LDAX D ;SIDE
LDAX D ;SECTOR
LDAX D
LDAX D ;CRC BYTE
LDAX D ;CRC BYTE
LDAX D
LDAX B ;GET RDSTAT
RAR RETRY1 ;CHECK ID CRC
JC RETRY1
ORI 0FFH ;ID CRC OK
RET ;RETURN FF

RETRY1 CALL ERRORCOUNT
JNZ SDLOOP1 ;FAILED BOTH DOUBLE AND SINGLE DENSITY
; TESTS 30H TIMES
LXI H,TESTMAX
INR M ;INCREMENT TESTMAX
MOV A,M
CPI 0
JNZ TEST1 ;FAILED TEST 10 TIMES
ORA A ;RETURN 0A
RET

40
SKEW:  ; COMPUTES PHYSICAL SECTOR FROM LOGICAL SECTOR
        ; SKEW FACTOR IS 8
        ; INPUT AND OUTPUT ARE IN C REG
        ; OUTPUT = (((INPUT) MOD 52)*8 - 7) MOD 52
        ; IF INPUT>52, SELECTS SIDE 1
LXI  H, 0
PUSH H
LDA CONTROLBYTE
ANI  7FH
MOV E, A
MOV A, C
SUI  52
MOV B, A
MOV A, E
JP SKIPI
ORI  80H
MOV B, C
SKIPI STA TWO SIDE
MOV A, B
MOV L, B
POP B
LOOP10 INR C
SUI 13
JP LOOP10
DAD H
DAD H
DAD H
MOV A, H
ORA A
MOV A, L
CNZ HIGH
LOOP11 CPI 52
JC SKIP12
ADI 204
JMP LOOP11
SKIP12 ADD C
MOV C, A
RET
HIGH ADI 48
RET

SETDMA MOV H, B
        MOV L, C
        SHLD DMA
        RET  ; STORE DMA ADDRESS

SETSEC MOV A, C
        STA SECTOR
        RET  ; STORE SECTOR NUMBER
```
SETTRK:  ;STEPS DRIVE TO TRACK (C)
   MOV  A,C
   CPI 44D   ;IF (C)<44
   MVI  A,10H ; THEN (ABOVE43)=10H
   JC   SKIP3
   MVI  A,50H ; ELSE (ABOVE43)=50H
SKIP3  STA  ABOVE43
   CALL DISKREADY

STEPLOOP LXI  H,TRACK
   MOV  A,M   ;GET (TRACK)
   CMP  C    ;DONE?
   JZ    DONESTEP
   CALL STEPHEAD   ;NO, STEP HEAD
   JMP   STEPLOOP   ;REPEAT

STEPHEAD JC  STEPIN   ;IF (TRACK)<(C) THEN STEP IN
STEPOUT LDA  CONTROLBYTE   ;ELSE STEP OUT
   DCR  M   ;(TRACK)=(TRACK)-1
   ORI  02H   ;DIR=OUT
   JMP   DOSTEP

STEPIN LDA  CONTROLBYTE
   INR  M   ;(TRACK)=(TRACK)+1
   ANI  0FDH   ;DIR=IN

DOSTEP STAX  D   ;STORE DIRECTION IN WRCONT
   DCR  A   ;-STEP=0
   STAX  D
   INR  A   ;-STEP=1
   STAX  D
   LDA  STEPTIME
   MOV  B,A   ;WAIT 8 MS FOR NEXT STEP
   JMP   DELAY   ;DELAY EXECUTES A RETURN

DONESTEP MVI  B,STEPSETTLE
   CALL DELAY   ;WAIT 8 MS FOR STEP SETTLE
   MOV  A,C
   CPI  2    ;IF (TRACK)<2 THEN SET TESTNEXT
   JC   SETTN
   LDA  TESTNEXT
   ORA  A
   MVI  A,0
   STA  TESTNEXT
   STC
   JNZ  SETDEN   ;IF TESTNEXT=55 TEST DENSITY
   RET

SETDEN:  ;TESTS DENSITY
   CALL TEST   ;TEST DENSITY
   MVI  A,4   ;IF Z IS SET (DOUBLE DENSITY)
   JZ  SKIP   ; THEN (DENBYTE)=4
   MVI  A,0   ; ELSE (DENBYTE)=0
```
SKIP STA DENBYTE
LXI H,DENMAP
PUSH PSW
LDA PRESDISK
MOV C,A
MVI B,Ø
DAD B
POP PSW ;SAVE FLAGS
MOV M,A
RET

SELDISK: ;SELECTS DRIVE POINTED TO BY C REG
;LOADS HEAD OF SELECTED DRIVE
LXI H,MASKTABLE
MVI B,Ø
DAD B ;C CONTAINS DRIVE NUMBER
MOV A, M ;MASKTABLE CONTAINS Ø FOR
SELDISK1 STA WRCONT ;SELECTED DRIVE, 1'S ELSEWHERE
STA TWOSIDE
STA CONTROLBYTE
LXI H,TRACKTAB
LDA PRESDISK
MOV E,A
MOV D,B
DAD D
LDA TRACK
MOV M,A ;(TRACKTAB(PRESDISK))=(TRACK)
MOV A,C
STA PRESDISK ;(PRESDISK)=(C)
STA DISK ;(DISK)=(C)
LXI H,TRACKTAB
DAD B
MOV A, M
STA TRACK ;(TRACK)=(TRACKTAB(C))
LXI H,LOGINTAB
DAD B
MOV A, M
ORA A ;HAS DRIVE BEEN LOGGED IN?
JNZ INOK ;NO. MARK AS LOGGED IN
MVI A,55H
MOV M,A
CALL HOME ;AND HOME THE HEAD
INOK CALL DISKREADY ;LOAD HEAD
MVI B,HEADSETTLE
CALL DELAY ;WAIT FOR HEAD SETTLING
LDA TRACK
CPI Ø2
JNC SETDEN

SETTN MVI A,55H ;ON TRACKS Ø AND 1, WE WANT
STA TESTNEXT ;TO TEST DENSITY OF NEXT TRACK
JMP SETDEN ;TEST DENSITY OF THIS TRACK
HEADLOAD LDAX D ;ASSUMES (D)=RDSTAT
ANI 20H ;HEAD ALREADY LOADED?
LDA RDMARK ;RESET HEAD LOAD COUNTER
MVI B,HEADDSETTLE
CNZ DELAY ;IF HEAD WASN'T LOADED
RET

DELAY: ;DELAYS (B) MILLISECONDS
PUSH H ;SAVE HL
DELAY2 LDA CONTROLBYTE
ANI 4 ;IF SINGLE DENSITY,
MVI L,31 ;31 BYTES * 32 USEC = 1 MS
JNZ DELAY1
MVI L,63 ;IN DD, 63 BYTES * 16 USEC = 1 MS
DELAY1 LDA RDDATA
DCR L
JNZ DELAY1
DCR B
JNZ DELAY2 ;END 1 MS LOOP
POP H ;RESTORE HL
RET

HOME CALL DISKREADY
LXI H,TRACK ;FOR STEPIN AND STEPOUT
ATHOME CALL STEPIN ;STEP TOWARD 76
LDAX D
ANI 02 ;UNTIL -TRK0 IS INACTIVE
JZ AT HOME
GOHOME CALL STEPOUT ;THEN STEP TOWARD 00
LDAX D
ANI 02 ;UNTIL -TRK0 IS ACTIVE
JNZ GOHOME
MVI A,10H
STA ABOVE43 ;(ABOVE43)=10H
STA TESTNEXT ;(TESTNEXT)=10H
XRA A
STA TRACK ;(TRACK)=00
JMP SETDEN ;TEST DENSITY
DISKREADY CALL LDA RDSTAT
MOV B,A
ANI 0A0H ;IF DRIVE READY AND HEAD LOADED
RZ
DISKREADY PUSH B
LXI D,WRCONT ;(D)=WRCONT=RDSTAT
CALL HEADLOAD ;LOAD HEAD
POP B
LDAX D
RLC
JC DISKREADY ;LOOP UNTIL DRIVE READY
RET
COLDBOOT LXI SP,STACK
XRA A
LXI B,BUFF
CBUFF STAX B ;ZERO OUT RAM BUFFER
INR C
JNZ CBUFF
MVI A,10 ;SET STEPTIME LONGER THAN IT
STA STEPTIME ;NEEDS TO BE TO BE SAFE, SINCE
;COLD BOOT LOADER RESETS IT
LXI B,0
CALL SETDMA ;SETDMA DOES NOT CHANGE C REG, SO
CALL SELDSK ;SELECT DRIVE A
MVI C,01 ;LOAD BOOTSTRAP LOADER
CALL SETSEC ; FROM TRACK 0 SECTOR 1
CALL READ
JNZ COLDBOOT ;ON READ FAILURE, TRY AGAIN
RST 0 ;EXECUTE BOOTSTRAP LOADER
;(SAVES 2 BYTES OVER JMP 0000)

MASKTABLE DB 0BFH,0DFH,0EFH,0F7H
Appendix C

CHANGING OPERATING SYSTEM CHARACTERISTICS

This section is divided into three subsections corresponding to three types of operating system alterations.

C-1 Changing the system size
C-2 Changing the IOBYTE and step time defaults
C-3 BIOS and BOOT Alteration

Section C-1 discusses use of the MOVCPM and SYSGEN transient commands to relocate the operating system to reside in a different part of memory. When shipped, the system is configured to use 62K of RAM, the maximum since the DOUBLER is located at F800H.

Section C-2 describes changing the IOBYTE, the byte of memory referred to by the system to set the logical to physical device assignments at cold boot. If you purchased Remex double-sided drives with the DOUBLER, refer to this section to change step time.

Section C-3 discusses altering the Micromation custom BIOS. MM2BIOS.ASM contains a driver for a Centronics 703/779 printer and all the software necessary to support the various configurations of disk drive types (single or double sided floppies with or without a hard disk) and serial I/O port assignments. A specific application may require additional driver routines or reconfigured I/O assignments.

C-1 CHANGING THE SYSTEM SIZE

Generally, the CP/M operating system should be located at the top of system memory since the O/S uses only that portion of memory below it. Circumstances may dictate a decrease in the amount of memory used, however. For instance, data base management systems typically set aside a portion of memory above the CP/M system area for reference and temporary data storage.

The MOVCPM and SYSGEN utilities (transient commands) are used to relocate the system to a lower portion of system memory. There are several different invocations of MOVCPM, each for a different purpose. Refer to the CP/M document, An Introduction to CP/M Features and Facilities, for a description of the options.

The MOVCPM utility contains the MM2BIOS.ASM and M2BOOT.ASM files in a somewhat altered form. This alteration allows for the size parameter to be specified by the user. All other values are held constant.

In the following example a 60K system size is specified. The re-
sultant system image is put on a formatted disk in drive B. In this and other examples the "@" symbol is used to indicate a carriage return. When it appears, press the RETURN key on your keyboard; do not press the @ key. All user entries are underscored.

A>MOVCPM 60 *@

CONSTRUCTING 60k CP/M - Micromation ver 2.x
READY FOR "SYSGEN" OR "SAVE 35 CPM60.COM"
A>SAVE 35 CPM60.COM@
A>SYSGEN@
SYSGEN VER 2.x
SOURCE DRIVE NAME (OR RETURN TO SKIP) @
DESTINATION DRIVE NAME (OR RETURN TO REBOOT) B
DESTINATION ON B, THEN TYPE RETURN @
FUNCTION COMPLETE
DESTINATION DRIVE NAME (OR RETURN TO REBOOT) @
A>

In all cases above, the 2.x indicates the current revision of the program.

Notice the MOVCPM transient command. In this mode, a system with a size of 60K is generated and left in the transient program area (TPA); the system originally loaded remains in control of the computer.

SAVE is invoked next. This writes the contents of memory, in this case the operating system, onto the disk. The file name "CPM60.COM" is given to the file. This step is not always necessary. It is included to demonstrate use of the SAVE command. Subsequently, CPM60.COM can be used with DDT to install a new BIOS and BOOT, if necessary (see section C-3 below).

The SYSGEN utility is invoked to write the system image from the TPA to the system tracks (0 - 1) on the designated disk. Notice that the example above puts the new system on the disk in drive B. "A" could have been specified as well. In this case, the write operation would replace the original system recording residing there with the system just created.

If several formatted disks require the new system, the operation can be repeated, without MOVCPM, by entering the destination drive name again instead of a RETURN in the last step.
C-2 CHANGING THE IOBYTE AND STEP TIME DEFAULTS

Changing the IOBYTE

A single byte at location 0003H sets the logical to physical device assignments in CP/M. It is set at cold boot (reset), but not warm boot (AC), by the BIOS section. The byte is broken down into four 2 bit sections corresponding to the four physical assignments. The bits are allocated as follows:

<table>
<thead>
<tr>
<th>MSB</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>bits</td>
<td>7 6</td>
</tr>
</tbody>
</table>

The following codes can be entered for each 2 bit location:

**LST** 00 - LIST is Teletype device (TTY:)
01 - LIST is terminal device (CRT:)
10 - LIST is a line printer device (LPT:)
11 - user defined list device (UL1:)

**PUN** 00 - PUNCH is Teletype device (TTY:)
01 - PUNCH is a high speed punch device (PUN:)
10 - user defined punch #1 (UP1:)
11 - user defined punch #2 (UP2:)

**RDR** 00 - READER is Teletype device (TTY:)
01 - READER is a high speed reader device (RDR:)
10 - user defined reader #1 (UR1:)
11 - user defined reader #2 (UR2:)

**CON** 00 - CONSOLE is Teletype device (TTY:)
01 - CONSOLE is a terminal (CRT:)
10 - batch mode: READER is CONSOLE input and LIST device is CONSOLE output (BAT:)
11 - user defined console device (UC1:)

In BIOS, the entire byte is entered in hexadecimal form. For instance, the value supplied in MM2BIOS.ASM is 81. In binary, this appears as 1000 0001 in the IOBYTE and means the LST is assigned to LPT:, PUN and RDR devices are coded for TTY: and the CON is coded for a terminal (CRT:). The replacement value used in the example below is 1. In binary, this is 0000 0001 which means LST: is assigned to TTY:, RDR: and PUN: are still assigned as TTY:, and CON: remains as CRT:.

The following assignments are made by MM2BIOS at cold boot. To have them displayed on your terminal, type STAT DEV:

- CON: is CRT:
- PUN: is TTY:
- RDR: is TTY:
- LST: is LPT:
NOTE: Although PUN: and RDR: are assigned to TTY:, MM2BIOS does not contain a driver routine. An attempt to output to a punch device or input from the reader device will not work.

The CRT: device is a typical terminal (keyboard with screen) attached to the serial port on the DOUBLER. The LPT: assignment for LST: references a Centronics 703/779 dot matrix printer driver included in the BIOS. Alternately TTY:, CRT: or UL1: may be assigned to LST:. TTY: provides output through the serial I/O port on the Micromation Multi I/O board. A driver routine is present in the BIOS to drive a serial printer. CRT: indicates that the terminal is the LST: device. UL1: references a custom driver written and installed by the user (see section C-3 below).

If your system has a serial printer attached to the Multi I/O Board serial port for the LST: device, you will need to type

```
STAT LST:=TTY:
```

after every cold boot (a warm boot does not affect the IOBYTE) to output to the printer. This may prove tedious in daily operation. If so, the procedure below describes how to change that portion of MOVCPM.COM that sets the IOBYTE. Subsequent use of MOVCPM renders TTY: as the LST: device. Of course, CRT: or UL1: can also be specified.

The procedure to change the IOBYTE uses the CP/M transients SUBMIT and XSUB. If you are not familiar with them, read through their descriptions in AN INTRODUCTION TO CP/M FEATURES AND FACILITIES and CP/M 2.x USER'S GUIDE FOR CP/M 1.4 OWNERS. To begin, place the back-up system disk in drive A, close the door and type

```
SUBMIT LIST xx
```

where xx is one of the following.

- 1 if you want TTY: as the default LST: assignment
- 41 if you want CRT: as the default LST: assignment
- 81 if you want LPT: as the default LST: assignment
- C1 if you want UL1: as the default LST: assignment

The following illustrates use of this procedure. In this example, the system size remains at the value shipped with the Z-PLUS (62K of memory) and the LST: assignment is changed from LPT: (the Centronics printer) to TTY: (the serial port on the Multi I/O board). As in other sections of this manual, user entries are underlined and a RETURN is indicated by the "@" character.
A>SUBMIT LIST 1@
A>XSUB
A>DDT
DDT VERS 2.x
-IMOVCPM.COM
-R
NEXT PC
2800 0100
-S20CA
20CA 81 1
20CB 32 .
-G0
(xsub active)
A>SAVE 39 MOVCPM.COM
A>

The only user entry in this example is the first. The remainder of the display is performed automatically under SUBMIT and XSUB. This SUBMIT file uses DDT to alter the location in MOVCPM that sets the IOBYTE. The value entered (1, 41, 81 or C1 for xx) replaces the default setting (81).

After the final prompt is displayed (after SAVE 39 MOVCPM.COM), press the RESET button on the front of the computer. This disengages XSUB. The CP/M sign-on and prompt are displayed. The following steps create a new system image using the new MOVCPM. After the RESET, perform the following.

A>MOVCPM 62 *@ Reminder: if you wish to generate a different size, replace 62.
CONSTRUCTING 62k CP/M - Micromation ver 2.x
READY FOR "SYSGEN" OR "SAVE 35 CPM62.COM"
A>

The new system is now in memory. The following transfers it to the disk in A. If you wish to preserve your current system, put a formatted disk in B and specify it as the destination disk.
To see (and ensure) that the change has occurred, press the RESET button again (if you wrote the new system to the disk in B, exchange the disks before resetting) and type

STAT DEV:@

The procedure described above renders the following response.

CON: is CRT:
RDR: is TTY:
PUN: is TTY:
LST: is TTY:

Changing the Step Time

IMPORTANT: This section has significance to those who have purchased Remex double sided drives with the DOUBLER. If your model has the standard Shugart drives, the BIOS portion of the O/S contains the appropriate step time setting. Do NOT execute this procedure if you have Shugart drives.

The step time value included in BIOS is used by the floppy disk drives to establish the rate at which the drive head moves from track to track. The appropriate rate for the different disk drives (from various manufacturers) is determined by noting the system performance with different settings; the good old trial and error method. Micromation has already done the leg-work and we've determined that the best setting for Shugart drives is 8 milliseconds and for Remex drives is 4 milliseconds. Since most of our systems are shipped with Shugart drives, MM2BIOS sets the step time at 8 ms. Remex drives will work with this setting, but system performance is enhanced by changing this value to 4 ms.

The STEP.SUB file is provided to change the step time value in MOVCPM for Remex drives much as the LIST.SUB file changes the IOBYTE default. In fact, execution of STEP.SUB is very similar. Again, STEP.SUB is SUBMITTED, and XSUB and DDT are called up. To invoke STEP.SUB, enter

SUBMIT STEP 4

when the A> prompt is displayed and terminate the command with a RETURN. As in the execution of LIST.SUB, do not make any additional entries until the final A> is displayed. It appears after A>SAVE 39 MOVCPM.COM has been displayed.
When the final A> prompt appears, hit the RESET button on the cabinet to disengage XSUB. Subsequently, perform MOVCPM exactly as illustrated above to generate a system image with the new step time installed.

C-3 BIOS AND BOOT ALTERATION

You will need to change and install MM2BIOS.ASM (the floppy disk only BIOS) if

- you need to insert a special printer driver (Note: several printer manufacturers use the Centronics conventions for data output. Check with your dealer to see if your printer falls into this category.)

- you do not have the Micromation Multi I/O Board. MM2BIOS is set-up to use the parallel and serial ports on this board for output to the Centronics printer and output to a serial interface printer. Code is present in MM2BIOS supporting serial output through a number of I/O boards from other manufacturers. This section describes MM2BIOS alteration to support these boards.

NOTE: If MM2BIOS is changed and inserted into the operating system as described in this section, the MOVCPM utility will not contain the change. Subsequent use of MOVCPM will render a system with the features of the O/S originally shipped with the unit. Consequently, plan ahead; determine the system size and IOBYTE default before installing the new system.

If you change the MSIZE equate in MM2BIOS.ASM, a corresponding change must be made in M2BOOT.ASM. There are two ways to do this:

1) Perform a MOVCPM as illustrated in Step 1 in NEW SYSTEM GENERATION below. The system size generated must match the value entered as MSIZE. In this case, there's no need to alter the MSIZE value in M2BOOT.

2) Change the MSIZE value in M2BOOT.ASM to agree with the value entered in MM2BIOS.ASM. Both must be assembled and inserted into the system as shown in NEW SYSTEM GENERATION. There's no need to perform Step 1 in this case; skip to Step 2.

Of course there are other changes that can be implemented. These should be left to programmers experienced in CP/M and BIOS alteration though. In fact, all changes to the operating system should be left to experienced programmers. This section is for reference to identify and define the significant labels. In addition, installation of the altered BIOS and BOOT with DDT is described.
Regardless of your hardware components, MM2BIOS.ASM and M2BOOT.ASM are the two programs to alter if a change is required. CBIOS.LIB, BIOS.LIB, and BOOT.LIB are examples of the form of these programs and should not be used as the source for changes. They are for reference only. Most alterations will affect MM2BIOS only.

Before demonstrating creation of a new system with altered BIOS and BOOT, the features and options of MM2BIOS are presented. Once the files have been edited and assembled, return to NEW SYSTEM GENERATION below to install them.

**MM2BIOS.ASM:** This program is supplied with systems that contain the Multi I/O Board. Four serial ports are available (though only one is installed). MM2BIOS assigns this port to the TTY: device. A driver routine for a Centronics 703/779 dot matrix printer is included to use a couple of the parallel ports. Flags are set to assemble only those features present in a particular system. The listing below is excerpted from MM2BIOS.ASM and shows the conditional flags.

```assembly
MSIZE EQU 48 ;SIZE OF OPERATING SYSTEM IN KILOBYTES
            ;(CURRENTLY 48K). THIS NUMBER MUST BE
            ;CHANGED FOR LARGER SYSTEMS.
NDRIVES EQU 4 ;NUMBER OF DISK DRIVES SUPPORTED BY
            ;THIS CBIOS

*=======================================================================
*I/O BYTE FOR LIST DEVICE IS IMPLEMENTED AS FOLLOWS:
*   "TTY" = MULTLIST (MM MULTI I/O BOARD SERIAL PORT#1)
*   "CRT" = CONOUT (MM DOUBLER SERIAL PORT)
*   "LPT" = CENTLIST (CENTRONICS 703/779 TYPE LIST DEVICE)
*   "UL1" = OPTIONAL DRIVER TO BE SELECTED BY USER (SEE BELOW)
*=======================================================================

===>UPDATED: 2-12-80

;LIST DEVICE EQUATES: (THESE COULD BE SET FALSE TO SAVE BIOS SPACE IF
; USER LIST DRIVER IS TOO LARGE TO FIT OTHERWISE)
MULTLIST EQU TRUE
CENTLIST EQU TRUE
```

53
;LIST DEVICE OPTIONS: SET ONLY ONE FLAG TRUE FOR DESIRED DRIVER AS "UL1"
;
NONE EQU TRUE ;NO "UL1" FUNCTION DESIRED
GODBIO EQU FALSE ;GODBOUT I/O BOARD AS LST:
SSMIO EQU FALSE ;SOLID STATE MUSIC 2S+P AS LST:
DPIO EQU FALSE ;DELTA PRODUCTS CPU BOARD AS LST:
USERLST EQU FALSE ;SET FLAG TO INSERT USER DEFINED LST:

The labels have the following meanings:

MSIZE: The amount of memory to be used by CP/M is set by this equate. Note that a change in MSIZE here necessitates a corresponding change in the MSIZE equate in MPBOOT.ASM (use one of the two methods described above). If only the system size is to be changed in the operating system, use MOVCPM instead. Notice that the value in MM2BIOS is 48. If you reset any flag, be sure to set this one to your current system size (62 in the standard configuration).

NDRIVES: The number of disk drives supported by the BIOS is indicated here. There is no need to change this equate even if your system has only two drives.

MULTLIST is assigned as the TTY: device and is set to use the Multi I/O board serial port (port 1) for output. It is set TRUE. If a user list device (UL1:) is to be incorporated, this flag can be set FALSE. Subsequent assembly will not include the MULTLIST related code. Thus, room can be made if the new code requires it.

CENTLIST is assigned as the LPT: device and is set to use the I/O board parallel ports for output. Subsequent code drives a Centronics printer. It, too, is set TRUE and can be set FALSE if the new code necessitates more room to fit in BIOS.

LIST DEVICE OPTIONS: One of the five labels shown can be assigned as the UL1: device. NONE is set TRUE since none apply. However, if you have one of these boards from other manufacturers or have a special printer driver you wish to incorporate, set the appropriate label TRUE. Note that only one can be TRUE.

If a driver is to be written (i.e., USERLST is to be used as UL1:), the code to assign the ports, to initialize the device, and to output data must be inserted in the appropriate places. The locations within MM2BIOS are identified by IF USERLST. (There are several, each for a separate portion of the code.) Insert the appropriate code after this entry and before the ENDIF statement.
If it is necessary to change the port assignments or install an additional USART on the Multi I/O Board, refer to that manual for a description of the changes necessary to the BIOS. Excerpts from MM2BIOS.ASM and program examples are provided.

To reiterate, the flags described above determine which options to implement. There is little need to alter other parts of MM2BIOS.ASM unless a special printer driver is to be inserted. Also, if you change the MSIZE equate in MM2BIOS, remember to change the MSIZE equate in M2BOOT.ASM.

Once any changes have been made, assemble MM2BIOS.ASM (and M2BOOT.ASM if necessary). Subsequently, proceed with the next part of this section to create a new system.

**M2BOOT.ASM ALTERATION**

M2BOOT.ASM will need alteration if you purchased REMEX doublesided floppy disk drives in your Z-PLUS system with Hard Disk. In Hard Disk systems, the STEP utility provided on the distribution diskette must NOT be used. Also, some users may want to change the MSIZE value in M2BOOT for convenience sake if the MSIZE in M(H)2BIOS was changed.

To review, changing M2BOOT isn't always necessary. In floppy disk only Z-PLUS systems, the values for MSIZE and STEPTIME can be changed with MOVCPM and SUBMIT STEP, respectively. If you choose either of these options be sure to perform the operation before installing the new BIOS. Subsequently, disregard the two commands for installing M2BOOT.HEX in Step 2 below.

The MSIZE equate appears in the beginning of the M2BOOT.ASM file. As distributed, a value of 48 is present. Change this value with your editor to agree with the value in MM2BIOS.ASM. The MSIZE equate in both MUST be the same.

To change the STEPTIME default for REMEX drives, look in the INIT section of M2BOOT.ASM for the following code.

```assembly
MVI    A,8
STA    STEPTIME ;STEPTIME FOR SHUGART DRIVES
```

Change the "8" in the first instruction to "4". This is the recommended steptime for REMEX drives.

These are the only changes that should be made to M2BOOT.ASM. After you are finished editing the file, assemble it and insert the new M2BOOT.HEX as described in Step 2 below.

If your intention is to change the STEPTIME only, you will need to change MSIZE as well. This is because the MSIZE equate is set for a 48K system which probable is not the amount of memory you have your BIOS set-up to use.

55
NEW SYSTEM GENERATION

After the programs have been assembled, MM2BIOS.HEX (and M2BOOT.HEX) must be inserted into the system. If the system size is 62K, you can use CPM62.COM as the source file when DDT is invoked. (Skip Step 1 below.) For another system size, proceed with Step 1.

In the examples that follow, "@" indicates a carriage return should be entered (press the RETURN key). Do not enter the @ character. All user entries are underscored.

To start, put a system diskette with the following files on it in drive A and a formatted (either single or double density) diskette in B.

M(H)2BIOS.HEX (assembled MM2BIOS.ASM or MH2BIOS.ASM)
MPBOOT.HEX (assembled MPBOOT.ASM, if necessary)
MOVCPM.COM
SYSGEN.COM
DDT.COM
CPM62.COM (unless a system with a different is created)

If a system size other than 62k is needed, proceed with the next step. Otherwise, skip to the Step 2.

STEP 1

A>MOVCPM xx *@
   where xx indicates the new system size

CONSTRUCTING xxK CP/M VERS 2.x
READY FOR "SYSGEN" OR
"SAVE 35 CPMxx.COM"

A>SAVE 35 CPMxx.COM

The MOVCPM program read the system image from the diskette, changed the values that specify the system size, and loaded it into memory, specifically the transient program area (TPA). SAVE was used to record the file CPMxx.COM from the contents of the TPA onto the disk in drive A:

To install the new BIOS and BOOT, DDT is used. In this example, the CPM62.COM file is used. If a different system size was created, substitute the CPMxx.COM SAVEd above for CPM62.COM.
**STEP 2**

A>DDT CPM62.COM

DDT VERS 2.x

NEXT PC

2400 0100

-L1F80

1F80 JMP F2C9
1F83 JMP F2FE
1F86 JMP F496

-H1F80 F200

1180 2D80

-IMM2BIOS.HEX

-R2D80

NEXT PC

2400 0000

-IM2BOOT.HEX

-R900

NEXT PC

2400 0000

-A>SAVE 35 CPM62X.COM

List the contents of the 10 locations starting at 1F80. This is the jump table for BIOS. Use F200 to determine offset in next step (see description below for reason).

Determine the offset.

2D80 is the offset.

Get MM2BIOS.

Insert it at 2D80.

Get new M2BOOT.

M2BOOT always goes at 900.

Exit to the system.

Save the new CP/M system.

* Installation of M2BOOT.HEX is only necessary if some value (e.g., MSIZE or STEPTIME) was changed. Recall that a change is system size can be performed by performing Step 1 above before installing the new BIOS with DDT. If no change was made to M2BOOT, skip this command and the next.

The system was SAVED as CPM62X.COM as a precaution in case the new BIOS doesn't work. Thus, CPM62.COM is available as a source if the operation needs to be done again.

Unlike M2BOOT, which always goes at 900H in the system, the location of MM2BIOS varies with the system size. Where to put it is determined by finding the difference between the its present location in memory and the location it resides at when in use. Memory location 1F80H is always the location of the jump table to the BIOS routines in CP/M. To determine where to put the new BIOS (MM2BIOS), list this portion of the system after executing MOVCPM to see where it has been placed. Since BIOS starts at an xx-hundred location, drop the last two digits, rounding down to the hundred hex number. The following table illustrates this procedure.

<table>
<thead>
<tr>
<th>Value Shown</th>
<th>Value Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>F2C9</td>
<td>F200</td>
</tr>
<tr>
<td>C2C9</td>
<td>C200</td>
</tr>
<tr>
<td>EAC9</td>
<td>EA00</td>
</tr>
</tbody>
</table>

57
The "Value Used" is then used to calculate the offset necessary. The offset is the negative difference between the location where BIOS resides during execution (in high memory) versus its present location under DDT in the TPA. Use the H command entering first the jump table address (1F80), a space, and then the location pointed to by the jump table. The numbers that result are the sum of the two and the difference. Disregard the sum and use the difference (2D80 in the example above) as the location for MM2BIOS in the DDT R (read) command.

Step 3 writes the new system on a disk. Again as a precaution, write the new system to a different formatted disk than the one currently in use. Since it is not known whether the new one works, it's not a good idea to erase the current system yet. To start, keep the system disk in drive A and put a formatted disk (either single or double density, it doesn't matter) in drive B. If you did not use CPM62.COM in the example above, substitute your CPMxx.COM file for CPM62X.COM in the examples below.

**STEP 3**

```
A>DDT CPM62X.COM@
DDT VERS 2.x
NEXT PC
2400 0100
-C
A>SYSGEN@
SYSGEN VERS 2.x
SOURCE DRIVE NAME (OR RETURN TO SKIP)@ DESTINATION DRIVE NAME (OR RETURN TO REBOOT)B
DESTINATION ON B, THEN TYPE RETURN@ FUNCTION COMPLETE
DESTINATION DRIVE NAME (OR RETURN TO REBOOT)@
A>
```

Notice that a RETURN was entered in response to the program question SOURCE DRIVE NAME. DDT had already transferred the system image (in this case CPM62X.COM) from disk into memory.

To test the new diskette, exchange the disks in A and B and hit the RESET button. The CP/M sign-on message and prompt should appear.

**STEP 4**

Assuming the new system worked, type

```
B:PIP A:=B:*.*
```

to transfer all the files from the disk in B (the former system disk) to the new system disk in drive A. "B:" had to precede PIP since the disk in A: is blank except for the system.
If the new system did not work, try the procedure for creating a new system again. (Perhaps you made a mistake the first time through.) If it doesn't work after the second try, the problem is most likely in the BIOS you wrote or patched.

Once MM2BIOS and M2BOOT have been patched and incorporated into your operating system, use of MOVCPM to change the system size will install the old system rather than the new one. MOVCPM contains the original MM2BIOS and M2BOOT with alterations to relocate the system size according to the value entered. All the other flags described above remain the same. To change the size of the new system, you will need to edit MM2BIOS.ASM and M2BOOT.ASM again changing the MSIZE equate, re-assemble the files, perform MOVCPM (specifying the new system size again to render the appropriate) locations in the jump table described above), and execute DDT to insert the altered files.
; PROM ROUTINES FOR MICROMATION DOUBLER, VERSION C.2

; THE C.1 VERSION HAS NOPs IN SYNC ROUTINE TO ALLOW MORE FREQUENT REFRESH
; OF DYNAMIC RAMS
; IT ALSO SETS UP THE SIDE BIT EARLIER TO MEET SETUP TIME FOR Y-E DATA DRIVES

; THIS VERSION HAS THE FOLLOWING CHANGES FROM C.1:
; HAS FIX FOR C.1 BUG IN SETTING UP DENSITY
; DISABLES INTERRUPTS AFTER FINDING CORRECT SECTOR
; HAS SLOWER STEP AND SETTLE TIMES

; FEB 11, 1980

FB00 BASE ORG OFB00H
FC00 = BUFF EQU BASE+400H ; SCRATCHPAD RAM

; START OF HARDWARE PORT DEFINITIONS

; FE00 = WRCONT EQU BASE+600H
FE01 = WRCLK EQU WRCONT+1
FE02 = WRUART EQU WRCONT+2
FE04 = WRIKRCRC EQU WRCONT+4
FE05 = WRIKRC EQU WRCONT+5
FE06 = WRDATA EQU WRCONT+6
FE07 = WRCRC EQU WRCONT+7
FE00 = RDSTAT EQU WRCONT
FE02 = RDUART EQU WRCONT+2
FE04 = RDRIKRCRC EQU WRCONT+4
FE05 = RDRIKRC EQU WRCONT+5
FE06 = RDRTDATA EQU WRCONT+6
FE07 = SYNCPORT EQU WRCONT+7

; START OF RAM VARIABLE DEFINITIONS

; FC00 = ERRORBYTE EQU BUFF ; NO. OF ERRORS DURING RETRIES
FC01 = DLNBYTE EQU BUFF+1 ; 0 FOR SINGLE DENSITY
; 14 FOR DOUBLE DENSITY
FC02 = READWRITE EQU BUFF+2 ; 0 FOR READ
; 10H FOR WRITE
FC03 = CONTROLBYTE EQU BUFF+3 ; RAM IMAGE OF RDSTAT OR WRCONT
FC04 = TRACK EQU BUFF+4
FC05 = PRESDISK EQU BUFF+5
FC06 = LOGINTAB EQU BUFF+6 ; FOR EACH DRIVE
; 0 IF DRIVE HAS NOT BEEN LOGGED IN
\texttt{FILE: \texttt{C2PROM PRN PAGE 002}}

\begin{verbatim}
FCOA = SECTOR   EQU    BUFF+0AH
FCOB = DMA     EQU    BUFF+0BH ;DMA ADDRESS
FCOD = DISK    EQU    BUFF+0DH
FCOE = TESTNEXT EQU    BUFF+0EH ;55H IF WANT TO TEST DENSITY
                    ;OF NEXT TRACK
FCOF = TWOSIDE  EQU    BUFF+0FH
FC10 = STEPTIME EQU    BUFF+10H
FC11 = ABOVE43  EQU    BUFF+11H ;10H IF (TRACK)<44D
                    ;50H OTHERWISE
FC12 = TRACKTAB EQU    BUFF+12H
FC16 = DENMAP   EQU    BUFF+16H ;SAME CONVENTION AS DENBYTE
FC20 = TRY1     EQU    BUFF+20H
FC21 = RETRYCOUNT EQU    BUFF+21H
FC22 = CURRDRIVE EQU    BUFF+22H
FC23 = TESTMAX  EQU    BUFF+23H ;NO. RETRIES FOR DENSITY TEST
000F = STEPSETTLE EQU    15
002B = HEADSETTLE EQU    40
FC40 = STACK EQU    BUFF+64H

;BEGIN WITH JUMP TABLE

F800 C303FB  JMP  COLDBOOT
F803 C307FB  JMP  HOME
F806 C311FB  JMP  SELDSK
F809 C3A6FB  JMP  SETTRK
F80C C3B0FB  JMP  SETSEC
F80F C3B4FB  JMP  SETDMA
FB12 C328FB  JMP  READ
FB15 C322FB  JMP  WRITE
FB18 C360FB  JMP  SKEW
FB1B C363FB  JMP  SETDEN

WRITEPROTECT:

FB1E CDBEFB  CALL  DISKREADY1 ;LOADS HEAD
              ;WAITS TILL DISK READY
              ;RETURNS (RDSTAT) IN B

FB21 7B  MOV  A, B
FB22 E604 ANI 04 ;WRITEPRT BIT FROM DRIVE
FB24 C0  RNZ
FB25 3A05FE LDA  RDARK  ;RESETS HEAD LOAD COUNTER
FB28 C9  RET

READ: ;ENTRY POINT FOR READ ROUTINE

FB29 AF  XRA  A ;(READWRITE)= 00 FOR READ
FB2A C322FB JMP  60

WRITE: ;ENTRY POINT FOR WRITE ROUTINE

FB2D 3E10  MVI  A, 10H ;(READWRITE)=10H FOR WRITE
\end{verbatim}
FILE: C2PROM  PRN  PAGE 003

FB2F 3202FC  6D;  STA  READWRITE  ;(L)=(DENBYTE)
FB32 2A01FC  LHLD  DENBYTE  
FB35 3A03FC  LDA  CONTROLBYTE  
FB3B 2F  CMA  
FB39 E6FB  ANI  OFBH  ;MASK OUT BIT 2 (SD/-DD = 0)
FB3B B5  ORA  L  
FB3C 2F  CMA  
FB3D 3200FE  STA  MRCONT  
FB40 CD8EFB  CALL  DISKREADY1  
FB43 3A03FC  LDA  SECTOR  
FB46 4F  "OF C,A  ;(C)=(SECTOR)
FB47 3A03FC  LDA  TRACK  
FB4A 47  MOV  B,A  ;(B)=(TRACK)
FB4B AF  XRA  A  
FB4C 3200FC  STA  ERRORBYTE  ;(ERRORBYTE)= 0
FB4F 7D  MOV  A,L  
FB50 B7  ORA  A  ;TEST FOR SINGLE DENSITY
FB51 CA70F9  JZ  SD  

;  DOUBLE DENSITY READ OR WRITE

READDD:

FB54 CD5FF9  BLOOP: CALL  SYNC  ;SYNC ON HEADER
          ;FOUND HEADER
FB57 360A  MVI  M,OAH  ;FIND OA CLOCK FOR ID MARK
FB59 1A  LDAX  D  ;SYNC WITH -EOM
FB5A 3A04FE  LDA  RDMARKRC
FB5D FEA1  CPI  OA1H  ;DATA FOR ID MARK
FB5F C254FB  JNZ  BLOOP  ;FOUND ID ADDRESS MARK
          ;
FB62 1A  LDAX  D  ;BYTE AFTER ID MARK SHOULD BE FE
FB63 FEFE  CPI  OFEH
FB65 C254FB  JNZ  BLOOP  ;FOUND FE BYTE
          ;
FB66 1A  LDAX  D  ;TRACK BYTE FROM DISK
FB69 BB  CMP  B  ;(B)=(TRACK)
FB6A C2F2FB  JC  ERROR1  ;TRACK ERROR
FB6D 1A  LDAX  D  ;SECTOR BYTE FROM DISK
FB6E B9  CMP  C  ;(C)=(SECTOR)
FB6F C254FB  JNZ  BLOOP  ;WRONG SECTOR. TRY AGAIN
          ;
FB72 1A  LDAX  D  ;DISABLE INTERRUPTS BEFORE CHECKING ID CRC
FB73 F3  DI  
FB74 1A  LDAX  D  
FB75 1A  LDAX  D  ;READ 1 BYTE PAST ID CRC
FB76 3A00FE  LDA  RSTAT  
FB79 1F  RAR  
FB7A 1A  LDAX  D  ;CHECK ID CRC
FB7B 0ADDFB  JC  ERROR  ;ID CRC ERROR
FB7E 1A  LDAX  D  
FB7F 3A11FC  LDA  ABOVE43
FILE: C2PRM  PRN  PAGE 004

FB02 47 MOV  B,A
FB03 1A LDAI  D
FB04 70 MOV  M,B ;(WRCLK)=(ABOVE43)
FB05 1A LDAI  D ;NOW 5 BYTES INTO GAP
FB06 0609 MVI  B,9
FB08 1A GLOOP: LDAI  D
FB09 05 DCR  B
FB0A C288F8 JNZ  GLOOP

FB0D 1A LDAI  D ;NOW 15 BYTES INTO GAP
FB0E 3A02FC LDA  READWRITE
FB0F 87 ORA  A ;CHECK FOR WRITE
FB10 1A LDAI  D ;16 BYTES INTO GAP
FB11 C203F9 JNZ  WRITEDD

FB16 1A LDAI  D
FB17 1A LDAI  D
FB18 36FF MVI  M,OFFH
FB19 1A LDAI  D
FB1A 1A LDAI  D
FB1C 1A LDAI  D ;21 BYTES INTO GAP
FB1D 13 INX  D ;(D)=SYNCPORT
FB1E 1A LDAI  D ;SYNC ON FF CLOCK PATTERN
FB1F 1B DCX  D ;(D)=RDRTA
FB20 360A MVI  M,OAH ;(WRCLK)=0A
FB22 C2DDFB JNZ  ERROR ;FINDING DATA MARK

FB25 1A LDAI  D ;DOUBLE DENSITY READ
FB26 1A LDAI  D
FB27 360F MVI  M,OFFH
FB28 1A LDAI  D
FB29 1A LDAI  D
FB2B 1A LDAI  D ;21 BYTES INTO GAP
FB2C 13 INX  D ;(D)=SYNCPORT
FB2D 1A LDAI  D ;SYNC ON FF CLOCK PATTERN
FB2E 1B DCX  D ;(D)=RDRTA
FB2F 360A MVI  M,OAH ;(WRCLK)=0A
FB32 2A08FC LHLDD  DMA
FB33 1A LDAI  D ;CLOCK PATTERN FOR DATA MARK
FB34 1A LDAI  D ;SYNC WITH -EDT!
FB35 3A04FE LDA  RDRTKRC
FB36 A1 CPI  OAH
FB37 C2D0F8 JNZ  ERRORR ;MISSING DATA MARK
FB38 1A RIFER: LDAI  D

FB40 77 MOY  M,A
FB41 23 INX  H
FB42 42 MOV  B,D
FB43 1A LDAI  D
FB45 77 MOV  M,A
FB46 23 INX  H
FB47 4B MOV  C,E
FB48 0A LDAI  B
FB49 77 MOV  M,A
FB50 23 INX  H
FB51 1EE1 MVI  E,OEH
FB52 0A LDAI  B
FB53 77 MOV  M,A ;4 BYTES OF DATA
FB54 23 INX  H
FB55 0A LDAI  B
FB56 77 RLOOP: MOV  M,A
FB57 0A LDAI  B
FB58 1C INR  E
FILE: C2PROM PRN PAGE 005

FBC2 23 INX H
FBC3 77 MOV M,A
FBC4 0A LDAX B
FBC5 23 INX H
FBC6 77 MOV M,A
FBC7 0A LDAX B
FBC8 23 INX H
FBC9 77 MOV M,A
FBCA 23 INX H
FBCC 0A LDAX B
F8CC C2BFF8 JNZ RLOOP ;HAVE TRANSFERRED 128 BYTES
;AND HAVE READ 129TH BYTE
F8CF 0A LDAX B
FBDO 0A LDAX B ;READ 1 BYTE PAST CRC
FB01 3A00FE LDA RDSTAT
FB04 1F RAR ;CHECK DATA CRC
FB05 DADD08 JC ERROR ;DATA CRC ERROR
;SUCCESSFUL SECTOR READ
;
FB08 AF XRA A ;RETURN 00 IN ACCUUMATOR
FB09 3201FE STA WRCLK
FB0C C9 RET

ERROR:

;ARRIVE HERE ON ANY OF FOLLOWING CONDITIONS
;30H TRACK ERRORS
;ID CRC ERROR
;MISSING DATA MARK
;DATA CRC ERROR
F80D 3EEF MVI A,0EFH ;RETURN EFH IN ACC
F80F 07 ORA A ;(UNSUCCESSFUL READ)
FB0E 3201FE STA WRCLK
FB03 C9 RET

TERROR:

;ARRIVE HERE ON TRACK ERROR IN SINGLE DENSITY
F84C CDFBF8 CALL ERRORCOUNT ;INCREMENT ERRORBYTE
F847 C270F9 JNZ ALOOP ;TRY AGAIN IF LESS THAN 30H
F8EB 3EEF NO MVI A,0EFH ;30H TRACK ERRORS
F8EE 07 ORA A ;RETURN EFH IN ACC
FBED 37 STC ;(UNSUCCESSFUL DISK OPERATION)
FBEE 3201FE STA WRCLK
FBF1 C9 RET

TERROR:

;ARRIVE HERE ON TRACK ERROR IN DOUBLE DENSITY
F8F2 CDFBF8 CALL ERRORCOUNT ;INCREMENT ERRORBYTE
F8F5 2C54F8 JNZ BLOOP ;TRY AGAIN IF LESS THAN 30H
FILE: C2PROM PRN PAGE 006

FBFB C3EAFB JMP NO
FBFB 2100FC ERRORCOUNT LXI H,ERRORBYTE
FBFE 34 INR M ;INCREMENT ERRORBYTE
F8FF 7E MOV A,M
F900 FE30 CPI 30H
F902 C9 RET

WRITEDD:

;DOUBLE DENSITY WRITE
;ARRIVE HERE 16 BYTES AFTER ID FIELD
F903 3E4E MVI A,0EH
F905 12 STAX D ;WRITE 4 BYTES OF 4E
F906 12 STAX D
F907 12 STAX D
F908 12 STAX D
F909 AF XRA A
F90A 12 STAX D ;WRITE 6BYTES OF 00
F90B 12 STAX D
F90C 2A0BFC LHLDD DMA
F90F 12 STAX D
F910 12 STAX D
F911 0104FE LXI B,WRMRKRC
F914 12 STAX D
F915 12 STAX D
F916 3EA1 MVI A,OA1H
F918 02 STAX B ;WRITE DATA MARK (A1)
F919 0EE1 MVI C,OE1H

;START WRITING DATA TO DISK FROM MEMORY
F91B 7E WIFER: MOV A,M
F91C 12 WLOOP: STAX D
F91D 23 INX H
F91E 0C INR C
F91F 7E MOV A,M
F920 12 STAX D
F921 23 INX H
F922 7E MOV A,M
F923 12 STAX D
F924 23 INX H
F925 7E MOV A,M
F926 23 INX H
F927 12 STAX D
F928 7E MOV A,M
F929 C21CF9 JNZ WLOOP

;WHEN WE ARRIVE HERE WE'VE WRITTEN
;31*4=124BYTES TO DISK
F92C 12 STAX D
F92D 23 INX H
F92E 7E MOV A,M
F92F 12 STAX D
F930 23 INX H
FILE: C2PRO I RTN PAGE 007

F931 7E MOV A,M
F932 23 INX H
F933 12 STAX D
F934 7E MOV A,M
F935 12 STAX D ;128TH BYTE TO DISK
F936 3EFF MVI A,OFFH
F938 3207FE STA WRCRC ;WRITE 2 BYTES OF DATA CRC
F939 3207FE STA WRCRC
F93A 12 STAX D ;WRITE 3 BYTES OF FF
F93B 12 STAX D
F93C 12 STAX D
F941 AF XRA A ;RETURN 00 IN ACC
F942 3201FE STA WRCLK ;(SUCCESSFUL WRITE)
F943 C9 RET

;SINGLE DENSITY ROUTINES
;ENTRY POINT IS SD (BELOW)

WRITESD: ;ARRIVE HERE 6 BYTES PAST ID FIELD
F946 3EFF MVI A,OFFH
F948 12 STAX D ;WRITE 3 BYTES FF (BYTES 7,8,9)
F949 12 STAX D
F94A 12 STAX D
F94B AF XRA A
F94C 12 STAX D ;WRITE 6 BYTES 00 (BYTES 10-15)
F94D 12 STAX D
F94E 2A08FC LHLD DMA
F950 12 STAX D
F951 12 STAX D
F952 12 STAX D
F953 12 STAX D
F954 12 STAX D ;BYTE 15 OF GAP
F955 3EFF MVI A,OFFH ;WRITE DATA MARK FOR SINGLE DEN
F957 3204FE STA WMRKRCRC
F958 0EE1 MVI C,0EH
F959 C31BF9 JMP WXFER ;JUMP TO COMMON WRITE ROUTINE

SYNC:

;ROUTINE TO SYNC ON HEADER
F95F 2101FE LIX H,WRCLK
F962 3EFF MVI H,OFFH
F964 1107FE LII D,SYNCPORT
F967 1A CLOOP: LDAX D ;SYNC ON FF CLOCK IN HEADER
F968 B7 ORA A ;SHOULD HAVE 00 DATA
F969 00 NOP ;FOUND SYNC PATTERN
F96A 00 NOP
F96B 1B DCI D ;(D)=WRDATA=READDATA
F96C CB RI
F96D C35FF9 JMP SYNC

;SINGLE DENSITY ENTRY POINT

SD:

F970 C05FF9 ALODP: CALL SYNC
;FOUND HEADER

F973 36C7 MLOOP: MVI M,OC7H ;CLOCK PATTERN FOR ID MARK
F975 3A04FE LOOP: LDA RDMRKRC
F97B 37F7 ORA A
F979 CA75F9 JZ LLOOP
F97C FEFE CPI OFEH
F97E CABDF9 JZ MLOOP
F981 36FF MVI M,OFFH
F983 3A07FE LDA SYNCPORT
F986 B7 ORA A
F987 CA73F9 JZ MLOOP
F98A C370F9 JMP ALOOP

NLOOP:

F980 1A LDAX D ;TRACK BYTE FROM DISK
F98B 1B CMP B
F98F C2E4FB JNZ AERROR ;TRACK ERROR
F992 1A LDAX D ;SIDE BYTE FROM DISK (IGNORE)
F993 1A LDAX D ;SECTOR BYTE FROM DISK
F994 1B CMP C
F995 C270F9 JNZ ALOOP ;WRONG SECTOR. TRY AGAIN

F998 1F DI ;DISABLE INTERRUPTS BEFORE CHECKING ID CRC
F999 1A LDAX D ;CRC BYTE
F99B 1A LDAX D ;CRC BYTE
F99C 1A LDAX D ;GAP BYTE 1
F99D 3A00FE LDA ROSTAT ;CHECK ID CRC
F9A0 1F RAR
F9A1 1A LDAX D ;GAP BYTE 2
F9A2 1A LDAX D ;GAP BYTE 3
F9A3 DADDFF JC ERROR ;ID CRC ERROR

F9A6 1A LDAX D ;GAP BYTE 4
F9A7 3A11FC LDA ABOVE43
F9A8 7F MOV M,A
F9A8 1A LDAX D ;GAP BYTE 5
F9AC 3A02FC LDA READWRITE
F9AF 1B ORA A ;CHECK FOR WRITE
F9B0 1A LDAX D ;GAP BYTE 6
F9B1 C246F9 JNZ WRITESD

;SINGLE DENSITY READ

F9B4 1A LDAX D ;READ 6 BYTES OF GAP
F9B5 1A LDAX D
F9B6 1A LDAX D
F9B7 1A LDAX D
F9B8 1A LDAX D
F9B9 1A LDAX D
F9BA 36FF MVI M,OFFH ;(WRCLK)=FF
F9BC 0106FE LXI B,RDDATA
F9BF 1A LDAX D ;GAP BYTE 14
F9C0 13 INX D ;(D)=SYNCPORT
F9C1 1A LDAX D
F9C2 36C7 MVI M,OC7H ;CLOCK PATTERN FOR DATA MARK
F9C4 1E04 MVI E,04 ;(D)=RDMRKRC
F9C6 0A LDA X B ;GAP BYTE 16
F9C7 1A LDA X D ;READ DATA MARK
F9CB E6FC ANI OFCH
F9CA FEF8 CPI OF8H ;DATA PATTERN FOR DATA MARK
F9CC C2DDFB JNZ ERROR ;MISSING DATA MARK

;FOUND SINGLE DENSITY DATA MARK

F9CF 1EE0 MVI E,0EH ;32#4=128 BYTE TRANSFER
F9D1 0A LDA X B
F9D2 2A08FC LHL D DMA
F9D5 C38FFB JMP RLOOP ;JUMP TO MAIN READ ROUTINE

TEST:

;TESTS DENSITY OF DISKETTE IN LOGGED-IN DRIVE
;RETURNS 00 IN ACC IF DOUBLE DENSITY
;RETURNS OF IN ACC IF SINGLE DENSITY
;RETURNS OA IN ACC IF TEST FAILS

F9DB AF XRA A
F9D9 3223FC STA TESTMAX ;(TESTMAX)=0
F9DC AF TEST1: XRA A
F9DD 3200FC STA ERRORBYTE ;(ERRORBYTE)=0
F9E0 CDC5FB CALL DISKREADY ;LOAD HEAD
F9E3 0100FE LXI B,MRCONT
F9E6 3A03FC LDA CONTROLBYTE
F9E9 F680 DRI 80H ;SET CONTROLLER FOR SIDE 0
F9EB E6FB ANI OF8H ;TRY DOUBLE DENSITY
F9ED 02 STAX B

LOOP6: ;DOUBLE DENSITY TEST
F9EE 2101FE LXI H,WRCLK
F9F1 36FF MVI M,OFFH
F9F3 1107FE LXI D,SYNCPOR ;SYNCPOR FF CLOCK IN HEADER
F9F6 1A LOOP7: LDA X D ;READ DATA PATTERN
F9F7 2C INR L ;ABORT AFTER 256 TRIES
F9FB CA1AFA JZ RETRY
F9FB 07 DRA A ;DATA SHOULD BE 00
F9FC C2F6F9 JNZ LOOP7 ;FOUND HEADER
F9FF 1B DCX D ;(D)=READDATA
FA00 2E01 MVI L,01 ;(H)=WRCLK
FA02 360A MVI M,0AH
FA04 1A LDA X D ;SYNC WITH -EOM
FA05 3A04FF LDA RMRKCR ;LOOK FOR ID MARK
FA08 FEA1 CPI OA1H
FA0A C21AFA JNZ RETRY ;FOUND ID MARK
FA0D 1A LDA X D ;FE BYTE
FA0E 1A LDA X D ;TRACK BYTE
FA0F 1A LDA X D ;SECTOR BYTE
FA10 1A LDA X D ;CRC BYTE
FA11 1A LDA X D ;CRC BYTE
FA12 1A LDA X D ;GAP BYTE 1
FA13 0A LDA X B
FA14 IF RAR ;CHECK ID CRC
FA15 DA1AFA JC RETRY ;ID CRC OK
FA18 AF IRA A ;RETURN 00
FA19 C9 RET

FA1A CDFBF8 RETRY: CALL ERRORCOUNT
FA1D C2EEF9 JNZ LOOP6

;SINGLE DENSITY TEST
;ARRIVE HERE AFTER 30H TRIES AT DOUBLE DENSITY

FA20 AF SDTEST: IRA A
FA21 3200FC STA ERRORBYTE ;(ERRORBYTE)=0
FA24 3A03FC LDA CONTROLBYTE
FA27 F604 ORI 04H ;SET UP SIDE 0, SINGLE DENSITY
FA29 02 STAX B ;TO WRCONT

SDL0OP1:
FA2A 1E07 MVI E,07 ;(D)=SYNCPORT
FA2C 2101FE LII H,WRCLK
FA2F 36FF MVI M,OFFH ;SYNC ON FF CLOCK PATTERN

SDL0OP2:
FA31 1A LDAX D ;GET CORRESPONDING DATA
FA32 2C INR L ;ABORT AFTER 256 TRIES
FA33 C57FA JI RETRY1
FA36 B7 ORA A ;DATA SHOULD BE 00
FA37 C31FA JNZ SDL00P2 ;FOUND HEADER

FA3A 18 DCX D ;(D)=READDATA
FA3B 2E01 MVI L,01 ;(H)=WRCLK
FA3D 36C7 MVI M,OC7H ;LOOK FOR C7 CLOCK
FA3F 1A LDAX D ;SYNC WITH -EOW
FA40 3A04FE LDA RDMRKRC
FA43 FF0E CPI OFEH ;DATA FOR ID MARK
FA45 C257FA JNZ RETRY1 ;FOUND ID MARK

FA48 1A LDAX D ;TRACK BYTE
FA49 1A LDAX D ;SIDE
FA4A 1A LDAX D ;SECTOR
FA4B 1A LDAX D
FA4C 1A LDAX D ;CRC BYTE
FA4D 1A LDAX D ;CRC BYTE
FA4E 1A LDAX D
FA4F 0A LDAX B ;GET RDSTAT
FA50 1F RAR ;CHECK ID CRC
FA51 D57FA JC RETRY1 ;ID CRC OK

FA54 F6FF ORI OFFH ;RETURN FF
FA56 C9 RET

FA57 CDFBF8 RETRY1: CALL ERRORCOUNT
FA5A C22AFA JNZ SDL00P1

;FAILED BOTH DOUBLE AND SINGLE DENSITY
;TESTS 30H TIMES

FA5D 2123FC LII H,TESTMAX
SKEW:
; COMPUTES PHYSICAL SECTOR FROM LOGICAL SECTOR
; SKEW FACTOR IS 8
; INPUT AND OUTPUT ARE IN C REG
; OUTPUT = ((INPUT) MOD 52) * 8 - 7) MOD 52
; IF INPUT>52, SELECTS SIDE 1

FA69 210000 LXI H,0
FA6C E5 PUSH H
FA6D 3A03FC LDA CONTROLBYTE
FA70 E67F AND 7FH ; SIDE 1
FA72 5F MOV E,A
FA73 79 MOV A,C
FA74 D634 SUI 52
FA76 47 MOV B,A ; (B) = (C)-52
FA77 7B MOV A,E ; (A) = (CONTROLBYTE)^7F
FA78 F27EFA JMP SKIPY ; INPUT WAS LESS THAN 52
FA7B 41 MOV B,C
FA7E 320FFC SKIPY: STA TMSIDE
FA81 78 MOV A,B ; (B) = (INPUT) MOD 52
FA82 6B MOV L,B
FA83 C1 POP B
FA84 0C LOOP10: INR C
FA85 D60D SUI 13
FA87 F284FA JP LOOP10
FA8A 29 DAD H
FA8B 29 DAD H
FA8C 29 DAD H
FA8D 7C MOV A,H
FA8E B7 ORA A
FA8F 7D MOV A,L
FA90 C4A0FA CNI HIGHE
FA93 FE34 LOOP11: CPI 52
FA95 DA9DFA JC SKIP12
FA98 C6CC ADI 204
FA9A 393FA JMP LOOP11
FA9D B1 SKIP12: ADD C
FA9E 4F MOV C,A
FA9F C9 RET
FAA0 C630 HIGHE: ADI 4B
FAA2 C9 RET

FAA3 60 SETDNA: MOV H,B
FAA4 69 MOV L,C
FAA5 220FFC SHLD DMA ; STORE DMA ADDRESS
FAAB C9   RET

FAA9 79   SETSEC: MOV A,C
FAAA 320AFC STA SECTOR ;STORE SECTOR NUMBER
FAAD C9   RET

FAAE 79   MOV A,C
FAAF FE2C  CPI  44D ;IF (C)<44
FAAB 3E10  MVI A,10H ; THEN (ABOVE43)=10H
FAAB DABBFA JC SKIP3
FAAB 3E50  MVI A,50H ; ELSE (ABOVE43)=50H
FAAB 3211FC STA ABOVE43
FAAB CDC5FB CALL DISKREADY

STEPTRK: ;STEPS DRIVE TO TRACK (C)
FAAE 79   MOV A,C
FAAF F62C  CPI  44D ;IF (C)<44
FAAB 3E10  MVI A,10H ; THEN (ABOVE43)=10H
FAAB 3E50  MVI A,50H ; ELSE (ABOVE43)=50H
FADD 3E50  STA ABOVE43
FAAE 79   MOV A,C
FAAF FE2C  CPI  44D ;IF (C)<44
FAAB 3E10  MVI A,10H ; THEN (ABOVE43)=10H
FAAB 3E50  MVI A,50H ; ELSE (ABOVE43)=50H
FAAB CDC5FB CALL DISKREADY

STEPLOOP:
FAAB 2104FC LXI H,TRACK
FAAC 17E   MOV A,M ;GET (TRACK)
FAAC 2B9   CMP C ;DONE?
FAAC CAE4FA JZ DONESTEP
FAAC CDCCFA CALL STEPHEAD ;NO, STEP HEAD
FAAD C3BEFA JMP STEPLOOP ;REPEAT

STEPHEAD:
FAAE DADBFA JC STEPIN ;IF (TRACK)<(C) THEN STEP IN

STEPOUT:
FAAB 30A3FC LDA CONTROLBYTE ;ELSE STEP OUT
FAAB 3502  DCR M ;(TRACK)=(TRACK)-1
FAAB F402  ORI 02H ;DIR=OUT
FAAB C30EFA JMP DOSTEP

FAAB 30A3FC STEPIN: LDA CONTROLBYTE
FAAE 34B4  INR M ;(TRACK)=(TRACK)+1
FAAE 66FD  ANI OFDH ;DIR=IN

FAAD 12 DOSTEP: STAI D ;STORE DIRECTION IN WRCONT
FAAD 30  DCR A ;-STEP=0
FAAE 12  STAX D
FAAE 3C  INR A ;-STEP=1
FAAE 12  STAX D
FAAE 3A0EFC LDA STEPTIME
FAAB 47   MOV B,A ;WAIT 0 MS FOR NEXT STEP
FAAB 37DFFB JMP DELAY ;DELAY EXECUTES A RETURN

DONESTEP:
FAAE 060F  MVI B,STEPSETTLE
FAAE CD7DFB CALL DELAY ;WAIT 0 MS FOR STEP SETTLE
FAAE 79   MOV A,C
FAAE FE02  CPI  2 ;IF (TRACK)<2 THEN SET TESTNEXT
FAAE DA69FB JC SETTN
FAAE 3A0EFC LDA TESTNEXT
FAAE B7   ORA A
FAAE 3E00  MVI A,0
FAAB 320EFC STA TESTNEXT
FAAE 37   STC
FILE: C2PROM   PRN   PAGE 013

FAF C203FB  JNZ SETDEN ; IF TESTNEXT=55 TEST DENSITY
FB02 C9    RET

SETDEN:

; TESTS DENSITY
; UPDATES DENBYTE AND DENMAP

FB03 CDDBF9 CALL TEST ; TEST DENSITY
FB06 3E04 MVI A,4 ; IF Z IS SET (DOUBLE DENSITY)
FB08 CA0DFB JZ SKIP ; THEN (DENBYTE)=4
FB0B 3E00 MVI A,0 ; ELSE (DENBYTE)=0
FB0D 3201FC SKIP STA DENBYTE
FB10 2116FC LXI H,DENMAP
FB13 F5 PUSH PSM
FB14 3A05FC LDA PRESDISK
FB17 4F MOV C,A
FB18 0600 MVI B,0
FB1A 09 DAD B
FB1B F1 POP PSM ; SAVE FLAGS
FB1C 77 MOV M,A ; (DENMAP(PRESDISK))=(DENBYTE)
FB1D C9 RET

SELDSK:

; SELECTS DRIVE POINTED TO BY C REG
; LOADS HEAD OF SELECTED DRIVE

FB1E 21F9FB LXI H,MASTERTABLE
FB21 0600 MVI B,0
FB23 09 DAD B ; C CONTAINS DRIVE NUMBER
FB24 7E MOV A,M ; MASKTABLE CONTAINS 0 FOR

SELDSK:

FB25 3200FE STA WRCONT ; SELECTED DRIVE, 1'S ELSEWHERE
FB28 320FFC STA TWOSIDE
FB2B 3203FC STA CONTROLBYTE
FB2E 2112FC LXI H,TRACKTAB
FB31 3A05FC LDA PRESDISK
FB34 5F MOV E,A
FB35 50 MOV D,B
FB36 19 DAD D
FB37 3A04FC LDA TRACK
FB3A 77 MOV M,A ; (TRACKTAB(PRESDISK))=(TRACK)
FB3B 79 MOV A,C
FB3C 3205FC STA PRESDISK ; (PRESDISK)=(C)
FB3F 3200FC STA DISK ; (DISK)=(C)
FB42 2112FC LXI H,TRACKTAB
FB45 09 DAD B
FB46 7E MOV A,M
FB47 3204FC STA TRACK ; (TRACK)=(TRACKTAB(C))
FB4A 2106FC LXI H,LOGINTAB
FB4D 09 DAD B
FB4E 7E MOV A,M
FB4F B7 ORA A ; HAS DRIVE BEEN LOGGED IN?
FB50 C259FB JNZ INOK
FB53 3E55 MVI A,5SH ; NO. MARK AS LOGGED IN
FB55 77 MOV M,A
FILE: C2PROM PRN PAGE 014

FB56 C097FB CALL HOME ; AND HOME THE HEAD
FB59 CD0CFB INOK: CALL DISKREADY ;LOAD HEAD
FB5C 062B MVI B,HEADSETTLE
FB5E CD0FB CALL DELAY ;WAIT FOR HEAD SETTLING
FB61 3A04FC LDA TRACK
FB64 FE02 CPI 02
FB66 D203FB JNC SETDEN
FB69 3E55 SETTN: MVI A,55H ;ON TRACKS 0 AND 1, WE WANT
FB6B 320EFC STA TESTMEXT ;TO TEST DENSITY OF NEXT TRACK
FB6E C303FB JMP SETDEN ;TEST DENSITY OF THIS TRACK

HEADLOAD:

FB71 1A LDAX D ;ASSUMES (D)=RDSTAT
FB72 E620 ANI 20H ;HEAD ALREADY LOADED?
FB74 3A05FE LDA RDMARK ;RESET HEAD LOAD COUNTER
FB77 062B MVI B,HEADSETTLE
FB79 C47DFB CNZ DELAY ;IF HEAD WASN'T LOADED
FB7C C9 RET

DELAY:

;DELAYS (B) MILLISECONDS

FB7D E5 PUSH H ;SAVE HL
FB7E 3A03FC DELAY2: LDA CONTROLBYTE
FB81 E604 ANI 4 ;IF SINGLE DENSITY,
FB83 2E1F MVI L,31 ;31 BYTES * 32 USEC = 1 MS
FB85 C28AFB JNZ DELAY1
FB88 2E3F MVI L,63 ;IN DD, 63 BYTES * 16 USEC = 1 MS
FB8A 3A06FE DELAY1: LDA RDDATA
FB8D 2D DCR L
FB8E C28AFB JNZ DELAY1
FB91 05 DCR B ;END 1 MS LOOP
FB92 C27EFB JNZ DELAY2
FB95 E1 POP H ;RESTORE HL
FB96 C9 RET

FB97 CDCSFB HOME: CALL DISKREADY
FB9A 2104FC LII H,TRACK ;FOR STEPIN AND STEPOUT
FB9D CD0BF ATHOME: CALL STEPIN ;STEP TOWARD 76
FB9E 1A LDAX D
FB9E A602 ANI 02 ; UNTIL -TRKO IS INACTIVE
FB9E C99FB JZ ATIDIE
FB9E CDCFFA GOHOME: CALL STEPOUT ;THEN STEP TOWARD 00
FBAA 1A LDAX D
FBA3 0602 ANI 02 ; UNTIL -TRKO IS ACTIVE
FBAC C2A6FB JNZ GOHOME
FBAF 3E10 MVI A,10H
FBBA 3211FC STA ABOVE43 ;(ABOVE43)=10H
FBBD 3204FC STA TESTNEXT ;(TESTNEXT)=10H
FBB7 AF XRA A
FBB8 3204FC STA TRACK ;(TRACK)=00
FBBB C303FB JMP SETDEN ;TEST DENSITY
DISKREADY:

```
FBBE 3A00FE  LDA  RDSAT
FBC1 47     MOV  B,A
FBC2 E6A0   ANI  0A0H ; IF DRIVE READY AND HEAD LOADED
FBC4 C8     RZ

IF DRIVE READY AND HEAD LOADED
THEN RETURN
```

DISKREADY:

```
FBC5 C5     PUSH  B
FBC6 1100FE  LDI  D,WRCONT ; (D)=WRCONT=RDSAT
FBC9 CD71FB  CALL  HEADLOAD ; LOAD HEAD
```

COLDBOOT:

```
FB03 3140FC  LII  SP,STACK
FB06 AF     XRA  A
FB07 0100FC  LII  B,BUFF
FB0A 02     CBUFF: STAX  B ; ZERO OUT RAM BUFFER
FB0D 0C     INR  C
FB0C 20AFBF  JNZ  CBUFF
FB0E 3E0A    MVI  A,10
FB0F 3210FC  STA  STEPTIME ; SET STEPTIME LONGER THAN IT NEEDS TO BE
                                ; TO BE SAFE, SINCE COLD BOOT LOADER RESETS IT
FB0E 010000  LII  B,0
FB0F 0B3FA    CALL  SETDMA ; SETDMA DOES NOT CHANGE C REG, SO...
FB0F 0C1FB    CALL  SELDSK ; SELECT DRIVE A
FB0F 0E01    MVI  C,01 ; LOAD BOOTSTRAP LOADER
FB0F 0E6FA    CALL  SETSEC ; FROM TRACK 0 SECTOR 1
FB0F 029FB    CALL  READ
FB0F 023FB    JNZ  COLDBOOT ; ON READ FAILURE, TRY AGAIN
FB0F 07C7    RST  0 ; EXECUTE BOOTSTRAP LOADER
                                ; (SAVES 2 BYTES OVER JMP 0000)
FB0F BDFEFF7  MASTABLE  DB  OBHF,ODFH,OEHF,OF7H
FC11 ABDVE43  F970 ALOOP  FB9D ATHOME  FB00 BASE  FB54 BLOOP
FB00 BUFF  FB0A CBUFF  F967 CLOOP  FB03 COLDBOOT
FC03 CONTROLBYTE  FC22 CURDRIVE  FB7D DELAY  FB0A DELAY1
FB07 DELAY2  FC01 DENBYTE  FC16 DENMAP  FC00 DISK  FB0E DISKREADY
FC05 DISKREADY  FC00 DMA  FAEA DONESTEP  FADE DOSTEP  FC00 ERRORBYTE
FB0D ERROR  FFBF ERRORCOUNT  FB88 GLOOP  FB2F GO  FB06 G0HOME
FB71 HEADLOAD  002B HEADSETTLE  FA00 HIGE  FB97 HOME  FB59 INK
FB75 LLOOP  FC06 LOGINTAB  FA80 LOOP10  FA93 LOOP11  F9EE LOOP6
FB96 LOOP7  FBFF MASTABLE  F973 MLOOP  F9BD NLOOP  F8EA NO
FC05 PRESDISK  FE06 RDDataTable  FE05 ROMMARK  FE04 RMRKCR  FB00 RDSAT
FE02 RDUART  FB29 READ  FB05 READADD  FC02 READWRITE  FA57 RETRY1
FC21 RETRYCOUNT  FA1A RETRY  FFBF RLOOP  F8AE RXFER  F970 SD
FA2A SDLPOP1  FA31 SDLPOP2  FA20 SDTEST  FC0A SECTOR  FB1E SELDSK
FB2B SELDSK1  FB03 SETDEN  FA93 SETDMA  FA99 SETSEC  FB69 SETTN
FAAE SETTRK  FA69 SKEW  FA9D SKIP12  F8B8 SKIP3  F80D SKIP
FATE SKIPIY  FC40 STACK  FACC STEPHED  FABA STEPEN  FABE STELOOP
FACF STEPOUT  000F STEPSETTLE  FC10 STEPTIME  F95F SYNC  FE07 SYNCPORT
FBF2 TERROR1  F8E4 TERROR  F90C TEST1  FC23 TESTMAX  FC0E TESTNEXT
FB9B TEST  FC04 TRACK  FC12 TRACKTAB  FC20 TRY1  FC0F TWOSIDE
```
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>F91C</td>
<td>WLDOP</td>
</tr>
<tr>
<td>F82D</td>
<td>WRITE</td>
</tr>
<tr>
<td>FE04</td>
<td>WRMRXRCRC</td>
</tr>
<tr>
<td>FE01</td>
<td>WRCLK</td>
</tr>
<tr>
<td>FE00</td>
<td>WRCONT</td>
</tr>
<tr>
<td>FE07</td>
<td>WRCRC</td>
</tr>
<tr>
<td>FE06</td>
<td>WRDATA</td>
</tr>
<tr>
<td>F903</td>
<td>WRITEDD</td>
</tr>
<tr>
<td>F81E</td>
<td>WRITEPROTECT</td>
</tr>
<tr>
<td>FE05</td>
<td>WRMRK</td>
</tr>
<tr>
<td>FE02</td>
<td>WRUART</td>
</tr>
<tr>
<td>F91B</td>
<td>WIXFER</td>
</tr>
<tr>
<td>F946</td>
<td>WRITESD</td>
</tr>
</tbody>
</table>