Force Computers

miniFORCE

USER's MANUAL

miniFORCE 2P/32

USER'S MANUAL

First Edition
February 1987

800133

FORCE COMPUTERS Inc./GmbH

NOTE

The information in this document has been carefully checked and is believed to be entirely reliable. FORCE COMPUTERS makes no warranty of any kind with regard to the material in this document, and assumes no responsibility for any errors that may appear in this document. FORCE COMPUTERS reserves the right to make changes without notice to this, or any of its products, to improve reliability, performance or design.

FORCE COMPUTERS assumes no responsibility for the use of any circuitry other than circuitry which is part of a product of FORCE COMPUTERS GmbH/Inc.

FORCE COMPUTERS does not convey to the purchaser of the product described herein any license under the patent rights of FORCE COMPUTERS GmbH/Inc. nor the rights of others.

FORCE COMPUTERS Inc.
727 University Avenue
Los Gatos, CA 95030
U.S.A.

Phone : (408) 354 34 10
Telex : 172465
FAX : (408) 395 77 18

FORCE COMPUTERS GmbH
Daimlerstrasse 9
D-8012 Ottobrunn/Munich
West Germany

Phone : (089) 600 91-0
Telex : 524190 forc-d
FAX : (089) 609 77 93

FORCE COMPUTERS FRANCE Sarl
11, rue Casteja
92100 Boulogne
France

Phone : (1) 4620 37 37
Telex : 206 304 forc-f
FAX : (1) 4621 35 19

FORCE Computers UK Ltd.
No. 1 Holly Court
3 Tring Road
Wendover
Buckinghamshire HP22 6NR
England

Phone : (0296) 625456
Telex : 838033
FAX : (0296) 624027

INTRODUCTION

TABLE OF CONTENTS

List of Figures

1.Ø GENERAL INFORMATION

The **miniFORCE 2P/32** systems family offers high end, low cost, 32 bit, high performance, high reliability, VMEbus based development/target system including the real time operating system PDOS*.

The main idea behind the design of this system is to allow the user the development of programs, combined with a minimum hardware investment, a low price and a high expansion capability.

The following sections and paragraphs of this manual describe in detail the features which are implemented on this VMEbus based Computer System family.

The first five sections (1,2,3,4,5) of this manual contain the description of the features of all miniFORCE 2P/32 based systems.

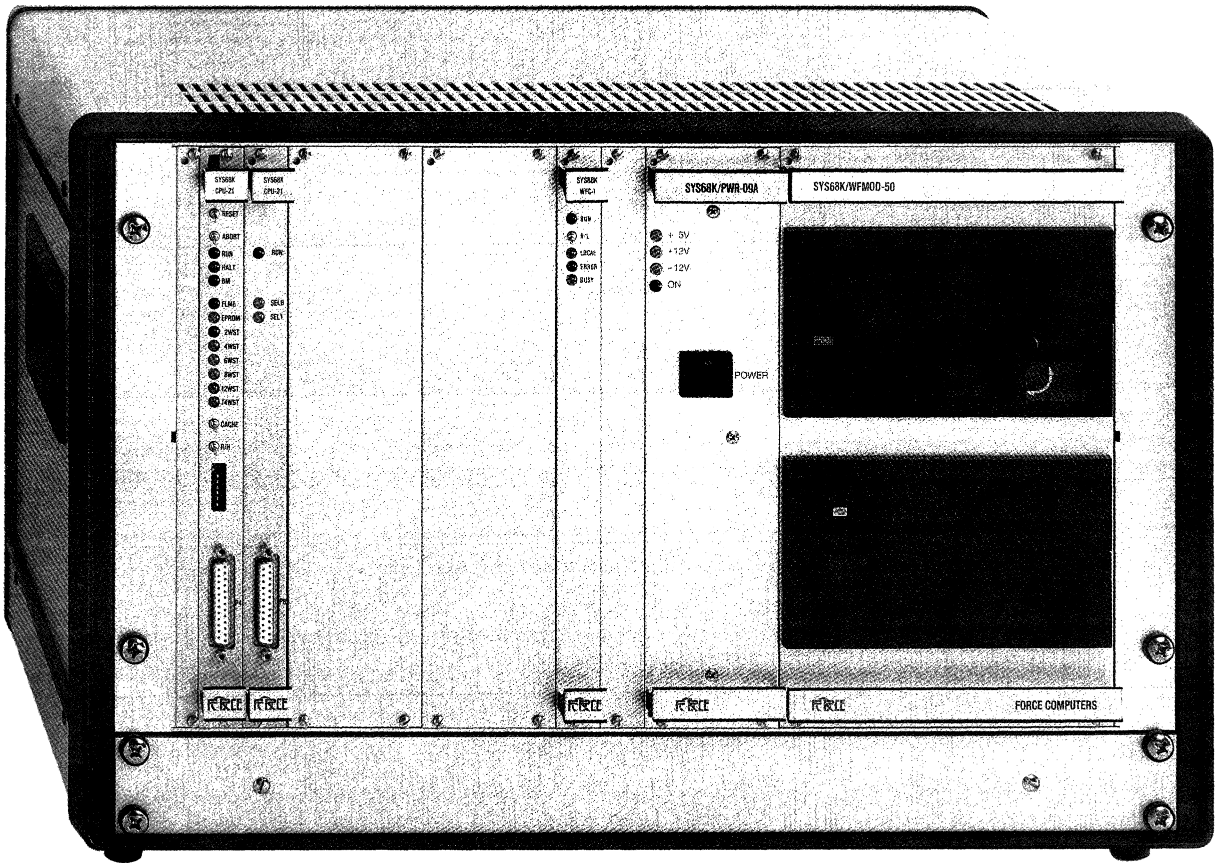
Section 6 describes the applications boards which can be installed in the system.

Section 7 describes the installation and the utilities of the PDOS* operating system.

These requirements are realized via a 19 inch standard enclosure and all included modules are 19 inch compatible. The system contains two backplanes (9 slot) for the P1 and P2 connector of the VMEbus based boards. Additional modules are the power supply and the mass storage drives enclosure.

* PDOS is a Trademark of Eyring Research Institutes.

Figure 1-1: Photo of the System



2.0 SYSTEM DESCRIPTION

2.1 Features of the System

- Desktop station for 32 bit VMEbus environments
- Two 9 slot motherboards for A32/D32 wide VMEbus (P1,P2)
- 280W power supply to drive VMEbus and mass storage memory
- 7HE 19 inch metal chassis including modules for drives, power supply and connectors (344mm x 520mm x 400mm).
- High modularity
- One 1HE fan module including 3 fans for optimal cooling
- Flexible mounting and very little time expenditure for repairs through modularity
- Status indicators and switches of the VME boards are directly accessible on the front of the system
- One 5 1/4" full height space for the floppy drive
- One 5 1/4" full height space for the winchester drive
- Up to 6 free slots for system expansion

2.2 Mechanical Modules

The **miniFORCE 2P/32** systems contain the following modules:

- Two 9 slot motherboards (VMEbus based)
- 28ØW power supply
- One 1HE fan module including three fans (1ØØ m3/h each)
- One module for drive mounting
- Internal basic construction for the mounting of the modules

All of these modules can be mounted/removed from the front very easily after removing up to four screws.

3.Ø APPLICATIONS FOR THE miniFORCE 2P/32 SYSTEMS

The **miniFORCE 2P/32** systems are designed for a variety of high speed real time applications. The enclosure itself, with its capabilities, allows the user to install a maximum of 9 VMEbus based boards. The Winchester drive, the floppy drive and the cabling are installed in the system by default.

The power supply included supports all modules in the system.

The VMEbus based boards installed and the implemented operating system PDOS* define the application of the system. Possible applications for the system are as follows:

- Development station/target (PDOS* based)
- Real time process controller
- Roboting and manufacturing automation
- For high speed scientific purposes

* PDOS is a Trademark of Eyring Research Institutes.

INSTALLATION

AND WARNINGS

TABLE OF CONTENTS

1.Ø GENERAL INFORMATION

The **miniFORCE 2P/32** systems contain switches, connectors, indicators, and different slots for board, drive installation and mounting.

This chapter contains a brief overview and description of the handling of the system.

The following paragraph describes the common modules of the system. In addition, the mechanical structure of the system, and important warnings will be described.

2.Ø FRONT PANEL DESCRIPTION

The 19 inch standard front panel of the system contains the openings for the VMEbus boards, the module for the storage drives and the power supply.

This front panel is fix-mounted via six screws on the system chassis.

2.1 Switches

The switches of the installed boards are directly accessible on the front.

Therefore the function of the switches is described in the hardware user's manual of the equivalent installed board.

Generally, these switches are marked on each front panel of the installed board. Each CPU board contains a RESET and a ABORT switch. Each additional VME board contains generally a RUN/LOCAL switch for isolation from the VMEbus.

2.2 Indicators

The indicators of each installed board are viewed on the front panel of the board.

Therefore the function of the indicators is described in the equivalent hardware user's manual.

Generally each CPU board contains a RUN and a HALT indicator and each additional VME board a RUN and a LOCAL indicator on the front panel.

2.3 The Drive Slots

The floppy drive which is installed in the system can be operated on the front. The drive module contains two 5 1/4 inch full height openings for the operation with the floppy and the Winchester drive.

The Winchester drive which is installed in the same enclosure can be removed or replaced by the user after removal of the drive module. The drive module is mounted with four screws on the chassis front panel. Therefore the drive module can be easily removed.

2.4 The VME Board Slots

The VME boards can be installed in the system via the front. The benefit of this mounting method is that the boards can be installed without opening the system. Additionally, the mounted I/O connectors on the front panel of the boards can be used without opening or making any changes to the mechanics of the system.

3.Ø BACK PANEL DESCRIPTION

The back panel is a fixed panel and contains different openings for power connector mounting, a power switch and the cooling area.

3.1 The Connectors

The back panel contains a power connector for 22Ø/11Ø input power, including fuse. This is the main power connector of the system to the 22Ø/11ØV input power.

3.2 The Power Switch

The main power-on switch for the 22Ø/11ØV is installed on the back panel. This switch is directly on top of the main power connector. This main power switch turns on the 22Ø/11ØV input power only. The DC power on switch of the system is included on the front panel of the power supply. (See Section 3, Chapter 4)

4.Ø WARNINGS

This paragraph contains a few very important check points which must be observed before the system is powered on.

- Visually inspect the product and the documentation for any physical damages. This product was shipped in perfect condition, therefore any damages have to be reported to the courier agent immediately.
- Check if the selected input power of the system corresponds with the used voltage in your country (11ØV or 22ØV).
- Never open the chassis before the main power is turned off, via the main power switch which is placed on the back panel of the system.
- Before the system is powered on, check if the cooling conditions (on the top and bottom) are uncovered. If this instruction has not been observed, then the system crashes and system damages are possible.

5.0 POWER UP PROCEDURE

To power on the **miniFORCE 2P/32** systems, the following procedures must be taken:

- Check if the system is marked to the 220V or 110V input voltage.

- Connect the 220V/110V power cable to the power connector on the back panel.

- Connect a terminal to the 25 pin D-sub connector of the main CPU board.

- Power on the main power switch on the back panel of the system.

- Power on the DC power switch on the front panel of the power supply.

- The system is now powered on.

- Check if the LED indicators marked with RUN on the CPU board lights on.

- In order to boot the installed operating system, see Section 7 "Operating System Installation".

SYSTEM LAYOUT AND INTERCONNECTION

TABLE OF CONTENTS

1.Ø GENERAL INFORMATION

The **miniFORCE 2P/32** series is a high speed, 32 bit wide Address-and-Data-Bus, low cost and flexible VMEbus based system. It contains functional modules which will be described separately. These functional modules are as follows:

- Two 9 slot VMEbus motherboards (MOTH-Ø9A, MOTH-EO9A).
- The power supply (SYS68K/PWR-Ø9A)
- The disk drive media module (WFMOD)
- The Cooling 1HE module
- The VMEbus area
- The back panel

The system is designed for 32 bit VMEbus based applications. The advantages of the system are the modularity, the standard 19 inch dimensions, the low cost and the easy operation. The interconnections of the different modules in the system are implemented via standard connectors and flexible flat cables.

2.Ø THE MOTHERBOARD LAYOUT

The **miniFORCE 2P/32** series systems contain a motherboard called **SYS68K/MOTH-Ø9A** which fully supports up to nine VME boards. It contains nine 96 pin connectors for the VMEbus (P1) as well as different power connectors. Additionally, an extension motherboard for the P2 connector called SYS68K/MOTH-EØ9A is included on the system.

The motherboards contain the VME termination resistor networks and fast-on connectors for the power control signals. This chapter describes the connectors, the connection and the functionality of the motherboards.

2.1 The VMEbus Slots

2.1.1 The SYS68K/MOTH-Ø9A

The **SYS68K/MOTH-Ø9A** supports nine VME slots for A24/D16 environment. All nine VMEbus boards have to be installed vertically. The opening for the installation of the VMEbus boards is on the front side of the system.

The slot number one is placed on the left corner, the next
slots continue from the left to the right until slot nine.

Figure 2-1 shows the principle slot numbering of the VMEbus motherboard. Regarding the VMEbus Specification Rev. C, all signals are terminated with resistor networks.

Figure 2-1: The VMEbus Slot Numbering

```
 _______________________________________________________________________
|   |   |   |   |   |   |   |   |   |          |          |
|   |   |   |   |   |   |   |   |   |          |          |
| S | S | S | S | S | S | S | S | S |          |          |
|   |   |   |   |   |   |   |   |   |          |          |
| L | L | L | L | L | L | L | L | L | PWR-Ø9A  |   DISK   |
|   |   |   |   |   |   |   |   |   |          |          |
| O | O | O | O | O | O | O | O | O |          |          |
|   |   |   |   |   |   |   |   |   |  POWER   |  DRIVES  |
| T | T | T | T | T | T | T | T | T |          |          |
|   |   |   |   |   |   |   |   |   |          |          |
|   |   |   |   |   |   |   |   |   |  SUPPLY  |  MODULE  |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |          |          |
|___|___|___|___|___|___|___|___|___|__________|__________|
|                                                          |
|              C O O L I N G   M O D U L E                 |
|__________________________________________________________|
```

The SYS68K/MOTH-Ø9A contains eight jumperfield arrays to support the interrupt acknowledge and the bus request daisy chain. The jumpers can be installed on the reverse side of the motherboard.

Figure 2-2 shows the jumper location on the motherboard componentside.

The following table 2-1 shows the exact correspondence between the jumperfields and the VMEbus slots for the IACK and bus grant daisy chain.

Table 2-1: **SYS68K/MOTH-Ø9A Jumper Description**

Slot #	Signals To Be Bridged Over				
	IACKIN to IACKOUT	BGØIN to BGØOUT	BG1IN to BG1OUT	BG2IN to BG2OUT	BG3IN to BG3OUT
1	*B1 connects IACK Slot 1 to IACKOUT of Slot 1	--	--	--	--
2	BØ	B5	B4	B3	B2
3	B6	B1Ø	B9	B8	B7
4	B11	B15	B14	B13	B12
5	B16	B2Ø	B19	B18	B17
6	B21	B25	B24	B23	B22
7	B26	B3Ø	B29	B28	B27
8	B31	B35	B34	B33	B32

Example for a possible jumper setting:

When in slot 3, the IACK daisy chain and the bus request level 2 daisy chain must be installed, then the following jumpers according to table 2-1 have to be inserted.

The B6 for the IACK daisy chain and the B8 for the Bus Grant level 2 must be inserted.

Figure 2-2: Jumper Layout of the MOTH - Ø9A

```
      __   __         __         __         __         __         __         __   __
o    |  | |  | o    |  | o    |  | o    |  | o    |  | o    |  | o    |  | |  | |o
oB5  |  | |  | oB1Ø |  | oB15 |  | oB2Ø |  | oB25 |  | oB3Ø |  | oB35 |  | |  | |o
     |  | |  |      |  |      |  |      |  |      |  |      |  |      |  | |  | |
o    |  | |  | o    |  | o    |  | o    |  | o    |  | o    |  | o    |  | |  | |o
oB4  |  | |  | o B9 |  | oB14 |  | oB19 |  | oB24 |  | oB29 |  | oB34 |  | |  | |o
     |  | |  |      |  |      |  |      |  |      |  |      |  |      |  | |  | |
o    |  | |  | o    |  | o    |  | o    |  | o    |  | o    |  | o    |  | |  | |o
oB3  |  | |  | o B8 |  | oB13 |  | oB18 |  | oB23 |  | oB28 |  | oB33 |  | |  | |o
     |  | |  |      |  |      |  |      |  |      |  |      |  |      |  | |  | |
o    |  | |  | o    |  | o    |  | o    |  | o    |  | o    |  | o    |  | |  | |o
oB2  |  | |  | o B7 |  | oB12 |  | oB17 |  | oB22 |  | oB27 |  | oB32 |  | |  | |o
     |  | |  |      |  | |  | |
     |  | |  |      |  | |  | |
o    |  | |  | o    |  | o    |  | o    |  | o    |  | o    |  | o    |  | |  | |o
oB1  | 1| | 2| o B6 | 3| oB11 | 4| oB16 | 5| oB21 | 6| oB26 | 7| oB31 | 8| | 9| |o
o    |  | |  |      |  |      |  |      |  |      |  |      |  |      |  | |  | |
oBØ  |__| |__|      |__|      |__|      |__|      |__|      |__|      |__| |__| |
```

2.1.2 The Motherboard MOTH-EØ9A

The extension motherboard SYS68K/MOTH-EØ9A is used for the A32/D32 mode extensions as well as for the VME boards which need more than 3A power consumption. Additionally this motherboard is needed for VME boards interconnections via the P2 connector.

2.2 The I/O Connectors

All I/O connectors which support the winchester, floppy and the RS232 Interfaces are directly connected to the equivalent front panel (SIO-1FP) or back panel (WFC-1BP). These panels are connected to the equivalent board either via the P2 connector of the MOTH-EØ9A (WFC-1BP) or via the connectors placed directly on the PCB (SIO-1FP).

2.3 The Power Connectors

The SYS68K/MOTH-Ø9A and the SYS68K/MOTH-EØ9A are the main consumers for the DC-power. The input powers are +5V, +12V, and -12V. The power lines are connected to the VMEbus slots according to the VMEbus specification.

The following tables give an overview about the connector type and the function description.

a) SYS68K/MOTH-Ø9A:

No. of Connectors	Type	Function
8	Gold plated fast on connectors	GND input to the motherboard
8	Gold plated fast on connectors	+5V input to the motherboard
2	Gold plated fast on connectors	+12V input to the motherboard
2	Fast on connector	-12V input to the motherboard

b) SYS68K/MOTH-EØ9A:

No. of Connectors	Type	Function
9	Gold plated fast on connectors	GND input to the motherboard
9	Gold plated fast on connectors	+5V input to the motherboard

2.4 The Power Control Signals

The motherboard called SYS68K/MOTH-Ø9A contains additional control signals which guarantee the stability of the DC power. Additionally, the motherboard contains power control signals which can be used optionally (SYSRESET,SYSFAIL,etc.,). The following table shows the functionality of the connectors available on the motherboard.

No. of Connectors	Marked With	Type	Function
1	ACFAIL	Gold plated fast on connector	VMEbus compatible ACFAIL signal placed on the top of the motherboard.
1	SYSRESET	Gold plated fast on connector	VMEbus compatible SYSRESET signal placed on the top of the motherboard.
1	SYSFAIL	Gold plated fast on connector	VMEbus compatible SYSFAIL signal placed on the top of the motherboard.
2	STDBY	Gold plated fast on connector	+5V standby. This power input can be used for systems with +5V standby.
1	SENSE +5V	Gold plated fast on connector	+5V sense line must always be connected.
1	SENSE GND	Gold plated fast on connector	GND sense line must always be connected.

3.Ø Mass Storage Media Module

3.1 The Disk Drive Module

The **miniFORCE 2P/32** series systems contain one Winchester drive with ST5Ø6 interface and a floppy drive with SA46Ø interface. The housing of the disk drives is a metal module. This module can be accessed directly on the front of the system. The module is mounted via four screws on the front of the system.

The drives are mounted into this module via four screws. The winchester and the floppy interface on the backside is connected to the back panel of the SYS68K/WFC-1 board.

Additionally, the power connectors of the drives are connected directly to the power supply connectors.

4.0 The Power Supply SYS68K/PWR-09A

The **miniFORCE 2P/32** series systems contain a highly flexible multi-output, switched mode power supply to fully support the VMEbus and both drives. The power supply is placed on the front of the system, directly between the VME area and the Winchester floppy module.

4.1 Specifications

The features of the SYS68K/PWR-09A are as follows:

- Input power	220V +20% -10%
	or 110V ±10% fuse selectable
	Frequency = 50-60Hz
- Output voltage	+5V +12V -12V
- Output current	36A 6A 2A
- Total power	280W
- Operating temperature	0-50 Degrees C

- The power supply is built according to the VDE 0804 and IEC-380.

- All output voltages are stabilized via the sense lines.

- All output voltages are separately controlled and no minimum load is required.

- The dimensions of the Power Supply:

Length:	210 mm	
Height:	6HE	(1HE ≙ 44,45mm)
Width:	12TE	(1TE ≙ 5,08mm)

- The power supply contains a slow-converter shut-down automatic soft restart.

- All output voltages are short protected and the foldback is supported.

- A power switch is included on the front panel.

4.2 Connector Description

The power supply contains two H15 connectors for all input and output signals. The signals to/from the power supply are divided into two groups. The first group contains all 22ØV/11ØV signals. The second group contains all power sense signals and the +5V, +12V, -12V and GND power lines. Each signal group described above is placed onto a separate connector area.

The connector on the bottom (P2) contains the connections for the 22ØV/11ØV input voltage power.

The connector on the top (P1) contains the connection points for the DC output power (GND, +12V, -12V, +5V).

The following schematic shows the location of the connectors on the back of the power supply.

The P1 Pin Assignment of the SYS68K/PWR-Ø9A

Pin No.

4	6	8	1Ø	12	14	16	18	2Ø	22	24	26	28	3Ø	32
+5V		+5V		+5V		-12V		GND		GND		+12V		GND
														SENSE z.
	+5V		+5V		+5V		GND		GND		GND		+12V	
					SENSE									d.

The P2 Pin Assignment of the SYS68K/PWR-Ø9A

Pin No.

4	6	8	1Ø	12	14	16	18	2Ø	22	24	26	28	3Ø	32
ACFAIL		NC		NC		NC		NC		NC		AC		AC-GND
	SYSRESET		NC		NC		NC		NC		NC		AC	

4.3 Mounting of the Power Supply

The power supply is placed on the frontside of the **miniFORCE 2P/32** systems. Four screws provide the fixing onto the front of the chassis. Therefore, for access of the power supply, no covers of the system have to be removed. For the replacement of the power supply, only four screws must be removed.

The cabling is realized via fast on connectors and via H15 connectors. Therefore an easy replacement of the power supply is guaranteed without having to remove cables.

5.Ø The Cooling of the System

The miniFORCE 2P/32 systems contain a 1HE cooling module including three fans with 1ØØm3/h air volume each.

The fans will be connected in parallel for 22Ø Voltage input power and the whole module must be replaced for a 11Ø Voltage input power. This fan module replacement is required because the three fans cannot be connected in a series with 11Ø Voltage input power.

STANDARD miniFORCE 2P/32 CONFIGURATIONS

TABLE OF CONTENTS

1.Ø GENERAL INFORMATION

The miniFORCE 2P/32 systems are 32 bit based systems. They contain modules which are the standard part of the whole system configuration. These modules, combined with different FORCE VMEbus based boards results in a configuration which will be described in this section. By using the standard system configuration, additional boards can be installed in the system for the final application.

All these systems named until now "miniFORCE 2P/32" contain the same modules, the same chassis (7HE) and the same operating system (PDOS*). Therefore the name of the system describes exactly the basics and the dimensions of the system.

For example the miniFORCE 2P21 means that the system is a product with a 7HE chassis (miniFORCE 2) and contains the above described modules. Additionally the "P" means that it is PDOS* operating system based and the "21" on the end means that the system contains the 68Ø2Ø based fully 32 bit main processor board SYS68K/CPU-21.

Therefore the name of the system informs the customer about the basics of the system.

2.Ø THE DEFAULT SYSTEM CONFIGURATIONS

2.1 The "miniFORCE 2P21"

The miniFORCE 2P21 is a multi-user development/target system for real time applications providing high performance and low cost. The used 68Ø2Ø off a real computing power of 2.5 to 3.5 MIPS running without wait states out of 512Kbyte on Board static RAM. The implemented operating system PDOS supports all features of the system.

The system features are as follows:

- Processor 68Ø2Ø (16.7MHz)
- Floating point Coprocessor 68881 (16.7MHz)
- 512Kbyte high speed local static RAM
- 2 serial I/O interfaces
- 51Mbyte winchester
- 1Mbyte floppy
- One Timer
- Real Time Operating System PDOS*
- Full support of all Interfaces under PDOS*

VMEbus boards used:

SYS68K/CPU-21
SYS68K/WFC-1

2.1.1 The Central Processor Unit

This high performance VMEbus-based CPU board combines the powerful 32 bit microprocessor, the 68Ø2Ø with 512Kbyte of static RAm, a maximum of 512Kbyte of ROM/EPROM, powerful I/O devices, and a Floating Point Coprocessor.

The local control, consisting of a Parallel Interface and Timer (PI/T) and a Bus Interrupter MOdule (BIM) supports software control for a wide variety of functions. the SYS68K/CPU-21 is equipped with two RS232-compatible serial I/O channels. The 68881 Floating Point Coprocessor included on the CPU-21 speeds up for number cruncher applications.

The SYS68K/CPU-21 boards contain a primary VMXbus master interface.

Easy access to the devices is provided by the ROM-resident SYSTEM MONITOR, called the SYS68K/2Ø FORCEbug. This powerful software package can be used for program development and debugging of application programs.

The SYS68K/CPU-21 has been developed as a sophisticated VMEbus boards which may be used in a multi-processor environment as a master processor, as a "standalone unit" or as a high performance co-processor board.

To provide all of these features, CPU-21 consists of two double Eurocard boards, the main board and the slave board. On the main board all the bus interfaces, the CPU and the EPROMs are installed.

The slave board contains on e serial I/O interface as well as the 512Kbyte Zero Wait State Static RAM.

Features of the SYS68K/CPU-21

- 68020 CPU with 16.7MHz Clock Frequency
- 68881 Floating Point Coprocessor with 16.7MHz Clock Frequency
- Static RAM 512Kbyte with 55ns access time
- 512Kbyte (max) of ROM/EPROM for the system
- 2 RS232 Multi Protocol Communication Interfaces (110-38400 Baud)
- Parallel Interface and Timer Chip provides local control and timer function
- VMXbus Primary Master Interface to P2 connector
- Local Interrupt handling via interrupt vectors
- Each VMEbus IRQ level can be enabled/disabled via software
- Address range for the short I/O address modifies (AM4)
- Address range for the standard address modifier
- Single level bus arbiter
- One level slave bus arbitration
- Power monitor
- RESET and SOFTWARE ABORT function switches
- Fully VMEbus, VMXbus and IEEE P1014 compatible

2.1.2 The Main Memory

The main memory of the miniFORCE 2P21 is placed on the CPU-21 slave board itself (512Kbyte). An increase of this memory is possible via additional boards on the VMEbus. This possibility is described in Section 6 ("System Expansion") of this manual.

The base address of the included local memory on the CPU-21 is jumpered to ØØØØØØ HEX. An additional memory board must be jumpered to the base address Ø8ØØØØ HEX if no additional local RAM is installed. If an additional RAM board is installed to the CPU-21, then the base address of the VME memory must be jumpered to 1ØØØØØ HEX.

2.1.3 The Mass Storage Controller

The SYS68K/WFC-1 is a high performance VMEbus interface controller board to control 5 1/4" Floppy SA46Ø Interface and Winchester drives(ST5Ø6 interface).

The SYS68K/WFC-1 board has been developed to serve as a controller board for high speed data transfers to and from 5 1/4" Floppy and Winchester drives via a 1Kbyte FIFO buffer. The board can communicate with a DMA controller to provide maximum bus transfer speed and is able to generate interrupts for complete operation, error and data request.

The VMEbus allows easy system design with extended I/O, RAM, CPU, ROM, and DMA cards.

The SYS68K/WFC-1 consists of a set of devices specifically designed for control of Winchester and Floppy disk drives. The heart of the control logic is the Control Processor Buffer Manager (WD1Ø15) that manages the on-board static RAM sector buffer (2Ø48-word by 8-bit). All bytes of data written to and read from disk are first stored on this sector buffer. When the buffer is full, the data is transferred, on command, to its intended destination.

The WD1Ø15, besides controlling the data flow between host, sector buffer, and disk controllers, also translates the host Winchester command format to Floppy disk format when addressing the Floppy Disk Controller (WD2797). This permits the host to maintain a single command format (Winchester) while in effect controlling two different disk command formats (Winchester vs. Floppy). This is possible, since the SDH register is used to select either type of drive.

The WD1Ø15 maintains the current copies of necessary host command data in the task files; a set of registers physically located in the Winchester Disk Control device (WD1Ø14).

The WD1Ø1Ø is the link between the host processor (via sector buffer) and the Winchester disk drives. During transfer of data from the host to the WD1Ø1Ø, the WD1Ø14 computers a 4-byte ECC which is appended to the end of the data being transferred to the WD1Ø1Ø and recorded on the disk. During data transfers from

WD1010 to the host (via the sector buffer), the WD1015 uses the ECC syndrome to validate the data. Retries and corrections are attempted automatically in case of corrupted data.

The WD1015 performs error correction in conjunction with WD1014 on data transferred to the disk. While the WD1015 controls the operation of the on-board error-correction logic, the WD1014 generates and checks the Error Correction Code (ECC) if SDH bit 7 = 0. Thus the WD1014 also provides the WD1015 with its real-time control capability.

If CRC format Winchester disks are used, CRC is selected by the WD1010 by setting SDH7 = 0. CRC for the floppy disk is performed by the WD2729, a device that furnishes all control functions for floppy disk drives, including necessary data separation and write precompensation. SDH7 must be set to zero for floppy disk operation.

The jumper setting for the installation on the miniFORCE 2P21 is described in Section 5 ("VME Boards Set Up").

The features of the SYS68K/WFC-1 are as follows:

- Fully VMEbus compatible
- Jumper selectable base address with address modifier
- Generation of two different interrupts
- Jumper selectable interrupt level
- Software programmable interrupt vectors
- Three VMEbus options (A31:D16), (A23:D16), (A15:D16) jumper selectable
- User selectable 5.25" Winchester or Floppy operation
- Controls up to 3 Winchester ST506 Interface and up to 4 Floppy drives SA460 compatible
- On-board data separation circuitry
- On-board write precompensation for floppy and hard disks
- On-board sector buffer supports up to 1Kbyte sectors
- Programmable sector sizes - 128, 256, 512, or 1024bytes
- Automatic track formatting on hard and floppy disks
- Multiple sector operations on all disks
- Data rates up to 5Mbits/sec on hard disk
- Single burst error correction up to 5bits on hard disk data

- CRC generation/verification for data and all I.D. fields
- Automatic retries on all errors with simulated completion
- ECC diagnostic commands included (READALONG & WRITEALONG)
- Internal diagnostics
- 16 different stepping rates for both hard and floppy drives

2.1.4 The Mass Storage Drives

The miniFORCE 2P21 contains one slim line 5 1/4" winchester drive and one slim line 5 1/4" floppy drive.

Both drives are installed in the module which is accessible on the front of the system. The drive select codes for these units are:

Winchester:	DS0
Floppy:	DS2

The installed Winchester is a half height disk drive which can store 51Mbyte of unformatted data. Winchester technology assures trouble-free performance, and the rugged, shock-resistant design makes the drive perfect for use in compact, portable systems. With only two required voltages (+12V and +5V) and its ST506 (ST412) type interface, the drive can be easily installed into the system.

The features of the drive are as follows:

Unformatted capacity:	51Mbyte
Formatted capacity: (256byte/sector)	39.2(40.96)Mbyte
Number of disks	3
Number of heads	5
Number of cylinders	981 (1024)
Data Transfer Rate	625Kbyte/s
Access Time Average	40ms
Recording Method	MFM
Non-operating Shock	40G at 10ms
Disk speed (rpm)	3564
Drive selections	4
Power Dissipation (average)	20W

Dimensions:

Width	146mm	(5.75 in)
Height	41mm	(1.62 in)
Depth	208mm	(8.2 in)
Weight	1.8kg	(4 lbs)

Reliability:

MTBF	20000 POH
MTTR	30 minutes

The installed TEAC FD-55FV in a highly reliable 5 1/4" slim line floppy disk drive.

The motor on the drive is equipped with precision digital locked control, on enhancement which ensures unerring accuracy under all operating conditions. The power consumption is an ultra-low 1.5W in the non-operating mode, and only 4.1W during actual operation.

A band actuator system works to position the magnetic heads rapidly and positively.

The features of the floppy disk drive are as follows:

Capacity unformatted	1Mbyte
Capacity formatted (256byte/sector)	655Kbyte
Tracks/Disk	160
Track Density	96tpi
Average Access Time	94ms
Track Access Time	3ms
Disk Rotation Speed	300rpm
Recording Method	MFM
MTBF	10000 POH
Power Requirements	+5V 0,3A(typ) 0,4A(max) +12V 0,22A(typ) 0,55A(max)
Safety Standard	UL and CSA
Power Consumption	
Operating:	4.1W
Non-Operating:	1.5W
Dimensions (WxHxD)	146 x 41.3 x 203mm 5 3/4 x 1 5/8 x 8 in

2.1.5 The Address Map

The following outline gives detailed information about the memory map including the on-board memory and the devices addresses, and standard addresses for the miniFORCE 2P21 cards.

Address	Definition
$00000000-$0007FFFF	Local SRAM
$00080000-$000FFFFF	Local SRAM on SRAM-22 board (optional)
$FF000000-$FF07FFFF	EPROM Area 1 on board I/O devices
$00080000-$FAFFFFFF	VMEbus memory if no additional SRAM-22 board is installed
$00100000-$FAFFFFFF	VMEbus memory if an additional SRAM-22 board is installed
$FCB01000-$FCB0100F	WFC-1 card
$FCB02000-$FCB022FF	ASCU-1/2 Card (optional)
$FCB00000-$FCB001FF	First SIO-1 Card (optional)

Interrupt Levels:

AV Level 7:	Abort switch
AV Level 5:	Timer chip
AV Level 4:	On-board serial I/O, WFC-1

To BAUD all the ports in the miniFORCE 2P1 system, use the following:

BP 2,1,1,$FF800200	Port 1 Terminal
BP 3,1,1,$FF800600	Port 2 Terminal (optional, if SRAM-22 is installed)

2.2 The miniFORCE 2P21S

The miniFORCE 2P21S is a downgrade of the miniFORCE 2P21. The features of the system are generally the same but only the coprocessor (68881) and the processor (68020) speed is reduced to 12.5MHz. The winchester drive capacity is also reduced to 20Mbyte. Therefore a low price can be offered for a 68020 based VMEbus system with a acceptable performance can be offered.

The features of the miniFORCE 2P21S system are as follows:

- Processor 68020 (12.5MHz)
- Floating point coprocessor 68881 (12.5MHz)
- Memory 512Kbyte local with zero wait state
- 2 serial I/O interfaces
- 20Mbyte Winchester
- 1Mbyte Floppy
- One Timer
- Real Time Operating System PDOS*
- Full support of all Interfaces under PDOS*

VMEbus boards used:

SYS68K/CPU-21S
SYS68K/WFC-1

2.2.1 The Central Processor Unit

This high performance VMEbus-based CPU board combines the powerful 32 bit microprocessor, the 68020 with 512Kbyte of static RAM, a maximum of 512Kbyte of ROM/EPROM, powerful I/O devices, and a Floating Point Coprocessor.

The local control, consisting of a Parallel Interface and Timer (P/IT) and a bus Interrupter Module (BIM) supports software control for a wide variety of functions. The SYS68K/CPU-21S is equipped with two RS232-compatible serial I/O channels. The 68881 Floating Point Coprocessor included on CPU-21S speeds up for number cruncher applications.

the SYS68K/CPU-21S boards contains a primary VMXbus master interface.

Easy access to the devices is provided by the ROM-resident SYSTEM MONITOR, called the SYS68K-20 FORCEbug. This powerful software package can be used for program development and debugging of application programs.

The SYS68K/CPU-21S has been developed as a sophisticated VMEbus board which may be used in a multi-processing environment as a master processor, as a "standalone unit" or as a high performance coprocessor board.

To provide all of these features, CPU-21S consists of two double Eurocard boards, the main board and the slave board. On the main board all the bus interfaces, the CPU and the EPROMs are installed.

The slave board contains one serial I/O interface as well as the 512Kbyte Zero Wait State Static RAM.

Features of the SYS68K/CPU-21S

- 68020 CPU with 12.5MHz Clock Frequency
- 68881 Floating Point Coprocessor with 12.5MHz Clock Frequency
- Static RAM 512Kbyte with 55ns access time
- 512Kbyte (max) of ROM/EPROM for the system
- 2 RS232 Multi Protocol Communication Interfaces (110-38400 Baud)
- Parallel Interface and Timer Chip provides local control and timer function
- VMXbus Primary Master Interface to P2 connector
- Local Interrupt handling via interrupt vectors
- Each VMEbus IRQ level can be enabled/disabled via software
- Address range for the short I/O address modifies (AM4)
- Address range for the standard address modifier
- Single level bus arbiter
- One level slave bus arbitration
- Power monitor
- RESET and SOFTWARE ABORT function switches
- Fully VMEbus, VMXbus and IEEE P1014 compatible

2.2.2 The Main Memory

The main memory of the miniFORCE 2P21S is placed on the CPU-21S (512Kbyte) by default. The memory is static RAM based and the processor runs with zero wait states out of this memory.

Two ways are possible of increasing the memory. First, the static RAM can be increase through an additional memory board call SRAM-22 which has 512Kbyte static RAM space. The benefit of this memory is that the processor runs with zero wait sates out of lMbyte memory. Second, an increase of the VMEbus memory capacity is possible via VMEbus based memory boards. In this case, the memory board must be jumpered with the address 80000 HEX.

If an SRAM-22 is already installed then the base address of the VMEbus memory board must be jumpered at 100000 HEX to guarantee contiguous memory.

2.2.3 The Mass Storage Controller

The SYS68K/WFC-1 is a high performance VMEbus interface controller board to control 5 1/4" Floppy SA460 Interface and Winchester drives(ST506 interface).

The SYS68K/WFC-1 board has been developed to serve as a controller board for high speed data transfers to and from 5 1/4" Floppy and Winchester drives via a 1Kbyte FIFO buffer. The board can communicate with a DMA controller to provide maximum bus transfer speed and is able to generate interrupts for complete operation, error and data request.

The VMEbus allows easy system design with extended I/O, RAM, CPU, ROM, and DMA cards.

The SYS68K/WFC-1 consists of set of devices specifically designed for control of Winchester and Floppy disk drives. The heart of the control logic is the Control Processor Buffer Manager (WD1015) that manages the on-board static RAM sector buffer (2048-word by 8-bit). All bytes of data written to and read from disk are first stored on this sector buffer. When the buffer is full, the data is transferred, on command, to its intended destination.

The WD1015, besides controlling the data flow between host, sector buffer, and disk controllers, also translates the host Winchester command format to Floppy disk format when addressing the Floppy Disk Controller (WD2797). This permits the host to maintain a single command format (Winchester) while in effect controlling two different disk command formats (Winchester vs. Floppy). This is possible, since the SDH register is used to select either type of drive.

The WD1015 maintains the current copies of necessary host command data in the task files; a set of register physically located in the Winchester Disk Control Device (WD1014).

The WD1010 is the link between the host processor (via sector buffer) and the Winchester disk drives. During transfer of data from the host to the WD1010, the WD1014 computers a 4-byte ECC which is appended to the end of the data being transferred to the WD1010 and recorded on the disk. During data transfers from WD1010 to the host (via the sector buffer), the WD1015 uses the ECC syndrome to validate the data. Retries and corrections are attempted automatically in case of corrupted data.

The WD1015 performs error correction in conjunction with WD1014 on data transferred to the disk. While the WD1015 controls the operation of the on-board error-correction logic, the WD1014 generate and checks the Error Correction Code (ECC) if SDH bit 7 = 0. Thus the WD1014 also provides the WD1015 with its real-time control capability.

If CRC format Winchester disks are used, CRC is selected by the WD1010 by setting SDH7 = 0. CRC for the floppy disk is performed by the WD2729, a device that furnishes all control functions for floppy disk drives, including necessary data separation and write precompensation. SDH7 must be set to zero for floppy disk operation.

The jumper setting for the installation on the miniFORCE 2P21S is described in Section 5 ("VME Boards Set Up").

The features of the SYS68K/WFC-1 are as follows:

- Fully VMEbus compatible
- Jumper selectable base address with address modifier
- Generation of two different interrupts
- Jumper selectable interrupt level
- Software programmable interrupt vectors
- Three VMEbus options (A31:D16), (A23:D16), (A15:D16) jumper selectable
- User selectable 5.25" Winchester or Floppy operation
- Controls up to 3 Winchester ST506 Interface and up to 4 Floppy drives SA460 compatible
- On-board data separation circuitry
- On-board write precompensation for floppy and hard disks
- On-board sector buffer supports up to 1Kbyte sectors
- Programmable sector sizes - 128, 256, 512, or 1024bytes
- Automatic track formatting on hard and floppy disks
- Multiple sector operations on all disks
- Data rates up to 5 Mbits/sec on hard disk
- Single burst error correction up to 5 bits on hard disk data
- CRC generation/verification for data and all I.D. fields
- Automatic retries on all errors with simulated completion
- ECC diagnostic commands included (READALONG & WRITEALONG)
- Internal diagnostics
- 16 different stepping rates for both hard and floppy drives

2.2.4 The Mass Storage Drives

The miniFORCE 2P21S contains one slim line 5 1/4 inch floppy drive and one 5 1/4 inch winchester drive.

Both drives are installed in the module which is accessible on the front of the system. The drive select codes for these units are:

Winchester:	DSØ
Floppy:	DS2

The installed drive Winchester is a half height disk drive which can store 25.6Mbyte of unformatted data. Winchester technology assures trouble-free performance, and the rugged, shock-resistant design makes the drive perfect for use in compact, portable systems. With only two required voltages (+12V and +5V) and its ST5Ø6 (ST412) type interface, the drive can be easily installed into the system.

The features of the drive are as follows:

Unformatted capacity:	25.6Mbyte
Formatted capacity: (256byte/sector)	19.68Mbyte
Number of disks	2
Number of heads	4
Number of cylinders	615
Data Transfer Rate	625Kbyte/s
Access Time Average	8Øms
Recording Method	MFM
Non-operating Shock	4ØG at 1Øms
Disk speed (rpm)	3564
Drive selections	4
Power Dissipation(average)	2ØW

Dimensions:

Width	146mm	(5.75 in)
Height	41mm	(1.62 in)
Depth	2Ø8mm	(8.2 in)
Weight	1.8kg	(4 lbs)

Reliability:

MTBF	2ØØØØ POH

MTTR	30 minutes

The installed TEAC FD-55FV is a highly reliable 5 1/4 inch slim line floppy disk drive.

The motor on the drive is equipped with precision digital locked control, an enhancement which ensures unerring accuracy under all operating conditions. The power consumption is an ultra-low 1.SW in the non-operating mode, and only 4.1W during actual operation.

A band actuator system works to position the magnetic heads rapidly and positively.

The features of the floppy disk drive are as follows:

Capacity unformatted	1Mbyte
Capacity formatted (256byte/sector)	655Kbyte
Tracks/Disk	160
Track Density	96tpi
Average Access Time	94ms
Track Access Time	3ms
Disk Rotation Speed	300rpm
Recording Method	MFM
MTBF	10000 POH
Power Requirements	+5V 0,3A(typ) 0,4A(max)
	+12V 0,22A(typ) 0,55A(max)
Safety Standard	UL and CSA
Power Consumption	
Operating:	4.1W
Non-Operating:	1.5W
Dimensions (WxHxD)	146 x 41.3 x 203mm 5 3/4 x 1 5/8 x 8 in

2.2.5 The Address Map

The following outline gives detailed information about the memory map including the on-board memory and the devices addresses, and standard addresses for the miniFORCE 2P21S cards.

Address	**Definition**
$00000000-$0007FFFF	Local SRAM
$00080000-$000FFFFF	Local SRAM on SRAM-22 board (optional)
$FF000000-$FF07FFFF	EPROM Area 1 on board I/O devices
$00080000-$FAFFFFFF	VMEbus memory if no additional SRAM-22 board is installed
$00100000-$FAFFFFFF	VMEbus memory if an additional SRAM-22 board is installed
$FCB01000-$FCB0100F	WFC-1 card
$FCB02000-$FCB022FF	ASCU-1/2 Card (optional)
$FCB00000-$FCB001FF	First SIO-1 Card (optional)

Interrupt Levels:

AV Level 7:	Abort switch
AV Level 5:	Timer chip
AV Level 4:	On-board serial I/O, WFC-1

To BAUD all the ports in the miniFORCE 2P1 system, use the following:

BP 2,1,1,$FF800200	Port 1 Terminal
BP 3,1,1,$FF800600	Port 2 Terminal (optional, if SRAM-22 is installed)

2.3 The miniFORCE 2P21A

The miniFORCE 2P21A is an upgrade of the miniFORCE 2P21. The features of the system are the same but the Processor Clock and the coprocessor clock are increased from 16.7MHz to 2Ø MHz. This speed increase is really a performance increase because the processor runs with zero wait states out of the local RAM by 2ØMHz. Therefore, actually a performance improvement of around 2Ø% is accomplished. The other features of the system are absolutely identical to the miniFORCE 2P21.

Additional information can be located in Section 2.1 ("The miniFORCE 2P21").

VME BOARDS SET UP

TABLE OF CONTENTS

1.Ø General Information

The described standard system configurations contain different CPU-, Memory-, and Controller-boards which run under PDOS* if the right jumper setting is installed.

This section describes the exact jumper setting changes to the default setup of each board. The jumper setting of the received system is already done by FORCE but the following description can be helpful for references.

The jumper setting changes which are required, are operating system (PDOS*) dependent. Therefore, some differences to the default jumper setting must be implemented, for example memory base addresses as well as interrupt request levels etc..

2.Ø The CPU Boards

No changes have to be made to the default configuration of the CPU-21, CPU-21S, or CPU-21A.

If you want to add an additional SRAM-22 board please refer to the Section 6 "System Expansions", Chapter 2.

Fore more detailed information of the default jumper setting and of the hardware please refer to the Hardware User's manual of the CPU-2Ø/21.

The table 2-1 shows the Memory Map of the CPU-21 board.

Table 2-1: **The Memory Map of CPU-21**

Address	Device
0000 0000 : 0007 FFFF	Local SRAM on slave board via FLME
0008 0000 : 000F FFFF	Local SRAM on SRAM-22 board (optional)
0008 0000 or 0010 0000 : FAFF FFFF	VMEbus (Extended Access) A32 : D32, D24, D16, D8
FB00 0000 : FBFE FFFF	VMEbus (Standard Access) A24 : D32, D24, D16, D8
FBFF 0000 : FBFF FFFF	VMEbus (Short I/O Access) A16 : D32, D24, D16, D8
FC00 0000 : FCFE FFFF	VMEbus (Standard Access) A24 : D16, D8
FCFF 0000 : FCFF FFFF	VMEbus (Short I/O Access) A16 : D16, D8
FD00 0000 : FDFF FFFF	VMXbus A24 : D32, D24, D16, D8
FE00 0000 : FEFF FFFF	VMXbus A24 : D16, D8
FF00 0000 : FF03 FFFF	EPROM Area 1
FF04 0000 : FF07 FFFF	EPROM Area 2
FF80 0000 : FFFF FFFF	On-board I/O Devices

3.Ø The Controller Boards

The controller board which is installed in the MINIFORCE-2P/32 series standard configuration is the SYS68K/WFC-1 board. Additional controller boards for application purposes are described in section 6.

3.1 The WFC-1 Setup

Winchester/Floppy controller card with the DATA REQ interrupt disabled and the COMPLETE interrupt enabled. We set the INTRQ interrupt to level 3, and the DRQ to level 4.

WFC-1 Jumper Configuration for PDOS*:

Name	**Connect**	**Default**	**Definition**
BR8	2-3	No	Enable A31 Comparison
BR2Ø	1-2	No	Enable COMPLETE ints
BR21	1-2	Yes	Disable DATA REQ ints
BR16	4-7,5-9	Yes	DRQ=IRQ4, INTRQ=IRQ3
BR18	3-4	Yes	INTRQ to level 3
BR19	1-6,2-5	Yes	DRQ to level 4
BR1Ø	7-1Ø	No	Base Address to FCBØ1ØØØ
BR1Ø	8-9	No	Base Address to FCBØ1ØØØ

Most importantly, be sure that the R/L switch on the WFC-1 card is in the DOWN or RUN position, and that the green `RUN' LED is lit and the red `Local' is not illuminated.

SYSTEM EXPANSIONS

TABLE OF CONTENTS

1.Ø General Information

Each of the described standard configurations can be expanded via additional VMEbus based boards (memory, controller) for memory expansion or for dedicated applications.

This section describes the possibilities of the system expansion via VMEbus based boards available by FORCE COMPUTERS.

2.0 Memory Expansion

The miniFORCE 2P/32 systems contain up to 512Kbyte memory by default.

To extend the memory capacity, additional 32 bit wide memory boards are required.

This expansion is possible via the SRAM-22 (512Kbyte) board, via the DRAM-E3xx or via other static RAMboards available by FORCE COMPUTERS.

The following table describes the base address which must be selected for the miniFORCE 2P/32 systems.

System	Base Address	Applicable VME Boards
miniFORCE 2P21	$080000 $100000 if an additional SRAM-22 is installed locally	32-bit VMEbus based RAM boards (DRAM-6,DRAM-7, DRAM-EXMX)
miniFORCE 2P21A	$080000 $100000 if an additional SRAM-22 is installed locally	32-bit VMEbus based RAM boards (DRAM-6, DRAM-7, DRAM-EXMX)
miniFORCE 2P21S miniFORCE 2P21S	$080000 $1000000 if an additional SRAM-22 is installed locally	32-bit VMEbus based RAM boards (DRAM-6, DRAM-7, DRAM-EXMX)

The detailed description of the jumper fields for the additional memory board can be found in the hardware user's manual of the particular board.

3.Ø The I/O Controller

The miniFORCE 2P/32 systems contain up to six free VMEbus slots. These free VMEbus slots can be used for additional controllers to increase the number of the RS232 channels or to add dedicated application controller (for example parallel I/O controller). Some of these boards are supported under PDOS* operating system and therefore the recommended jumper setting must be observed.

3.1 The SYS68K/SIO-1

The AM decoder of the SIO-1 card must be set for "Standard Data Access". To do so , two jumpers have to be inserted in jumperfield BR-3. Figure 3-1 shows the jumperfiled and jumpers which have to be inserted.

The access address of the SIO board is $FCBØØØØØ. If a second SIO board is used, the access address must be set to $FCBØØ2ØØ.

Note that the real base address is $BØØØØØ but is manipulated by the AM-Decoder to be $FCBØØØØ from the CPU-21.

3.2 The SYS68K/ASCU-1/2

This version of the PDOS supports the Real Time Clock of the ASCU-1/2. The ASCU must be inserted in slot 1 of the VMEbus backplane for the arbiter to function correctly.

If an ASCU is used the arbiter on the CPU-21 processor board must be disabled. To do so, jumper B21 must be reused. Please refer to the CPU=21 User's Manual for details.

In addition the SYSCLK driver on the CPU-2Ø/21 has to be disabled by removing jumper B31.

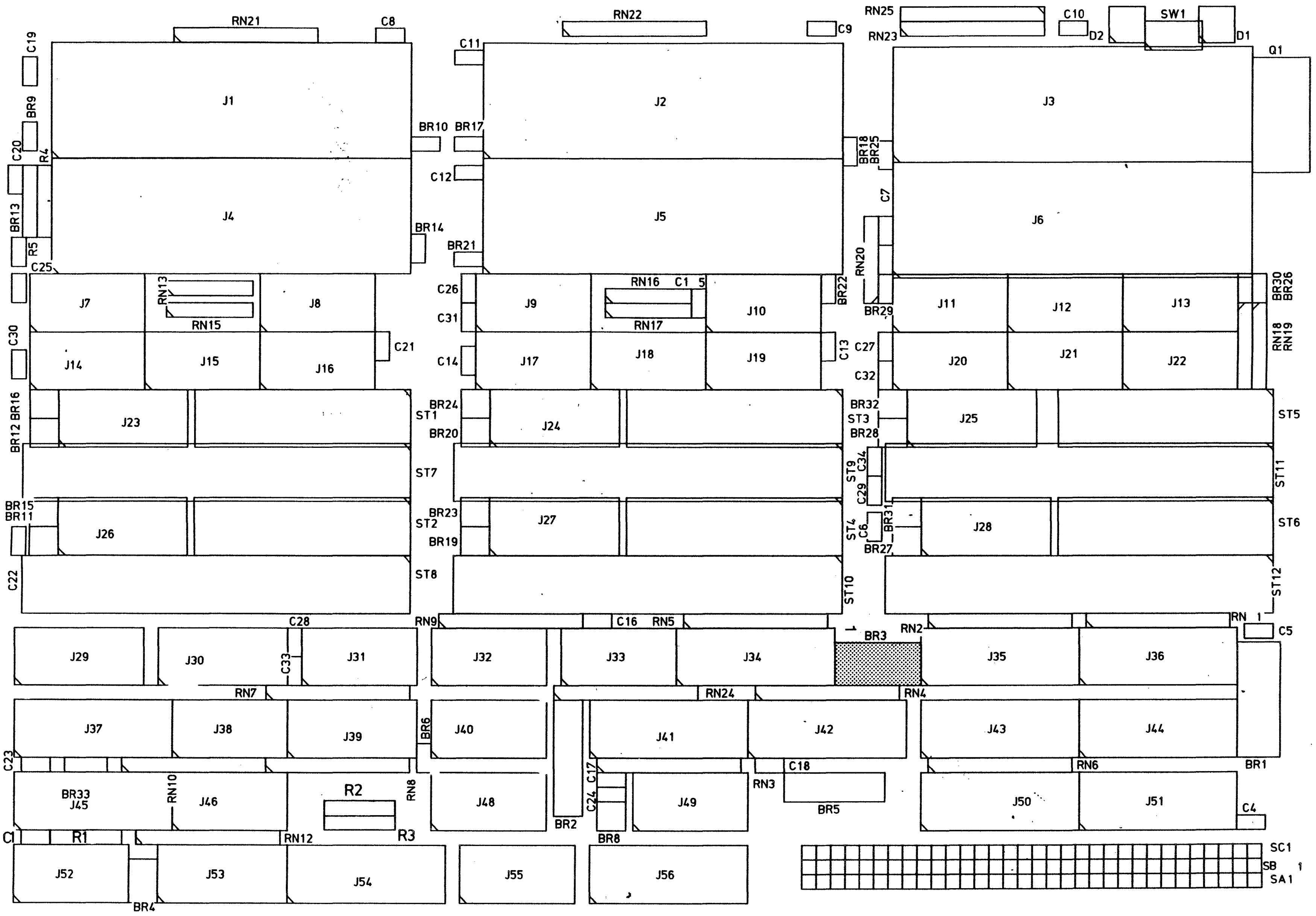
The following jumpers on the ASCU-1/2 must be changed:

Jumper	Pins	Conditions
B7	1-8	inserted
B7	2-7	inserted
B8	2-3	removed

3.3 The SYS68K/ISIO-1

The intelligent serial I/O bard called SYS68K/ISIO-1 supports eight serial I/O channels. It contains a dual ported RAM on the VMEbus for commands and data transfers between the main CPU and the local 68ØØØ processor. The default base address of the board is $FC98ØØØØ. To install this board in a miniFORCE 2P/32, system, no jumper changes are required. PDOS* supports the board by default, a driver and the standard set ups are installed in the operating system.

Figure 3-1: Jumper Location Diagram of the AM Decoder Jumperfield on SIO-1



3.4 Dedicated I/O Controller

FORCE COMPUTERS offers additional I/O controllers which can be implemented in the system for dedicated applications. The installation of the boards under the PDOS* operating system is easy because no drivers are required. The controller boards can be accessed directly via application programs. The base access address and the generated interrupts of the controller must be jumpered so that no base address and no interrupt collisions are possible on the VMEbus.

For more information about the jumper setting and about the interrupt levels and vectors see the Hardware User's Manual of the controller board.

OPERATING SYSTEM INSTALLATION

TABLE OF CONTENTS

INTRODUCTION

This guide describes the initial booting, the installation of the PDOS on the miniFORCE series systems, and how to use and customize the system to fit your needs. The system is fixed configured and few files are supplied with PDOS which are written specifically for this system. The guide is divided into five chapters.

The first chapter, "Initial Operation", contains a step-by-step procedure of booting PDOS after power-on . The second chapter comprises the "System Management Guide", which describes how to format both floppy disks and Winchester drives. The format utility examples are included, along with an explanation of Winchester disk numbering, bad track mapping and partitioning. Also included are the hows and whys of customizing PDOS for your system and making bootable floppy or Winchester disks. Autostart possibilities are also considered for turnkey installations.

The next chapter contains a step-by-step procedure of installing PDOS floppies on the Winchester. After that, a power-up and boot from floppy procedure is described along with sample screen output and trouble shooting suggestions.

Chapter four describes the FORCE specific utilities for the miniFORCE 2P/32 series systems.

The final chapter, "Hardware Implementation", expands the BIOS section of the PDOS Reference Manual with specific information on the miniFORCE 2P/32 systems. The BIOS startup code, the system interrupt structure, and the clock operations are described, along with the usual DISK and UART device service routines. An I/O address map and the boot ROMs are next. Finally, some hints on attaching terminals or serial printers are given along with the commands under PDOS to get output to them.

Since this INSTALLATION GUIDE is very important to first time users of PDOS or FORCE Computers, please report to FORCE any errors, inconsistencies, or ambiguities as soon as possible.

1.0 INITIAL OPERATION

This section describes for the first time PDOS user a step-by-step procedure for initially booting PDOS on the miniFORCE 2P/32 series systems. The user should consult the PDOS Reference Manual and Section 2 of this guide for a more comprehensive look at each utility.

The version 3.0 release of PDOS is meant to run in a single processor system using the equivalent FORCE CPU board.

1.1 SOFTWARE CONFIGURATION

The received system operates with the operating system PDOS. PDOS is already installed on the Winchester disk partition two and three. The system can boot from both disk partitions. All utilities including the assembler are installed on the Winchester. Therefore the system can be booted directly from the Winchester.

1.2 LICENSE AGREEMENT

Sign the PDOS license agreements. You are to keep one copy and immediately send the other copy to FORCE. This agreement licenses your use of PDOS to one machine.

1.3 HARDWARE CONFIGURATION

In order for your system to boot PDOS, the hardware is configured to support PDOS. Therefore no jumpers have to be changed. The default jumper settings for the basic configuration are described in section 5.

1.4 THE BOOTING PDOS FROM WINCHESTER

Attach a 9600 BAUD terminal the terminal port, apply power, activate the RESET switch and start the PDOS bootstrap program. The PDOS Bootstrap program can be started as follows:

> Type "BOOT" when the FORCEbug prompt ("?") appears on the screen.

The bootstrap first initializes the disk controller in the system. It then starts to boot, first from the floppy, and next from the hard disk. For example:

```
FORCE SYS68K/CPU-x PDOS Bootstrap Revision x.x   Date
     WFC-1 installed.
```

If the bootstrap program cannot find a disk controller, the following will be displayed:

```
  FORCE SYS68K/CPU-x PDOS Bootstrap Revision x.x   Date
       No Disk Controller Installed ???
```

If you see the previous message and yet you know that there is a disk controller board installed, please check the address jumpers on the disk card and then test that the card is really accessible under FORCEbug.

After printing the banner and initializing the disk controller, the bootstrap program begins reading the header sectors of disks, number Ø through 99, for PDOS boot information. You should see some messages printed out that indicate disk errors on disks Ø and 1, since the Winchester drive gets up to speed or you insert a floppy disk.

```
  Disk #Ø:  Header sector error = 116
  Disk #1:  Header sector error = 116
  Out of PDOS boot disk table entries.
  I'll retry them all.
```

After speed up of the Winchester, the bootstrap should find the PDOS boot on disk #2 and display:

```
Disk #2:  FORCE CPU-x PDOS was found  >>  Sector xxxx,  Addr xxxx
           Execute it  ? _
```

You do not need to respond ot the question, simply wait for a moment and see the result:

```
Disk #2:  FORCE CPU-x PDOS was found  >>  Sector xxx, Addr xxx
           Execute it  ? _  Yes.  Booting...SUCCESS!
```

After "Booting...SUCCESS! is displayed, the system will enter an initialization routine. The name of this initialization program is "SY$STRT". It initialises the available serial I/O channels, prints a welcome message and then requests the input of date and time. The time should be entered in a 24 hour format. PDOS is now running on you system.

If at any time your system "crashes", push the reset button located on the front panel of the CPU and restart the PDOS boot program. You will then be prompted with the same boot message as above. Proceed in the same manner as a cold startup.

2.Ø SYSTEM MANAGEMENT GUIDE

Your PDOS system contains a collection of files and utilities to assist you in your development work. The purpose of the "System Management Guide" is to acquaint you with those files and utilities you will need to use in operating your system. Not all utilities are discussed in this section; for further information and additional utilities you should use the **PDOS Reference Manual.**

2.1 DISK PREPARATION OVERVIEW

Since the FORCE PDOS has adopted the PDOS Winchester disk standard, you can use many different disk controller and drive combinations without changing the operating system code at all. PDOS finds what controllers are in the system, what Winchester drives are attached, and what drive information is written on each drive. It then sets up tables for the file manager to communicate with the drives. Before you can use the Winchester drives correctly, you need to prepare them by marking bad tracks, defining drive parameters and partitions, formatting all the tracks on the drive, writing PDOS Winchester information onto the drive, and finally initializing each partition with PDOS file system and, if desired, a PDOS boot.

This section tells you how to prepare Winchester drives for use with PDOS. It also gives a brief explanation of the disk format utility that deals with bad track mapping, disk parameters, PDOS disk partitions and disk numbering.

2.2 PREPARING FLOPPY DISKETTES FOR USE WITH PDOS

Floppy Disk preparation consists of two parts. The first part formats the floppy with the sector ID and data fields required by the controller to access the diskette. The second part is the installation of the PDOS directory structure onto the floppy diskette.

The standard FORCE system requires diskettes that are:

5 1/4"
Double sided
Double density
96 tpi (8Ø tracks are used)

Diskettes should match these specifications or you may encounter problems with reading the disks.

2.2.1 Hardware Formatting

Hardware formatting is done by running the FFRMT utility that is on your boot disk.

Note: FFRMT should only be run when no other tasks are running.

```
>FFRMT
68K PDOS FORCE Format Drive Utility  Ø8-JAN-87
  Possible Disk Controllers in this System are:
    Controller #1 is a FORCE SASI-1
    Controller #2 is a FORCE WFC-1
    Controller #3 is a FORCE ISCSI-1
  Drives that are currently defined in system are:
    FØ is controller #3, drive select byte $73
    F1 is controller #3, drive select byte $74
    WØ is controller #3, drive select byte $ØØ
    W1 is undefined.
    W2 is undefined.
    W3 is undefined.
  Select Menu:  W,WØ,W1,W2,W3=Winch; F,FØ,F1=Floppy; Q=Quit
       Select Drive:
```

You select the drive to format. The floppy disk is F or FØ. If you have two floppy drives, the second drive is F1. We will assume that all formatting will be to the `F' or first floppy drive. To select floppy drive Ø enter an F followed by a [CR].

The floppy Ø drive menu is now displayed. Most of the options on the menu only apply to Winchester disks and should be ignored while formatting floppies.

```
  FØ Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command: _
```

The option for formatting diskettes is selected by typing a `3' followed by a [CR].

```
       Command: 3[CR]
```

At this point you should insert the floppy you wish to format into the disk drive. Remember, formatting will destroy all data on the diskette so double check that the disk in the drive is the disk you wish to format.

You are now prompted for the `Sector Interleave' of the format.

Sector interleaving allows you to map the sequence of the logical sectors onto non-adjacent physical sectors. The optimal sector interleave is a factor of many actions that may be occurring on the system. Such actions could include the number of tasks running, number of tasks accessing the disks, etc. You can use a disk of any interleave on your system, but the proper interleave will give you a better performance. FFRMT will display a good interleave for most applications.

If you wish to optimize the performance, you may want to experiment by formatting your disks at different interleaves. To use the default interleave just type a [CR]. To enter a different interleave, type the new number followed by a [CR].

```
  Sector Interleave = 1[CR]
```

You are next prompted for the physical tracks to format. The default display is every track on the diskette or a total of 160 tracks (80 tracks per side). Track 0 is the outermost track. Normally you will just type a [CR] to format the entire diskette.

```
  Physical Tracks to Format = 0,159[CR]
```

FFRMT waits for confirmation to begin formatting. A `Y' followed by a [CR] will begin the format process.

An [ESC] will return you to the `Floppy Drive 0 Menu' ending the format option. Any other response will restart the Format option with the Sector interleave question.

WARNING: **FORMAT WILL DESTROY ALL DATA ON YOUR DISKETTE. If you do begin formatting and wish to abort you may type a [CTRL-C].**

Type a `Y[CR]' to begin the format process.

The Sector Interleave Table will be displayed along with the current track formatting. The sector interleave table is meaningless on the ISCSI-1 since FFRMT just passes the interleave value on to the controller. However, the WFC controller will display the logical to physical sector map. During the formatting, the floppy disk light should be on. If you are using a WFC-1 the track numbers will be printed on the screen.

```
  Sector Interleave Table:
1,9,4,12,7,15,2,10,5,13,8,16,3,11,6,14

  Issuing Format Drive Command

  FORMAT Successful!
```

You will then be returned to the `Floppy Drive 0 Menu'.

Errors may occur when formatting disks, the most common error is that the drive is not ready. This may indicate that the door is open or the floppy has been placed into the drive incorrectly.

We recommend that you verify the diskettes after format. This is done by selecting option 4 `Verify tracks' by typing a `4[CR]'.

FFRMT will display the default tracks which will be the same as you selected under the format option. If the track number is correct, you may enter a [CR] or if you wish, enter two new numbers separated by a comma and ended with a [CR].

You will next be prompted with:

```
  Ready to VERIFY Floppy Drive Ø ?
```

A `Y[CR]' response will begin verifying the diskette reading each track. Any other response will return you to the Floppy Drive Ø Menu. If an error occurs, FFRMT will ask you if you wish to add the track to the bad track table. Since there are no bad track tables for floppies, you should either respond with `N[CR]' which will proceed to the next track or a `R[CR]' which will retry the track.

If it cannot verify after a couple of retries, you should abort with the [ESC] key and verify another diskette which you have formatted. If the problem is resolved, then the diskette was probably bad and should not be used. If the problem persists, then you may be using an incorrect diskette type (4Ø track or single density) or you may have a hardware problem with your drive or controller.

After you have successfully formatted and verified your diskettes you are now ready to install a PDOS directory.

2.2.2 Installation of PDOS Directory Structure

The installation of the PDOS Directory structure may only be done on diskettes that have been hardware formatted. This installation is the second process of the preparation of diskettes. This process will also destroy any data that may have been on the diskette so care should be taken.

PDOS diskettes may be initialized in one of two ways. The first way is to backup a current PDOS disk onto the new diskettes. This procedure is done by using the MBACK utility.

MBACK performs a sector to sector transfer. The new disk will look just like the source disk when the backup is completed. Disk name, sector bit map, directory entries, files, and boot information are all copied. MBACK is fully described in the **PDOS Reference Manual.**

The second method is to set up a NEW PDOS diskette. This method is done with the MINIT utility which is described in more detail in the **PDOS Reference Manual.**

The initialization process is started by running the utility MINIT.

```
>MINIT[CR]
68K PDOS Disk Initialize Utility Ø7/29/85
  Disk # =
```

You are first prompted for the PDOS disk number to initialize. The floppy on a standard FORCE system is disk number Ø. Normally MINIT will write to each sector on the disk to verify the media for data use unless you specify the `/Q' option. With the `/Q' option, MINIT will only write out to every 32nd sector. This is a reliable and a much faster method, provided you did the verify option during hardware formatting.

```
  Disk # = Ø/Q[CR]
```

After the disk number has been entered, you are asked for the number of sides and the density of the diskette. Standard PDOS on the FORCE systems only supports double sided double density diskettes. A [CR] should be entered to retain those two defaults.

You will next be asked for the number of directory entries for the disk. This is the maximum number of unique files that may be created on the disk. Each directory entry requires 32 bytes of storage allowing 8 directory entries per disk sector. It is recommended that you make this number a multiple of 8, since the directory always ends on a sector boundary. You should realise that every eight directory entries will reduce your file storage area by 1 sector.

In addition, you will not be able to create more files on the disk than this maximum without re-running MINIT which will also destroy all data on the disk. The default for floppies is 128. You may enter in your new number followed by a [CR] or just a [CR] if you wish to use the default.

```
  Maximum Directory Size = 128[CR]
```

The total number of PDOS sectors will be asked. The default is 2336 which is the correct number for a bootable diskette. If you only want to use the diskette for data, then you can enter 2528 which is the maximum storage on the diskette. (Note: the total number of floppy sectors is 256Ø but under the PDOS disk standard, track Ø is always reserved for hardware dependent formats. 256Ø - 32 sectors per track is 2528. You may enter any number of sectors up to 2528. If you enter a larger number you will get disk errors as PDOS attempts to access non-existent sectors. If you enter a smaller number, it means that those sectors will not be available for PDOS file allocation.

```
  Total Number of PDOS Sectors = 2336 [CR]
```

A disk name is next asked for. The disk name is any string with up to 16 characters. The disk name is not used by PDOS in any command function, but it is displayed whenever a file listing of the disk is requested by the LS command or is output by several utilities such as MASM on list files. The name can be changed at any time without destroying the data on the disk by the MDNAME utility.

After the disk name has been entered, a summary of the initialization parameters will be displayed along with a prompt to proceed with the initialization process.

```
INIT:   Disk #Ø
        Double Sided
        Double Density
        128 Files
        2336 Sectors
DESTROY DISK? Y[CR]
```

If you wish to proceed with the initialization, type a `Y[CR]'. To return to the PDOS monitor type an [ESC]. Any other response will return you to the `Disk #' prompt and allow you to re-enter the parameters.

WARNING: **MINIT WILL DESTROY ALL DATA ON YOUR DISKETTE. Reconfirm the disk numbers to make sure that you are initializing the disk number you want to. A [CTRL-C] during the initialization of the diskette will abort further initialization.**

After the destroy disk confirmation is given, the initialization will display the current sector number. The header sector information will be written out to the disk. The message "INITIALIZATION Successful!" will be displayed.

The diskette is now ready to be used with the PDOS system.

2.3 PREPARING WINCHESTER DRIVES FOR USE WITH PDOS

CAUTION: Your FORCE PDOS System comes with PDOS already installed on the Winchester on Disk #2. When formatting the Winchester or changing the partitions, the installed system will be destroyed.

Winchester Drive preparation consists of two parts. The first part is the hardware preparation. Hardware preparation is the format information that the controllers need to access the disk. The second part is the installation of the PDOS directory structure onto the disk. Winchester disk preparation is very similar to the floppy diskette preparation except that PDOS allows you to handle bad tracks and to define several logical disks, or directories, to reside on a single physical drive.

The standard FORCE system will support either a ISCSI-1 or a WFC controller from FORCE. Through the PDOS Winchester disk standard, PDOS will also support various 5 1/4" drives up to the maximum heads and cylinders handled by the controller.

2.3.1 Hardware Formatting

First let us outline the procedure for hardware format. This is done by running the FFRMT utility that is on your PDOS FORCE boot disk. A complete description of the FFRMT utility is found in Appendix B.

NOTE: FFRMT should only be run when no other tasks are running.

```
>FFRMT
68K PDOS FORCE Format Drive Utility Ø8-Jan-87
  Possible Disk Controllers in this System are:
       Controller #1 is a FORCE SASI-1
       Controller #2 is a FORCE WFC-1
       Controller #3 is a FORCE ISCSI-1
  Drives that are currently defined in system are:
       FØ is controller #3, drive select byte $73
       F1 is controller #3, drive select byte $74
       WØ is controller #3, drive select byte $ØØ
       W1 is undefined.
       W2 is undefined.
       W3 is undefined.
  Select Menu:  W,WØ,W1,W2,W3=Winch; F,FØ,F1=Floppy; Q=Quit
       Select Drive:
```

PDOS supports up to 3 Winchester drives and 2 floppy drives.

WØ will be the first Winchester in a standard system regardless of the controller being used. If all three Winchesters are undefined, then the Winchester may not have been up to speed when PDOS booted from your floppy. Just reboot and rerun FFRMT. If you still have problems then check if your controller is on-line or that it has been installed correctly.

To select your disk, enter W or WØ followed by a [CR].

If the controller is not defined, you will need to define it by selecting WØ and answering the additional questions for the controller number and drive select number.

The "WØ Main Menu" is now displayed. To set up the disk, you first select option 1 and then sequence through to option 6. If you wish to log your work, you may do so by using the `P' option, which will toggle output to your unit 2. All entries are terminated with a [CR].

```
  WØ Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:
```

Option 1 is used to reconfigure your drive. That is to display and/or change the number of heads, cylinders, shipping cylinder, etc. If you are using any other than the standard Winchester disk, then you need to select option 1. The following gives an example of the default parameters which apply to the Micropolis 1375 SCSI Winchester disk. Please note that the parameters of the SCSI Winchesters must match the total number of blocks, but not the physics of the drive, because the interface is sector oriented. Please refer to Section 4 of this Manual for a detailed description of the physics of your Winchester.

```
Current Winch Drive Ø Parameters:
               # of Heads = 16      Number of heads on drive
           # of Cylinders = 1ØØØ    Number of cylinders on drive
Physical Blocks per Track = 32      Number of 256 sectors per track
 Physical Bytes per Block = 256     Number in sector
         Shipping Cylinder = Ø      Not needed
                Step rate = Ø       Not used
Reduced write current cyl = Ø       Not used
  Write Precompensate cyl = Ø       Not used
```

If you wish to change any of the defaults, enter `A' followed by a carriage return and then you will be prompted for them one at a time. The parameters most often changed are heads and cylinders which vary according to the size of the disk. This information is included in the hardware drive specifications. Enter a [CR] to preserve the default information or type in the new number followed by a [CR] to change the information.

Note: The drive parameters have to be changed only when you want to use another Winchester type.

Option 2 will allow you to enter the bad tracks. This means you can logically map out tracks that are unsuitable for data storage. Most disks will have some media defects when manufactured. The bad track map can be found on the disk drive or with the documentation provided with your system. This option should not be used on SCSI Winchester drives as they map out the bad blocks internally.

If you do not know the bad tracks for your drive, just format and verify your drive, adding bad tracks if they are found.

PDOS maps logical tracks to physical tracks skipping those physical tracks that are marked as bad. You should complete the bad track table before you partition and initialize the drive. If you add a bad track, PDOS remaps all subsequent tracks, resulting in a loss of data.

When the Bad Track option is selected, enter `D' to display the current bad track list:

```
  WØ Bad Tracks Menu:  Bad Track, D)isplay, C)lear, H)elp, Q)uit
       Command: D[CR]

  Current Winch Drive Ø Bad Tracks:
231 613 697 7ØØ 7Ø3
```

To display instructions on entering bad tracks enter `H' followed by a [CR]. The following help menu is displayed and you are returned to the Bad Track menu.

```
  To add a bad track, enter either {track} or {Head,Cylinder}. To
  delete a bad track from the table, precede the track number with
  a minus sign. Enter [ESC] or `Q' to exit.

  Current Winch Drive Ø Bad Tracks:
231 613 697 7ØØ 7Ø3

  WØ Bad Tracks Menu: Bad Track, D)isplay, C)lear, H)elp, Q)uit
       Command: _
```

Enter the bad track numbers by typing in the number of the track. For example, typing `7Ø1' enters track 7Ø1 into the table. `3,5Ø' enters the track on head 3, cylinder 5Ø into the table. When you have completed entering the bad tracks in, type a `Q' to return to the "Winchester Drive Ø Main Menu."

Option 3 is the format option. Remember, formatting will destroy all data on the drive so double check that you are formatting the correct drive number.

You are now prompted for the `Sector Interleave' of the format.

Sector interleaving allows us to map the sequence of logical sectors onto non-adjacent physical sectors. The optimal sector interleave is a factor of many actions that may be occurring on the system such as the number of tasks running, number of tasks accessing the disks, etc. You can use a disk of any interleave on your system, but the proper interleave will give you better performance. FFRMT will display a good interleave for most applications. If you wish to optimize the performance, you may want to experiment by formatting your disks at different interleaves. Note that the interleave of SCSI Winchester drives cannot be changed.

To use the default format you may just type a [CR]. To enter a different interleave, type the new number followed by a [CR].

You are next prompted for the physical tracks to format. The default display are all tracks on the drive. The default number is calculated from the number of heads and cylinders you specified in option 1 - Display/Alter Drive parameters. Track 0 is the outer-most track. Normally just type a [CR] to format the entire drive.

```
 Sector Interleave = 1                       (just hit [CR])
 Physical Tracks to Format = 0,16000         (just hit [CR])
 Ready to FORMAT Winchester Drive 0 ?
```

FFRMT waits for confirmation whether to begin formatting or not. A `Y' followed by a [CR] will begin the format process.

An [ESC] will return you to the `Winchester Drive 0 Menu' ending the Format option. Any other response will restart the Format option with the Sector interleave question.

WARNING: **FORMAT WILL DESTROY ALL DATA ON YOUR DRIVE. If you do begin formatting and wish to abort you may type a [CTRL-C].**

The Sector Interleave Table will be displayed along with the current track formatting. The sector interleave table is meaningless on the ISCSI-1, and no track numbers are displayed. The WFC controller however will display the logical to physical sector map. During the formatting, the drive select light should be on and the track numbers will be printed on the screen.

```
  Sector Interleave Table:
1,9,4,12,7,15,2,10,5,13,8,16,3,11,6,14
  Issuing Format Drive Command.
```

When all tracks are formatted, the following display will be shown on your console:

```
  FORMAT Successful!
```

You will then be returned to the `Winchester Drive Menu'. Errors rarely occur when formatting drives.

Option 4 will verify the format by reading each track on the drive. This will allow you to detect any bad tracks that are unsuitable for data storage. We recommend that you verify the drive after format. This is done by selecting option 4 `Verify tracks'.

FFRMT will display the default tracks which will be the same as you selected under the format option. If the track numbers are correct, you may type a [CR] or if you wish, enter two new numbers separated by a comma and ended with a [CR]. You will next be prompted with:

```
  Ready to VERIFY Winchester Drive Ø ?
```

A `Y[CR]' response will begin verifying the drive reading each track. Any other response will return you to the Winchester Drive Ø Menu.

If an error occurs, FFRMT will ask you if you wish to add the track to the bad track table. This is done by typing a `Y' followed by a [CR].

WARNING: **You will change the logical track map of the drive any time you enter a bad track which may require reinitializing the PDOS file structure. You should only enter bad tracks on a drive with PDOS file structure after you backup all data on the drive. Otherwise verify may be run without affecting the data on the disk.**

If you do not want to add the track into the bad track table, type a `N' followed by a [CR]. `R[CR]' will retry the track. We recommend that if at first you cannot verify the track and the error number is "bad data on read" error, that you enter the track as bad. If many of your tracks are bad then you should verify the installation of the drive.

The message `VERIFY Done!' will be displayed when the verify option is completed, and you will be returned to the Winchester Drive Menu.

Select Option 5 next to setup your logical PDOS disk partitions on your Winchester. You have the option of defining large partitions and floppy sized partitions. PDOS does limit you to 16MB per partition, so most Winchester drives will need to be defined into appropriately sized partitions.

The floppy partitions are about 660KB each and correspond to the PDOS standard floppy disk size. These partitions are allocated from the end of the Winchester tracks. After all floppy partitions have been allocated, the remaining tracks are allocated evenly between the number of large partitions defined. Since the floppy disk drives are 0 and 1 on standard PDOS, the default starting partition or disk number is 2. The following diagram shows how four large disk partitions and two floppy sized partition would be allocated on the tracks.

```
track 0 ..............................................track n

large disk|large disk|large disk|large disk|FloppyDisk|FloppyDisk
   #2     |    #3    |    #4    |    #5    |    #6    |    #7
```

When this option is selected you will see the following menu appear on your screen. There may be some variation due to the number of bad tracks and size of your drive.

```
  Current Winch Drive 0 Disk Partitions:
     # of Large partitions = 10
    # of Floppy partitions = 12
First track for PDOS Parts = 0
 Last track for PDOS Parts = 15979
         First PDOS disk # = 2
 Total # of Logical Tracks = 16000
 Disk #    Logical Trks  Physical Trks  PDOS sectors
           Base,Top      Base,Top       Total/{boot}
  2        0,1499        0,1500         47968/47776
  3        1500,2999     1501,3000      47968/47776
  4        3000,4499     3001,4500      47968/47776
  5        4500,5999     4501,6000      47968/47776
  6        6000,7499     6001,7500      47968/47776
  7        7500,8999     7501,9000      47968/47776
  9        9000,10499    9001,10500     47968/47776
 10        10500,11999   10501,12000    47968/47776
 11        12000,13499   12001,13500    47968/47776
 12        13500,14999   13501,15000    47968/47776
 13        15000,15079   15001,15080    2528/2336
 14        15080,15159   15081,15160    2528/2336
 15        15160,15239   15161,15240    2528/2336
 16        15240,15319   15241,15320    2528/2336
 17        15320,15399   15321,15400    2528/2336
 18        15400,15479   15401,15481    2528/2336
 19        15480,15559   15482,15564    2528/2336
 20        15560,15639   15565,15647    2528/2336
 21        15640,15719   15648,15730    2528/2336
 22        15720,15799   15731,15813    2528/2336
 23        15800,15879   15814,15896    2528/2336
 24        15880,15959   15897,15979    2528/2336
  W0 Partitions menu: A)lter, D)isplay, R)ecalc, Q)uit
       Command:  Q
```

The standard ISCSI-1 system will assume ten large partitions and 12 floppy sized partitions. The number of partitions and types you want will depend on your needs. You may find it helpful to create mostly floppy sized partitions since they easily backup to floppy diskettes with MBACK, a sector to sector image backup utility. You should also keep one floppy partition as a temporary disk partition to copy diskettes. This is because it is much faster to do a transfer of files to a floppy Winchester area followed by a sector for sector backup to a diskette with MBACK, than it is to transfer files directly to a floppy diskette with MTRANS or the monitor command TF.

The first large partition (defaults to disk 2) is generally referred to as the system disk. This partition should contain all the system files that have been distributed with PDOS along with other utilities and languages. You may use any disk for your system disk, but the installation guide will treat disk 2 as the system disk. The installation guide will also treat floppy partition 20 as the temporary disk partition.

To set up your partitions, type an `A' followed by a [CR]. You will be prompted for the number of large partitions:

```
# of Large partitions = 1Ø
```

You may type a [CR] to use the default or enter in a new number followed by a [CR]. You are next prompted for the number of floppy sized partitions.

```
# of FLoppy Partitions = 12
```

You may type a [CR] to use the default or enter in a new number followed by a [CR]. You are next prompted for the first track to use for PDOS followed by the last track for PDOS. You should respond to both with a [CR] unless you wish to reserve some tracks for non-PDOS use.

```
  First track for PDOS Parts = Ø
  Last track for PDOS Parts  = 15979
```

By default the last track for PDOS parts is the maximum number of tracks on the Winchester minus the number of bad tracks (which is Ø in the above example). The format utility automatically subtracts the bad tracks from the maximum number of tracks. Note that bad tracks are mapped automatically on the SCSI drives.

Next you are asked for the starting PDOS disk number for the first partition on the drive. Normally, the Floppy diskette drives are 0 and 1, the RAM disk is 8, and the Winchester disk partitions start at 2. FFRMT will assign disk numbers from the starting number on up skipping 8. This number should be in the range of 0 to 99. You should select the number for starting the Winchester disk partitions that is low enough to define all the partition numbers to be under 99.

```
  First PDOS disk # = 2
```

Enter your number or just type [CR] for the default. You will then have the Winchester partitions displayed.

```
  Current Winch Drive 0 Disk Partitions:
     # of Large Partitions = 10
    # of Floppy Partitions = 12
First track for PDOS Parts = 0
 Last track for PDOS Parts = 15979
         First PDOS disk # = 2
 Total # of Logical Tracks = 16000
 Disk #    Logical Trks  Physical Trks   PDOS sectors
           Base,Top      Base,Top        Total/{boot}
  2        0,1499        0,1500          47968/47776
  3        1500,2999     1501,3000       47968/47776
  4        3000,4499     3001,4500       47968/47776
  5        4500,5999     4501,6000       47968/47776
  6        6000,7499     6001,7500       47968/47776
  7        7500,8999     7501,9000       47968/47776
  9        9000,10499    9001,10500      47968/47776
 10        10500,11999   10501,12000     47968/47776
 11        12000,13499   12001,13500     47968/47776
 12        13500,14999   13501,15000     47968/47776
 13        15000,15079   15001,15080     2528/2336
 14        15080,15159   15081,15160     2528/2336
 15        15160,15239   15161,15240     2528/2336
 16        15240,15319   15241,15320     2528/2336
 17        15320,15399   15321,15400     2528/2336
 18        15400,15479   15401,15481     2528/2336
 19        15480,15559   15482,15564     2528/2336
 20        15560,15639   15565,15647     2528/2336
 21        15640,15719   15648,15730     2528/2336
 22        15720,15799   15731,15813     2528/2336
 23        15800,15879   15814,15896     2528/2336
 24        15880,15959   15897,15979     2528/2336
 W0 Partitions Menu:  A)lter, D)isplay, R)ecalc, Q)uit
     Command:  Q
```

You may notice that the `Total # of Logical Tracks' is greater than the `Last track for PDOS parts'. This is because of an odd number of tracks that could not be allocated evenly between the large partitions. You may also notice that the logical track numbers may not exactly correspond to the physical tracks.

Disk #	Logical Trks	Physical Trks	PDOS sectors
	Base,Top	Base,Top	Total/{boot}
2	Ø,1499	Ø,15ØØ	47968/47776

This is because of bad tracks that may have been mapped out. The final information that is important to note is the PDOS Sector column. The number to the left is the number of sectors for the partition if you want it to be a data only disk. The number to the right is the number of sectors for the partition if you wish to install a boot on the partition. This information is needed when setting up the PDOS directory structure on the partition with MINIT. In the preceding example, we have 47968 sectors for data only or 47776 for a bootable partition.

You should now exit with the `Q' option back to the Winchester Drive Menu.

The drive parameters, bad track list and disk partition information must now be saved onto the Winchester drive so that future accesses will have the information. Select "Write out Header info to disk". When you select option 6 you will be prompted with:

```
  WØ Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  6[CR]
  Write to Disk Y)es, N)o, F)ile : F[CR]
  Write Parameter File: #TEMP/Ø[CR]
  WØ Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  Q[CR]
  Exit to Select Drive.  Update Param RAM (Y/N) ? Y[CR]
  Select Menu: W,WØ,W1,W2,W3=Winch; F,FØ,F1=Floppy; Q=Quit
       Select Drive:  Q[CR]
```

A `Y' followed by a [CR] will update the drive by writing the tables out. An `F' followed by a [CR] will write the tables out to a file of your choice. Any other response will return you to the Winchester Drive Menu.

Now that you have set up your Winchester drive, you can exit back to the Drive Selection Menu. This is done by selecting option Q. As part of the Exit procedure you will also be asked if you want to update the current BIOS tables for the Drives with the parameters you have set up.

Answer with a `Y' followed by a [CR] to update the current BIOS tables. The BIOS table must always be updated if you have made any changes. An `N' will not update the BIOS tables.

You may now proceed to set up additional Winchester drives or type a `Q' for quit followed by a [CR] to exit back to the PDOS monitor.

You are ready to install a PDOS directory onto each partition.

2.3.2 Installation of PDOS Directory Structure on Winchesters

The installation of the PDOS Directory structure may only be done on Winchester partitions that have been hardware formatted and defined as outlined in section 2.3.1. You must initialize each partition before you may use it for PDOS. Before you begin you have to know the disk partition number and number of blocks for the partition as provided by FFRMT under "Disp/Alter Disk Partitions Option." The initialization process is done with the MINIT utility which is described in detail in the **PDOS Reference Manual.**

```
>MINIT
68K PDOS Disk Initialize Utility Ø7/29/85
  Disk # =
```

You are first prompted for the PDOS disk number to initialize. Enter in the number for one of the partitions.

Normally MINIT will write zeros in each sector on the disk to verify the media for data use unless you specify the `/Q' option.

With the `Q' option MINIT will only write out zeros to every 32nd sector. This is a much faster method and is reliable provided you did the verify option during hardware formatting.

```
  Disk # = 2/Q[CR]
```

After the disk number has been entered, you will be asked for the number of sides and the density of the disk. These have no meaning to Winchester partitions. We recommend that you use the default, especially if the disk is going to backup to a floppy disk later. The default is selected by typing a [CR].

You will next be asked for the number of directory entries for the disk. This is the maximum number of unique files that may be created on the disk. Each directory entry requires 32 bytes of storage allowing 8 directory entries per disk sector. It is recommended that you make this number a multiple of 8, since the directory always ends on a sector boundary. You should also keep in mind that every 8 directory entries will reduce your file storage area by 1 sector. You will not be able to create more files on the disk than this maximum without rerunning MINIT which will also destroy all data on the disk. 128 is the default and is suitable for floppy sized partitions. For large partitions, you may want to use 1Ø24. You may enter in your new number followed by a [CR] or a [CR] if you wish to use the default.

```
  Maximum Directory Size = 1Ø24[CR]
```

The total number of PDOS sectors will next be asked. The default is 2336 which is the correct number for a bootable floppy diskette. You will need to enter in either the total number of blocks if you want the disk to be data only, or the boot number of blocks if you are going to make the disk a bootable disk. The number of sectors is the same as the blocks given in the FFRMT program.

Sample table from FFRMT Disp/Alter Disk Partitions:

```
 Disk #    Logical Trks  Physical Trks  PDOS sectors
           Base,Top      Base,Top       Total/{boot}
    2      Ø,1499        Ø,15ØØ         47968/47776
```

You may enter any number up to the total number of sectors. Use the total sector number (i.e. 47968) for a data only disk, or the {boot} number (i.e. 47776) for a bootable disk. If you enter a larger number you will get disk errors as MINIT attempts access to non-existent sectors. If you enter a smaller number, it will mean that those sectors will not be available for PDOS file allocation. For this example we will make disk 2 a boot disk.

```
  Total Number of PDOS Sectors = 47776
```

If you can't remember the total number of PDOS sectors, simply type `MAX[CR]' for a data only disk and `BOOT[CR]' for a boot disk. MINIT will then calculate the correct number of sectors and display them for you as the default and you can then enter a [CR].

```
  Total Number of PDOS Sectors = BOOT[CR]
  Total Number of PDOS Sectors = 47776[CR]
```

You are then asked to name the disk. The disk name is any string up to 16 characters which is not used by PDOS in any command function. However, the disk name is displayed when file listings of the disk are requested by the `LS' monitor command, or it is output by several utilities such as MASM on the list file. The name can be changed by the MDNAME utility at any time without destroying the data on the disk.

```
  Disk Name = My New Disk[CR]
```

After you have entered the disk name, a summary of the initialization parameters and a prompt to confirm the initialization process is displayed.

```
  INIT:    Disk #2
           Double Sided
           Double Density
           1Ø24 Files
           47776 Sectors
  DESTROY DISK?  Y[CR]
```

If you wish to proceed with the initialization, type a `Y[CR]'. To return to PDOS monitor type an [ESC]. Any other response will return you to the `Disk #' prompt and allow you to re-enter the parameters.

WARNING: MINIT WILL DESTROY ALL DATA ON YOUR DISK. Reconfirm the disk parameters to make sure that you are initializing the right disk number. A [CTRL-C] during the initialization of the diskette aborts further initialization.

After the destroy disk confirmation is given, the initialization displays the current sector numbers. These are printed out every 32 sectors, up to the total minus 1. Finally, the header sector information is written out to the disk. The message "INITIALIZATION Successful!" is displayed.

The disk partition is now ready to be used with the PDOS system. You need to repeat the procedure for each partition on the Winchester drive.

2.4 PDOS GENERATION

PDOS provides you with a procedure file to assist you in the generation of the PDOS system. The file is named F32DOS:GEN and is on your boot disk.

F32DOS:GEN will assemble all user-alterable portions of PDOS and then proceed to link a new bootable system file.

The following files must be accessible when running F32DOS:GEN:

MASM20	Rev. 3.2 assembler (or later)
QLINK	The PDOS linker
MSYRAM:SR	PDOS RAM definitions
MBIOS:SR	Generic PDOS BIOS file (used as included file to FxBIOS:SR)
FPARM:SR	Parameter definitions/memory map for absolute locations
F32PARM:SR	System Hardware Addresses
F32BIOS:SR	BIOS file
F32BIOSU:SR	Character I/O (UARTs) service routines - for the CPU-board UARTs
FBIOSU:SR	Character I/O drivers for System UARTs
FBIOSW:SR	Block I/O (disk read/write) routines
MPDOS:OBJ	PDOS system
or	
MPDOSBAS:OBJ	PDOS system with BASIC

F32DOS:GEN will create the following files as output:

MSYRAM:OBJ	Object of MSYRAM:SR
F32BIOS:OBJ	Object if FxBIOS:SR and MBIOS:SR
F32BIOSU:OBJ	Object of FxBIOSU:SR
FBIOSU:OBJ	Object of FBIOSU:SR
FBIOSW:OBJ	Object of FBIOSW:SR
F32DOS:MAP	Load map of the new PDOS system
F32DOS	SY file image of new PDOS system

2.4.1 Generation Arguments and Switches

F32DOS:GEN has two optional arguments to assist you in building a new PDOS system. Argument one is either the word `LINK' or switches to change the generation. Argument two is either blank, the word `BASIC', or `BASIC/68881'.

If argument one is the word LINK then F32DOS:GEN will build the new PDOS system without reassembly.

```
  F32DOS:GEN LINK
```

This will use the object files that were sent with PDOS. You can optionally input switches to change parameters in the generation files. The most notable are:

MAIL ARRAY SETUP:

This will dynamically setup the mail array and PDOS will place a pointer to this sized array in location `MAIL.' in SYRAM (see file MSYRAM:SR).

Switch Name	Default Value	
MZ	256	Number of bytes allocated to the Mail array

RAM DISK SETUP:

This is the specification of a RAM disk area. If RZ is set to Ø then no RAM disk will be allocated.

Switch Name	Default Value	
RU	8	Default RAM disk unit number
RZ	255	Default sectors allocated to RAM disk. 4 sectors equal 1KB
RE	(RZ/8)!7+1	Default number of directory entries on RAM disk (if RZ is 255 then RE will default to 32)
RA	Ø	RAM disk address. If Ø, (default), then allocate the RAM disk dynamically from top of memory, right below the MAIL array. If not Ø, then place RAM disk at the given address.
IRD	1	If Ø, don't initialize RAM disk area. If 1, then initialize RAM disk area on startup if it is not a PDOS disk image already.

LINE EDITING KEY DEFAULTS:

These are the key definitions for the get line primitives.

They are limited to a single character.

Switch Name	Default Value	
B.ADD	[CTRL-A]	Recall last line
B.LFT	[CTRL-H]	Move cursor left
B.RGT	[CTRL-L]	Move cursor right
B.DRT	[CTRL-D]	Delete character cursor is under (Delright)
B.DLT	[RUB]	Delete character to left of cursor

SYSTEM PARAMETERS:

Switch Name	Default Value	
SD	Ø	Default disk unit number (if Ø, then setup by the boot ROMs; if not Ø, then system will start with this disk number).
BR	Ø	Default baud rate. This is a number from Ø to 7 which represents initial baud rate for the character I/O ports. Ø = 19200 4 = 12ØØ 1 = 96ØØ 5 = 6ØØ 2 = 48ØØ 6 = 3ØØ 3 = 24ØØ 7 = 11Ø
CPSC	$AAØØ9B3D	Default clear screen and position codes. The default is for a SOROC terminal type. This will select the system startup default. You can select a different type for each task by running the MTERM utility after the system is up. PDOS can output 1 to 4 characters to clear the screen or callout to a BIOS driver. The first 16 bits of CPSC is the clear screen command. The format for clear screen is (where each letter is a bit): E111 1111 E222 2222 E = if 1 then precede with [ESC] 1 = 7 bits for first ASCII character 2 = 7 bits for second ASCII character If all 16 bits are Ø then callout to BIOS label B$CLS.

PDOS can position the cursor with 1 of the leading characters followed by row/column or column/row optional based by $20 or PDOS call callout to a BIOS driver. The second 16 bits of CPSC is the position cursor leading characters. The format is (where each letter is a bit):

```
        B111 1111 0222 2222
```

```
B = 0 then $00 bias
    1 then $20 bias
O = 0 then row first then column 1 then
      row
1 = 7 bits for first ASCII lead in
      character
2 = 7 bits for second ASCII lead in
      character
```

If all 16 bits are 0 then callout to BIOS label B$PSC.

ANS 0 — If 1, then the BIOS callout for clear screen and position cursor for ANSI 3.64 terminal support is included.

CLKADJ 0 — Clock adjust factor. This is used to fine tune the system clock. On each tic, PDOS will add this value to the adjustment factor. Each time the adjustment factor rolls over PDOS will add one tic. The formula is $80000000/N*2 where N is the number of tics to pass before adding an additional tic to the PDOS tics.

For example, if you needed to add 1 extra tic every 100,000 tics to keep the clock accurate, then you would set CLKADJ to $80000000/100000*2. This will cause a roll over every 100,000 tics.

FSTEP 3 — Floppy disk drive stepping rate. The default is 3msec, but full height drives may require 6msec stepping rate.

The above switches may be passed into the generation procedure file by specifying as the first argument a `/', the switch name, followed by an `=', then by the value of the switch. Note: The command line length is limited to 78 characters, so if you must set many switches,you will need to edit file F32BIOS:SR file and place the switch name in as EQUATES. The following shows some examples for defining the switches on a FORCE CPU-5 system:

```
Example 1 -- Set ANSI terminal support on:
             >F32DOS:GEN /ANS=1

Example 2 -- RAM disk size of 2000 sectors and at address $800000
             >F32DOS:GEN /RZ=2000/RA=$800000
```

The second argument of F32DOS:GEN is the word BASIC. If you have licensed BASIC then you need to specify the BASIC interpreter with your system. Failure to do so will create a system without BASIC. For 68020 CPU-boards with the 68881 co-processor, the string /68881 can be appended to the word BASIC to use the HW co-processor together with the BASIC interpreter. For example to build a system with BASIC and 68881 support, you need to type the command:

```
             >F32DOS:GEN ,BASIC/68881
```

2.4.2 Loading the New System

After F32DOS:GEN has been run without errors,you are now ready to try your new system. You can do this by using the FLDGO utility. FLDGO has the format:

```
            FLDGO <load address>,<filename>
```

The load address is the location in memory where the program is to be located. The default of $8ØØ is assumed on all FORCE Systems when no address is given. The filename is the name of your system file that you have just generated. After FLDGO has loaded your new system it will jump to the load address and begin execution. To try the new system type in (example for a CPU-2 system):

```
            FLDGO ,F32DOS
```

You should then see something similar to the following:

```
                   DOS File Loaded: F32DOS
              Found PDOS at address $ØØØ2ØEB6
                        DOS size is $ØØØØ8BD4

   (Execution of the startup file on the new version)
```

FLDGO allows you to try a new version of PDOS without modifying your disk boot image. When you reboot, via the reset switch or by powering down and up, your boot will come off the disk unit selected by the boot ROMs. To make this new system into a disk boot you need to follow the instructions in chapter 4, description of the MMKBT utility, or refer to chapter 2.5.

You can verify that you are executing the new version of PDOS with the PDOS ID monitor command. You will notice that the BIOS date will be the current date.

2.5 WRITING BOOTS TO DISK

The MMKBT utility is used to install bootable PDOS files onto your disks. The disk should first be prepared for use with the PDOS system as outlined in either section 2.2 for floppy disks or section 2.3 for your Winchester partitions. You need to have a PDOS bootable file as outlined in section 2.4.

To install a new boot you run MMKBT from the PDOS monitor:

```
>MMKBT
68K PDOS Make Boot Disk Utility Ø2/26/86
```

You are prompted with 3 options. The first is (F)ile for creating a new boot disk from a file. The second is (M)emory for creating a new boot disk from a memory area. The last is (B)ootstrap for the outputting of special sector information onto physical track Ø of the disk. The FORCE systems do not use the (B)ootstrap option since PDOS boot ROMs are supplied. To select any option, type the letter followed by a [CR].

2.5.1 Installing From a File

To install from a file type `F' followed by a [CR].

```
(F)ile, (M)emory, or (B)ootstrap? F[CR]
```

You are next asked for the name of the new system file. The default name is `F32DOS', which is the same filename generated from the F32DOS:GEN procedure file. You may enter in the name if different or just type a [CR] to use the default.

Next you are asked how large the boot is in bytes. MMKBT will size the file you specified and print the appropriate default for you in HEX, or you may enter in your own size in bytes followed by a [CR]. To use the default just type a [CR].

```
Boot size = $ØØØØ54ØC[CR]
```

The load address is specified next. The default load address is what you will normally use. It is selected by just typing a [CR]. PDOS is also position independent, and can be loaded at any address. To load PDOS at some other location in memory enter `$' if you are entering it in HEX.

```
Load address = $ØØØØØ8ØØ[CR]
```

You are asked for the PDOS disk number for writing the boot. Your current working disk is used as the default. If it is correct, then type a [CR], otherwise enter in your new disk number followed by a [CR].

```
Disk = 2[CR]
```

The boot sector number is next requested. The default is the current boot sector number if a boot is presently on the disk or it is the sector after the last initialized PDOS sector if no boot is present. You may generally use the default which is selected by typing a [CR]. You may optionally place the boot at any sector on the disk by typing the new sector number in.

You may wish to verify the boot sector number with the disk partition table from the FFRMT program to confirm that the boot is at the recommended location for disk partitions. This will give you about a 160 sector boot area or enough for a 40KB system. If you have a larger system then you will need to initialize your disk for fewer PDOS sectors which will allow a larger boot area. The boot sector number is generally the same as the number of PDOS sectors initialized in the MINIT system. Bootable diskettes should normally be initialized to 2336 PDOS sectors, and have the boot sector at 2336.

```
             Sector = 47776
```

After you have entered the sector number, you will be asked to confirm the parameters for writing the boot. If you respond with a `Y' followed by a [CR], the boot will be written out. An [ESC] will exit the program back to PDOS. Any other response will restart the MMKBT program.

```
  Write 85 sector boot beginning at address $0000D500 with load
  address $00000800 to disk 2, sector 47776. Ready (Y/N)? Y[CR]
```

WARNING: DOUBLE CHECK THAT YOU ARE WRITING TO THE CORRECT DISK AND SECTOR NUMBER. MMKBT will destroy any data at the specified sector and disk when this question is verified.

When a `Y[CR]' is typed you will be informed that the boot was or was not written successfully.

```
     One moment, Please...
        Wrote out header information.
        PDOS Written Successfully!!
```

2.5.2 Installing from Memory

Boots may also be written to the disk from memory. This is typically done to build a boot image of the currently executing system. After executing the MMKBT program you select the (M)emory option by typing an `M' followed by a [CR].

You are asked for the memory start address and end address for the boot. The defaults are according to your standard system set up. You may select the default by typing a [CR] or enter new addresses. (Remember to use the $ to prefix hexadecimal numbers):

```
Start address = $ØØØØØ8ØØ[CR]
End address   = $ØØØØ98ØØ[CR]
```

The disk number is next asked for and the rest of the make boot proceeds the same as with the (F)ile option in section 2.5.1.

2.6 AUTO-START FILE

Standard PDOS will look for the Auto-Start procedure file to execute by task Ø on startup. The file must be found on the boot disk, be named `SY$STRT' and must have an executable file attribute (i.e. AC, SY, OB or EX, BX if BASIC is resident). Any program or monitor command may be placed in the procedure file. The printout below shows the default contents of the startup file SY$STRT.

Feel free to customize your startup file to set up terminal types, correct flags for the ports (that is XON/XOFF, DTR, etc.) and baud rates. You may also want to create other tasks on the system with the startup file. This can be done by editing the startup file SY$STRT.

Listing of the startup file SY$STRT:

```
FTIME P
EE 1
MTERM D
CT (WIND1 1/14/15,1),32
CT MABORT,2
CT ,1ØØ,64,$F
CT ,1ØØ,64,$E
BP 2,1,1,$FF8ØØ2ØØ
BP 3,1,1,$FF8ØØ6ØØ
FBAUD
```

3.Ø INSTALLATION OF PDOS ON THE WINCHESTER

The PDOS operating system with all utilities is already installed by default on the Winchester on logical disk 2. This chapter, which describes the installation on the Winchester, is only to be used when you change the Winchester.

3.1 PDOS FLOPPIES

Three floppies are delivered with every PDOS System. The files are already installed on the Winchester and so the floppies are only a backup. Keep these floppies in a safe place. They will be needed if the Winchester crashes and a new one has to be installed in the system.

The three PDOS floppies contain the following files:

1. BOOT — This floppy contains a bootable PDOS operating system, all hardware implementation and some useful utilities.

2. UTIL — This is the standard PDOS utility disk and contains all utilities and the sources of most of the utilities.

3. FORCE UTIL — This disk holds some FORCE specific utilities and their sources.

3.2 BOOTING PDOS FROM FLOPPY

Attach a 9600 BAUD terminal to the Terminal port, apply power, activate the RESET switch and start the PDOS Bootstrap program (see chapter 1 for details).

The bootstrap first initializes the disk controllers in the system. It then starts to boot, first from the floppy, and next from the hard disk. For example:

```
   FORCE SYS68K/CPU-20/21 PDOS Bootstrap Revision 2.1   01/15/85
       Disk Controller Installed
```

If the bootstrap program cannot find a disk controller, the following will be displayed:

```
   FORCE SYS68K/CPU-20/21 PDOS Bootstrap Revision 2.1    01/15/86
    No Disk Controller Installed ???
```

If you see the previous message and yet you know that there is a disk controller board installed, please check the address jumpers on the disk card and then test that the card is really accessible under the monitor.

After printing the banner and initializing the disk controller, the bootstrap program begins reading the header sectors of disks, number 0 through 99, for PDOS boot information. You should see some messages printed out that indicate disk errors on disks 0 and 1, since the floppy drives have no diskettes in them:

```
   Disk #0:  Header sector error = 116
   Disk #1:  Header sector error = 116
   Out of PDOS boot disk table entries.
   I'll Retry them all.
```

These errors are normal until the Winchester gets up to speed or you insert a floppy disk. The bootstrap reinitializes the disk controller and starts all over again, looking at disk 0 to 99.

Now, insert the PDOS disk marked `BOOT' in the main floppy drive and close the drive door. The next time around, the bootstrap should find the PDOS boot on disk #0 and display:

```
Disk #0:
  FORCE CPU-20/21 PDOS was found  >>  Sector 2336, Addr $800
            Execute it ? _
```

You do not need to respond to the question, simply wait for a moment and see the result:

```
Disk #0:
  FORCE CPU-20/21 PDOS was found  >>  Sector 2336, Addr $800
            Execute it ? Yes.  Booting...SUCCESS!
```

After "Booting...SUCCESS!" is displayed, the system will enter an initialization routine. The name of this initialization program is "SY$STRT". It initializes the available serial I/O channels, initializes the date and time (if a real-time clock is available), and prints a welcome message.

If at any time your system "crashes", push the reset button located on the front panel of your system and restart the PDOS boot program. You will then be prompted with the same boot message as above. Proceed in the same manner as a cold startup.

3.3 WINCHESTER SET UP

Next you must format and set up the partitions for your Winchester drive. The number of partitions and sizes will depend on your needs. If you wish to change the partitioning of the drive, you will need to backup the entire Winchester drive and rebuild the files. Complete instructions on Winchester setup are given in the system management portion of this guide under the heading "Preparing Winchester Drives for use with PDOS" (chapter 2).

3.4 TRANSFERRING PDOS SYSTEM FILES TO YOUR WINCHESTER

After you have formatted and partitioned your Winchester drive, you will need to move all files from the PDOS distribution disks to a system disk on the Winchester. By PDOS standard this is disk 2, but you may use any other disk unit on your Winchester. It is also a PDOS standard that the last Floppy image on the Winchester is a temporary disk that can be used to quickly transfer files. It is assumed in here that disk 2Ø is a floppy image for these instructions. You may have to substitute your temporary floppy number if it is different.

First backup the PDOS boot disk to temporary floppy disk 2Ø. This is done by first inserting the distribution disk marked PDOS BOOT into floppy drive Ø and then running the MBACK utility with the following arguments:

```
>MBACK Ø,2Ø,2528
```

This will backup 2528 sectors of disk Ø to floppy image disk 2Ø. It also copies the boot tracks from floppy to disk 2Ø. Next transfer all files from disk 2Ø to the system disk 2. This can be done with the PDOS monitor command TF:

```
>TF ;@/2Ø,2,A
```

Next backup the PDOS UTILITY disk to temporary floppy disk 2Ø by inserting the utility disk into drive Ø and running MBACK with the following arguments:

```
>MBACK Ø,2Ø,F
```

This will now copy all files of the utility floppy onto the floppy image disk 2Ø on the Winchester. Next we have to transfer these files to disk 2.

```
>TF ;@/2Ø,2,A
```

The last floppy to transfer is the FORCE utility disk (disk #3). You can use the same commands as outlined in the description of the PDOS utility disk:-

```
>MBACK Ø,2Ø,F
>TF ;@/2Ø,2,A
```

If you purchased PDOS BASIC, you may want to also copy those files onto disk 2 so that BASIC can be installed into your system for the next section. Use MBACK and TF commands as used with the previous 2 disks.

```
>MBACK Ø,2Ø,2528
>TF ;@/2Ø,2,A
```

Now you will have to generate a PDOS System with BASIC before you install the boot tracks. You can use following command line:

```
>F32DOS:GEN <parameters>,BASIC</68881>
```

The parameters are described in detail in chapter 2. The option /68881 should only be specified if your system contains a 68881 co-processor.

3.5 INSTALL PDOS SYSTEM BOOT ONTO WINCHESTER

The PDOS system boot can reside on one or more Winchester disks. You will need to have initialized any disk to the boot sector numbers instead of the total sector numbers. This is under discussion in the Winchester drive preparation of Chapter 2.3. Upon startup, the PDOS boot ROMs will look for a PDOS boot first at disk Ø then 1 (the two floppy disks). The ROMs will start with the lowest Winchester disk partition number up to disk partition number 99. The process will repeat until a valid PDOS boot is found.

For this section we will assume that disk 2 is the disk you wish to install the PDOS boot on. You will want to run the F32DOS:GEN procedure file followed by the MMKBT utility. The process of BOOT making is covered in detail in section 2.3. For the initial set up you may want to follow one of the following examples: (MMKBT will display the correct default parameters for you which may be different than shown):

1) Install standard system:

```
>F32DOS:GEN
>MMKBT
68K PDOS Make Boot Disk Utility Ø5/Ø6/86
(F)ile, (M)emory, or (B)ootstrap?        (answer F[CR])
        File name = F32DOS               (just type [CR])
        Boot size = $ØØØØ54ØC            (just type [CR])
     Load address = $ØØØØØ8ØØ            (just type [CR])
             Disk = 2                    (enter your disk
                                         number then [CR])
           Sector = 47776                (just type [CR])
Write 85 sector boot beginning at address $ØØØØB5ØØ
with load address $ØØØØØ8ØØ to disk 2, sector 47776.
Ready (Y/N)? Y                           (type Y[CR])
One moment, please...
  Wrote out header information
  PDOS written successfully
>
```

2) Install standard system with ANSI terminal support and BASIC with 68881 support:

```
>F32DOS:GEN /ANS=1,BASIC/68881
>MMKBT
68K PDOS Make Boot Disk Utility Ø5/Ø6/86
(F)ile, (M)emory, or (B)ootstrap?          (answer F[CR])
       File name = F32DOS                  (just type [CR])
       Boot size = $ØØØØ54ØC               (just type [CR])
    Load address = $ØØØØØ8ØØ               (just type [CR])
            Disk = 2                       (enter your disk
                                           number then [CR])
          Sector = 47776                   (just type [CR])

Write 85 sector boot beginning at address $ØØØØB5ØØ
with load address $ØØØØØ8ØØ to disk 2, sector 47776.
Ready (Y/N)? Y                             (type Y[CR])
One moment, please...
  Wrote out header information.
  PDOS written successfully!
>
```

You should now be able to remove the boot floppy from drive Ø and to have the system booted from the ROMs. This can be done by pressing the reset switch on the CPU card. If you have installed the boot correctly, the system will boot from disk 2.

Your PDOS system is now installed and ready to use.

4.0 FORCE UTILITIES

This chapter describes all the utilities which are specific for the FORCE FOCUS Systems.

It covers the following utilities:

1.)	MMKBT	- Make bootable disk
2.)	FTIME	- Set /read PDOS clock or Battery clock
3.)	FFRMT	- Disk format utility
4.)	FLDGO	- Load and/or Go To a new system
5.)	FPARK	- Park drives for shipping
6.)	BLOCKING	- Enable/disable blocking mode (ISCSI-1 Systems)
7.)	FLUSH	- Flush ISCSI-1 buffers
8.)	TIME	- Run-time measurement utility
9.)	RR-2	- EPROM programming utility (needs RR-2/RR-3 board)

The utilities numbered 1-8 are located on the PDOS boot floppy. Number 9 is located on the disk labeled "FORCE Utils".

In addition, the following loadable drivers are distributed with the FORCE PDOS System. They are located on the boot disk.

1.)	PR	- Driver for the parallel printer interface of the ASCU board
2.)	PRS	- Driver for the serial printer interface of the ASCU board

4.1 MMKBT - MAKE DISK BOOT

Name: MMKBT
Function: Installs PDOS system and bootstrap onto disks
Format: >MMKBT

Restrictions: May not be re-entered with the monitor command "GO."

Description: The MMKBT utility is used to install bootable PDOS files onto your disks. The disk should first be prepared for use with the PDOS system.

You are prompted with three options. The first option is (F)ile for creating a new boot disk from a file. The second is (M)emory for creating a new boot disk from a memory area. The last option is (B)ootstrap for putting out special sector information onto physical track 0 of the disk. The FORCE systems do not use the (B)ootstrap option since PDOS boot ROMs are supplied. To select any option, type the letter followed by a [CR].

This utility provides defaults which are the correct parameters in most cases.

(F)ile Option: When the (F)ile option is selected, you can install a boot from a file on your disk . You are prompted for the filename. The file will be read in and sized. You are then asked for the boot size, the load address, the disk number, and the boot sector number. You must confirm the installation before any data is written to the disk.

```
  > MMKBT
  68K PDOS Make Boot Disk Utility
  (F)ile, (M)emory, or (B)ootstrap? F
       Filename = F32DOS
      Boot size = $0000540C
   Load address = $00000800
           Disk = 0
         Sector = 2336

Write 85 sector boot beginning at address $0000D500
with load address $00000800 to disk 0, sector 2336.
Ready (Y/N)?Y
One moment, Please...
  Wrote out header information.
  PDOS Written Successfully!!
```

(M)emory Option: The (M)emory option allows you to copy memory onto the boot area of a disk. This is typically done to build a boot image of the currently executing system. After executing the MMKBT program, you select the (M)emory option by typing an "M[CR]".

You are asked for the memory start address and end address for the boot. The defaults are determined by your standard system setup. You may select the defaults by typing a [CR] or you may enter new addresses followed by a [CR]. (Remember to use the $ to prefix hexadecimal numbers). The rest of the questions are the same as the file option.

```
  > MMKBT
  68K PDOS Make Boot Disk Utility
  (F)ile, (M)emory, or (B)ootstrap? M
       Start address = $00000800
         End address = $00009800
        Load address = $00000800
                Disk = 0
              Sector = 2336

Write 144 sector boot beginning at address
$00000800 with load address  $00000800  to disk 0,
sector 2336.
Ready (Y/N)?Y
One moment, Please...
  Wrote out header information.
  PDOS Written Successfully!!
```

(B)ootstrap option: The (B)ootstrap option is not used with FORCE systems.

4.2 FTIME - SET PDOS/BATTERY CLOCK

Name: FTIME
Function: Set the PDOS and/or Battery clocks
Format: >FTIME {P | B }

Restrictions: FTIME only sets the PDOS clock correctly when a battery clock is present and the battery is installed. This utility is only available for the CPU-5 and for all 68020 based CPU-boards. For all other 16-Bit boards MTIME should be used.

Description: Either the letter "P" or the letter "B" may be omitted from the first parameter. If only the letter "P" is specified, then the PDOS clock is set from the battery clock. For example, to set the PDOS clock from the battery clock you would type:

```
>FTIME P
```

When the first parameter is the letter "B", the battery clock is set to the current PDOS clock values. This is usually done after the PDOS clock has been set with the "ID" monitor command.

```
>ID
PDOS/68020 R3.2 14-Jan-87
ERII, Copyright 1983-1986
FORCE CPU-20/21 BIOS 04-Feb-87
DATE=00/00/00 02 05 86
TIME=00/00/01 10 15 00
>FTIME B
>_
```

When no parameters are specified, the values of both the battery and PDOS clocks are displayed.

```
>FTIME
ASCU CLOCK = 05-Feb-87 14:47:54
PDOS CLOCK = 05-Feb-87 14:47:56
```

Note: The year is kept in the RAM of the real-time clock. If the year rolls over (i.e. from 1987 to 1988), the year has to be set with the ID Monitor Command and the real-time clock has to be loaded.

The following Battery clocks are supported by FTIME:

1. CPU-5 ASCU-1/2 board at address $B02000

2. CPU-20/21 ASCU-1/2 board at address $FCB02000 (Standard VME access, user and supervisor data.)

3. CPU-24/25 ASCU-1/2 board at address $FCB02000 (Standard VME access, user and supervisor data.)

4.3 FFRMT - DISK HARDWARE FORMAT

Name: FFRMT
Function: Hardware format disks and set up PDOS partitions
Format: >FFRMT

Restrictions: FFRMT may only be run when no other tasks are running.

Description: FFRMT allows you to define drives and to format and partition disk drives. This utility is hardware dependent. PDOS will support up to four drives on one or two controllers. This utility is menu driven. Refer to section 2 in this guide for complete details on its use. The following is only a summary.

When you run this utility, you may select a drive to access (i.e. F, FØ, F1 for the floppy diskette drives or W, WØ, W1, W2, W3 for up to 4 Winchester drives). Enter the drive letters followed by a [CR] to access the drive. If the drive is undefined, you will be prompted with the drive select byte for the controller.

```
                   >FFRMT
                   68K PDOS FORCE Format Drive Utility Ø5/23/85
                    Possible Disk Controllers in this System are:
                        Controller #1 is a FORCE SASI-1
                        Controller #2 is a FORCE WFC-1
                        Controller #3 is a FORCE ISCSI-1
                    Drives that are currently defined in system are:
                        FØ is controller #3, drive select byte $73
                        F1 is controller #3, drive select byte $74
                        WØ is controller #3, drive select byte $ØØ
                        W1 is undefined.
                        W2 is undefined.
                        W3 is undefined.

Select Menu: W,WØ,W1,W2,W3=Winch; F,FØ,F1=Floppy; Q=Quit
     Select Drive: _
```

If you select either a floppy drive or a Winchester drive that is already defined, FFRMT directly enters the Drive Command Menu. If you are installing a new Winchester drive which is currently undefined, then you must enter the controller number and drive select jumpering (Ø-3). The Drive Command Menu tells you which drive you are currently dealing with and has the following commands:

```
  Select Menu: W,WØ,W1,W2,W3=Winch; F,FØ,F1=Floppy; Q=Quit
       Select Drive:  WØ[CR]
  WØ Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  [CR]

  Winchester Drive Ø Menu:
       1) Display/Alter drive Parameters.
       2) Display/Alter Bad Track List.
       3) Format tracks.
       4) Verify tracks.
       5) Display/Alter PDOS Disk Partitions.
       6) Write out Header info to disk.
       P) Toggle Unit 2.
       Q) Quit & Select another Drive.
  WØ Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  _
```

When dealing with a floppy drive, the display/alter commands do not allow you to alter the drive parameters, the bad track table, or the disk partitions, and you may not write out header information to a floppy disk. To exit to PDOS, you must first return to the Select Drive Menu with the Q) command. Following is a detailed description of the Drive Command Menu commands:

1) <u>Display/Alter Drive Parameters:</u>

The Display/Alter Drive Parameters menu allows you to D)isplay the currently defined drive parameters, A)lter them, R)ead them in from a file, or Q)uit and exit to the Select Drive Menu:

```
  WØ Parameters Menu:  A)lter, D)isplay, R)ead file, Q)uit
       Command: _
```

To display the current drive parameters on a Winchester, enter the `D' command. The parameters are displayed to the screen. The Drive Parameters that are displayed, and that can be altered are:

```
       Current (type) Drive N Parameters:
                # of Heads = Number of heads on drive
            # of Cylinders = Number of cylinders on drive
 Physical Blocks per Track = Actual blocks on a track
  Physical Bytes per Block = Actual bytes per physical block
         Shipping Cylinder = Where to position head before
                             moving drive
                 Step rate = Controller dependent definition
 Reduced write current cyl = Cylinder to apply reduced
                             write current
   Write Precompensate cyl = Cyl to apply write precompensation
```

To alter them, enter the `A' command. In the alter mode, you enter either: 1) a carriage return to leave the parameter the same and go to the next prompt; 2) a number and a carriage return to change the parameter and go to the previous parameter prompt. The Drive Parameters are displayed one at a time for you to either alter or leave alone.

If you have previously saved out the drive parameters to a disk file, you can restore them by entering the `R' command, followed by the name of the file. This file may be created using the F)ile command of Drive Command Menu option 6) Write to disk, or it can be created with a PDOS editor. The format and syntax of the parameter file is discussed under option 6). Reading this information destroys all other information; replaces the parameters, the bad track table, and the partition definitions.

The `Q' command returns you to the Drive Command Menu.

For example, look at floppy drive FØ parameters:

```
  Select Menu: W,WØ,W1,W2,W3=Winch; F,FØ,F1=Floppy; Q=Quit
       Select Drive:  FØ[CR]
  FØ Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  1[CR]

  Current Floppy Drive Ø Parameters:
                # of Heads = 2
            # of Cylinders = 8Ø
 Physical Blocks per Track = 16
  Physical Bytes per Block = 256
         Shipping Cylinder = Ø
                 Step rate = Ø
 Reduced write current cyl = Ø
   Write Precompensate cyl = Ø
  FØ Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command: _
```

As another example, select the W0 Winchester and display the current parameters:

```
  W0 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  1[CR]
  W0 Parameters: A)lter, D)isplay, R)ead file, Q)uit
       Command:  D[CR]

  Current Winch Drive 0 Parameters:
                # of Heads = 16
            # of Cylinders = 1000
 Physical Blocks per Track = 32
  Physical Bytes per Block = 256
         Shipping Cylinder = 0
                 Step rate = 0
 Reduced write current cyl = 0
   Write Precompensate cyl = 0

  W0 Parameters Menu:  A)lter, D)isplay, R)ead file, Q)uit
       Command:  Q[CR]
  W0 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command: _
```

2) Display/Alter Bad Track List:

The Display/Alter Bad Track menu allows you to D)isplay the currently defined bad tracks on the drive (if any), add or delete tracks, C)lear the bad track table, get a H)elp message, or Q)uit and exit to the Drive Command Menu:

```
  W0 Bad Tracks Menu:  Bad Track, D)isplay, C)lear, H)elp, Q)uit
       Command:  _
```

To display the current bad tracks on a Winchester, enter the `D' command. The tracks are displayed on the screen in ascending order as a physical track number followed by the head and cylinder number, separated by a comma and enclosed in parentheses.

To add a bad track to the list, enter either the actual physical track number and a carriage return, or the head and cylinder number desired, separated by a comma and followed by a carriage return. To delete a track, precede the track or head number with a minus sign (-).

Sometimes the bad track table may be incorrect or spoiled. You can start all over again by entering the C)lear table command. The `Q' command returns you to the Drive Command Menu. In case you have added or deleted some bad tracks, FFRMT asks if you want to recalculate the disk partitions on the drive before returning to the drive menu. By altering the number of bad tracks, you also alter the number of logical tracks available for PDOS disk partitions. Answer `Y' or `N' to the query, as you like.

Note that the SCSI Winchesters handle bad blocks internally. So when you are using the ISCSI-1 controller, the bad blocks defined by the manufacturer are already spared on the disk.

For example, look at the Winchester drive Ø bad track list:

```
  WØ Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  2[CR]
  WØ Bad Tracks Menu: Bad Track, D)isplay, C)lear, H)elp, Q)uit
       Command:  D[CR]

  Current Winch Drive Ø Bad Tracks:
231(Ø,77)    613(1,2Ø4)    697(1,232)    7ØØ(1,233)    7Ø3(1,234)

  WØ Bad Tracks Menu: Bad Track, D)isplay, C)lear, H)elp, Q)uit
       Command:  Q[CR]
  WØ Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  _
```

3) <u>Format Drive/Tracks:</u>

```
       Sector Interleave = {default from MCONTB table is listed}
 Physical Tracks to Format = {[CR] for beg,end tracks listed}
   Ready to Format Drive Ø ? {`Y' or `N'}
```

This routine first calls the INFMT routine which sets up the format. Then F)ormat makes one or more calls to the TKFMT routine until all the specified tracks are formatted. Between calls, a check for user break ([CTRL-C]) is made, and the track number just formatted is printed to the terminal. If there are errors, you can select either R)etry, Y)es -- add the track to the bad track list, or N)o -- ignore the error and go on.

For example, format a floppy disk with the default sector interleave, 5, and do tracks Ø to 159, inclusive:

```
       Sector Interleave = 5[CR]
       Physical Tracks to Format = Ø,159[CR]
       Ready to FORMAT Floppy Drive Ø ? Y[CR]
       Sector Interleave Table:
     1,9,4,12,7,15,2,1Ø,5,13,8,16,3,11,6,14

       Issuing Format Drive Command

       FORMAT Successful!
```

Note that the interleave is "Don't care" for SCSI Winchester drives.

4) Verify Tracks:

```
    Physical Tracks to Verify = {default from last format command}
                        Ready ? {`Y' or `N'}
```

This routine, after calling INFMT, reads every sector on each track specified. Errors are reported to the terminal. Between calls a check for user break ([CTRL-C]) is made, and the track just verified is printed to the terminal. If there are errors, you can select either R)etry, Y)es -- add the track to the bad track list, or N)o -- ignore the error and go on.

5) Display/Alter Disk Partitions:

The Display/Alter Partitions menu allows you to D)isplay the currently defined disk partitions, A)lter them, R)ecalculate them from the current values, or Q)uit and exit to the Drive Command Menu:

```
  WØ Partitions Menu: A)lter, D)isplay, R)ecalc, Q)uit
       Command: _
```

To display the current disk partitions on a Winchester, enter the `D' command. The partitions are displayed on the screen. The Disk Partitions that are displayed are based on a few parameters, which you can change:

```
     # of Large partitions = How many large divisions on the drive
    # of Floppy partitions = How many small divisions on the drive
First track for PDOS Parts = Where to begin the disk partitions
 Last track for PDOS Parts = Where to end the disk partitions
         First PDOS disk # = What is first PDOS disk # of
                             partitions
```

To alter them, enter the `A' command. In the alter mode, you enter either: 1) a carriage return to leave the parameter the same and go to the next prompt; 2) a number and a carriage return to change the parameter and go to the next prompt; or 3) an escape to go to the previous parameter prompt. The disk partitions parameters are displayed one at a time for you to either alter or leave alone. If you alter the number of disks or the tracks for partitions, then you are asked if you would like to recalculate the partitions. Enter either `Y' or `N'. If you only change the beginning PDOS disk number then only the disk numbers are reassigned, leaving the base and top tracks of the partitions alone.

You can make the partition information consistent by simply entering the `R' command. This recalculates the drive partition information using the current values of drive parameters, bad track table, and partition parameters. The `Q' command returns you to the Drive Command Menu.

For example, look at floppy drive FØ partitions:

```
       Current Floppy Drive Ø Disk Partitions:
     # of Large partitions = Ø
    # of Floppy partitions = 1
First track for PDOS Parts = Ø
 Last track for PDOS Parts = 159
         First PDOS disk # = Ø
 Total # of Logical Tracks = 16Ø
 Disk #     Base Trk    Top Trk    # of PDOS sectors
    Ø          Ø          159       2528/2336{boot}
```

As another example, examine the partitions on WØ:

```
  WØ Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  5[CR]
  WØ Partitions Menu: A)lter, D)isplay, R)ecalc, Q)uit
       Command:  D[CR]

  Current Winch Drive 1 Disk Partitions:
     # of Large Partitions = 1Ø
    # of Floppy Partitions = 12
First track for PDOS Parts = Ø
 Last track for PDOS Parts = 15979
         First PDOS disk # = 2
 Total # of Logical Tracks = 16ØØØ
 Disk #    Logical Trks  Physical Trks   PDOS sectors
           Base,Top      Base,Top        Total/{boot}
  2        Ø,1499        Ø,15ØØ          47968/47776
  3        15ØØ,2999     15Ø1,3ØØØ       47968/47776
  4        3ØØØ,4499     3ØØ1,45ØØ       47968/47776
  .         .             .               .
  .         .             .               .
  .         .             .               .

 24        1588Ø,15959   15897,15979     2528/2336
  WØ Partitions Menu: A)lter, D)isplay, R)ecalc, Q)uit
       Command:  Q[CR]
  WØ Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  _
```

6) Write Header Information to Drive:

The Write Header Information to Drive menu allows you to 1) `Y' write the information to the drive header, 2) `N' abort the command and return to the Drive Command Menu, or 3) `F' write drive information to a file. After assigning the correct parameters to a drive, entering the bad tracks, formatting it, and partitioning it into PDOS disk numbers, you still need to write this information to the drive's header. This information must reside on the disk and is used by for the BOOT ROMs and by PDOS.

This routine calls the INFMT subroutine to initialize the controller for the new number of heads and cylinders, and then calls the WTHED subroutine which writes out the drive data block (DDB) to the correct sector on the drive, usually sector Ø.

To write this information to the drive, enter the `Y' command. If you have second thoughts, enter the `N' command.

You should save the information out to a floppy disk file for each Winchester drive. This file makes recovering from Winchester disasters easier. You can either select to only write out the file with the `F' command, or write the file out after writing out the header information to the drive.

The file syntax is that:

1) lines starting with an asterisk (*) are ignored as comments;
2) parameter keywords are four characters long and appear as the first four characters of the line;
3) keywords are followed by an equal sign (=) and the value (hex must be preceded with dollar sign ($));
4) bad tracks use the keyword TRACK, are followed by an equals sign, and are designated by either the track number or the head and cylinder numbers (separated by a comma);
5) order of the keywords is not significant, except that the HEDS definition must precede any TRACK specification using the head, cylinder format; and
6) any unspecified keyword parameters are reset to system defaults, and not left as previously entered values.

The drive parameter keywords are defined as follows:

HEDS = # of Heads

CYLS = # of Cylinders

BPTK = Physical Blocks per Track

BPBK = Physical Bytes per Block

SHIP = Shipping Cylinder

STEP = Step Rate

REDU = Reduced Write Current Cylinder

WRTP = Write Precompensate Cylinder

The disk partition keywords are defined as follows:

WPRT = # of Large Partitions

FPRT = # of Floppy Partitions

BTRK = First Track for PDOS Parts

ETRK = Last Track for PDOS Parts

BDKN = First PDOS disk #

While reading in the file using the R)ead command of the 1)Parameter menu, FFRMT outputs a `Found:' message, followed by the parameter value when a successful keyword match and number conversion is made. This indicates that the parameter was loaded. If a keyword match is not made or if the conversion fails, FFRMT echoes the line to the terminal preceded by two question marks (??). This indicates that the parameter was not loaded.

In the following example, the file, TEMP is first written, shown, and then re-read with FFRMT:

```
  WØ Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  6[CR]
  Write to Disk Y)es, N)o, F)ile : F[CR]
  Write Parameter File:#TEMP/Ø[CR]
  WØ Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  Q[CR]
  Exit to Select Drive.  Update Param RAM (Y/N) ? N[CR]
  Select Menu: W,WØ,W1,W2,W3=Winch; F,FØ,F1=Floppy; Q=Quit
       Select Drive:  Q[CR]

2>SF -TEMP[CR]
* Drive parameters:
HEDS=16
CYLS=1ØØØ
BPTK=32
BPBK=256
SHIP=Ø
STEP=Ø
REDU=Ø
WRTP=Ø
* Drive partitions:
WPRT=1
FPRT=16
BTRK=Ø
ETRK=2489
BDKN=2
2>FFRMT[CR]
68K PDOS FORCE Format Drive Utility Ø9/15/86
  Select Menu: W,WØ,W1,W2,W3=Winch; F,FØ,F1=Floppy; Q=Quit
       Select Drive:  W1[CR]
  Drive not defined in system.
    What controller # ? 3[CR]
    What is Winch wired for (Ø-3) ? 1[CR]
  W1 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  1[CR]
  W1 Parameters Menu:  A)lter, D)isplay, R)ead file, Q)uit
       Command: R[CR]
  Read Parameter File:TEMP[CR]
  Processing file...
```

```
Found:                    # of Heads = 16
Found:                # of Cylinders = 1ØØØ
Found:     Physical Blocks per Track = 32
Found:      Physical Bytes per Block = 256
Found:             Shipping Cylinder = Ø
Found:                     Step rate = Ø
Found:Reduced write current cylinder = Ø
Found:       Write Precompensate cyl = Ø
Found:         # of Large partitions = 1Ø
Found:        # of Floppy partitions = 12
Found:     First track for PDOS Parts = Ø
Found:      Last track for PDOS Parts = 15979
Found:             First PDOS disk # = 2

  W1 Parameters Menu:  A)lter, D)isplay, R)ead file, Q)uit
       Command: Q[CR]
  W1 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  2[CR]
  W1 Bad Tracks Menu: Bad Track, D)isplay, C)lear, H)elp, Q)uit
       Command:  D[CR]

  Current Winch Drive 1 Bad Tracks:
  W1 Bad Tracks Menu: Bad Track, D)isplay, C)lear, H)elp, Q)uit
       Command:  Q[CR]
  W1 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  5[CR]
  W1 Partitions Menu: A)lter, D)isplay, R)ecalc, Q)uit
       Command:  D[CR]

  Current Winch Drive 1 Disk Partitions:
     # of Large Partitions = 1Ø
    # of Floppy Partitions = 12
First track for PDOS Parts = Ø
 Last track for PDOS Parts = 15979
         First PDOS disk # = 2
 Total # of Logical Tracks = 16ØØØ
 Disk #    Logical Trks  Physical Trks   PDOS sectors
           Base,Top      Base,Top        Total/{boot}
  2        Ø,1499        Ø,15ØØ          47968/47776
  3        15ØØ,2999     15Ø1,3ØØØ       47968/47776
  4        3ØØØ,4499     3ØØ1,45ØØ       47968/47776
  .        .             .               .
  .        .             .               .
  .        .             .               .

 24        1588Ø,15959   15897,15979     2528/2336

  W1 Partitions Menu: A)lter, D)isplay, R)ecalc, Q)uit
       Command:  Q[CR]
  W1 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  _
```

Q) Select Another Drive:

If you were working with a floppy drive, the Q)uit command simply returns you to the Drive Select Menu. If you were working with a Winchester, then the Q)uit command asks whether or not to write the new drive data block down to low parameter RAM. Enter either `Y' or `N' to this query. If you answer `N', your configuring session will be lost. It then exits to the Drive Select Menu.

```
  WØ Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  Q[CR]
  Exit to Select Drive.  Update Param RAM (Y/N) ? Y[CR]
  System Parameter RAM Updated!!
  Select Menu: W,WØ,W1,W2,W3=Winch; F,FØ,F1=Floppy; Q=Quit
       Select Drive:  Q[CR]

2>_
```

4.4 FLDGO - LOAD AND/OR GO TO A NEW SYSTEM

Name: FLDGO
Function: Load into memory and/or execute new system.
Format: >FLDGO {<load address>}{,<filename>}

Restrictions: FLDGO will replace your current PDOS operating system and execute a new system terminating all tasks.

Description: FLDGO is used to load and execute new PDOS systems.

The load address is the location in memory where the program is to be located. The default of $800 is assumed.

The filename is the name of your system file. If a filename is not given, then FLDGO will look for a PDOS system in your task space. FLDGO will only load a file in which the PDOS ID characters are found. After FLDGO has loaded your new system, it will jump to the load address and begin execution.

The following is an example on a FORCE FOCUS system 21A:

```
>FLDGO ,F32DOS
```

You then see something similar to the following:

```
        DOS File Loaded: F32DOS
  Found PDOS at address: $00100EB6
            DOS size is $00008BD4
```

(The startup file on the new version is now executed).

```
....
```

FLDGO allows you to try a new version of PDOS without modifying your disk boot image. When you reboot via the reset switch or by powering down then up, your boot will come off the disk unit selected by the boot ROMs. To make this new system into a disk boot you need to follow the instruction for MMKBT.

4.5 FPARK - PARK DRIVES FOR SHIPPING

Name: FPARK
Function: Flush buffers and park drives for shipping
Format: >FPARK

Description: This utility does nothing on the standard FORCE system. You need to use this utility if you use drives that must be parked before they are shipped. Define a shipping cylinder under option 1 of the FFRMT utility.

4.6 BLOCK - ENABLE/DISABLE BLOCKING MODE (ISCSI-1 ONLY)

Name: BLOCK

Function: This utility toggles the blocking enable/disable mode of the ISCSI-1 controller.

Format: `>BLOCK`

Restrictions: This utility is only useful in systems with an ISCSI-1 controller at the default address.

Description: This utility toggles the blocking enable/disable mode of the ISCSI-1 controller (i.e. blocking enabled or blocking disabled). This utility should be put in the startup file SY$STRT if you want to use the blocking mode. If blocking is enabled you must make sure that all buffers which are kept in the RAM of the ISCSI-1 controller are updated to disk. To do so, the "FLUSH" utility or the "FPARK" utility can be used. One of the two must be invoked whenever the system is shut down.

In addition, the ISCSI-1 controller is doing an automatic flush every 20 seconds. So when blocking is enabled when you crashed your system and the processor comes to a halt, wait at least 20 seconds before you press reset or turn the power off, so that the ISCSI-1 controller has a chance to write the buffer to disk.

For detailed information please refer to the ISCSI-1 hardware user's manual.

Example:

```
>BLOCK
  Blocking enabled
>BLOCK
  Blocking disabled
```

4.7 FLUSH - FLUSH ALL ISCSI-1 BUFFERS TO DISK

Name: FLUSH

Function: This utility writes all modified buffers of the ISCSI-1 controller to disk.

Format: >FLUSH

Restrictions: This utility is only useful in systems with an ISCSI-1 controller at the default address.

Description: If blocking was enabled on the ISCSI-1 disk controller (see 4.6, BLOCK), this utility can be used to write all modified buffers, which are kept local on the ISCSI-1, to the disk. The command has no effect if blocking is disabled, or if no ISCSI-1 controller is used in the system.

Example: >FLUSH

4.8 TIME - MEASURE THE RUN-TIME OF A PROGRAM

Name: TIME
Function: Measure the run time of a user program.
Format: >TIME <program>

Restrictions: This utility can only be used for location independent programs which reside in Main Memory. Therefore, all programs which are loaded from disk (or RAM disk) can be used, but ROM resident programs or PDOS monitor commands (or command files) are not allowed.

Description: This utility loads the program to RAM using the XLDF primitive, preserves the current TIC counter value and sets a new XEXT and XERR address; then the program is invoked. When the program terminated control is passed back to the time utility, the TIC counter is read again, then the difference is calculated and printed to the terminal.

Example:

```
>TIME CC SIEVE
 USER TIME = 18.70

>
```

4.9 RR-2 - EPROM PROGRAMMING UTILITY

This utility requires a SYS68K/RR-2/3 board at the default address. A complete documentation of this utility is included in the RR-2/3 manual and it can also be found on the FORCE utility floppy.

4.10 PR - DRIVER FOR THE PARALLEL PRINTER PORT OF THE ASCU-1/2

Name: PR
Function: Loadable driver for the parallel printer port of the ASCU-1/2.
Format: >CF <textfile>, PR

Restrictions: Only the data lines, Data Strobe and Busy are used and supported.

Description: PR is a loadable PDOS driver for the parallel printer port of the ASCU-1/2 board. It requires one ASCU-1/2 board on the default address.

For a detailed description of the PDOS loadable Driver mechanism please refer to the Appendix of the PDOS Operating System Manual.

Example: >CF PR:SR,PR this prints a listing of the Driver on the parallel printer

4.11 PRS - DRIVER FOR THE SERIAL PRINTER PORT OF THE ASCU-1/2

Name: PRS

Function: Loadable driver for the serial printer port of the ASCU-1/2.

Format: >CF <textfile>,PRS

Restrictions: None

Description: PRS is a loadable PDOS driver for the serial printer port of the ASCU-1/2 board. It requires one ASCU-1/2 board on the default address. The driver supports both hardware and software (XON/XOFF) handshakes.

For a detailed description of the PDOS loadable driver mechanism please refer to the Appendix on the PDOS Operating System Manual.

Example: >CF PRS:SR,PRS this prints a listing of the Driver on the serial printer

5.Ø FORCE HARDWARE IMPLEMENTATION

FORCE BIOS Modules for 32-Bit CPU-Boards

The Basic Input/Output Systems (BIOS) consists of code in the operating system which is specific to the FORCE hardware. This code is generated from source files supplied to the user for inspection and alteration. The following files comprise the FORCE BIOS:

FILENAME	DESCRIPTION
FPARM:SR	Global memory Definitions. These are common for all FORCE CPU-boards.
F32PARM:SR	Device addresses and register offsets for all FORCE 32-Bit CPU-boards.
F32BIOS:SR	Startup, Interrupt, and clock routines.
MBIOS:SR	General startup & tables for all PDOS systems.
F32BIOSU:SR	UART (Character I/O) code for the devices on the CPU-boards.
FBIOSU:SR	UART Code for the SYS68K/SIO-1/2 and the SYS68K/ISIO-1/2
FBIOSW:SR	Disk code.

5.1 FORCE PARAMETER FILES

The file FPARM:SR defines low ram locations for flags and the disk partition table. This file is used with the BIOS modules, as well as by other special routines (e.g. FFRMT and F32BOOT). Some low memory locations are used in the BIOS for flags, counters in disk parameters. These are outside of the standard PDOS. The labels that are defined externally for linking by some programs are prefixed with a `P$' as a help in remembering where the label is defined; That is the Parameter file. These parameters are located at absolute memory locations starting at address $4ØØ. A definition of FPARM:SR labels along with their meanings follow:

Label	Address	Meaning
	$4ØØ	1 byte. Reserved.
P$WFCF	$4Ø1	1 byte. Number of WFC controller cards in system. If Ø then no WFC controller cards were found by boot ROMS.
P$SIOF	$4Ø2	1 byte. Number of SIO cards found by PDOS startup code.
P$ASCF	$4Ø3	1 byte. Number of ASCU cards in the system.

Label	Address	Meaning
P$FPCPF	$4Ø4	1 byte. This flag is Ø if no 68881 Co-processors are available. Otherwise it is set to one.
P$SCSIF	$4Ø5	1 byte. The number of ISCSI-1 controller cards in the system.
P$ISIOF	$4Ø6	1 byte. The number of ISIO-1/2 controller cards in the system.
	$4Ø7-$4ØF	These locations are reserved for future use.
P$DRW	$41Ø	1 long word. This location is used to hold the address of the disk read/write driver in the F32BIOSW module. The system will crash if this location is destroyed.
P$DFORM	$414	1 long word. This location holds the entry of a specific disk format routine in the FBIOSW driver. It is only used by FFRMT. Currently there is only an entry if the ISCSI-1 controller is used.
P$DMAC	$418	1 long word. This location holds the address of a 6845Ø DMA controller to be used for disk I/O. It must be Ø if no DMA controller is available.
B$SRAM	$41C	1 long word. This location holds the begin address of the System RAM.
P$CPU	$42Ø	1 word. This word is used to hold an identification of the CPU-board. Currently, only the CPU-2Ø/21 is supported by a value of Ø.
P$ISINT	$422	1 word. This word contains the high/low water flags for up to two ISIO-1/2 boards.
STOLOC	$424	1 byte. Time out value for WFC controller. This byte flag is the Select Time-Out counter used only by the WFC-1 code. It is set to 5 by the FBIOSW module when leaving a read or write sector call. Then it is decremented once a second by the W$XDOF routine until it goes to zero. When it reaches zero, the disk is deselected.
P$BDSK	$425	1 byte. This byte variable is loaded by the Boot ROMs with the disk number from which PDOS was booted. The BIOS module reads this location to set the default system disk for PDOS. The FLDGO utility also loads P$BDSK with SDK$ from the user's TCB to simulate booting from a disk.

Label	Address	Meaning
P$INTF	$426	1 byte. Flags for FBIOSW and F32BIOS to indicate if interrupts or DMA are to be used on either the disk controllers or the ISIO board. The flags are single bits defined as followed: ISAS BIT Ø SASI Interrupts if 1 IWFC BIT 1 WFC Interrupts if 1 DSAS BIT 2 SASI DMA move if 1 DWFC BIT 3 WFC DMA move if 1 ISCSI BIT 4 ISCSI-1 Interrupts if 1 DSCSI BIT 5 ISCSI-1 DMA move if 1
P$LEDB	$427	1 byte. Reserved.
SAVE	$428	2 long words. These are 2 long words used by W$XDIT.
P$UADR	$43Ø	1 long word to save the address of the interrupting UART.
P$PARM	$434	6 long words. Disk drive definition tables for PDOS disk standard. The drive parameters are stored in the locations following P$PARM.

FPARM also has definitions for the offsets and addresses of the various control registers to the hardware which is supported on the standard FORCE system. The hardware includes: SIO and ISIO cards, ISCSI, WFC, ASCU and DMA controller.

The file F32PARM:SR holds definitions which are specific for the FORCE 32-bit CPU-boards. It contains the base definitions and register offsets for the devices on the CPU-boards.

5.2 FORCE BIOS MODULE : F32BIOS:SR

The file F32BIOS:SR has machine-specific, interrupt, clock and mapper routines. Assembling this file includes the files MBIOS:SR, which has general startup code common to all systems, the file FPARM:SR which provides global definitions of the low memory locations, and the file F32PARM:SR, which holds CPU-board specific device addresses and register offsets. The file F32BIOS:SR is the main module for the PDOS BIOS and is assembled into the file F32BIOS:OBJ.

5.2.1 BIOS Labels

The labels that are defined externally for linking with other modules are prefixed with a `B$'as a help in remembering where the label is defined. The following describes the labels as they are used in F32BIOS:SR:

B$CPU: The code of this label performs the FORCE hardware dependent setup. It first tries to find out which CPU-board is used in the system and then initialzes the interrupt structure and timer of the board.

Then the availability of the 68881 Floating Point Co-processor is checked and a flag is set in low memory.

Finally the VMEbus is checked for the availability of different FORCE hardware.

The following boards are tested and counted:

ISCSI-1	SCSI/floppy controller
✗ WFC-1	Winchester/Floppy Controller
SASI-1	SASI Bus Controller
SIO-1/2	6-Channel Serial I/O Board
ISIO-1/2	Intelligent 8 Channel Serial I/O Board
✗ ASCU-1/2	System Control Unit

B$RAM: This routine reads the front panel switches of the CPU-board and sets the data bus width to the corresponding size. Then the memory is initialized to Ø and checked for read/write capability until a Bus error is generated. The so found top of memory is handed over to PDOS for end of tasking memory.

B$RSW: Here, the front panel switches of the CPU-board are read in to determine the baud rate and character format for the terminal port. The default is 96ØØ Baud, 8 data bits and no parity. If a different setup has to be used, the front panel switches have to be changed. Note that in this case the PDOS system is not romable.

SYS68K/CPU-2Ø/21 front panel switch definitions under PDOS:

SW1: Not used by PDOS.

SW2: This switch defines the default Data bus width on the VMEbus/VMXbus. If in "On" position, the data-bus width is 16 bit, if in "Off" position, the size is 32 bit.

SW3: If this switch is set on "On", PDOS is booted automatically after reset has been pressed.

SW4: Not used by PDOS.

SW5: This switch defines the default setting for the serial I/O devices. If in "Off" position, the serial I/O is initialized to 96ØØ Baud, 8 bits per character and no parity. In this case, switches 6-8 are free for user definition. If this switch is in "On" position, the switches 6-8 define the setup for the serial I/O devices.

SW6: The switches 6-8 are only valid if SW5 is set on "On". Switch 6 defines the bits per character. If in "On" position, 7 bits/character with 2 stop- bit are used, if in "Off" position, 8 bits/character with 1 stop-bit is used. Parity is always disabled.

SW7:
SW8: These two switches defines the baud rate for the serial I/O if SW5 is set to "On". The following summarizes the possible baud rate setups:

SW7	SW8	Baud rate
OFF	OFF	96ØØ
OFF	ON	48ØØ
ON	OFF	24ØØ
ON	ON	12ØØ

B$SAV: This routine is used to save the context of the 68881 co-processor. If the co-processor flag in the task control block (SVF$) is set, this routine is called on every context switch.

The register A1 holds the user stack and A7 the supervisor stack. As the supervisor stack area is too small, it is recommended that the co-processor contents are saved on the user stack.

B$RES: This is the complementary subroutine to B$SAV. It is called if the next task to be rescheduled has the flag SVF$ on the task control block set.

The values to be loaded to the co-processor have to be popped off the stack. The same stack as in B$SAV has to be used.

B$ACK: This is the acknowledge routine for the system timer interrupt.

B$LED: This performs the fine adjustment of the PDOS System Clock.

BPRT, BMAP: There is neither a protect PDOS (B$PRT) routine nor a mapper (B$MAP) routine in the BIOS for the FORCE series of CPU-boards.

5.2.2 BIOS Interrupts

All interrupts of the default PDOS configuration are handled in the BIOS module. All interrupts are programmed to use different interrupt vectors.

The following interrupt sources are supported by the standard PDOS BIOS:

- On-board I/O devices (including the ABORT switch)
- Disk I/O
- Up to two SIO-1/2 boards for serial I/O
- Up to two ISIO-1/2 boards for serial I/O

The on-board I/O devices are programmed to use the auto vector part of the vector table to leave the user interrupt vectors as free as possible.

The disk controller uses vector #119. The SIO-1 serial I/O controller uses the vectors #64 - #75, while the ISIO-1/2 board uses the vectors #76 - 83. Each serial channel has its own interrupt vector for the SIO-1/2 boards, but the two serial I/O channels of the ISIO-1/2 board share one vector.

5.2.3 On-board Interrupts

The handling of on-board interrupts differs for the different CPU-boards. For a detailed description please refer to the source of the file F32BIOS:SR.

5.2.4 Disk Interrupt Vector

Exception vector 119 is used by the disk module FBIOSW. The interrupt service routine W$DINT sets the PDOS event 119 and returns. The event number and vector address is defined by the W$EVNT equate in the disk module FBIOSW.

5.3 COMMON BIOS MODULE : MBIOS:SR

The common BIOS module, MBIOS:SR contains the main PDOS cold entry point and startup code, the BIOS table, some default definitions, and a RAM disk initialize routine. MBIOS:SR is included when the file F32BIOS:SR is assembled.

5.3.1 MBIOS System Constants

If certain system constants are not defined either by the user in the MASM command line or in the F32BIOS:SR file itself, MBIOS defines them to some default values. A user can alter these temporarily by either adding a divide sign `/', the label, and the desired value to the source file parameter of the MASM assembler. (See section 2.4.1 for an example and details.) More permanent values can be assigned by inserting equate (EQU) statements in the F32BIOS:SR file.

5.3.2 Task Startup Table

The following labels define the task Ø values to place in the task startup table. This table is used by PDOS to determine what tasks to start up initially, on what ports, at which priorities and what size:

R$TPØØ	64	Task priority
R$TMØØ	1	Task time
R$DZØØ	Ø	DSEG size (Ø=use all of memory)
R$MPØØ	Ø	Map # (unused)
R$PMØØ	*-*	PSEG relative start addr (Ø=Monitor)
R$PNØØ	1	Port #

Note that the B$RSW routine in the F32BIOS module changes the startup table at the label R$TASK to set the baud rate of the terminal according to the front panel switches. So the BIOS module is not romable.

To make the code romable, just remove the lines of the baud rate setting, but in this case the baud rate cannot be set by the front panel switches.

5.3.3 Kernel Callout Subroutines

There are eight subroutines in the BIOS called by the PDOS kernel while running:

BACK, BLED, BMAP, BCTB, BKTB, BPRT, B$SAV and B$RES

The kernel makes these calls to addresses in the BIOS table. These eight routines are not necessarily implemented in a particular PDOS system, but if they are, they reside in the F32BIOS:SR file for that system. If the label is defined in F32BIOS:SR, the MBIOS:SR will assemble a long branch (BRA.L B$xxx) to that subroutine. If the label is not defined, MBIOS:SR will assemble a return (RTS) instruction. The details of these routines are contained in section 5.2 and in the **PDOS Reference Manual.**

5.3.4 Screen Command Table

MBIOS also contains a screen command table for BASIC PRINT support. The table is outlined below:

```
        Screen Command Table

SCRNTB  DC.B    $ØB, `U'         ;U = UP
        DC.B    $ØA, `D'         ;D = DOWN
        DC.B    $ØC, `R'         ;R = RIGHT

        DC.B    $Ø8, `L'         ;L = LEFT
        DC.B    $ØD, `B'         ;B = BEGINNING
        DC.B    $1E, `H'         ;H = HOME
        DC.B    $D9, `S'         ;S = CLEAR TO END OF SCREEN
        DC.B    $D4, `E'         ;E = CLEAR TO END OF LINE
        DC.B    $A7, `W'         ;W = RESET WRITE PROTECT
        DC.B    $A6, `P'         ;P = SET WRITE PROTECT
        DC.B    $A9, `('         ;( = START WRITE PROTECT
        DC.B    $A8, `)'         ;) = END WRITE PROTECT
        DC.B    $AB, `Z'         ;Z = CLEAR UNPROTECTED
        DC.B    $Ø9, `N'         ;N = SKIP TO NEXT FIELD
        DC.B    Ø                ;END-OF-TABLE
```

5.3.5 System String Constants

Two PDOS system string constants are defined in the MBIOS:SR file: the system auto start file and the RAM disk name. They may be altered as the user desires. The default names are:

```
SY$STRT = Auto start file name
SY$DSK  = RAM disk name
```

5.4 UART MODULES

5.1.4 FORCE UART Module : F32BIOSU:SR

The file F32BIOSU:SR is the UART module for the CPU-board UARTs and contains routines that handle the character I/O devices on the FORCE CPU-boards. Various entry points and assembly definitions are provided in F32BIOSU:SR. The labels that are defined externally for linking with other modules are prefixed with a `U$'as a help in remembering where the label is defined.

The terminal port on the processor board along with all additional UART devices are defined as type 1 UARTs.

Only port 1 is bauded by PDOS after booting. The procedure file SY$STRT is provided on the boot disk to baud all additional CPU-ports.

5.4.2 FORCE UART Module: FBIOSU:SR

5.4.2.1 Type 2 -- SIO-1 Boards

Up to two SIO-1/2 boards are supported by the standard PDOS. They are defined as type 2 UARTs.

These boards have to be set to default address for the first board and to default +$200 for the second board.

Each channel has its own interrupt vector defined. The used vectors are #64 - 75. The interrupts are handled in the BIOS module.

The SIO-1/2 boards allow many different combinations for the I/O signal assignment. Please refer to chapter 5.2, or to the Hardware User's Manual of the SIO-1/2 boards for detailed information.

The program FBAUD is provided to baud a maximum of 15 serial I/O channels for the PDOS operating system.

5.4.2.2 Type 3 -- ISIO-1/2 Boards

Up to two ISIO-1/2 boards are supported by the standard PDOS. They are defined as type 3 UARTs.

These boards have to be set to default address for the first board and to default address +$20000 for the second board.

Each ISIO-1/2 board supports 4 different interrupt vectors. Therefore two channels share the same vector. The used vector numbers are #76 - 83. The interrupts are handled in the BIOS module.

For a detailed description of the ISIO-1/2 board, and for I/O pin assignment description, please refer to the User's Manual of the ISIO-1/2 board, or to chapter 5.2 of this manual.

The program FBAUD is provided to baud a maximum of 15 serial I/O channels for the PDOS Operating System.

5.4.2.3 Type 4 -- Currently Not Used

5.5 FORCE DISK MODULE : FBIOSW:SR

The file FPARM:SR is included when FBIOSW:SR is assembled to provide global definitions of disk parameters and low memory locations. In addition the physical driver files "ISCSI1:SR" (for the ISCSI-1 board), "WFC1:SR" (for the WFC-1 board) and "SASI-1" (for the SASI-1 controller) are included at assembly time. The labels that are defined externally for linking with other modules are prefixed with a `W' as a help in remembering where the label is defined .

Label W$XDIT is called to initialize the disk controllers. Up to two floppies and 3 Winchester drives are supported by the by the FBIOSW module.

5.5.1 PDOS Unit Defaults and Partitions

Normally PDOS disk numbers Ø and 1 are for the Floppy disk drives. The default floppy disk is disk unit Ø in a single floppy system. Standard PDOS floppy disks treat track Ø as hardware dependent. So units Ø and 1 will access the floppy disks starting with track 1. Units 1ØØ and 1Ø1 may be used to access the floppy disk starting with sector Ø. The Winchester partitioning must use units less than 1ØØ. Units 2ØØ through 299 are used by FBIOSW to access partition information on drives.

PDOS DISK	DESCRIPTION
Ø	First Floppy Disk
1	Second Floppy Disk
2 through 99	Unit numbers for Winchester
1ØØ	Unbiased floppy disk Ø
1Ø1	Unbiased floppy disk 1
2ØØ through 299	Winchester Partition table access.

The Winchester may be partitioned into multiple disks without regenerating PDOS. This is fully described in section 2 of this manual. The partitioning is handled by tables that are read off track Ø of each Winchester drive. If no valid table is found or no Winchester is installed, a default table is loaded. The default Winchester type and partitioning depends on the controller used.

The parameters are easily changed by using the FFRMT utility provided with your system.

The FORCE PDOS system allocates, in its system parameter RAM at $434, a table (P$PARM) of six Drive Data Block addresses. Two drive data block addresses are for floppy drives (FØ & F1) and four are for Winchester drives (WØ to W3). The addresses of the drive data blocks (or DDBs) are stored by the FBIOSW disk initialize routine, W$XDIT, when PDOS first starts up. The actual DDB's are stored in the systems Parameter RAM area by W$XDIT immediately following the six addresses of the P$PARM table.

The two Floppy Drive Data Blocks are built without accessing the drives, using default parameters, since the floppy drives are common to each system, they have only one partition and do not have bad tracks.

The only difference between the FØ and F1 tables is usually the drive select bytes and the disk number is set respectively to Ø and 1. The drive data block right after the six address pointers corresponds to floppy drive FØ. The next DDB in RAM defines floppy drive F1.

The disk initialize routine next searches each possible Winchester drive controller for attached drives by reading the first sector (or block) using all the possible select codes. If a Winchester drive is found installed (no read error), then W$XDIT determines if the header data is actually PDOS Winchester information. If it is correct, then the data is moved into a DDB in system parameter RAM, and the address is saved in the next available P$PARM table location. If the drive is installed but the header data is not a valid PDOS Winchester information, then W$XDIT moves down some default drive data into the DDB in P$PARM. This allows booting a new system and still allowing access to files on non-PDOS Winchester standard drives.

The four Winchester Drive Data Blocks are filled as W$XDIT finds them in the system, altering the controller number and drive select bytes to match where the drive was found. The first Winchester's Drive Data Block is usually read into the systems Parameter RAM area by W$XDIT immediately following the 2 floppy DDB's. We refer to it as drive `WØ', but it may be attached to any controller. W1, W2, and W3 are also loaded if drives are found.

5.5.2 Error handling

FBIOSW performs no retries before an error is reported back to PDOS.

The following Disk errors detected are reported to PDOS. Errors concerning the state of the controller are also reported.

DISK ERROR DEFINITIONS:

Common Errors

100	Illegal Drive
101	Sector too big
102	Timeout

WFC-1 Errors

103	Write Protected
104	AM not found
105	No TK00
106	Not ready
107	DMA error
108	Sect Not Found
109	ID read
110	Uncorrectable data

SASI-1 Errors

111	Bad Block
112	Channel active
113	No index
114	No seek
115	Write Fault
116	Not ready
117	Not selected
118	No TK00
119	Multiple select
125	Seek IP
128	ID read
129	Uncorrectable data
130	IDAM not found
131	DTAM not found
132	RNF
133	Seek error
135	Write protected
136	Corrected data
137	Bad track
138	Format error
143	Seq TO
144	Bad cmd
145	Illegal disk adr
146	Illegal function

ISCSI-1 Errors

147	SCSI Parity error
153	Illegal Command
157	Address Error
158	ISCSI-1 Exception
159	Target Mode
161	SCSI timeout
162	LUN not ready
163	SCSI read error
164	SCSI write error
165	Block not found
166	Target reservation conflict
167	Format error
172	Floppy not ready
173	FDC read error
174	FDC write error
175	FDC record not found
177	FDC format error

5.5.3 Data movement and Interrupts

The disk driver uses programmed I/O with interrupts. This allows lower priority tasks to execute during disk accesses. A flag in low memory, P$INTF, signals the FBIOSW routine whether or not to use interrupts. If the bit ISCSI (for a ISCSI-1 controller) in P$INTF is zero, then the command RAM is polled for Channel Operation Complete to be set. This mode is used by the W$XDITS routine and the boot EPROMs since using interrupts requires that PDOS is in control. Upon exit, W$XDITS sets the bit P$INTF to 1 enabling completion interrupts. The read/write routine checks this flag and when it is set, sets the local event (128) to delay for three seconds. This is a timeout in case the controller goes away or fails to interrupt. Then suspends on event 128 and system event 119. When the controller is interrupting or after three seconds. When the task awakens, event 119 is automatically reset for the next disk request. If the timeout awoke the task, then the flag in P$INTF is reset and regular polling of the command completion is entered.

FORCE COMPUTERS®

March 7,1988

miniFORCE 2P/32 User's Manual Update

from PDOS Rev. 3.2a to Rev. 3.3a

Please replace the following chapter in your User **Manual:**

Chapter 4: Standard miniFORCE 2P/32 Configuration

Chapter 7: OPERATING SYSTEM **INSTALLATION**

Chapter 8: USER NOTES

Thank you.

FORCE COMPUTERS GMBH

miniFORCE 2P/32

USER'S MANUAL

Second Edition
February 1988

800133

FORCE COMPUTERS Inc./GmbH
All Rights Reserved

STANDARD miniFORCE 2P/32
CONFIGURATIONS

TABLE OF CONTENTS

1.0 GENERAL INFORMATION

The miniFORCE 2P/32 systems are 32 bit based systems. They contain modules which are the standard part of the whole system configuration. These modules, combined with different FORCE VMEbus based boards, result in a configuration which will be described in this section. By using the standard system configuration, additional boards can be installed in the system for the final application.

All of these systems, named until now "miniFORCE 2P/32", contain the same modules, the same chassis (7HE), and the same operating system (PDOS*). Therefore, the name of the system exactly describes the basics and the dimensions of the system.

For example, the miniFORCE 2P21 means that the system is a product with a 7HE chassis (miniFORCE 2) and contains the above described modules. Additionally, the "P" means that it is PDOS* operating system based and the "21" on the end means that the system contains the 68020 based fully 32 bit main processor board SYS68K/CPU-21.

2.0 THE DEFAULT SYSTEM CONFIGURATIONS

2.1 The "miniFORCE 2P21"

The miniFORCE 2P21 is a multiuser development/target system for real time applications, providing high performance and low cost. The 68020 offers a real time computing power of 2.5 to 3.5 MIPS, running without wait states out of 512 Kbyte on-board static RAM. The implemented operating system, PDOS, supports all features of the system.

The System Features

- Processor 68020 (16.7 MHz)
- Floating Point Coprocessor 68881 (16.7 MHz)
- 512 Kbyte high speed local static RAM
- 2 serial I/O interfaces
- 51 Mbyte Winchester
- 1 Mbyte floppy
- One Timer
- Real Time Operating System PDOS*
- Full support of all Interfaces under PDOS*

VMEbus boards used:

SYS68K/CPU-21
SYS68K/WFC-1

2.1.1 The Central Processor Unit

This high performance VMEbus-based CPU board combines the powerful 32 bit microprocessor, the 68020 with 512 Kbyte of static RAM, a maximum of 512 Kbyte of ROM/EPROM, powerful I/O devices, and a Floating Point Coprocessor.

The local control, consisting of a Parallel Interface and Timer (PI/T) and a Bus Interrupter Module (BIM), supports software control for a wide variety of functions. The SYS68K/CPU-21 is equipped with two RS232 compatible serial I/O channels. The 68881 Floating Point Coprocessor included on the CPU-21 speeds up for number cruncher applications.

The SYS68K/CPU-21 boards contain a primary VMXbus master interface.

Easy access to the devices is provided by the ROM-resident SYSTEM MONITOR, called the SYS68K/20 FORCEbug. This powerful software package can be used for program development and debugging of application programs.
The SYS68K/CPU-21 has been developed as a sophisticated VMEbus boards which may be used in a multiprocessor environment as a master processor, as a standalone unit or as a high performance coprocessor board.

To provide all of these features, CPU-21 consists of two double Eurocard boards, the main board and the slave board. All of the bus interfaces, the CPU and the EPROMs are installed on the main board.

The slave board contains on e serial I/O interface as well as the 512 Kbyte Zero Wait State Static RAM.

Features of the SYS68K/CPU-21

- 68020 CPU with 16.7 MHz Clock Frequency
- 68881 Floating Point Coprocessor with 16.7 MHz Clock Frequency
- Static RAM 512 Kbyte with 55ns access time
- 512 Kbyte (max) of ROM/EPROM for the system
- 2 RS232 Multiprotocol Communication Interfaces (110-38400 Baud)
- Parallel Interface and Timer Chip provides local control and timer function
- VMXbus Primary Master Interface to P2 connector
- Local interrupt handling via interrupt vectors
- Each VMEbus IRQ level can be enabled/disabled via software
- Address range for the short I/O address modifiers (AM4)
- Address range for the standard address modifier
- Single level bus arbiter
- Single level slave bus arbitration
- Power monitor
- RESET and SOFTWARE ABORT function switches
- Fully VMEbus, VMXbus and IEEE 1014 compatible

2.1.2 The Main Memory

The main memory of the miniFORCE 2P21 is placed on the CPU-21 slave board itself (512 Kbyte). An increase of·this memory is possible via additional boards on the VMEbus. This possibility is described in Section 6 of this manual, "System Expansion".

The base address of the included local memory on the CPU-21 is jumpered to 000000 HEX. An additional memory board must be jumpered to the base address 080000 HEX if no additional local RAM is installed. If an additional RAM board is installed on the CPU-21, then the base address of the VME memory must be jumpered to 100000 HEX.

2.1.3 The Mass Storage Controller

The SYS68K/WFC-1 is a high performance VMEbus interface controller board to control 5 1/4 inch Floppy SA460 Interface and Winchester drives (ST506 interface).

The SYS68K/WFC-1 board has been developed to serve as a controller board for high speed data transfers to and from 5 1/4 inch floppy and Winchester drives via a 1 Kbyte FIFO buffer. The board can communicate with a DMA controller to provide maximum bus transfer speed and is able to generate interrupts for complete operation, error and data request.

The VMEbus allows easy system design with extended I/O, RAM, CPU, ROM, and DMA cards.

The SYS68K/WFC-1 consists of a set of devices specifically designed for control of Winchester and floppy disk drives. The heart of the control logic is the Control Processor Buffer Manager (WD1015) that manages the on-board static RAM sector buffer (2048-word by 8-bit). All bytes of data written to and read from disk are first stored on this sector buffer. When the buffer is full, the data is transferred, on command, to its intended destination.

The WD1015, besides controlling the data flow between host, sector buffer, and disk controllers; also translates the host Winchester command format to floppy disk format when addressing the Floppy Disk Controller (WD2797). This permits the host to maintain a single command format (Winchester) while in effect controlling two different disk command formats (Winchester vs. floppy). This is possible, since the SDH register is used to select either type of drive.

The WD1015 maintains the current copies of necessary host command data in the task files; a set of registers physically located in the Winchester Disk Control device (WD1014).

The WD1010 is the link between the host processor (via sector buffer) and the Winchester disk drives. During transfer of data from the host to the WD1010, the WD1014 computers a 4 byte ECC, which is appended to the end of the data being transferred to the WD1010 and recorded on the disk. During data transfers from WD1010 to the host (via the sector buffer), the WD1015 uses the ECC syndrome to validate the data. Retries and corrections are attempted automatically in case of corrupted data.

The WD1015 performs error correction in conjunction with WD1014 on data transferred to the disk. While the WD1015 controls the operation of the on-board error correction logic, the WD1014 generates and checks the Error Correction Code (ECC) if SDH bit 7 = 0. Thus the WD1014 also provides the WD1015 with its real time control capability.

If CRC format Winchester disks are used, CRC is selected by the WD1010 by setting SDH7 = 0. CRC for the floppy disk is performed by the WD2729, a device that furnishes all control functions for floppy disk drives, including necessary data separation and write precompensation. SDH7 must be set to zero for floppy disk operation.

The jumper setting for the installation on the miniFORCE 2P21 is described in Section 5 ("VME Boards Set Up"). Features of the WFC-1 are shown below.

Features of the SYS68K/WFC-1

- Fully VMEbus compatible
- Jumper selectable base address with address modifier
- Generation of two different interrupts
- Jumper selectable interrupt level
- Software programmable interrupt vectors
- Three VMEbus options (A31:D16), (A23:D16), (A15:D16) jumper selectable
- User selectable 5.25 inch Winchester or floppy operation
- Controls up to 3 Winchester ST506 Interface and up to 4 floppy drives SA460 compatible
- On-board data separation circuitry
- On-board write precompensation for floppy and hard disks
- On-board sector buffer supports up to 1 Kbyte sectors

Features of the SYS68K/WFC-1 con't

- Programmable sector sizes - 128, 256, 512, or 1024 bytes
- Automatic track formatting on hard and floppy disks
- Multiple sector operations on all disks
- Data rates up to 5 Mbits/sec on hard disk
- Single burst error correction up to 5bits on hard disk data
- CRC generation/verification for data and all I.D. fields
- Automatic retries on all errors with simulated completion
- ECC diagnostic commands included (READALONG & WRITEALONG)
- Internal diagnostics
- 16 different stepping rates for both hard and floppy drives

2.1.4 The Mass Storage Drives

The miniFORCE 2P21 contains one slim line 5 1/4 inch Winchester drive and one slim line 5 1/4 inch floppy drive.

Both drives are installed in the module, which is accessible on the front of the system. The drive select codes for these units are:

Winchester:	DS0
Floppy:	DS2

The installed Winchester is a half height disk drive which can store 51 Mbyte of unformatted data. Winchester technology assures trouble-free performance, and the rugged, shock resistant design makes the drive perfect for use in compact, portable systems. With only two required voltages (+12V and +5V) and its ST506 (ST412) type interface, the drive can easily be installed into the system.

Features of the Winchester Drive

Unformatted capacity:	51 Mbyte
Formatted capacity: (256 byte/sector)	39.2(40.96) Mbyte
Number of disks	3
Number of heads	5
Number of cylinders	981 (1024)
Data Transfer Rate	625 Kbyte/s
Access Time Average	40ms
Recording Method	MFM
Non-operating Shock	40G at 10ms
Disk speed (rpm)	3564
Drive selections	4
Power Dissipation (average)	20W

Dimensions:

Width	146mm	(5.75 in)
Height	41mm	(1.62 in)
Depth	208mm	(8.2 in)
Weight	1.8kg	(4 lbs)

Reliability:

MTBF	20000 POH
MTTR	30 minutes

The installed TEAC FD-55FV is a highly reliable 5 1/4 inch slim line floppy disk drive.

The motor on the drive is equipped with precision digital locked control, an enhancement which ensures unerring accuracy under all operating conditions. The power consumption is an ultralow 1.5W in the nonoperating mode, and only 4.1W during actual operation.
A band actuator system works to position the magnetic heads rapidly and positively.

Features of the TEAC FD-55FV Floppy Disk Drive

Capacity unformatted	1 Mbyte
Capacity formatted (256byte/sector)	655 Kbyte
Tracks/Disk	160
Track Density	96tpi
Average Access Time	94ms
Track Access Time	3ms
Disk Rotation Speed	300rpm
Recording Method	MFM
MTBF	10000 POH
Power Requirements	+5V 0,3A(type) 0.4A(max) +12V 0,22A(typ) 0.55A(max)
Safety Standard	UL and CSA
Power Consumption	
Operating:	4.1W
Non-Operating:	1.5W
Dimensions (WxHxD)	146 x 41.3 x 203mm 5 3/4 x 1 5/8 x 8 in

2.1.5 The Address Map

The following outline gives detailed information about the memory map, including the on-board memory, device addresses, and standard addresses for the miniFORCE 2P21 cards.

Address	**Definition**
$00000000-$0007FFFF	Local SRAM
$00080000-$000FFFFF	Local SRAM on SRAM-22 board (optional)
$FF000000-$FF07FFFF	EPROM Area 1 on board I/O devices

$00080000-$FAFFFFFF	VMEbus memory if no additional SRAM-22 board is installed
$00100000-$FAFFFFFF	VMEbus memory if an additional SRAM-22 board is installed
$FCB01000-$FCB0100F	WFC-1 card
$FCB02000-$FCB022FF	ASCU-1/2 Card (optional)
$FCB00000-$FCB001FF	First SIO-1 Card (optional)

Interrupt Levels:

AV Level 7:	Abort switch
AV Level 5:	Timer chip
AV Level 4:	On-board serial I/O, WFC-1

To BAUD all the ports in the miniFORCE 2P1 system, use the following:

BP 2,1,1,$FF800200	Port 1 Terminal
BP 3,1,1,$FF800600	Port 2 Terminal (optional, if SRAM-22 is installed)

2.2 The miniFORCE 2P21S

The miniFORCE 2P21S is a downgrade of the miniFORCE 2P21. The features of the system are generally the same but only the coprocessor (68881) and the processor (68020) speed is reduced to 12.5 MHz. The winchester drive capacity is also reduced to 20 Mbytes. Therefore, a low price can be offered for a 68020 based VMEbus system with an acceptable performance.

The features of the miniFORCE 2P21S system are as follows:

- Processor 68020 (12.5 MHz)
- Floating point coprocessor 68881 (12.5 MHz)
- 512 Kbyte local memory with zero wait state
- 2 serial I/O interfaces
- 20 Mbyte Winchester
- 1 Mbyte Floppy
- One Timer
- Real Time Operating System PDOS*
- Full support of all Interfaces under PDOS*

VMEbus boards used:

SYS68K/CPU-21S
SYS68K/WFC-1

2.2.1 The Central Processor Unit

This high performance VMEbus-based CPU board combines the powerful 32 bit microprocessor, the 68020 with 512 Kbytes of static RAM, a maximum of 512 Kbyte of ROM/EPROM, powerful I/O devices, and a Floating Point Coprocessor.

The local control, consisting of a Parallel Interface and Timer (PI/T) and a Bus Interrupter Module (BIM), supports software control for a wide variety of functions. The SYS68K/CPU-21S is equipped with two RS232-compatible serial I/O channels. The 68881 Floating Point Coprocessor included on CPU-21S speeds up for number cruncher applications.

The SYS68K/CPU-21S boards contain a primary VMXbus master interface.

Easy access to the devices is provided by the ROM-resident SYSTEM MONITOR, called the SYS68K-20 FORCEbug. This powerful software package can be used for program development and debugging of application programs.

The SYS68K/CPU-21S has been developed as a sophisticated VMEbus board which may be used in a multiprocessing environment as a master processor, as a standalone unit or as a high performance coprocessor board.

To provide all of these features, CPU-21S consists of two double Eurocard boards, the main board and the slave board. On the main board all the bus interfaces, the CPU and the EPROMs are installed.

The slave board contains one serial I/O interface as well as the 512 Kbyte Zero Wait State Static RAM.

Features of the SYS68K/CPU-21S

- 68020 CPU with 12.5 MHz Clock Frequency
- 68881 Floating Point Coprocessor with 12.5 MHz Clock Frequency
- 512 Kbyte static RAM with 55ns access time
- 512 Kbyte (max) of ROM/EPROM for the system
- 2 RS232 Multiprotocol Communication Interfaces (110-38400 Baud)
- Parallel Interface and Timer Chip provides local control and timer function
- VMXbus Primary Master Interface to P2 connector
- Local Interrupt handling via interrupt vectors
- Each VMEbus IRQ level can be enabled/disabled via software
- Address range for the short I/O address modifiers (AM4)
- Address range for the standard address modifier
- Single level bus arbiter
- Single level slave bus arbitration
- Power monitor
- RESET and SOFTWARE ABORT function switches
- Fully VMEbus, VMXbus and IEEE 1014 compatible

2.2.2 The Main Memory

The main memory of the miniFORCE 2P21S is placed on the CPU-21S (512 Kbyte) by default. The memory is static RAM based and the processor runs with zero wait states out of this memory.

There are two possible ways of increasing the memory. First, the static RAM can be increased through an additional memory board call SRAM-22 which has 512 Kbyte static RAM space. The benefit of this is that the processor runs with zero wait states out of 1 Mbyte of memory. Second, an increase in the VMEbus memory capacity is possible via VMEbus based memory boards. In this case, the memory board must be jumpered with the address 80000 HEX.

If an SRAM-22 is already installed, the base address of the VMEbus memory board must be jumpered at 100000 HEX to guarantee contiguous memory.

2.2.3 The Mass Storage Controller

The SYS68K/WFC-1 is a high performance VMEbus interface controller board to control 5 1/4 inch Floppy SA460 Interface and Winchester drives (ST506 interface).

The SYS68K/WFC-1 board has been developed to serve as a controller board for high speed data transfers to and from 5 1/4 inch floppy and Winchester drives via a 1 Kbyte FIFO buffer. The board can communicate with a DMA controller to provide maximum bus transfer speed and is able to generate interrupts for complete operation, error and data request.

The VMEbus allows easy system design with extended I/O, RAM, CPU, ROM, and DMA cards.

The SYS68K/WFC-1 consists of set of devices specifically designed for control of Winchester and floppy disk drives. The heart of the control logic is the Control Processor Buffer Manager (WD1015) that manages the on-board static RAM sector buffer (2048-word by 8-bit). All bytes of data written to and read from disk are first stored on this sector buffer. When the buffer is full, the data is transferred on command to its intended destination.

The WD1015, besides controlling the data flow between host, sector buffer, and disk controllers; also translates the host Winchester command format to floppy disk format when addressing the Floppy Disk Controller (WD2797). This permits the host to maintain a single command format (Winchester) while in effect controlling two different disk command formats (Winchester vs. floppy). This is possible since the SDH register is used to select either type of drive.

The WD1015 maintains the current copies of necessary host command data in the task files; a set of registers physically located in the Winchester Disk Control Device (WD1014).

The WD1010 is the link between the host processor (via sector buffer) and the Winchester disk drives. During transfer of data from the host to the WD1010, the WD1014 computers a 4 byte ECC which is appended to the end of the data being transferred to the WD1010 and recorded on the disk. During data transfers from WD1010 to the host (via the sector buffer), the WD1015 uses the ECC syndrome to validate the data. Retries and corrections are attempted automatically in case of corrupted data.

The WD1015 performs error correction in conjunction with WD1014 on data transferred to the disk. While the WD1015 controls the operation of the on-board error correction logic, the WD1014 generate and checks the Error Correction Code (ECC) if SDH bit 7 = 0. Thus the WD1014 also provides the WD1015 with its real-time control capability.

If CRC format Winchester disks are used, CRC is selected by the WD1010 by setting SDH7 = 0. CRC for the floppy disk is performed by the WD2729, a device that furnishes all control functions for floppy disk drives, including necessary data separation and write precompensation. SDH7 must be set to zero for floppy disk operation.

The jumper setting for the installation on the miniFORCE 2P21S is described in Section 5, VME Boards Set Up.

Features of the SYS68K/WFC-1

- Fully VMEbus compatible
- Jumper selectable base address with address modifier
- Generation of two different interrupts
- Jumper selectable interrupt level
- Software programmable interrupt vectors
- Three VMEbus options (A31:D16), (A23:D16), (A15:D16) jumper selectable
- User selectable 5.25 inch Winchester or floppy operation
- Controls up to 3 Winchester ST506 Interface and up to 4 floppy drives SA460 compatible
- On-board data separation circuitry

- On-board write precompensation for floppy and hard disks
- On-board sector buffer supports up to 1 Kbyte sectors
- Programmable sector sizes - 128, 256, 512, or 1024 bytes
- Automatic track formatting on hard and floppy disks
- Multiple sector operations on all disks
- Data rates up to 5 Mbits/sec on hard disk
- Single burst error correction up to 5 bits on hard disk data
- CRC generation/verification for data and all I.D. fields
- Automatic retries on all errors with simulated completion
- ECC diagnostic commands included (READALONG & WRITEALONG)
- Internal diagnostics
- 16 different stepping rates for both hard and floppy drives

2.2.4 The Mass Storage Drives

The miniFORCE 2P21S contains one slim line 5 1/4 inch floppy drive and one 5 1/4 inch Winchester drive.

Both drives are installed in the module which is accessible on the front of the system. The drive select codes for these units are:

Winchester: DS0
Floppy: DS2

The installed drive Winchester is a half height disk drive which can store 25.6 Mbytes of unformatted data. Winchester technology assures trouble free performance, and the rugged, shock-resistant design makes the drive perfect for use in compact, portable systems. With only two required voltages (+12V and +5V) and its ST506 (ST412) type interface, the drive can be easily installed into the system.

Features of the Winchester drive

Unformatted capacity:	25.6 Mbyte
Formatted capacity: (256byte/sector)	19.68 Mbyte
Number of disks	2
Number of heads	4
Number of cylinders	615
Data Transfer Rate	625 Kbyte/s
Access Time Average	80ms
Recording Method	MFM
Nonoperating Shock	40G at 10ms
Disk speed (rpm)	3564
Drive selections	4
Power Dissipation(average)	20W

Dimensions:

Width	146mm	(5.75 in)
Height	41mm	(1.62 in)
Depth	208mm	(8.2 in)
Weight	1.8kg	(4 lbs)

Reliability:

MTBF	20000 POH
MTTR	30 minutes

The installed TEAC FD-55FV is a highly reliable 5 1/4 inch slim line floppy disk drive.

The motor on the drive is equipped with precision digital locked control, an enhancement which ensures unerring accuracy under all operating conditions. The power consumption is an ultra-low 1.SW in the nonoperating mode, and only 4.1W during actual operation.

A band actuator system works to position the magnetic heads rapidly and positively.

Features of the Floppy Disk Drive

Capacity unformatted	1 Mbyte
Capacity formatted (256 byte/sector)	655 Kbyte
Tracks/Disk	160
Track Density	96tpi
Average Access Time	94ms
Track Access Time	3ms
Disk Rotation Speed	300rpm
Recording Method	MFM
MTBF	10000 POH
Power Requirements	+5V 0,3A(typ) 0,4A(max)
	+12V 0,22A(typ) 0,55A(max)
Safety Standard	UL and CSA
Power Consumption	
Operating:	4.1W
Non-Operating:	1.5W
Dimensions (WxHxD)	146 x 41.3 x 203mm 5 3/4 x 1 5/8 x 8 in

2.2.5 The Address Map

The following outline gives detailed information about the memory map including the on-board memory and the devices addresses, and standard addresses for the miniFORCE 2P21S cards.

Address	Definition
$00000000-$0007FFFF	Local SRAM
$00080000-$000FFFFF	Local SRAM on SRAM-22 board (optional)
$FF000000-$FF07FFFF	EPROM Area 1 on board I/O devices
$00080000-$FAFFFFFF	VMEbus memory if no additional SRAM-22 board is installed

$00100000-$FAFFFFFF	VMEbus memory if an additional SRAM-22 board is installed
$FCB01000-$FCB0100F	WFC-1 card
$FCB02000-$FCB022FF	ASCU-1/2 Card (optional)
$FCB00000-$FCB001FF	First SIO-1 Card (optional)

Interrupt Levels:

AV Level 7:	Abort switch
AV Level 5:	Timer chip
AV Level 4:	On-board serial I/O, WFC-1

To BAUD all the ports in the miniFORCE 2P1 system, use the following:

BP 2,1,1,$FF800200	Port 2 Terminal
BP 3,1,1,$FF800600	Port 3 Terminal (optional, if SRAM-22 is installed)

2.3 The miniFORCE 2P21A

The miniFORCE 2P21A is an upgrade of the miniFORCE 2P21. The features of the system are the same but the processor clock and the coprocessor clock are increased from 16.7 MHz to 20 MHz. This speed increase is really a performance increase because the processor runs with zero wait states out of the local RAM by 20 MHz. Therefore, a performance improvement of around 20% is accomplished. The other features of the system are absolutely identical to the miniFORCE 2P21.

Additional information can be located in Section 2.1, The miniFORCE 2P21.

2.4 The miniFORCE 2P29X

The miniFORCE 2P29X is a PDOS based development system for the SYS68K/CPU-29. The features of the system are generally the same as with the miniFORCE 2P21AI, but the CPU board will be replaced by the CPU-29X. The Winchester drive capacity is 175 Mbytes. A low price and acceptable performance can be offered for this 68020 based VMEbus system because of the structure of the CPU-29X (single board solution).

The features of the miniFORCE 2P29X system are as follows:

- Processor 68020 (12.5, 16, 25 MHz)
- Floating point coprocessor 68882 (12.5, 16, 25 MHz)
- 1 Mbyte local memory with zero wait state at 16 MHz and one wait state at 25 MHz
- 2 serial I/O interfaces on the CPU
- 175 Mbyte Winchester
- 1 Mbyte Floppy
- Two Timers
- Real Time Operating System PDOS*
- Full support of all interfaces under PDOS*

VMEbus boards used:

SYS68K/CPU-29X (CPU-29XS, CPU-29XB)
SYS68K/ISCSI-1

2.4.1 The Central Processor Unit

This high performance VMEbus-based CPU board combines the powerful 32 bit microprocessor, the 68020 with 1 Mbyte of static RAM, a maximum of 4 Mbytes of ROM/EPROM, powerful I/O devices, and a Floating Point Coprocessor.

The local control, consisting of two Parallel Interface and Timer (PI/T) devices and two Bus Interrupter Modules (BIM) supports software control for a wide variety of functions. The SYS68K/CPU-29X is equipped with two RS232-compatible serial I/O channels. The 68881 Floating Point Coprocessor included on CPU-29X speeds up for number cruncher applications.

The SYS68K/CPU-29X board contains a primary VSBbus master interface.

Easy access to the devices is provided by the ROM-resident Real Time Kernel based package called VMEPROM. This powerful software package can be used for program development, debugging and execution of programs in a real time environment.

The SYS68K/CPU-29X has been developed as a sophisticated VMEbus board which may be used in a multiprocessing environment as a master processor, as a "standalone unit" or as a high performance coprocessor board.

All of these features are implemented on one double Eurocard board.

Features of the SYS68K/CPU-29XS

- 68020 CPU with 12.5, 16, and 25 MHz Clock Frequencies
- 68882 Floating Point Coprocessor with 12.5, 16, and 20 MHz Clock Frequencies
- 1 Mbyte static RAM with 100ns access time
- 4 Mbyte (max) of ROM/EPROM for the system
- 2 RS232 Multi Protocol Communication Interfaces (110-38400 Baud)
- Parallel Interface and Timer chips provide local control and timer function
- A full 32 bit VSB interface including bus arbitration and interrupt handling is installed on all CPU-29 board versions
- Local Interrupt handling via interrupt vectors

- Each VMEbus IRQ level can be enabled/disabled via software
- Address range for the short I/O address modifiers (AM4)
- Address range for the standard address modifier
- Single level bus arbiter
- One level slave bus arbitration
- Power monitor
- RESET and SOFTWARE ABORT function switches
- Fully VMEbus, VSBbus and IEEE P1014 compatible
- Real Time Clock RTC 62421 with battery backup

2.4.2 The Main Memory

The main memory of the miniFORCE 2P29X is placed on the CPU-29X (1 Mbyte) by default. The memory is static RAM based and the processor runs with zero wait states out of this memory.

The RAM capacity can be increased through an additional memory board on the VMEbus. The memory board must be jumpered with the address 100000 HEX.

2.4.3 The Mass Storage Controller

The SYS68K/ISCSI-1 is an intelligent, high performance VMEbus interface controller board to control 5 1/4 inch Floppy SA460 Interface and SCSI Winchester drives.

A description of this board can be found in register 10 (OPTIONS) of this manual.

2.4.4 The Mass Storage Drives

The miniFORCE 2P29X contains one slim line 5 1/4 inch floppy drive and one 5 1/4 inch Winchester drive.

Both drives are installed in the module which is accessible on the front of the system. The drive select codes for these units are:

Winchester:	DS0
Floppy:	DS2

The installed Winchester drive is a full height disk drive which can store 175 Mbytes of unformatted data. Winchester technology assures trouble free performance, and the rugged, shock resistant design makes the drive perfect for use in compact, portable systems. With only two required voltages (+12V and +5V) and its SCSI type interface, the drive can be easily installed into the system.

Features of the Winchester Drive

Unformatted capacity:	175 Mbytes
Formatted capacity: (256 byte/sector)	128 Mbytes
Number of disks	5
Number of heads	8
Number of cylinders	1016
Data Transfer Rate	1.25 Mbyte/s
Access Time Average	23ms
Recording Method	MFM
Non-operating Shock	40G at 5ms
Disk speed (rpm)	3600
Drive selections	4
Power Dissipation (average)	20W

Dimensions:

Width	146mm	(5.75 in)
Height	82.6mm	(3.25 in)
Depth	208mm	(8.2 in)
Weight	2.7kg	(6 lbs)

Reliability:

MTBF	20000 POH
MTTR	15 minutes

The installed TEAC FD-55FV is a highly reliable 5 1/4 inch slim line floppy disk drive. The motor on the drive is equipped with precision digital locked control, an enhancement which ensures unerring accuracy under all operating conditions. The power consumption is extremely low; 1.5W in the nonoperating mode, and only 4.1W during actual operation.

A band actuator system works to position the magnetic heads rapidly and positively.

Features of the Floppy Disk Drive

Capacity unformatted	1 Mbyte
Capacity formatted (256 byte/sector)	655 Kbyte
Tracks/Disk	160
Track Density	96tpi
Average Access Time	94ms
Track Access Time	3ms
Disk Rotation Speed	300rpm
Recording Method	MFM
MTBF	10000 POH
Power Requirements	+5V 0,3A(typ) 0,4A(max)
	+12V 0,22A(typ) 0,55A(max)
Safety Standard	UL and CSA
Power Consumption	
Operating:	4.1W
Non-Operating:	1.5W
Dimensions (WxHxD)	146 x 41.3 x 203mm 5 3/4 x 1 5/8 x 8 in

2.4.5 The Address Map

The following outline gives detailed information about the memory map including the on-board memory and the devices addresses, and standard addresses for the miniFORCE 2P29X cards.

Address	Definition
$00000000-$00010000	Local SRAM on the CPU
$FF000000-$FF3FFFFF	EPROM Areas 1 and 2
$00100000-$FEFFFFFF	VMEbus memory
$FCB01000-$FCB0100F	WFC-1 card
$FCA00000-$FCA1FFFF	ISCSI-1 card
$FCB02000-$FCB022FF	ASCU-1/2 card (optional)
$FCB00000-$FCB001FF	First SIO-1 card (optional)
$FC960000-$FC97FFFF	First ISIO-1 card (optional)

Interrupt Levels:

AV Level 7:	Abort switch
AV Level 5:	Timer chip
AV Level 4:	On-board serial I/O

To BAUD all the ports in the miniFORCE 2P29X system, use the following:

BP 2,1,1,$FF800200	Port 2 Terminal (second RS232 on the CPU-29X)

For a detailed memory map description of local devices, please refer to the CPU-29 Hardware User's Manual.

2.5 The miniFORCE 2P32X

The miniFORCE 2P32 series is a PDOS development system for the SYS68K/CPU-32. The features of the systems are generally the same as for the miniFORCE 2P29. The SYS68K/CPU-32 is an upgrade of the SYS68K/CPU-29. The CPU has been changed to a 68030 for data caching and for a better all-round performance; therefore, there are other additions to the SYS68K/CPU-29. For more detailed information, including jumper settings, please refer to the CPU-32 Hardware User's Manual. Except where reference is made to specific hardware on the SYS68K/CPU-29, all other information described in section 2.4 of this manual is valid.

OPERATING SYSTEM INSTALLATION

TABLE OF CONTENTS

TABLE OF CONTENTS

INTRODUCTION

This guide describes the initial booting, the installation of the PDOS on the miniFORCE series systems, and how to use and customize the system to fit your needs. The system is fixed configured and a few files are supplied with PDOS which are written specifically for this system. The guide is divided into five chapters.

The first chapter, "Initial Operation", contains a step by step procedure for booting PDOS after power on . The second chapter comprises the "System Management Guide", which describes how to format both floppy disks and Winchester drives. The format utility examples are included, along with an explanation of Winchester disk numbering, bad track mapping and partitioning. Also included are the hows and whys of customizing PDOS for your system and making bootable floppy or Winchester disks. Autostart possibilities are also considered for turnkey installations.

The next chapter contains a step by step procedure of installing PDOS floppies on the Winchester. After that, a power up and boot from floppy procedure is described along with sample screen output and trouble shooting suggestions.

Chapter four describes the FORCE specific utilities for the miniFORCE 2P/32 series systems.

The final chapter, "Hardware Imp]ementation", expands the BIOS section of the PDOS Reference Manual with specific information on the miniFORCE 2P/32 systems. The BIOS startup code, the system interrupt structure, and the clock operations are described, along with the usual DISK and UART device service routines. An I/O address map and the boot ROMs are next. Finally, some hints on attaching terminals or serial printers are given along with the commands under PDOS to get output to them.

Since this INSTALLATION GUIDE is very important to first time users of PDOS or FORCE products, please report any errors, inconsistencies, or ambiguities to FORCE COMPUTERS as soon as possible.

1.0 INITIAL OPERATION

This section describes for the first time PDOS* user a step by step procedure for initially booting PDOS on the miniFORCE 2P/32 series systems. The user should consult the PDOS Reference Manual and Section 2 of this guide for a more comprehensive look at each utility.

The version 3.3 release of PDOS is meant to run in a multiprocessor system using the equivalent FORCE CPU boards.

1.1 Software Configuration

The system runs under the operating system PDOS. PDOS is already installed on the Winchester disk, partitions two and three. The system can boot from both disk partitions. All utilities, including the assembler, are installed on the Winchester; therefore, the system can be booted directly from the Winchester.

1.2 License Agreement

Please sign the PDOS license agreements. You are to keep one copy and immediately send the other copy to FORCE COMPUTERS. This agreement licenses your use of PDOS to one machine.

1.3 Hardware Configuration

In order for your system to boot PDOS, the hardware is configured to support PDOS. Therefore no jumpers have to be changed. The default jumper settings for the basic configuration are described in section 5.

1.4 Booting PDOS from the Winchester

Attach a 9600 BAUD terminal the terminal port, apply power, activate the RESET switch and start the PDOS bootstrap program. The PDOS Bootstrap program can be started as follows:

> Type "BOOT" when the VMEPROM prompt ("?") appears on the screen.

The bootstrap first initializes the disk controller in the system. It then starts to boot, first from the floppy, and next from the hard disk. For example:

```
FORCE SYS68K/CPU-x PDOS Bootstrap Revision x.x   Date
     WFC-1 installed.
```

If the bootstrap program cannot find a disk controller, the following will be displayed:

```
  FORCE SYS68K/CPU-x PDOS Bootstrap Revision x.x   Date
       No Disk Controller Installed ???
```

If you see the previous message and yet you know that there is a disk controller board installed, please check the address jumpers on the disk card and then test that the card is really accessible under FORCEbug.

After printing the banner and initializing the disk controller, the bootstrap program begins reading the header sectors of disks, number 0 through 99, for PDOS boot information. You should see some messages printed out that indicate disk errors on disks 0 and 1, since the Winchester drive gets up to speed or you insert a floppy disk.

```
  Disk #0:  Header sector error = 161
  Disk #1:  Header sector error = 161
  Out of PDOS boot disk table entries.
  I'll retry them all.
```

After speed up of the Winchester, the bootstrap should find the PDOS boot on disk #2 and display:

```
Disk #2:  FORCE CPU-x PDOS was found  >>  Sector xxxx,  Addr xxxx
           Execute it  ? _
```

You do not need to respond ot the question, simply wait for a moment and see the result:

```
Disk #2:  FORCE CPU-x PDOS was found  >>  Sector xxx, Addr xxx
           Execute it  ? _  Yes.  Booting...SUCCESS!
```

After "Booting...SUCCESS!" is displayed, the system will enter an initialization routine. The name of this initialization program is "SY$STRT". It initializes the available serial I/O channels, prints a welcome message, and then requests the input of date and time. The time should be entered in a 24 hour format. PDOS is now running on your system.

If at any time your system crashes, push the reset button located on the front panel of the CPU and restart the PDOS boot program. You will then be prompted with the same boot message as above. Proceed in the same manner as a cold startup.

2.0 SYSTEM MANAGEMENT GUIDE

Your PDOS system contains a collection of files and utilities to assist you in your development work. The purpose of the System Management Guide is to acquaint you with those files and utilities you will need to use in operating your system. Not all utilities are discussed in this section; for further information and additional utilities you should use the **PDOS Reference Manual.**

2.1 DISK PREPARATION OVERVIEW

Since FORCE PDOS has adopted the PDOS Winchester disk standard, you can use many different disk controller and drive combinations without changing the operating system code at all. PDOS finds which controllers are in the system, what Winchester drives are attached, and what drive information is written on each drive. It then sets up tables for the file manager to communicate with the drives. Before you can use the Winchester drives correctly, you need to prepare them by marking bad tracks, defining drive parameters and partitions, formatting all the tracks on the drive, writing PDOS Winchester information onto the drive, and finally initializing each partition with PDOS file system and, if desired, a PDOS boot.

This section tells you how to prepare Winchester drives for use with PDOS. It also gives a brief explanation of the disk format utility dealing with bad track mapping, disk parameters, PDOS disk partitions and disk numbering.

2.2 PREPARING FLOPPY DISKETTES FOR USE WITH PDOS

Floppy disk preparation consists of two parts. The first part formats the floppy with the sector ID and data fields required by the controller to access the disk. The second part is the installation of the PDOS directory structure onto the floppy disk.

The standard FORCE system requires diskettes that are:

5 1/4 inch
Double sided
Double density
96 tpi (80 tracks are used)

Floppy disks should match these specifications or you may encounter problems with reading the disks.

2.2.1 Hardware Formatting

Hardware formatting is done by running the FFRMT utility that is on your boot disk.

Note: FFRMT should only be run when no other tasks are running.

```
>FFRMT
68K PDOS FORCE Format Drive Utility  08-JAN-87
  Possible Disk Controllers in this System are:
    Controller #1 is a FORCE SASI-1
    Controller #2 is a FORCE WFC-1
    Controller #3 is a FORCE ISCSI-1
  Drives that are currently defined in system are:
    F0 is controller #3, drive select byte $73
    F1 is controller #3, drive select byte $74
    W0 is controller #3, drive select byte $00
    W1 is undefined.
    W2 is undefined.
    W3 is undefined.
  Select Menu:  W,W0,W1,W2,W3=Winch; F,F0,F1=Floppy; Q=Quit
       Select Drive:
```

You select the drive to format. The floppy disk is F or F0. If you have two floppy drives, the second drive is F1. We will assume that all formatting will be to the `F' or first floppy drive. To select floppy drive 0 enter an F followed by a [CR].

The floppy 0 drive menu is now displayed. Most of the options on the menu only apply to Winchester disks and should be ignored while formatting floppies.

```
  F0 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command: _
```

The option for formatting diskettes is selected by typing a `3' followed by a [CR].

```
       Command: 3[CR]
```

At this point, you should insert the floppy you wish to format into the disk drive. Remember, formatting will destroy all data on the diskette so double check that the disk in the drive is the disk you wish to format.

You are now prompted for the `Sector Interleave' of the format.

Sector interleaving allows you to map the sequence of the logical sectors onto nonadjacent physical sectors. The optimal sector interleave is a factor of many actions that may be occurring on the system. Such actions could include the number of tasks running, number of tasks accessing the disks, etc. You can use a disk of any interleave on your system, but the proper interleave will give you a better performance. FFRMT will display a good interleave for most applications.

If you wish to optimize the performance, you may want to experiment by formatting your disks at different interleaves. To use the default interleave just type a [CR]. To enter a different interleave, type the new number followed by a [CR].

```
  Sector Interleave = 1[CR]
```

You are next prompted for the physical tracks to format. The default display is every track on the diskette or a total of 160 tracks (80 tracks per side). Track 0 is the outermost track. Normally you will just type a [CR] to format the entire diskette.

```
  Physical Tracks to Format = 0,159[CR]
```

FFRMT waits for confirmation to begin formatting. A `Y' followed by a [CR] will begin the format process.

An [ESC] will return you to the `Floppy Drive 0 Menu' ending the format option. Any other response will restart the Format option with the Sector interleave question.

WARNING: FORMAT WILL DESTROY ALL DATA ON YOUR DISKETTE. If you do begin formatting and wish to abort you may type a [CTRL-C].

Type a `Y[CR]' to begin the format process.

The Sector Interleave Table will be displayed along with the current track formatting. The sector interleave table is meaningless on the ISCSI-1 since FFRMT just passes the interleave value on to the controller. However, the WFC controller will display the logical to physical sector map. During the formatting, the floppy disk light should be on. If you are using a WFC-1 the track numbers will be printed on the screen.

```
  Sector Interleave Table:
1,9,4,12,7,15,2,10,5,13,8,16,3,11,6,14

  Issuing Format Drive Command

  FORMAT Successful!
```

You will then be returned to the `Floppy Drive 0 Menu'.

Errors may occur when formatting disks, the most common error is that the drive is not ready. This may indicate that the door is open or the floppy has been placed into the drive incorrectly.

We recommend that you verify the disks after formatting. This is done by selecting option 4 `Verify tracks' by typing a `4[CR]'.

FFRMT will display the default tracks which will be the same number as the one you selected under the format option. If the track number is correct, you may enter a [CR] or if you wish, enter two new numbers separated by a comma and ended with a [CR].

You will next be prompted with:

```
  Ready to VERIFY Floppy Drive 0 ?
```

A `Y[CR]' response will begin verifying the diskette reading each track. Any other response will return you to the Floppy Drive 0 Menu. If an error occurs, FFRMT will ask you if you wish to add the track to the bad track table. Since there are no bad track tables for floppies, you should either respond with `N[CR]' which will proceed to the next track or a `R[CR]' which will retry the track.

If it cannot verify after a couple of retries, you should abort with the [ESC] key and verify another diskette which you have formatted. If the problem is resolved, then the diskette was probably bad and should not be used. If the problem persists, then you may be using an incorrect diskette type (40 track or single density) or you may have a hardware problem with your drive or controller.

After you have successfully formatted and verified your diskettes you are ready to install a PDOS directory.

2.2.2 Installation of PDOS Directory Structure

The installation of the PDOS Directory structure may only be done on disks that have been hardware formatted. This installation is the second process of the preparation of diskettes. This process will also destroy any data that may have been on the disk so care should be taken.

PDOS floppy disks may be initialized in one of two ways. The first way is to backup a current PDOS disk onto the new disks. This procedure is done by using the MBACK utility.

MBACK performs a sector to sector transfer. The new disk will look just like the source disk when the backup is completed. Disk name, sector bit map, directory entries, files, and boot information are all copied. MBACK is fully described in the **PDOS Reference Manual.**

The second method is to set up a NEW PDOS disk. This method is done with the MINIT utility which is described in more detail in the **PDOS Reference Manual.**

The initialization process is started by running the utility MINIT.

```
>MINIT[CR]
68K PDOS Disk Initialize Utility 07/29/85
  Disk # =
```

You are first prompted for the PDOS disk number to initialize. The floppy on a standard FORCE system is disk number 0. Normally MINIT will write to each sector on the disk to verify the media for data use unless you specify the `/Q' option. With the `/Q' option, MINIT will only write out to every 32nd sector. This is a reliable and a much faster method, provided you did the verify option during hardware formatting.

```
  Disk # = 0/Q[CR]
```

After the disk number has been entered, you are asked for the number of sides and the density of the diskette. Standard PDOS on the FORCE systems only supports double sided double density diskettes. A [CR] should be entered to retain those two defaults.

You will next be asked for the number of directory entries for the disk. This is the maximum number of unique files that may be created on the disk. Each directory entry requires 32 bytes of storage allowing 8 directory entries per disk sector. It is recommended that you make this number a multiple of 8, since the directory always ends on a sector boundary. You should realise that every eight directory entries will reduce your file storage area by 1 sector.

In addition, you will not be able to create more files on the disk than this maximum without rerunning MINIT which will also destroy all data on the disk. The default for floppies is 128. You may enter your new number followed by a [CR] or just a [CR] if you wish to use the default.

```
  Maximum Directory Size = 128[CR]
```

You will be asked for the total number of PDOS sectors. The default is 2336 which is the correct number for a bootable disk. If you only want to use the diskette for data, then you can enter 2528 which is the maximum storage on the diskette. (Note: the total number of floppy sectors is 2560 but under the PDOS disk standard, track 0 is always reserved for hardware dependent formats. 2560 - 32 sectors per track is 2528. You may enter any number of sectors up to 2528. If you enter a larger number you will get disk errors as PDOS attempts to access nonexistent sectors. If you enter a smaller number, it means that those sectors will not be available for PDOS file allocation.

```
  Total Number of PDOS Sectors = 2336 [CR]
```

Nest, a disk name is requested. The disk name is any string with up to 16 characters. The disk name is not used by PDOS in any command function, but it is displayed whenever a file listing of the disk is requested by the LS command or is output by several utilities such as MASM on list files. The name can be changed at any time without destroying the data on the disk by the MDNAME utility.

After the disk name has been entered, a summary of the initialization parameters will be displayed along with a prompt to proceed with the initialization process.

```
INIT:   Disk #0
        Double Sided
        Double Density
        128 Files
        2336 Sectors
DESTROY DISK? Y[CR]
```

If you wish to proceed with the initialization, type a `Y[CR]'. To return to the PDOS monitor type an [ESC]. Any other response will return you to the `Disk #' prompt and allow you to re-enter the parameters.

WARNING: **MINIT WILL DESTROY ALL DATA ON YOUR DISKETTE. Reconfirm the disk numbers to make sure that you are initializing the disk number you want to. A [CTRL-C] during the initialization of the diskette will abort further initialization.**

After the destroy disk confirmation is given, the initialization will display the current sector number. The header sector information will be written out to the disk. The message "INITIALIZATION Successful!" will be displayed.

The disk is now ready to be used with the PDOS system.

2.3 PREPARING WINCHESTER DRIVES FOR USE WITH PDOS

CAUTION: **Your FORCE PDOS System comes with PDOS already installed on the Winchester on Disk #2. When formatting the Winchester or changing the partitions, the installed system will be destroyed.**

Winchester drive preparation consists of two parts. The first part is the hardware preparation. Hardware preparation is the format information that the controllers need to access the disk. The second part is the installation of the PDOS directory structure onto the disk. Winchester disk preparation is very similar to the floppy diskette preparation except that PDOS allows you to handle bad tracks and to define several logical disks, or directories, to reside on a single physical drive.

The standard FORCE system will support either an ISCSI-1 or a WFC controller from FORCE. Through the PDOS Winchester disk standard, PDOS will also support various 5 1/4 inch drives up to the maximum heads and cylinders handled by the controller.

2.3.1 Hardware Formatting

First let us outline the procedure for hardware format. This is done by running the FFRMT utility that is on your PDOS FORCE boot disk. A complete description of the FFRMT utility is found in Appendix B.

NOTE: FFRMT should only be run when no other tasks are running.

```
>FFRMT
68K PDOS FORCE Format Drive Utility 08-Jan-87
  Possible Disk Controllers in this System are:
       Controller #1 is a FORCE SASI-1
       Controller #2 is a FORCE WFC-1
       Controller #3 is a FORCE ISCSI-1
  Drives that are currently defined in system are:
       F0 is controller #3, drive select byte $73
       F1 is controller #3, drive select byte $74
       W0 is controller #3, drive select byte $00
       W1 is undefined.
       W2 is undefined.
       W3 is undefined.
  Select Menu:  W,W0,W1,W2,W3=Winch; F,F0,F1=Floppy; Q=Quit
       Select Drive:
```

PDOS supports up to 3 Winchester drives and 2 floppy drives.

W0 will be the first Winchester in a standard system regardless of the controller being used. If all three Winchesters are undefined, then the Winchester may not have been up to speed when PDOS booted from your floppy. Just reboot and rerun FFRMT. If you still have problems then check if your controller is on-line or that it has been installed correctly.

To select your disk, enter W or W0 followed by a [CR].

If the controller is not defined, you will need to define it by selecting W0 and answering the additional questions for the controller number and drive select number.

The "W0 Main Menu" is now displayed. To set up the disk, you first select option 1 and then sequence through to option 6. If you wish to log your work, you may do so by using the `P' option, which will toggle output to your unit 2. All entries are terminated with a [CR].

```
  W0 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:
```

Option 1 is used to reconfigure your drive; i.e., to display and/or change the number of heads, cylinders, shipping cylinder, etc. If you are using any other than the standard Winchester disk, then you need to select option 1. The following gives an example of the default parameters which apply to the Micropolis 1375 SCSI Winchester disk. Please note that the parameters of the SCSI Winchesters must match the total number of blocks, but not the physics of the drive, because the interface is sector oriented. Please refer to Section 4 of this Manual for a detailed description of the physics of your Winchester.

```
Current Winchester Drive 0 Parameters:
               # of Heads = 16       Number of heads on drive
           # of Cylinders = 1000     Number of cylinders on drive
Physical Blocks per Track = 32       Number of 256 sectors per track
 Physical Bytes per Block = 256      Number in sector
        Shipping Cylinder = 0        Not needed
                Step rate = 0        Not used
Reduced write current cyl = 0        Not used
  Write Precompensate cyl = 0        Not used
```

If you wish to change any of the defaults, enter `A' followed by a carriage return and then you will be prompted for them one at a time. The parameters most often changed are heads and cylinders which vary according to the size of the disk. This information is included in the hardware drive specifications. Enter a [CR] to preserve the default information or type in the new number followed by a [CR] to change the information.

Note: The drive parameters only have to be changed when you want to use another Winchester type.

Option 2 will allow you to enter the bad tracks. This means you can logically map out tracks that are unsuitable for data storage. Most disks will have some media defects when manufactured. The bad track map can be found on the disk drive or with the documentation provided with your system. This option should not be used on SCSI Winchester drives as they map out the bad blocks internally.

If you do not know the bad tracks for your drive, just format and verify your drive, adding bad tracks if they are found.

PDOS maps logical tracks to physical tracks skipping those physical tracks that are marked as bad. You should complete the bad track table before you partition and initialize the drive. If you add a bad track, PDOS remaps all subsequent tracks, resulting in a loss of data.

When the Bad Track option is selected, enter `D' to display the current bad track list:

```
  W0 Bad Tracks Menu:  Bad Track, D)isplay, C)lear, H)elp, Q)uit
       Command: D[CR]

  Current Winch Drive 0 Bad Tracks:
231 613 697 700 703
```

To display instructions on entering bad tracks enter `H' followed by a [CR]. The following help menu is displayed and you are returned to the Bad Track menu.

```
  To add a bad track, enter either {track} or {Head,Cylinder}.  To
  delete a bad track from the table, precede the track number with
  a minus sign.  Enter [ESC] or `Q' to exit.

  Current Winch Drive 0 Bad Tracks:
231 613 697 700 703

  W0 Bad Tracks Menu: Bad Track, D)isplay, C)lear, H)elp, Q)uit
       Command: _
```

Enter the bad track numbers by typing in the number of the track. For example, typing `701' enters track 701 into the table. `3,50' enters the track on head 3, cylinder 50 into the table. When you have completed entering the bad tracks in, type a `Q' to return to the "Winchester Drive 0 Main Menu."

Option 3 is the format option. Remember, formatting will destroy all data on the drive so double check that you are formatting the correct drive number.

You are now prompted for the `Sector Interleave' of the format.

Sector interleaving allows us to map the sequence of logical sectors onto non-adjacent physical sectors. The optimal sector interleave is a factor of many actions that may be occurring on the system such as the number of tasks running, number of tasks accessing the disks, etc. You can use a disk of any interleave on your system, but the proper interleave will give you better performance. FFRMT will display a good interleave for most applications. If you wish to optimize the performance, you may want to experiment by formatting your disks at different interleaves. Note that the interleave of SCSI Winchester drives cannot be changed.

To use the default format you may just type a [CR]. To enter a different interleave, type the new number followed by a [CR].

You are next prompted for the physical tracks to format. The default display are all tracks on the drive. The default number is calculated from the number of heads and cylinders you specified in option 1 - Display/Alter Drive parameters. Track 0 is the outermost track. Normally just type a [CR] to format the entire drive.

```
 Sector Interleave = 1                        (just hit [CR])
 Physical Tracks to Format = 0,16000          (just hit [CR])
 Ready to FORMAT Winchester Drive 0 ?
```

FFRMT waits for confirmation whether to begin formatting or not. A `Y' followed by a [CR] will begin the format process.

An [ESC] will return you to the `Winchester Drive 0 Menu' ending the Format option. Any other response will restart the Format option with the Sector interleave question.

WARNING: FORMAT WILL DESTROY ALL DATA ON YOUR DRIVE. If you do begin formatting and wish to abort you may type a [CTRL-C].

The Sector Interleave Table will be displayed along with the current track formatting. The sector interleave table is meaningless on the ISCSI-1, and no track numbers are displayed. The WFC controller, however, will display the logical to physical sector map. During the formatting, the drive select light should be on and the track numbers will be printed on the screen.

```
  Sector Interleave Table:
1,9,4,12,7,15,2,10,5,13,8,16,3,11,6,14
  Issuing Format Drive Command.
```

When all tracks are formatted, the following display will be shown on your console:

```
  FORMAT Successful!
```

You will then be returned to the `Winchester Drive Menu'. Errors rarely occur when formatting drives.

Option 4 will verify the format by reading each track on the drive. This will allow you to detect any bad tracks that are unsuitable for data storage. We recommend that you verify the drive after format. This is done by selecting option 4 `Verify tracks'.

FFRMT will display the default tracks which will be the same numbers as those you selected under the format option. If the track numbers are correct, you may type a [CR] or if you wish, enter two new numbers separated by a comma and ended with a [CR]. You will next be prompted with:

```
  Ready to VERIFY Winchester Drive 0 ?
```

A `Y[CR]' response will begin verifying the drive reading each track. Any other response will return you to the Winchester Drive 0 Menu.

If an error occurs, FFRMT will ask you if you wish to add the track to the bad track table. This is done by typing a `Y' followed by a [CR].

WARNING: **You will change the logical track map of the drive any time you enter a bad track which may require reinitializing the PDOS file structure. You should only enter bad tracks on a drive with PDOS file structure after you backup all data on the drive. Otherwise verify may be run without affecting the data on the disk.**

If you do not want to add the track into the bad track table, type an `N' followed by a [CR]. `R[CR]' will retry the track. We recommend that if at first you cannot verify the track and the error number is "bad data on read" error, that you enter the track as bad. If many of your tracks are bad then you should verify the installation of the drive.

The message `VERIFY Done!' will be displayed when the verify option is completed, and you will be returned to the Winchester Drive Menu.

Select Option 5 next to setup your logical PDOS disk partitions on your Winchester. You have the option of defining large partitions and floppy sized partitions. PDOS does limit you to 16 Mbytes per partition, so most Winchester drives will need to be defined into appropriately sized partitions.

The floppy partitions are about 660 Kbytes each and correspond to the PDOS standard floppy disk size. These partitions are allocated from the end of the Winchester tracks. After all floppy partitions have been allocated, the remaining tracks are allocated evenly between the number of large partitions defined. Since the floppy disk drives are 0 and 1 on standard PDOS, the default starting partition or disk number is 2. The following diagram shows how four large disk partitions and two floppy sized partitions would be allocated on the tracks.

```
track 0 ..................................................track n

large disk|large disk|large disk|large disk|FloppyDisk|FloppyDisk
  #2      |   #3     |   #4     |   #5     |   #6     |   #7
```

When this option is selected you will see the following menu appear on your screen. There may be some variation due to the number of bad tracks and size of your drive.

```
  Current Winch Drive 0 Disk Partitions:
     # of Large partitions = 10
    # of Floppy partitions = 12
First track for PDOS Parts = 0
 Last track for PDOS Parts = 15979
         First PDOS disk # = 2
 Total # of Logical Tracks = 16000
 Disk #    Logical Trks  Physical Trks  PDOS sectors
           Base,Top      Base,Top       Total/{boot}
  2        0,1499        0,1500         47968/47776
  3        1500,2999     1501,3000      47968/47776
  4        3000,4499     3001,4500      47968/47776
  5        4500,5999     4501,6000      47968/47776
  6        6000,7499     6001,7500      47968/47776
  7        7500,8999     7501,9000      47968/47776
  9        9000,10499    9001,10500     47968/47776
 10        10500,11999   10501,12000    47968/47776
 11        12000,13499   12001,13500    47968/47776
 12        13500,14999   13501,15000    47968/47776
 13        15000,15079   15001,15080    2528/2336
 14        15080,15159   15081,15160    2528/2336
 15        15160,15239   15161,15240    2528/2336
 16        15240,15319   15241,15320    2528/2336
 17        15320,15399   15321,15400    2528/2336
 18        15400,15479   15401,15481    2528/2336
 19        15480,15559   15482,15564    2528/2336
 20        15560,15639   15565,15647    2528/2336
 21        15640,15719   15648,15730    2528/2336
 22        15720,15799   15731,15813    2528/2336
 23        15800,15879   15814,15896    2528/2336
 24        15880,15959   15897,15979    2528/2336
  W0 Partitions menu: A)lter, D)isplay, R)ecalc, Q)uit
       Command:  Q
```

The standard ISCSI-1 system will assume ten large partitions and 12 floppy sized partitions. The number of partitions and types you want will depend on your needs. You may find it helpful to create mostly floppy sized partitions since they easily backup to floppy diskettes with MBACK, a sector to sector image backup utility. You should also keep one floppy partition as a temporary disk partition to copy diskettes. This is because it is much faster to do a transfer of files to a floppy Winchester area followed by a sector for sector backup to a diskette with MBACK, than it is to transfer files directly to a floppy diskette with MTRANS or the monitor command TF.

The first large partition (defaults to disk 2) is generally referred to as the system disk. This partition should contain all the system files that have been distributed with PDOS along with other utilities and languages. You may use any disk for your system disk, but the installation guide will treat disk 2 as the system disk. The installation guide will also treat floppy partition 20 as the temporary disk partition.

To set up your partitions, type an `A' followed by a [CR]. You will be prompted for the number of large partitions:

```
# of Large partitions = 10
```

You may type a [CR] to use the default or enter in a new number followed by a [CR]. Next, you are prompted for the number of floppy sized partitions.

```
# of Floppy Partitions = 12
```

You may type a [CR] to use the default or enter in a new number followed by a [CR]. You are next prompted for the first track to use for PDOS followed by the last track for PDOS. You should respond to both with a [CR] unless you wish to reserve some tracks for non-PDOS use.

```
  First track for PDOS Parts = 0
  Last track for PDOS Parts  = 15979
```

By default the last track for PDOS parts is the maximum number of tracks on the Winchester minus the number of bad tracks (which is 0 in the above example). The format utility automatically subtracts the bad tracks from the maximum number of tracks. Note that bad tracks are mapped automatically on the SCSI drives.

Next you are asked for the starting PDOS disk number for the first partition on the drive. Normally, the Floppy disk drives are 0 and 1, the RAM disk is 8, and the Winchester disk partitions start at 2. FFRMT will assign disk numbers from the starting number on up skipping 8. This number should be in the range of 0 to 99. You should select the number for starting the Winchester disk partitions that is low enough to define all the partition numbers to be under 99.

```
  First PDOS disk # = 2
```

Enter your number or just type [CR] for the default. You will then have the Winchester partitions displayed.

```
  Current Winch Drive 0 Disk Partitions:
     # of Large Partitions = 10
    # of Floppy Partitions = 12
First track for PDOS Parts = 0
 Last track for PDOS Parts = 15979
         First PDOS disk # = 2
 Total # of Logical Tracks = 16000
 Disk #    Logical Trks  Physical Trks   PDOS sectors
           Base,Top      Base,Top        Total/{boot}
  2        0,1499        0,1500          47968/47776
  3        1500,2999     1501,3000       47968/47776
  4        3000,4499     3001,4500       47968/47776
  5        4500,5999     4501,6000       47968/47776
  6        6000,7499     6001,7500       47968/47776
  7        7500,8999     7501,9000       47968/47776
  9        9000,10499    9001,10500      47968/47776
 10        10500,11999   10501,12000     47968/47776
 11        12000,13499   12001,13500     47968/47776
 12        13500,14999   13501,15000     47968/47776
 13        15000,15079   15001,15080     2528/2336
 14        15080,15159   15081,15160     2528/2336
 15        15160,15239   15161,15240     2528/2336
 16        15240,15319   15241,15320     2528/2336
 17        15320,15399   15321,15400     2528/2336
 18        15400,15479   15401,15481     2528/2336
 19        15480,15559   15482,15564     2528/2336
 20        15560,15639   15565,15647     2528/2336
 21        15640,15719   15648,15730     2528/2336
 22        15720,15799   15731,15813     2528/2336
 23        15800,15879   15814,15896     2528/2336
 24        15880,15959   15897,15979     2528/2336
 W0 Partitions Menu:  A)lter, D)isplay, R)ecalc, Q)uit
      Command:  Q
```

You may notice that the `Total # of Logical Tracks' is greater than the `Last track for PDOS parts'. This is because of an odd number of tracks that could not be allocated evenly between the large partitions. You may also notice that the logical track numbers may not exactly correspond to the physical tracks.

```
 Disk #     Logical Trks  Physical Trks   PDOS sectors
            Base,Top      Base,Top        Total/{boot}
  2         0,1499        0,1500          47968/47776
```

This is because of bad tracks that may have been mapped out. The final information that is important to note is the PDOS Sector column. The number to the left is the number of sectors for the partition if you want it to be a data only disk. The number to the right is the number of sectors for the partition if you wish to install a boot on the partition. This information is needed when setting up the PDOS directory structure on the partition with MINIT. In the preceding example, we have 47968 sectors for data only or 47776 for a bootable partition.

You should now exit with the `Q' option back to the Winchester Drive Menu.

The drive parameters, bad track list and disk partition information must now be saved onto the Winchester drive so that future accesses will have the information. Select "Write out Header info to disk". When you select option 6 you will be prompted with:

```
  W0 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  6[CR]
  Write to Disk Y)es, N)o, F)ile : F[CR]
  Write Parameter File: #TEMP/0[CR]
  W0 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  Q[CR]
  Exit to Select Drive.  Update Param RAM (Y/N) ? Y[CR]
  Select Menu: W,W0,W1,W2,W3=Winch; F,F0,F1=Floppy; Q=Quit
       Select Drive:  Q[CR]
```

A `Y' followed by a [CR] will update the drive by writing the tables out. An `F' followed by a [CR] will write the tables out to a file of your choice. Any other response will return you to the Winchester Drive Menu.

Now that you have set up your Winchester drive, you can exit back to the Drive Selection Menu. This is done by selecting option Q. As part of the Exit procedure you will also be asked if you want to update the current BIOS tables for the Drives with the parameters you have set up.

Answer with a `Y' followed by a [CR] to update the current BIOS tables. The BIOS table must always be updated if you have made any changes. An `N' will not update the BIOS tables.

You may now proceed to set up additional Winchester drives or type a `Q' for quit followed by a [CR] to exit back to the PDOS monitor.

You are ready to install a PDOS directory onto each partition.

2.3.2 Installation of PDOS Directory Structure on Winchesters

Installation of the PDOS directory structure may only be done on Winchester partitions that have been hardware formatted and defined as outlined in section 2.3.1. You must initialize each partition before you may use it for PDOS. Before you begin you have to know the disk partition number and number of blocks for the partition as provided by FFRMT under "Disp/Alter Disk Partitions Option." The initialization process is done with the MINIT utility which is described in detail in the **PDOS Reference Manual.**

```
>MINIT
68K PDOS Disk Initialize Utility 07/29/85
  Disk # =
```

You are first prompted for the PDOS disk number to initialize. Enter in the number for one of the partitions.

Normally MINIT will write zeros in each sector on the disk to verify the media for data use unless you specify the `/Q' option.

With the `Q' option MINIT will only write out zeros to every 32nd sector. This is a much faster method and is reliable provided you did the verify option during hardware formatting.

```
  Disk # = 2/Q[CR]
```

After the disk number has been entered, you will be asked for the number of sides and the density of the disk. These have no meaning to Winchester partitions. We recommend that you use the default, especially if the disk is going to backup to a floppy disk later. The default is selected by typing a [CR].

You will next be asked for the number of directory entries for the disk. This is the maximum number of unique files that may be created on the disk. Each directory entry requires 32 bytes of storage allowing 8 directory entries per disk sector. It is recommended that you make this number a multiple of eight, since the directory always ends on a sector boundary. You should also keep in mind that every eight directory entries will reduce your file storage area by one sector. You will not be able to create more files on the disk than this maximum without rerunning MINIT which will also destroy all data on the disk. The default is 128, and is suitable for floppy sized partitions. For large partitions, you may want to use 1024. You may enter your new number followed by a [CR] or enter a [CR] if you wish to use the default.

```
  Maximum Directory Size = 1024[CR]
```

The total number of PDOS sectors will be asked. The default is 2336 which is the correct number for a bootable floppy diskette. You will need to enter in either the total number of blocks if you want the disk to be data only, or the boot number of blocks if you are going to make the disk a bootable disk. The number of sectors is the same as the blocks given in the FFRMT program.

Sample table from FFRMT Disp/Alter Disk Partitions:

```
 Disk #     Logical Trks   Physical Trks   PDOS sectors
            Base,Top       Base,Top        Total/{boot}
    2       0,1499         0,1500          47968/47776
```

You may enter any number up to the total number of sectors. Use the total sector number (i.e. 47968) for a data only disk, or the {boot} number (i.e. 47776) for a bootable disk. If you enter a larger number you will get disk errors as MINIT attempts access to nonexistent sectors. If you enter a smaller number, it will mean that those sectors will not be available for PDOS file allocation. For this example we will make disk 2 a boot disk.

```
  Total Number of PDOS Sectors = 47776
```

If you can't remember the total number of PDOS sectors, simply type `MAX[CR]' for a data only disk and `BOOT[CR]' for a boot disk. MINIT will then calculate the correct number of sectors and display them for you as the default and you can then enter a [CR].

```
  Total Number of PDOS Sectors = BOOT[CR]
  Total Number of PDOS Sectors = 47776[CR]
```

You are then asked to name the disk. The disk name is any string up to 16 characters which is not used by PDOS in any command function. However, the disk name is displayed when file listings of the disk are requested by the `LS' monitor command, or it is output by several utilities such as MASM on the list file. The name can be changed by the MDNAME utility at any time without destroying the data on the disk.

```
  Disk Name = My New Disk[CR]
```

After you have entered the disk name, a summary of the initialization parameters and a prompt to confirm the initialization process is displayed.

```
  INIT:    Disk #2
           Double Sided
           Double Density
           1024 Files
           47776 Sectors
  DESTROY DISK?  Y[CR]
```

If you wish to proceed with the initialization, type a `Y[CR]'. To return to PDOS monitor type an [ESC]. Any other response will return you to the `Disk #' prompt and allow you to re-enter the parameters.

WARNING: MINIT WILL DESTROY ALL DATA ON YOUR DISK. Reconfirm the disk parameters to make sure that you are initializing the right disk number. A [CTRL-C] during the initialization of the disk aborts further initialization.

After the destroy disk confirmation is given, the initialization displays the current sector numbers. These are printed out every 32 sectors, up to the total minus 1. Finally, the header sector information is written out to the disk. The message "INITIALIZATION Successful!" is displayed.

The disk partition is now ready to be used with the PDOS system. You need to repeat the procedure for each partition on the Winchester drive.

2.4 PDOS Generation

PDOS provides you with a procedure file to assist you in the generation of the PDOS system. The file is named FxxDOS:GEN and is on your boot disk.

FxxDOS:GEN will assemble all user alterable portions of PDOS and then proceed to link a new bootable system file.

The following files must be accessible when running FxxDOS:GEN:

MASM20	Rev. 3.3 assembler (or later)
QLINK	The PDOS linker
LINKIN:COM	LINKIN command for additional resident programs
MSYRAM:SR	PDOS RAM definitions
MBIOS:SR	Generic PDOS BIOS file (used as included file to FxBIOS:SR)
FPARM:SR	Parameter definitions/memory map for absolute locations
FxxPARM:SR	System Hardware Addresses
FxxBIOS:SR	BIOS file
FxxBIOSU:SR	Character I/O (UARTs) service routines - for the CPU-board UARTs
FBIOSU:SR	Character I/O drivers for System UARTs
FBIOSW:SR	Block I/O (disk read/write) routines
MPDOS:OBJ	PDOS system
or	
MPDOSBAS:OBJ	PDOS system with BASIC
and	
ISCSIx:SR	Include file for FBIOSW:SR

FxxDOS:GEN will create the following files as output:

MSYRAM:OBJ	Object of MSYRAM:SR
FxxBIOS:OBJ	Object if FxBIOS:SR and MBIOS:SR
FxxBIOSU:OBJ	Object of FxBIOSU:SR
FBIOSU:OBJ	Object of FBIOSU:SR
FBIOSW:OBJ	Object of FBIOSW:SR
FxxDOS:MAP	Load map of the new PDOS system
FxxDOS	SY file image of new PDOS system

NOTE: **'xx' as used above and in the next chapters of this manual is generally the type or the number of the FORCE CPU. For example, for the CPU-21, CPU-29 or CPU-32, 'xx' = 32. For the CPU-26, 'xx' = 26.**

2.4.1 Generation Arguments and Switches

FxxDOS:GEN has two optional arguments to assist you in building a new PDOS system. Argument one is either the word `LINK' or switches to change the generation. Argument two is either blank, the word `BASIC', or `BASIC/68881'.

If argument one is the word LINK then FxxDOS:GEN will build the new PDOS system without reassembly.

```
  FxxDOS:GEN LINK
```

This will use the object files that were sent with PDOS. You can optionally input switches to change parameters in the generation files. The most notable are:

MAIL ARRAY SETUP:

This will dynamically setup the mail array and PDOS will place a pointer to this sized array in location `MAIL.' in SYRAM (see file MSYRAM:SR).

Switch Name	Default Value	
MSZ	256	Number of bytes allocated to the Mail array

RAM DISK SETUP:

This is the specification of a RAM disk area. If RZ is set to 0 then no RAM disk will be allocated.

Switch Name	Default Value	
RU	8	Default RAM disk unit number
RZ	255	Default sectors allocated to RAM disk. 4 sectors equal 1KB
RE	((RZ+1)/16)&~7	Default number of directory entries on RAM disk (if RZ is 255 then RE will default to 16)
RA	0	RAM disk address. If 0, (default), then allocate the RAM disk dynamically from top of memory, right below the MAIL array. If not 0, then place RAM disk at the given address.
IRD	1	If 0, don't initialize RAM disk area. If 1, then initialize RAM disk area on startup if it is not a PDOS disk image already.

LINE EDITING KEY DEFAULTS:

These are the key definitions for the get line primitives.

They are limited to a single character.

Switch Name	Default Value	
B.ADD	[CTRL-A]	Recall last line
B.LFT	[CTRL-H]	Move cursor left
B.RGT	[CTRL-L]	Move cursor right
B.DRT	[CTRL-D]	Delete character cursor is under (Delright)
B.DLT	[RUB]	Delete character to left of cursor

SYSTEM PARAMETERS:

Switch Name	Default Value	
SD	0	Default disk unit number (if 0, then setup by the boot ROMs; if not 0, then system will start with this disk number).
BR	1	Default baud rate. This is a number from 0 to 7 which represents initial baud rate for the character I/O ports. 0 = 19200 4 = 1200 1 = 9600 5 = 600 2 = 4800 6 = 300 3 = 2400 7 = 110 8 = 38400
CPSC	$AA009B3D	Default clear screen and position codes. The default is for a SOROC terminal type. This will select the system startup default. You can select a different type for each task by running the MTERM utility after the system is up. PDOS can output 1 to 4 characters to clear the screen or callout to a BIOS driver. The first 16 bits of CPSC is the clear screen command. The format for clear screen is (where each letter is a bit): E111 1111 E222 2222 E = if 1 then precede with [ESC] 1 = 7 bits for first ASCII character 2 = 7 bits for second ASCII character If all 16 bits are 0 then callout to BIOS label B$CLS.

PDOS can position the cursor with 1 of the leading characters followed by row/column or column/row optional based by $20 or PDOS call callout to a BIOS driver. The second 16 bits of CPSC is the position cursor leading characters. The format is (where each letter is a bit):

```
B111 1111 0222 2222
```

B = 0 then $00 bias
    1 then $20 bias
O = 0 then row first then column 1 then row
1 = 7 bits for first ASCII lead in character
2 = 7 bits for second ASCII lead in character

If all 16 bits are 0 then callout to BIOS label B$PSC.

ANS 0

If 1, then the BIOS callout for clear screen and position cursor for ANSI 3.64 terminal support is included.

CLKADJ 0

Clock adjust factor. This is used to fine tune the system clock. On each tic, PDOS will add this value to the adjustment factor. Each time the adjustment factor rolls over PDOS will add one tic. The formula is $80000000/N*2 where N is the number of tics to pass before adding an additional tic to the PDOS tics.

For example, if you needed to add 1 extra tic every 100,000 tics to keep the clock accurate, then you would set CLKADJ to $80000000/100000*2. This will cause a roll over every 100,000 tics.

FSTEP 3

Floppy disk drive stepping rate. The default is 3msec, but full height drives may require 6msec stepping rate.

The above switches may be passed into the generation procedure file by specifying as the first argument a `/', the switch name, followed by an `=', then by the value of the switch. Note: The command line length is limited to 78 characters, so if you must set many switches,you will need to edit file FxxBIOS:SR file and place the switch name in as EQUATES. The following shows some examples for defining the switches on a FORCE CPU-5 system:

```
Example 1 -- Set ANSI terminal support on:
             >FxxDOS:GEN /ANS=1

Example 2 -- RAM disk size of 2000 sectors and at address $800000
             >FxxDOS:GEN /RZ=2000/RA=$800000
```

The second argument of FxxDOS:GEN is the word BASIC. If you have licensed BASIC then you need to specify the BASIC interpreter with your system. Failure to do so will create a system without BASIC. For 68020 CPU-boards with the 68881 co-processor, the string /68881 can be appended to the word BASIC to use the HW co-processor together with the BASIC interpreter. For example to build a system with BASIC and 68881 support, you need to type the command:

```
             >FxxDOS:GEN LINK,BASIC/68881
```

2.4.2 Loading the New System

After FxxDOS:GEN has been run without errors,you are now ready to try your new system. You can do this by using the FLDGO utility. FLDGO has the format:

```
             FLDGO <load address>,<filename>
```

The load address is the location in memory where the program is to be located. The default of $800 is assumed on all FORCE systems when no address is given. The filename is the name of the system file that you have just generated. After FLDGO has loaded your new system it will jump to the load address and begin execution. To try the new system type in (example for a CPU-2 system):

```
             FLDGO ,FxxDOS
```

You should then see something similar to the following:

```
                    DOS File Loaded: FxxDOS
               Found PDOS at address $00020EB6
                         DOS size is $00008BD4

   (Execution of the startup file on the new version)
```

FLDGO allows you to try a new version of PDOS without modifying your disk boot image. When you reboot, via the reset switch or by powering down and up, your boot will come off the disk unit selected by the boot ROMs. To make this new system into a disk boot you need to follow the instructions in chapter 4, description of the MMKBT utility, or refer to chapter 2.5.

You can verify that you are executing the new version of PDOS with the PDOS ID monitor command. You will notice that the BIOS date will be the current date.

2.5 WRITING BOOTS TO DISK

The MMKBT utility is used to install bootable PDOS files onto your disks. The disk should first be prepared for use with the PDOS system as outlined in either section 2.2 for floppy disks or section 2.3 for your Winchester partitions. You need to have a PDOS bootable file as outlined in section 2.4.

To install a new boot you run MMKBT from the PDOS monitor:

```
>MMKBT
68K PDOS Make Boot Disk Utility 02/26/86
```

You are prompted with 3 options. The first is (F)ile for creating a new boot disk from a file. The second is (M)emory for creating a new boot disk from a memory area. The last is (B)ootstrap for the outputting of special sector information onto physical track 0 of the disk. The FORCE systems do not use the (B)ootstrap option since PDOS boot ROMs are supplied. To select any option, type the letter followed by a [CR].

2.5.1 Installing From a File

To install from a file type `F' followed by a [CR].

```
(F)ile, (M)emory, or (B)ootstrap? F[CR]
```

You are next asked for the name of the new system file. The default name is `FxxDOS', which is the same filename generated from the FxxDOS:GEN procedure file. You may enter in the name if different or just type a [CR] to use the default.

Next you are asked how large the boot is in bytes. MMKBT will size the file you specified and print the appropriate default for you in HEX, or you may enter in your own size in bytes followed by a [CR]. To use the default just type a [CR].

```
             Boot size = $00007B14[CR]
```

The load address is specified next. The default load address is what you will normally use. It is selected by just typing a [CR]. PDOS is also position independent, and can be loaded at any address. To load PDOS at some other location in memory enter `$' if you are entering it in HEX.

```
             Load address = $00000800[CR]
```

You are asked for the PDOS disk number for writing the boot. Your current working disk is used as the default. If it is correct, then type a [CR], otherwise enter your new disk number followed by a [CR].

```
             Disk = 2[CR]
```

The boot sector number is then requested. The default is the current boot sector number if a boot is presently on the disk or it is the sector after the last initialized PDOS sector if no boot is present. You may generally use the default which is selected by typing a [CR]. You may optionally place the boot at any sector on the disk by typing the new sector number.

You may wish to verify the boot sector number with the disk partition table from the FFRMT program to confirm that the boot is at the recommended location for disk partitions. This will give you about a 160 sector boot area or enough for a 40 Kbyte system. If you have a larger system, you will need to initialize your disk for fewer PDOS sectors which will allow a larger boot area. The boot sector number is generally the same as the number of PDOS sectors initialized in the MINIT system. Bootable diskettes should normally be initialized to 2336 PDOS sectors, and have the boot sector at 2336.

```
             Sector = 47776
```

After you have entered the sector number, you will be asked to confirm the parameters for writing the boot. If you respond with a `Y' followed by a [CR], the boot will be written out. An [ESC] will exit the program back to PDOS. Any other response will restart the MMKBT program.

```
  Write 124 sector boot beginning at address $0000CD00 with
  load address $00000800 to disk 2, sector 47776.
  Ready (Y/N)? Y[CR]
```

WARNING: DOUBLE CHECK THAT YOU ARE WRITING TO THE CORRECT DISK AND SECTOR NUMBER. MMKBT will destroy any data at the specified sector and disk when this question is verified.

When a `Y[CR]' is typed you will be informed that the boot was or was not written successfully.

```
     One moment, Please...
        Wrote out header information.
        PDOS Written Successfully!!
```

2.5.2 Installing from Memory

Boots may also be written to the disk from memory. This is typically done to build a boot image of the currently executing system. After executing the MMKBT program you select the (M)emory option by typing an `M' followed by a [CR].

You are asked for the memory start address and end address for the boot. The defaults are according to your standard system set up. You may select the default by typing a [CR] or enter new addresses. (Remember to use the $ to prefix hexadecimal numbers):

```
Start address = $00000800[CR]
End address   = $00009800[CR]
```

The disk number is next asked for and the rest of the make boot proceeds the same as with the (F)ile option in section 2.5.1.

2.6 Auto-Start File

Standard PDOS will look for the Auto-Start procedure file to execute by task 0 on startup. The file must be found on the boot disk, be named `SY$STRT' and must have an executable file attribute (i.e. AC, SY, OB or EX, BX if BASIC is resident). Any program or monitor command may be placed in the procedure file. The printout below shows the default contents of the startup file SY$STRT.

Feel free to customize your startup file to set up terminal types, correct flags for the ports (that is XON/XOFF, DTR, etc.) and baud rates. You may also want to create other tasks on the system with the startup file. This can be done by editing the startup file SY$STRT.

Listing of the startup file SY$STRT:

```
FTIME P
EE 1
MTERM D
CT (WIND1 1/14/15,1),32
CT MABORT,2
CT ,100,64,$F
CT ,100,64,$E
BP 2,1,1,$FF800200
BP 3,1,1,$FF800600   For CPU-21 with SYRAM 22
FBAUD
```

3.0 INSTALLATION OF PDOS ON THE WINCHESTER

The PDOS operating system with all utilities is already installed by default on the Winchester on logical disk 2. This chapter, which describes the installation on the Winchester, is only to be used when you change the Winchester.

3.1 PDOS FLOPPIES

Three floppies are delivered with every PDOS System. The files are already installed on the Winchester and so the floppies are only a backup. Keep these floppies in a safe place. They will be needed if the Winchester crashes and a new one has to be installed in the system.

The three PDOS floppies contain the following files:

1. BOOT — This floppy contains a bootable PDOS operating system, all hardware implementation and some useful utilities.

2. UTIL — This is the standard PDOS utility disk and contains all utilities and the sources of most of the utilities.

3. FORCE UTIL — This disk holds some FORCE specific utilities and their sources.

3.2 Booting PDOS From Floppy

Attach a 9600 BAUD terminal to the Terminal port, apply power, activate the RESET switch and start the PDOS Bootstrap program (see chapter 1 for details).

The bootstrap first initializes the disk controllers in the system. It then starts to boot, first from the floppy, and next from the hard disk. For example:

```
   FORCE PDOS Bootstrap Revision 2.1   01/15/85
       Disk Controller Installed
```

If the bootstrap program cannot find a disk controller, the following will be displayed:

```
   FORCE PDOS Bootstrap Revision 2.1    01/15/86
    No Disk Controller Installed ???
```

If you see the previous message and yet you know that there is a disk controller board installed, please check the address jumpers on the disk card and then test that the card is really accessible under the monitor.

After printing the banner and initializing the disk controller, the bootstrap program begins reading the header sectors of disks, number 0 through 99, for PDOS boot information. You should see some messages printed out that indicate disk errors on disks 0 and 1, since the floppy drives have no diskettes in them:

```
   Disk #0:  Header sector error = 161
   Disk #1:  Header sector error = 161
   Out of PDOS boot disk table entries.
   I'll Retry them all.
```

These errors are normal until the Winchester gets up to speed or you insert a floppy disk. The bootstrap reinitializes the disk controller and starts all over again, looking at disk 0 to 99.

Now, insert the PDOS disk marked `BOOT' in the main floppy drive and close the drive door. The next time around, the bootstrap should find the PDOS boot on disk #0 and display:

```
Disk #0:
  FORCE PDOS was found  >>  Sector 2336, Addr $800
            Execute it ? _
```

You do not need to respond to the question, simply wait for a moment and see the result:

```
Disk #0:
  FORCE PDOS was found  >>  Sector 2336, Addr $800
            Execute it ? Yes.  Booting...SUCCESS!
```

After "Booting...SUCCESS!" is displayed, the system will enter an initialization routine. The name of this initialization program is "SY$STRT". It initializes the available serial I/O channels, initializes the date and time (if a real-time clock is available), and prints a welcome message.

If at any time your system "crashes", push the reset button located on the front panel of your system and restart the PDOS boot program. You will then be prompted with the same boot message as above. Proceed in the same manner as a cold startup.

3.3 WINCHESTER SET UP

Next you must format and set up the partitions for your Winchester drive. The number of partitions and sizes will depend on your needs. If you wish to change the partitioning of the drive, you will need to backup the entire Winchester drive and rebuild the files. Complete instructions on Winchester setup are given in the system management portion of this guide under the heading "Preparing Winchester Drives for use with PDOS" (chapter 2).

3.4 TRANSFERRING PDOS SYSTEM FILES TO YOUR WINCHESTER

After you have formatted and partitioned your Winchester drive, you will need to move all files from the PDOS distribution disks to a system disk on the Winchester. By PDOS standard this is disk 2, but you may use any other disk unit on your Winchester. It is also a PDOS standard that the last Floppy image on the Winchester is a temporary disk that can be used to quickly transfer files. It is assumed in here that disk 20 is a floppy image for these instructions. You may have to substitute your temporary floppy number if it is different.

First backup the PDOS boot disk to temporary floppy disk 20. This is done by first inserting the distribution disk marked PDOS BOOT into floppy drive 0 and then running the MBACK utility with the following arguments:

```
>MBACK 0,20,2528
```

This will backup 2528 sectors of disk 0 to floppy image disk 20. It also copies the boot tracks from floppy to disk 20. Next transfer all files from disk 20 to the system disk 2. This can be done with the PDOS monitor command TF:

```
>TF ;@/20,2,A
```

Next backup the PDOS UTILITY disk to temporary floppy disk 20 by inserting the utility disk into drive 0 and running MBACK with the following arguments:

```
>MBACK 0,20,F
```

This will now copy all files of the utility floppy onto the floppy image disk 20 on the Winchester. Next we have to transfer these files to disk 2.

```
>TF ;@/20,2,A
```

The last floppy to transfer is the FORCE utility disk (disk #3). You can use the same commands as outlined in the description of the PDOS utility disk:

```
>MBACK 0,20,F
>TF ;@/20,2,A
```

If you purchased PDOS BASIC, you may want to also copy those files onto disk 2 so that BASIC can be installed into your system for the next section. Use MBACK and TF commands as used with the previous 2 disks.

```
>MBACK 0,20,2528
>TF ;@/20,2,A
```

Now you will have to generate a PDOS System with BASIC before you install the boot tracks. You can use following command line:

```
>FxxDOS:GEN <parameters>,BASIC</68881>
```

The parameters are described in detail in chapter 2. The option /68881 should only be specified if your system contains a 68881 co-processor.

3.5 INSTALL PDOS SYSTEM BOOT ONTO WINCHESTER

The PDOS system boot can reside on one or more Winchester disks. You will need to have initialized any disk to the boot sector numbers instead of the total sector numbers. This is under discussion in the Winchester drive preparation of Chapter 2.3. Upon startup, the PDOS boot ROMs will look for a PDOS boot first at disk 0 then 1 (the two floppy disks). The ROMs will start with the lowest Winchester disk partition number up to disk partition number 99. The process will repeat until a valid PDOS boot is found.

For this section we will assume that disk 2 is the disk you wish to install the PDOS boot on. You will want to run the FxxDOS:GEN procedure file followed by the MMKBT utility. The process of BOOT making is covered in detail in section 2.3. For the initial set up you may want to follow one of the following examples: (MMKBT will display the correct default parameters for you which may be different than shown):

1) Install standard system:

```
        >FxxDOS:GEN
        >MMKBT
        68K PDOS Make Boot Disk Utility 05/06/86
        (F)ile, (M)emory, or (B)ootstrap?        (answer F[CR])
                File name = FxxDOS               (just type [CR])
                Boot size = $00007B14            (just type [CR])
             Load address = $00000800            (just type [CR])
                     Disk = 2                    (enter your disk
                                                 number then [CR])
                   Sector = 47776                (just type [CR])
        Write 124 sector boot beginning at address $0000CD00
        with load address $00000800 to disk 2, sector 47776.
        Ready (Y/N)? Y                           (type Y[CR])
        One moment, please...
          Wrote out header information
          PDOS written successfully
        >
```

2) Install standard system with ANSI terminal support and BASIC with 68881 support:

```
>FxxDOS:GEN /ANS=1,BASIC/68881
>MMKBT
68K PDOS Make Boot Disk Utility 05/06/86
(F)ile, (M)emory, or (B)ootstrap?          (answer F[CR])
        File name = FxxDOS                 (just type [CR])
        Boot size = $000007B14             (just type [CR])
     Load address = $00000800              (just type [CR])
             Disk = 2                      (enter your disk
                                           number then [CR])
           Sector = 47776                  (just type [CR])

Write 124 sector boot beginning at address $0000CD00
with load address $00000800 to disk 2, sector 47776.
Ready (Y/N)? Y                             (type Y[CR])
One moment, please...
  Wrote out header information.
  PDOS written successfully!
>
```

You should now be able to remove the boot floppy from drive 0 and to have the system booted from the ROMs. This can be done by pressing the reset switch on the CPU card. If you have installed the boot correctly, the system will boot from disk 2.

Your PDOS system is now installed and ready to use.

4.0 FORCE UTILITIES

This chapter describes all the utilities which are specific for the FORCE FOCUS Systems.

It covers the following utilities:

1.)	MMKBT	- Make bootable disk
2.)	FTIME	- Set/read PDOS clock or Battery clock
3.)	FFRMT	- Disk format utility
4.)	FLDGO	- Load and/or Go To a new system
5.)	FPARK	- Park drives for shipping
6.)	BLOCKING	- Enable/disable blocking mode (ISCSI-1 Systems)
7.)	FLUSH	- Flush ISCSI-1 buffers
8.)	TIME	- Run time measurement utility
9.)	RR-2	- EPROM programming utility (needs RR-2/RR-3 board)

The utilities numbered 1-8 are located on the PDOS boot floppy. Number 9 is located on the disk labeled "FORCE Utils".

In addition, the following loadable drivers are distributed with the FORCE PDOS System. They are located on the boot disk.

1.)	PR	- Driver for the parallel printer interface of the ASCU board
2.)	PRS	- Driver for the serial printer interface of the ASCU board

* Starting with PDOS 3.3, a new PDOS utility is available called INSTALL. To test it, two installable UART drivers are on the disk "FORCE Utils". They are samples for the ISIO-1/2 and the SIO-1/2.

1)	USISIO1:SR/OBJ	- installable driver for the ISIO-1/2
2)	USSIO1:SR/OBJ	- installable driver for the SIO-1/2

4.1 MMKBT - MAKE DISK BOOT

Name: MMKBT
Function: Installs PDOS system and bootstrap onto disks
Format: >MMKBT

Restrictions: May not be re-entered with the monitor command "GO."

Description: The MMKBT utility is used to install bootable PDOS files onto your disks. The disk should first be prepared for use with the PDOS system.

You are prompted with three options. The first option is (F)ile for creating a new boot disk from a file. The second is (M)emory for creating a new boot disk from a memory area. The last option is (B)ootstrap for putting out special sector information onto physical track 0 of the disk. The FORCE systems do not use the (B)ootstrap option since PDOS boot ROMs are supplied. To select any option, type the letter followed by a [CR].

This utility provides defaults which are the correct parameters in most cases.

(F)ile Option: When the (F)ile option is selected, you can install a boot from a file on your disk . You are prompted for the filename. The file will be read in and sized. You are then asked for the boot size, the load address, the disk number, and the boot sector number. You must confirm the installation before any data is written to the disk.

```
  > MMKBT
  68K PDOS Make Boot Disk Utility
  (F)ile, (M)emory, or (B)ootstrap? F
       Filename = FxxDOS
      Boot size = $0000540C
   Load address = $00000800
           Disk = 0
         Sector = 2336

Write 85 sector boot beginning at address $0000D500
with load address $00000800 to disk 0, sector 2336.
Ready (Y/N)?Y
One moment, Please...
  Wrote out header information.
  PDOS Written Successfully!!
```

(M)emory Option: The (M)emory option allows you to copy memory onto the boot area of a disk. This is typically done to build a boot image of the currently executing system. After executing the MMKBT program, you select the (M)emory option by typing an "M[CR]".

You are asked for the memory start address and end address for the boot. The defaults are determined by your standard system setup. You may select the defaults by typing a [CR] or you may enter new addresses followed by a [CR]. (Remember to use the $ to prefix hexadecimal numbers). The rest of the questions are the same as the file option.

```
  > MMKBT
  68K PDOS Make Boot Disk Utility
  (F)ile, (M)emory, or (B)ootstrap? M
       Start address = $00000800
         End address = $00009800
        Load address = $00000800
                Disk = 0
              Sector = 2336

Write 144 sector boot beginning at address
$00000800 with load address $00000800 to disk 0,
sector 2336.
Ready (Y/N)?Y
One moment, Please...
  Wrote out header information.
  PDOS Written Successfully!!
```

(B)ootstrap option: The (B)ootstrap option is not used with FORCE systems.

4.2 FTIME - Set PDOS/Battery Clock

Name: FTIME
Function: Set the PDOS and/or Battery clocks
Format: >FTIME {P | B }

Restrictions: FTIME only sets the PDOS clock correctly when a battery clock is present and the battery is installed. This utility is only available for the CPU-5 and for all 68020 based CPU-boards. For all other 16-Bit boards MTIME should be used.

Description: Either the letter "P" or the letter "B" may be omitted from the first parameter. If only the letter "P" is specified, then the PDOS clock is set from the battery clock. For example, to set the PDOS clock from the battery clock you would type:

```
>FTIME P
```

When the first parameter is the letter "B", the battery clock is set to the current PDOS clock values. This is usually done after the PDOS clock has been set with the "ID" monitor command.

```
>ID
PDOS/68020 R3.3 14-Jan-88
ERII, Copyright 1983-1988
FORCE CPU-20/21 BIOS 04-Feb-88
DATE=00/00/00 02 05 88
TIME=00/00/01 10 15 00
>FTIME B
>_
```

When no parameters are specified, the values of both the battery and PDOS clocks are displayed.

```
>FTIME
ASCU CLOCK = 05-Feb-88 14:47:54
PDOS CLOCK = 05-Feb-88 14:47:56
```

Note: The year is kept in the RAM of the real-time clock. If the year rolls over (i.e. from 1987 to 1988), the year has to be set with the ID Monitor Command and the real-time clock has to be loaded.

The following Battery clocks are supported by FTIME:

1. CPU-5 ASCU-1/2 board at address $B02000

2. CPU-20/21 ASCU-1/2 board at address $FCB02000 (Standard VME access, user and supervisor data.)

3. CPU-29 ASCU-1/2 board at address $FCB02000 (Standard VME access, user and supervisor data.)

4.3 FFRMT - DISK HARDWARE FORMAT

Name: FFRMT
Function: Hardware format disks and set up PDOS partitions
Format: >FFRMT

Restrictions: FFRMT may only be run when no other tasks are running.

Description: FFRMT allows you to define drives and to format and partition disk drives. This utility is hardware dependent. PDOS will support up to four drives on one or two controllers. This utility is menu driven. Refer to section 2 in this guide for complete details on its use. The following is only a summary.

When you run this utility, you may select a drive to access (i.e. F, F0, F1 for the floppy diskette drives or W, W0, W1, W2, W3 for up to 4 Winchester drives). Enter the drive letters followed by a [CR] to access the drive. If the drive is undefined, you will be prompted with the drive select byte for the controller.

```
               >FFRMT
               68K PDOS FORCE Format Drive Utility 05/23/85
                Possible Disk Controllers in this System are:
                    Controller #1 is a FORCE SASI-1
                    Controller #2 is a FORCE WFC-1
                    Controller #3 is a FORCE ISCSI-1
                Drives that are currently defined in system are:
                    F0 is controller #3, drive select byte $73
                    F1 is controller #3, drive select byte $74
                    W0 is controller #3, drive select byte $00
                    W1 is undefined.
                    W2 is undefined.
                    W3 is undefined.

Select Menu: W,W0,W1,W2,W3=Winch; F,F0,F1=Floppy; Q=Quit
     Select Drive: _
```

If you select either a floppy drive or a Winchester drive that is already defined, FFRMT directly enters the Drive Command Menu. If you are installing a new Winchester drive which is currently undefined, then you must enter the controller number and drive select jumpering (0-3). The Drive Command Menu tells you which drive you are currently dealing with and has the following commands:

```
Select Menu: W,W0,W1,W2,W3=Winch; F,F0,F1=Floppy; Q=Quit
     Select Drive: .W0[CR]
W0 Main Menu:
     1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
     Command:  [CR]

Winchester Drive 0 Menu:
     1) Display/Alter drive Parameters.
     2) Display/Alter Bad Track List.
     3) Format tracks.
     4) Verify tracks.
     5) Display/Alter PDOS Disk Partitions.
     6) Write out Header info to disk.
     P) Toggle Unit 2.
     Q) Quit & Select another Drive.
W0 Main Menu:
     1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
     Command:  _
```

When dealing with a floppy drive, the display/alter commands do not allow you to alter the drive parameters, the bad track table, or the disk partitions, and you may not write out header information to a floppy disk. To exit to PDOS, you must first return to the Select Drive Menu with the Q) command. Following is a detailed description of the Drive Command Menu commands:

1) <u>Display/Alter Drive Parameters:</u>

The Display/Alter Drive Parameters menu allows you to D)isplay the currently defined drive parameters, A)lter them, R)ead them in from a file, or Q)uit and exit to the Select Drive Menu:

```
W0 Parameters Menu:  A)lter, D)isplay, R)ead file, Q)uit
     Command: _
```

To display the current drive parameters on a Winchester, enter the `D' command. The parameters are displayed to the screen. The Drive Parameters that are displayed, and that can be altered are:

```
      Current (type) Drive N Parameters:
                # of Heads = Number of heads on drive
            # of Cylinders = Number of cylinders on drive
 Physical Blocks per Track = Actual blocks on a track
  Physical Bytes per Block = Actual bytes per physical block
         Shipping Cylinder = Where to position head before
                             moving drive
                 Step rate = Controller dependent definition
 Reduced write current cyl = Cylinder to apply reduced
                             write current
   Write Precompensate cyl = Cyl to apply write precompensation
```

To alter them, enter the `A' command. In the alter mode, you enter either: 1) a carriage return to leave the parameter the same and go to the next prompt; 2) a number and a carriage return to change the parameter and go to the previous parameter prompt. The Drive Parameters are displayed one at a time for you to either alter or leave alone.

If you have previously saved out the drive parameters to a disk file, you can restore them by entering the `R' command, followed by the name of the file. This file may be created using the F)ile command of Drive Command Menu option 6) Write to disk, or it can be created with a PDOS editor. The format and syntax of the parameter file is discussed under option 6). Reading this information destroys all other information; replaces the parameters, the bad track table, and the partition definitions.

The `Q' command returns you to the Drive Command Menu.

For example, look at floppy drive F0 parameters:

```
  Select Menu: W,W0,W1,W2,W3=Winch; F,F0,F1=Floppy; Q=Quit
       Select Drive:  F0[CR]
  F0 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  1[CR]

  Current Floppy Drive 0 Parameters:
                # of Heads = 2
            # of Cylinders = 80
 Physical Blocks per Track = 16
  Physical Bytes per Block = 256
         Shipping Cylinder = 0
                 Step rate = 0
 Reduced write current cyl = 0
   Write Precompensate cyl = 0
  F0 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command: _
```

As another example, select the W0 Winchester and display the current parameters:

```
  W0 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  1[CR]
  W0 Parameters: A)lter, D)isplay, R)ead file, Q)uit
       Command:  D[CR]

  Current Winch Drive 0 Parameters:
                # of Heads = 16
            # of Cylinders = 1000
 Physical Blocks per Track = 32
  Physical Bytes per Block = 256
         Shipping Cylinder = 0
                 Step rate = 0
 Reduced write current cyl = 0
   Write Precompensate cyl = 0

  W0 Parameters Menu:  A)lter, D)isplay, R)ead file, Q)uit
       Command:  Q[CR]
  W0 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command: _
```

2) Display/Alter Bad Track List:

The Display/Alter Bad Track menu allows you to D)isplay the currently defined bad tracks on the drive (if any), add or delete tracks, C)lear the bad track table, get a H)elp message, or Q)uit and exit to the Drive Command Menu:

```
  W0 Bad Tracks Menu:  Bad Track, D)isplay, C)lear, H)elp, Q)uit
       Command:  _
```

To display the current bad tracks on a Winchester, enter the `D' command. The tracks are displayed on the screen in ascending order as a physical track number followed by the head and cylinder number, separated by a comma and enclosed in parentheses.

To add a bad track to the list, enter either the actual physical track number and a carriage return, or the head and cylinder number desired, separated by a comma and followed by a carriage return. To delete a track, precede the track or head number with a minus sign (-).

Sometimes the bad track table may be incorrect or spoiled. You can start all over again by entering the C)lear table command. The `Q' command returns you to the Drive Command Menu. In case you have added or deleted some bad tracks, FFRMT asks if you want to recalculate the disk partitions on the drive before returning to the drive menu. By altering the number of bad tracks, you also alter the number of logical tracks available for PDOS disk partitions. Answer `Y' or `N' to the query, as you like.

Note that the SCSI Winchesters handle bad blocks internally. So when you are using the ISCSI-1 controller, the bad blocks defined by the manufacturer are already spared on the disk.

For example, look at the Winchester drive 0 bad track list:

```
  W0 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  2[CR]
  W0 Bad Tracks Menu: Bad Track, D)isplay, C)lear, H)elp, Q)uit
       Command:  D[CR]

  Current Winch Drive 0 Bad Tracks:
231(0,77)    613(1,204)    697(1,232)    700(1,233)    703(1,234)

  W0 Bad Tracks Menu: Bad Track, D)isplay, C)lear, H)elp, Q)uit
       Command:  Q[CR]
  W0 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  _
```

3) Format Drive/Tracks:

```
          Sector Interleave = {default from MCONTB table is listed}
 Physical Tracks to Format = {[CR] for beg,end tracks listed}
   Ready to Format Drive 0 ? {`Y' or `N'}
```

This routine first calls the INFMT routine which sets up the format. Then F)ormat makes one or more calls to the TKFMT routine until all the specified tracks are formatted. Between calls, a check for user break ([CTRL-C]) is made, and the track number just formatted is printed to the terminal. If there are errors, you can select either R)etry, Y)es -- add the track to the bad track list, or N)o -- ignore the error and go on.

For example, format a floppy disk with the default sector interleave, 5, and do tracks 0 to 159, inclusive:

```
       Sector Interleave = 5[CR]
       Physical Tracks to Format = 0,159[CR]
       Ready to FORMAT Floppy Drive 0 ? Y[CR]
       Sector Interleave Table:
     1,9,4,12,7,15,2,10,5,13,8,16,3,11,6,14

       Issuing Format Drive Command

       FORMAT Successful!
```

Note that the interleave is "Don't care" for SCSI Winchester drives.

4) Verify Tracks:

```
    Physical Tracks to Verify = {default from last format command}
                         Ready ? {`Y' or `N'}
```

This routine, after calling INFMT, reads every sector on each track specified. Errors are reported to the terminal. Between calls a check for user break ([CTRL-C]) is made, and the track just verified is printed to the terminal. If there are errors, you can select either R)etry, Y)es -- add the track to the bad track list, or N)o -- ignore the error and go on.

5) Display/Alter Disk Partitions:

The Display/Alter Partitions menu allows you to D)isplay the currently defined disk partitions, A)lter them, R)ecalculate them from the current values, or Q)uit and exit to the Drive Command Menu:

```
  W0 Partitions Menu: A)lter, D)isplay, R)ecalc, Q)uit
       Command: _
```

To display the current disk partitions on a Winchester, enter the `D' command. The partitions are displayed on the screen. The Disk Partitions that are displayed are based on a few parameters, which you can change:

# of Large partitions	= How many large divisions on the drive
# of Floppy partitions	= How many small divisions on the drive
First track for PDOS Parts	= Where to begin the disk partitions
Last track for PDOS Parts	= Where to end the disk partitions
First PDOS disk #	= What is first PDOS disk # of partitions

To alter them, enter the `A' command. In the alter mode, you enter either: 1) a carriage return to leave the parameter the same and go to the next prompt; 2) a number and a carriage return to change the parameter and go to the next prompt; or 3) an escape to go to the previous parameter prompt. The disk partitions parameters are displayed one at a time for you to either alter or leave alone. If you alter the number of disks or the tracks for partitions, then you are asked if you would like to recalculate the partitions. Enter either `Y' or `N'. If you only change the beginning PDOS disk number then only the disk numbers are reassigned, leaving the base and top tracks of the partitions alone.

You can make the partition information consistent by simply entering the `R' command. This recalculates the drive partition information using the current values of drive parameters, bad track table, and partition parameters. The `Q' command returns you to the Drive Command Menu.

For example, look at floppy drive F0 partitions:

```
       Current Floppy Drive 0 Disk Partitions:
     # of Large partitions = 0
    # of Floppy partitions = 1
First track for PDOS Parts = 0
 Last track for PDOS Parts = 159
         First PDOS disk # = 0
 Total # of Logical Tracks = 160
 Disk #     Base Trk    Top Trk    # of PDOS sectors
    0          0          159       2528/2336{boot}
```

As another example, examine the partitions on W0:

```
  W0 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  5[CR]
  W0 Partitions Menu: A)lter, D)isplay, R)ecalc, Q)uit
       Command:  D[CR]

  Current Winch Drive 1 Disk Partitions:
     # of Large Partitions = 10
    # of Floppy Partitions = 12
First track for PDOS Parts = 0
 Last track for PDOS Parts = 15979
         First PDOS disk # = 2
 Total # of Logical Tracks = 16000
 Disk #    Logical Trks  Physical Trks   PDOS sectors
           Base,Top      Base,Top        Total/{boot}
  2        0,1499        0,1500          47968/47776
  3        1500,2999     1501,3000       47968/47776
  4        3000,4499     3001,4500       47968/47776
  .        .             .               .
  .        .             .               .
  .        .             .               .

 24        15880,15959   15897,15979     2528/2336
  W0 Partitions Menu: A)lter, D)isplay, R)ecalc, Q)uit
       Command:  Q[CR]
  W0 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  _
```

6) Write Header Information to Drive:

The Write Header Information to Drive menu allows you to 1) `Y' write the information to the drive header, 2) `N' abort the command and return to the Drive Command Menu, or 3) `F' write drive information to a file. After assigning the correct parameters to a drive, entering the bad tracks, formatting it, and partitioning it into PDOS disk numbers, you still need to write this information to the drive's header. This information must reside on the disk and is used by for the BOOT ROMs and by PDOS.

This routine calls the INFMT subroutine to initialize the controller for the new number of heads and cylinders, and then calls the WTHED subroutine which writes out the drive data block (DDB) to the correct sector on the drive, usually sector 0.

To write this information to the drive, enter the `Y' command. If you have second thoughts, enter the `N' command.

You should save the information out to a floppy disk file for each Winchester drive. This file makes recovering from Winchester disasters easier. You can either select to only write out the file with the `F' command, or write the file out after writing out the header information to the drive.

The file syntax is that:

1) lines starting with an asterisk (*) are ignored as comments;
2) parameter keywords are four characters long and appear as the first four characters of the line;
3) keywords are followed by an equal sign (=) and the value (hex must be preceded with dollar sign ($));
4) bad tracks use the keyword TRACK, are followed by an equals sign, and are designated by either the track number or the head and cylinder numbers (separated by a comma);
5) order of the keywords is not significant, except that the HEDS definition must precede any TRACK specification using the head, cylinder format; and
6) any unspecified keyword parameters are reset to system defaults, and not left as previously entered values.

The drive parameter keywords are defined as follows:

HEDS = # of Heads
CYLS = # of Cylinders
BPTK = Physical Blocks per Track
BPBK = Physical Bytes per Block
SHIP = Shipping Cylinder
STEP = Step Rate
REDU = Reduced Write Current Cylinder
WRTP = Write Precompensate Cylinder

The disk partition keywords are defined as follows:

WPRT = # of Large Partitions
FPRT = # of Floppy Partitions
BTRK = First Track for PDOS Parts
ETRK = Last Track for PDOS Parts
BDKN = First PDOS disk #

While reading in the file using the R)ead command of the 1)Parameter menu, FFRMT outputs a `Found:' message, followed by the parameter value when a successful keyword match and number conversion is made. This indicates that the parameter was loaded. If a keyword match is not made or if the conversion fails, FFRMT echoes the line to the terminal preceded by two question marks (??). This indicates that the parameter was not loaded.

In the following example, the file, TEMP is first written, shown, and then re-read with FFRMT:

```
  W0 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  6[CR]
  Write to Disk Y)es, N)o, F)ile : F[CR]
  Write Parameter File:#TEMP/0[CR]
  W0 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  Q[CR]
  Exit to Select Drive.  Update Param RAM (Y/N) ? N[CR]
  Select Menu: W,W0,W1,W2,W3=Winch; F,F0,F1=Floppy; Q=Quit
       Select Drive:  Q[CR]

2>SF -TEMP[CR]
* Drive parameters:
HEDS=16
CYLS=1000
BPTK=32
BPBK=256
SHIP=0
STEP=0
REDU=0
WRTP=0
* Drive partitions:
WPRT=1
FPRT=16
BTRK=0
ETRK=2489
BDKN=2
2>FFRMT[CR]
68K PDOS FORCE Format Drive Utility 09/15/86
  Select Menu: W,W0,W1,W2,W3=Winch; F,F0,F1=Floppy; Q=Quit
       Select Drive:  W1[CR]
  Drive not defined in system.
    What controller # ? 3[CR]
    What is Winch wired for (0-3) ? 1[CR]
  W1 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  1[CR]
  W1 Parameters Menu:  A)lter, D)isplay, R)ead file, Q)uit
       Command: R[CR]
  Read Parameter File:TEMP[CR]
  Processing file...
```

```
Found:                    # of Heads = 16
Found:                # of Cylinders = 1000
Found:     Physical Blocks per Track = 32
Found:      Physical Bytes per Block = 256
Found:             Shipping Cylinder = 0
Found:                     Step rate = 0
Found:Reduced write current cylinder = 0
Found:       Write Precompensate cyl = 0
Found:         # of Large partitions = 10
Found:        # of Floppy partitions = 12
Found:    First track for PDOS Parts = 0
Found:     Last track for PDOS Parts = 15979
Found:             First PDOS disk # = 2

  W1 Parameters Menu:  A)lter, D)isplay, R)ead file, Q)uit
       Command: Q[CR]
  W1 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  2[CR]
  W1 Bad Tracks Menu: Bad Track, D)isplay, C)lear, H)elp, Q)uit
       Command:  D[CR]

  Current Winch Drive 1 Bad Tracks:
  W1 Bad Tracks Menu: Bad Track, D)isplay, C)lear, H)elp, Q)uit
       Command:  Q[CR]
  W1 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  5[CR]
  W1 Partitions Menu: A)lter, D)isplay, R)ecalc, Q)uit
       Command:  D[CR]

  Current Winch Drive 1 Disk Partitions:
     # of Large Partitions = 10
    # of Floppy Partitions = 12
First track for PDOS Parts = 0
 Last track for PDOS Parts = 15979
         First PDOS disk # = 2
 Total # of Logical Tracks = 16000
 Disk #    Logical Trks  Physical Trks   PDOS sectors
           Base,Top      Base,Top        Total/{boot}
  2        0,1499        0,1500          47968/47776
  3        1500,2999     1501,3000       47968/47776
  4        3000,4499     3001,4500       47968/47776
  .        .             .               .
  .        .             .               .
  .        .             .               .

 24        15880,15959   15897,15979     2528/2336

  W1 Partitions Menu: A)lter, D)isplay, R)ecalc, Q)uit
       Command:  Q[CR]
  W1 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  _
```

Q) Select Another Drive:

If you were working with a floppy drive, the Q)uit command simply returns you to the Drive Select Menu. If you were working with a Winchester, then the Q)uit command asks whether or not to write the new drive data block down to low parameter RAM. Enter either `Y' or `N' to this query. If you answer `N', your configuring session will be lost. It then exits to the Drive Select Menu.

```
  W0 Main Menu:
       1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
       Command:  Q[CR]
  Exit to Select Drive.  Update Param RAM (Y/N) ? Y[CR]
  System Parameter RAM Updated!!
  Select Menu: W,W0,W1,W2,W3=Winch; F,F0,F1=Floppy; Q=Quit
       Select Drive:  Q[CR]

2>_
```

4.4 FLDGO - LOAD AND/OR GO TO A NEW SYSTEM

Name: FLDGO

Function: Load into memory and/or execute new system.

Format: >FLDGO {<load address>}{,<filename>}

Restrictions: FLDGO will replace your current PDOS operating system and execute a new system terminating all tasks.

Description: FLDGO is used to load and execute new PDOS systems.

The load address is the location in memory where the program is to be located. The default of $800 is assumed.

The filename is the name of your system file. If a filename is not given, then FLDGO will look for a PDOS system in your task space. FLDGO will only load a file in which the PDOS ID characters are found. After FLDGO has loaded your new system, it will jump to the load address and begin execution.

The following is an example on a FORCE FOCUS system 21A:

```
    >FLDGO ,FxxDOS
```

You then see something similar to the following:

```
             DOS File Loaded: FxxDOS
    Found PDOS at address: $00100EB6
                  DOS size is $00008BD4
```

(The startup file on the new version is now executed).

```
    ....
```

FLDGO allows you to try a new version of PDOS without modifying your disk boot image. When you reboot via the reset switch or by powering down then up, your boot will come off the disk unit selected by the boot ROMs. To make this new system into a disk boot you need to follow the instruction for MMKBT.

4.5 FPARK - PARK DRIVES FOR SHIPPING

Name: FPARK
Function: Flush buffers and park drives for shipping
Format: >FPARK

Description: This utility does nothing on the standard FORCE system. You need to use this utility if you use drives that must be parked before they are shipped. Define a shipping cylinder under option 1 of the FFRMT utility.

4.6 BLOCK - ENABLE/DISABLE BLOCKING MODE (ISCSI-1 ONLY)

Name: BLOCK

Function: This utility toggles the blocking enable/disable mode of the ISCSI-1 controller.

Format: >BLOCK

Restrictions: This utility is only useful in systems with an ISCSI-1 controller at the default address.

Description: This utility toggles the blocking enable/disable mode of the ISCSI-1 controller (i.e. blocking enabled or blocking disabled). This utility should be put in the startup file SY$STRT if you want to use the blocking mode. If blocking is enabled you must make sure that all buffers which are kept in the RAM of the ISCSI-1 controller are updated to disk. To do so, the "FLUSH" utility or the "FPARK" utility can be used. One of the two must be invoked whenever the system is shut down.

In addition, the ISCSI-1 controller is doing an automatic flush every 20 seconds. So when blocking is enabled when you crashed your system and the processor comes to a halt, wait at least 20 seconds before you press reset or turn the power off, so that the ISCSI-1 controller has a chance to write the buffer to disk.

For detailed information please refer to the ISCSI-1 hardware user's manual.

Example:

```
>BLOCK
  Blocking enabled
>BLOCK
  Blocking disabled
```

4.7 FLUSH - FLUSH ALL ISCSI-1 BUFFERS TO DISK

Name: FLUSH

Function: This utility writes all modified buffers of the ISCSI-1 controller to disk.

Format: >FLUSH

Restrictions: This utility is only useful in systems with an ISCSI-1 controller at the default address.

Description: If blocking was enabled on the ISCSI-1 disk controller (see 4.6, BLOCK), this utility can be used to write all modified buffers, which are kept local on the ISCSI-1, to the disk. The command has no effect if blocking is disabled, or if no ISCSI-1 controller is used in the system.

Example: >FLUSH

4.8 TIME - MEASURE THE RUN-TIME OF A PROGRAM

Name: TIME
Function: Measure the run time of a user program.
Format: >TIME <program>

Restrictions: This utility can only be used for location independent programs which reside in Main Memory. Therefore, all programs which are loaded from disk (or RAM disk) can be used, but ROM resident programs or PDOS monitor commands (or command files) are not allowed.

Description: This utility loads the program to RAM using the XLDF primitive, preserves the current TIC counter value and sets a new XEXT and XERR address; then the program is invoked. When the program terminated control is passed back to the time utility, the TIC counter is read again, then the difference is calculated and printed to the terminal.

Example:

```
>TIME CC SIEVE
 USER TIME = 18.70

>
```

4.9 RR-2 - EPROM PROGRAMMING UTILITY

This utility requires a SYS68K/RR-2/3 board at the default address. A complete documentation of this utility is included in the RR-2/3 manual and it can also be found on the FORCE utility floppy.

4.10 PR - DRIVER FOR THE PARALLEL PRINTER PORT OF THE ASCU-1/2

Name: PR

Function: Loadable driver for the parallel printer port of the ASCU-1/2.

Format: >CF <textfile>, PR

Restrictions: Only the data lines, Data Strobe and Busy are used and supported.

Description: PR is a loadable PDOS driver for the parallel printer port of the ASCU-1/2 board. It requires one ASCU-1/2 board on the default address.

For a detailed description of the PDOS loadable Driver mechanism please refer to the Appendix of the PDOS Operating System Manual.

Example: >CF PR:SR,PR this prints a listing of the Driver on the parallel printer

4.11 PRS - DRIVER FOR THE SERIAL PRINTER PORT OF THE ASCU-1/2

Name:	PRS
Function:	Loadable driver for the serial printer port of the ASCU-1/2.
Format:	>CF <textfile>,PRS
Restrictions:	None

Description: PRS is a loadable PDOS driver for the serial printer port of the ASCU-1/2 board. It requires one ASCU-1/2 board on the default address. The driver supports both hardware and software (XON/XOFF) handshakes.

For a detailed description of the PDOS loadable driver mechanism please refer to the Appendix on the PDOS Operating System Manual.

Example: >CF PRS:SR,PRS this prints a listing of the Driver on the serial printer

5.0 FORCE HARDWARE IMPLEMENTATION

FORCE BIOS Modules for 32-Bit CPU-Boards

The Basic Input/Output Systems (BIOS) consists of code in the operating system which is specific to the FORCE hardware. This code is generated from source files supplied to the user for inspection and alteration. The following files comprise the FORCE BIOS:

FILENAME	DESCRIPTION
FPARM:SR	Global memory Definitions. These are common for all FORCE CPU-boards.
FxxPARM:SR	Device addresses and register offsets for all FORCE 32-Bit CPU-boards.
FxxBIOS:SR	Startup, Interrupt, and clock routines.
MBIOS:SR	General startup & tables for all PDOS systems.
FxxBIOSU:SR	UART (Character I/O) code for the devices on the CPU-boards.
FBIOSU:SR	UART Code for the SYS68K/SIO-1/2 and the SYS68K/ISIO-1/2
FBIOSW:SR	Disk code.

5.1 FORCE PARAMETER FILES

The file FPARM:SR defines low ram locations for flags and the disk partition table. This file is used with the BIOS modules, as well as by other special routines (e.g. FFRMT and FxxBOOT). Some low memory locations are used in the BIOS for flags, counters in disk parameters. These are outside of the standard PDOS. The labels that are defined externally for linking by some programs are prefixed with a `P$' as a help in remembering where the label is defined; That is the Parameter file. These parameters are located at absolute memory locations starting at address $400. A definition of FPARM:SR labels along with their meanings follow:

Label	Address	Meaning
	$400	1 byte. Reserved.
P$WFCF	$401	1 byte. Number of WFC controller cards in system. If 0 then no WFC controller cards were found by boot ROMS.
P$SIOF	$402	1 byte. Number of SIO cards found by PDOS startup code.
P$ASCF	$403	1 byte. Number of ASCU cards in the system.

Label	Address	Meaning
P$FPCPF	$404	1 byte. This flag is 0 if no 68881 Co-processors are available. Otherwise it is set to one.
P$SCSIF	$405	1 byte. The number of ISCSI-1 controller cards in the system.
P$ISIOF	$406	1 byte. The number of ISIO-1/2 controller cards in the system.
	$407-$40F	These locations are reserved for future use.
P$DRW	$410	1 long word. This location is used to hold the address of the disk read/write driver in the FxxBIOSW module. The system will crash if this location is destroyed.
P$DFORM	$414	1 long word. This location holds the entry of a specific disk format routine in the FBIOSW driver. It is only used by FFRMT. Currently there is only an entry if the ISCSI-1 controller is used.
P$DMAC	$418	1 long word. This location holds the address of a 68450 DMA controller to be used for disk I/O. It must be 0 if no DMA controller is available.
B$SRAM	$41C	1 long word. This location holds the begin address of the System RAM.
P$CPU	$420	1 word. This word is used to hold an identification of the CPU-board. Currently, only the CPU-20/21 is supported by a value of 0.
P$ISINT	$422	1 word. This word contains the high/low water flags for up to two ISIO-1/2 boards.
STOLOC	$424	1 byte. Time out value for WFC controller. This byte flag is the Select Time-Out counter used only by the WFC-1 code. It is set to 5 by the FBIOSW module when leaving a read or write sector call. Then it is decremented once a second by the W$XDOF routine until it goes to zero. When it reaches zero, the disk is deselected.
P$BDSK	$425	1 byte. This byte variable is loaded by the Boot ROMs with the disk number from which PDOS was booted. The BIOS module reads this location to set the default system disk for PDOS. The FLDGO utility also loads P$BDSK with SDK$ from the user's TCB to simulate booting from a disk.

Label	Address	Meaning
P$INTF	$426	1 byte. Flags for FBIOSW and FxxBIOS to indicate if interrupts or DMA are to be used on either the disk controllers or the ISIO board. The flags are single bits defined as followed: ISAS BIT 0 SASI Interrupts if 1 IWFC BIT 1 WFC Interrupts if 1 DSAS BIT 2 SASI DMA move if 1 DWFC BIT 3 WFC DMA move if 1 ISCSI BIT 4 ISCSI-1 Interrupts if 1 DSCSI BIT 5 ISCSI-1 DMA move if 1
P$LEDB	$427	1 byte. Reserved.
SAVE	$428	2 long words. These are 2 long words used by W$XDIT.
P$UADR	$430	1 long word to save the address of the interrupting UART.
P$PARM	$434	6 long words. Disk drive definition tables for PDOS disk standard. The drive parameters are stored in the locations following P$PARM.

FPARM also has definitions for the offsets and addresses of the various control registers to the hardware which is supported on the standard FORCE system. The hardware includes: SIO and ISIO cards, ISCSI, WFC, ASCU and DMA controller.

The file FxxPARM:SR holds definitions which are specific for the FORCE 32-bit CPU-boards. It contains the base definitions and register offsets for the devices on the CPU-boards.

5.2 FORCE BIOS MODULE : FxxBIOS:SR

The file FxxBIOS:SR has machine-specific, interrupt, clock and mapper routines. Assembling this file includes the files MBIOS:SR, which has general startup code common to all systems, the file FPARM:SR which provides global definitions of the low memory locations, and the file FxxPARM:SR, which holds CPU-board specific device addresses and register offsets. The file FxxBIOS:SR is the main module for the PDOS BIOS and is assembled into the file FxxBIOS:OBJ.

5.2.1 BIOS Labels

The labels that are defined externally for linking with other modules are prefixed with a `B$'as a help in remembering where the label is defined. The following describes the labels as they are used in FxxBIOS:SR:

B$CPU: The code of this label performs the FORCE hardware dependent setup. It first tries to find out which CPU-board is used in the system and then initialzes the interrupt structure and timer of the board.

Then the availability of the 68881 Floating Point Co-processor is checked and a flag is set in low memory.

Finally the VMEbus is checked for the availability of different FORCE hardware.

The following boards are tested and counted:

ISCSI-1	SCSI/floppy controller
WFC-1	Winchester/Floppy Controller
SASI-1	SASI Bus Controller
SIO-1/2	6-Channel Serial I/O Board
ISIO-1/2	Intelligent 8 Channel Serial I/O Board
ASCU-1/2	System Control Unit

B$RAM: This routine reads the front panel switches of the CPU-board and sets the data bus width to the corresponding size. Then the memory is initialized to 0 and checked for read/write capability until a Bus error is generated. The so found top of memory is handed over to PDOS for end of tasking memory.

B$RSW: Here, the front panel switches of the CPU-board are read in to determine the baud rate and character format for the terminal port. The default is 9600 Baud, 8 data bits and no parity. If a different setup has to be used, the front panel switches have to be changed. Note that in this case the PDOS system is not romable.

SYS68K/CPU-20/21 Front Panel Switch Definitions under PDOS:

SW1: Not used by PDOS.

SW2: This switch defines the default Data bus width on the VMEbus/VMXbus. If in "On" position, the data-bus width is 16 bit, if in "Off" position, the size is 32 bit.

SW3: Not used by PDOS.

SW4: Not used by PDOS.

SW5: If this switch is set to "On", PDOS is booted automatically after reset has been pressed.

SW6: Not used by PDOS.

SW7: Not used by PDOS.

SW8: Not used by PDOS.

SYS68K/CPU-29 Front Panel Switch Definitions

Upper Rotary Switch (SW2):

Bit 7: Not used for PDOS.

Bit 6: This bit defines the default data bus size on the VMEbus. If the bit is set to "0", 16 bits are selected, if it is set to "1", 32 bits are selected.

Bit 5: Not used for PDOS.

Bit 4: Not used for PDOS.

Lower Rotary Switch (SW1):

Bit 3: Autoboot PDOS

Bit 2: Not used for PDOS.

Bit 1: Not used for PDOS.

Bit 0: Not used for PDOS.

For example, to autoboot PDOS set SWITCH 2 to F and SWITCH 1 to 7.

B$SAV: This routine is used to save the context of the 68881 coprocessor. If the coprocessor flag in the task control block (SVF$) is set, this routine is called on every context switch.

The register A1 holds the user stack and A7 the supervisor stack. As the supervisor stack area is too small, it is recommended that the coprocessor contents are saved on the user stack.

B$RES: This is the complementary subroutine to B$SAV. It is called if the next task to be rescheduled has the flag SVF$ on the task control block set.

The values to be loaded to the coprocessor have to be popped off the stack. The same stack as in B$SAV has to be used.

B$ACK: This is the acknowledge routine for the system timer interrupt.

B$LED: This performs the fine adjustment of the PDOS System Clock.

BPRT, BMAP: There is neither a protect PDOS (B$PRT) routine nor a mapper (B$MAP) routine in the BIOS for the FORCE series of CPU-boards.

5.2.2 BIOS Interrupts

All interrupts of the default PDOS configuration are handled in the BIOS module. All interrupts are programmed to use different interrupt vectors.

The following interrupt sources are supported by the standard PDOS BIOS:

- On-board I/O devices (including the ABORT switch)
- Disk I/O
- Up to two SIO-1/2 boards for serial I/O
- Up to two ISIO-1/2 boards for serial I/O

The on-board I/O devices are programmed to use the auto vector part of the vector table to leave the user interrupt vectors as free as possible.

The disk controller uses vector #119. The SIO-1 serial I/O controller uses the vectors #64 - #75, while the ISIO-1/2 board uses the vectors #76 - 83. Each serial channel has its own interrupt vector for the SIO-1/2 boards, but the two serial I/O channels of the ISIO-1/2 board share one vector.

5.2.3 On-board Interrupts

The handling of on-board interrupts differs for the different CPU-boards. For a detailed description please refer to the source of the file FxxBIOS:SR.

5.2.4 Disk Interrupt Vector

Exception vector 119 is used by the disk module FBIOSW. The interrupt service routine W$DINT sets the PDOS event 119 and returns. The event number and vector address is defined by the W$EVNT equate in the disk module FBIOSW.

5.3 COMMON BIOS MODULE : MBIOS:SR

The common BIOS module, MBIOS:SR contains the main PDOS cold entry point and startup code, the BIOS table, some default definitions, and a RAM disk initialize routine. MBIOS:SR is included when the file FxxBIOS:SR is assembled.

5.3.1 MBIOS System Constants

If certain system constants are not defined either by the user in the MASM command line or in the FxxBIOS:SR file itself, MBIOS defines them to some default values. A user can alter these temporarily by either adding a divide sign `/', the label, and the desired value to the source file parameter of the MASM assembler. (See section 2.4.1 for an example and details.) More permanent values can be assigned by inserting equate (EQU) statements in the FxxBIOS:SR file.

5.3.2 Task Startup Table

The following labels define the task 0 values to place in the task startup table. This table is used by PDOS to determine what tasks to start up initially, on what ports, at which priorities and what size:

R$TP00	64	Task priority
R$TM00	1	Task time
R$DZ00	0	DSEG size (0=use all of memory)
R$MP00	0	Map # (unused)
R$PM00	*-*	PSEG relative start addr (0=Monitor)
R$PN00	1	Port #

Note that the B$RSW routine in the FxxBIOS module changes the startup table at the label R$TASK to set the baud rate of the terminal according to the front panel switches. So the BIOS module is not romable.

To make the code romable, just remove the lines of the baud rate setting, but in this case the baud rate cannot be set by the front panel switches.

5.3.3 Kernel Callout Subroutines

There are eight subroutines in the BIOS called by the PDOS kernel while running:

BACK, BLED, BMAP, BCTB, BKTB, BPRT, B$SAV and B$RES

The kernel makes these calls to addresses in the BIOS table. These eight routines are not necessarily implemented in a particular PDOS system, but if they are, they reside in the FxxBIOS:SR file for that system. If the label is defined in FxxBIOS:SR, the MBIOS:SR will assemble a long branch (BRA.L B$xxx) to that subroutine. If the label is not defined, MBIOS:SR will assemble a return (RTS) instruction. The details of these routines are contained in section 5.2 and in the **PDOS Reference Manual.**

5.3.4 Screen Command Table

MBIOS also contains a screen command table for BASIC PRINT support. The table is outlined below:

Screen Command Table

```
SCRNTB  DC.B     $0B, `U'             ;U = UP
        DC.B     $0A, `D'             ;D = DOWN
        DC.B     $0C, `R'             ;R = RIGHT

        DC.B     $08, `L'             ;L = LEFT
        DC.B     $0D, `B'             ;B = BEGINNING
        DC.B     $1E, `H'             ;H = HOME
        DC.B     $D9, `S'             ;S = CLEAR TO END OF SCREEN
        DC.B     $D4, `E'             ;E = CLEAR TO END OF LINE
        DC.B     $A7, `W'             ;W = RESET WRITE PROTECT
        DC.B     $A6, `P'             ;P = SET WRITE PROTECT
        DC.B     $A9, `('             ;( = START WRITE PROTECT
        DC.B     $A8, `)'             ;) = END WRITE PROTECT
        DC.B     $AB, `Z'             ;Z = CLEAR UNPROTECTED
        DC.B     $09, `N'             ;N = SKIP TO NEXT FIELD
        DC.B     0                    ;END-OF-TABLE
```

5.3.5 System String Constants

Two PDOS system string constants are defined in the MBIOS:SR file: the system auto start file and the RAM disk name. They may be altered as the user desires. The default names are:

```
SY$STRT = Auto start file name
SY$DSK  = RAM disk name
```

5.4 UART MODULES

5.1.4 FORCE UART Module : FxxBIOSU:SR

The file FxxBIOSU:SR is the UART module for the CPU-board UARTs and contains routines that handle the character I/O devices on the FORCE CPU-boards. Various entry points and assembly definitions are provided in FxxBIOSU:SR. The labels that are defined externally for linking with other modules are prefixed with a `U$'as a help in remembering where the label is defined.

The terminal port on the processor board along with all additional UART devices are defined as type 1 UARTs.

Only port 1 is bauded by PDOS after booting. The procedure file SY$STRT is provided on the boot disk to baud all additional CPU-ports.

5.4.2 FORCE UART Module: FBIOSU:SR

5.4.2.1 Type 2 -- SIO-1 Boards

Up to two SIO-1/2 boards are supported by the standard PDOS. They are defined as type 2 UARTs.

These boards have to be set to default address for the first board and to default +$200 for the second board.

Each channel has its own interrupt vector defined. The used vectors are #64 - 75. The interrupts are handled in the BIOS module.

The SIO-1/2 boards allow many different combinations for the I/O signal assignment. Please refer to chapter 5.2, or to the Hardware User's Manual of the SIO-1/2 boards for detailed information.

The program FBAUD is provided to baud a maximum of 15 serial I/O channels for the PDOS operating system.

5.4.2.2 Type 3 -- ISIO-1/2 Boards

Up to two ISIO-1/2 boards are supported by the standard PDOS. They are defined as type 3 UARTs.

These boards have to be set to default address for the first board and to default address +$20000 for the second board.

Each ISIO-1/2 board supports 4 different interrupt vectors. Therefore two channels share the same vector. The used vector numbers are #76 - 83. The interrupts are handled in the BIOS module.

For a detailed description of the ISIO-1/2 board, and for I/O pin assignment description, please refer to the User's Manual of the ISIO-1/2 board, or to chapter 5.2 of this manual.

The program FBAUD is provided to baud a maximum of 15 serial I/O channels for the PDOS Operating System.

5.4.2.3 Type 4 -- Currently Not Used

5.5 FORCE DISK MODULE : FBIOSW:SR

The file FPARM:SR is included when FBIOSW:SR is assembled to provide global definitions of disk parameters and low memory locations. In addition the physical driver files "ISCSI1:SR" (for the ISCSI-1 board), "WFC1:SR" (for the WFC-1 board) and "SASI-1" (for the SASI-1 controller) are included at assembly time. The labels that are defined externally for linking with other modules are prefixed with a `W' as a help in remembering where the label is defined .

Label W$XDIT is called to initialize the disk controllers. Up to two floppies and 3 Winchester drives are supported by the by the FBIOSW module.

5.5.1 PDOS Unit Defaults and Partitions

Normally PDOS disk numbers 0 and 1 are for the Floppy disk drives. The default floppy disk is disk unit 0 in a single floppy system. Standard PDOS floppy disks treat track 0 as hardware dependent. So units 0 and 1 will access the floppy disks starting with track 1. Units 100 and 101 may be used to access the floppy disk starting with sector 0. The Winchester partitioning must use units less than 100. Units 200 through 299 are used by FBIOSW to access partition information on drives.

PDOS DISK	DESCRIPTION
0	First Floppy Disk
1	Second Floppy Disk
2 through 99	Unit numbers for Winchester
100	Unbiased floppy disk 0
101	Unbiased floppy disk 1
200 through 299	Winchester Partition table access.

The Winchester may be partitioned into multiple disks without regenerating PDOS. This is fully described in section 2 of this manual. The partitioning is handled by tables that are read off track 0 of each Winchester drive. If no valid table is found or no Winchester is installed, a default table is loaded. The default Winchester type and partitioning depends on the controller used.

The parameters are easily changed by using the FFRMT utility provided with your system.

The FORCE PDOS system allocates, in its system parameter RAM at $434, a table (P$PARM) of six Drive Data Block addresses. Two drive data block addresses are for floppy drives (F0 & F1) and four are for Winchester drives (W0 to W3). The addresses of the drive data blocks (or DDBs) are stored by the FBIOSW disk initialize routine, W$XDIT, when PDOS first starts up. The actual DDB's are stored in the systems Parameter RAM area by W$XDIT immediately following the six addresses of the P$PARM table.

The two Floppy Drive Data Blocks are built without accessing the drives, using default parameters, since the floppy drives are common to each system, they have only one partition and do not have bad tracks.

The only difference between the F0 and F1 tables is usually the drive select bytes and the disk number is set respectively to 0 and 1. The drive data block right after the six address pointers corresponds to floppy drive F0. The next DDB in RAM defines floppy drive F1.

The disk initialize routine next searches each possible Winchester drive controller for attached drives by reading the first sector (or block) using all the possible select codes. If a Winchester drive is found installed (no read error), then W$XDIT determines if the header data is actually PDOS Winchester information. If it is correct, then the data is moved into a DDB in system parameter RAM, and the address is saved in the next available P$PARM table location. If the drive is installed but the header data is not a valid PDOS Winchester information, then W$XDIT moves down some default drive data into the DDB in P$PARM. This allows booting a new system and still allowing access to files on non-PDOS Winchester standard drives.

The four Winchester Drive Data Blocks are filled as W$XDIT finds them in the system, altering the controller number and drive select bytes to match where the drive was found. The first Winchester's Drive Data Block is usually read into the systems Parameter RAM area by W$XDIT immediately following the 2 floppy DDB's. We refer to it as drive `W0', but it may be attached to any controller. W1, W2, and W3 are also loaded if drives are found.

5.5.2 Error handling

FBIOSW performs no retries before an error is reported back to PDOS.

The following Disk errors detected are reported to PDOS. Errors concerning the state of the controller are also reported.

DISK ERROR DEFINITIONS:

Common Errors

100	Illegal Drive
101	Sector too big
102	Timeout

WFC-1 Errors

103	Write Protected
104	AM not found
105	No TK00
106	Not ready
107	DMA error
108	Sect Not Found
109	ID read
110	Uncorrectable data

SASI-1 Errors

111	Bad Block
112	Channel active
113	No index
114	No seek
115	Write Fault
116	Not ready
117	Not selected
118	No TK00
119	Multiple select
125	Seek IP
128	ID read
129	Uncorrectable data
130	IDAM not found
131	DTAM not found
132	RNF
133	Seek error
135	Write protected
136	Corrected data
137	Bad track
138	Format error
143	Seq TO
144	Bad cmd
145	Illegal disk adr
146	Illegal function

ISCSI-1 Errors

147	SCSI Parity error
153	Illegal Command
157	Address Error
158	ISCSI-1 Exception
159	Target Mode
161	SCSI timeout
162	LUN not ready
163	SCSI read error
164	SCSI write error
165	Block not found
166	Target reservation conflict
167	Format error
172	Floppy not ready
173	FDC read error
174	FDC write error
175	FDC record not found
176	Floppy write protected
177	FDC format error

5.5.3 Data Movement and Interrupts

The disk driver uses programmed I/O with interrupts. This allows lower priority tasks to execute during disk accesses. A flag in low memory, P$INTF, signals the FBIOSW routine whether or not to use interrupts. If the bit ISCSI (for a ISCSI-1 controller) in P$INTF is zero, then the command RAM is polled for Channel Operation Complete to be set. This mode is used by the W$XDITS routine and the boot EPROMs since using interrupts requires that PDOS is in control. Upon exit, W$XDITS sets the bit P$INTF to 1 enabling completion interrupts. The read/write routine checks this flag and when it is set, sets the local event (128) to delay for three seconds. This is a timeout in case the controller goes away or fails to interrupt. Then suspends on event 128 and system event 119. When the controller is interrupting or after three seconds. When the task awakens, event 119 is automatically reset for the next disk request. If the timeout awoke the task, then the flag in P$INTF is reset and regular polling of the command completion is entered.

USER NOTES

USER NOTE

Dear Customer,

The miniFORCE 2P systems, such as 2P2, 2P4, or 2P5; contain a SYS68K/SIO-2 serial I/O board instead of the SYS68K/SIO-1, as described in this manual.

FORCE COMPUTERS has decided to implement the SYS68K/SIO-2 board as standard for the above systems. The SYS68K/SIO-2 is fully functional compatible to the SYS68K/SIO-1, but for detailed information, please refer to the delivered SYS68K/SIO-2 User's Manual.

Thank you.

FORCE COMPUTERS GmbH